НАЦІОНАЛЬНА АКАДЕМІЯ НАУК УКРАЇНИ
ІНСТИТУТ ПРОГРАМНИХ СИСТЕМ

ISSN 1727-4907

П Р О Б Л Е М И ПРОГРАМУВАННЯ

НАУКОВИЙ ЖУРНАЛ

P R O B L E M S IN PROGRAMMING

SCIENTIFIC JOURNAL

2016

№ 1

Теми випуску:

- ***Теоретичні та методологічні основи програмування***
- ***Моделі та засоби систем баз даних і знань***
- ***Експертні та інтелектуальні інформаційні системи***
- ***Прикладні засоби програмування та програмне забезпечення***
- ***Математичне моделювання об'єктів та процесів***

НАЦІОНАЛЬНА АКАДЕМІЯ НАУК УКРАЇНИ
ІНСТИТУТ ПРОГРАМНИХ СИСТЕМ

П Р О Б Л Е М И ПРОГРАМУВАННЯ

науковий журнал

Головний редактор
Андон Пилип Іларіонович
академік НАН України,
директор Інституту програмних систем
НАН України
✉ Інститут програмних систем
НАН України
Проспект Академіка Глушкова, 40
03187, Київ-187
☎ Тел.+380 (44) 526 5507
E-mail: ANDON@ISOFTS.KIEV.UA
http://www.progproblems.org.ua

Адреса для кореспонденції

✉ Інститут програмних систем
НАН України
Проспект Академіка Глушкова, 40
03187, Київ-187

☎ Тел.: +380 (44) 526 5065
Факс: +380 (44) 526 6263
E-mail: ISS@ISOFTS.KIEV.UA

Редактор *В.П. Замула*
Комп'ютерна верстка *В.П. Замула*

Підписано до друку 05.02.2016. Формат 60x84/8. Папір офс. Ум. друк. арк. 15,58. Обл.-вид. арк. 13,00. Тираж 120 прим. Ціна договірна. Замовл.

Друкарня Видавничого дому «Академперіодика» НАН України
01004 Київ-4, вул. Терещенківська, 4

Свідоцтво про внесення до Державного реєстру суб'єкта видавничої справи
серії ДК № 544 від 27.07.2001 р.

НАЦІОНАЛЬНА АКАДЕМІЯ НАУК УКРАЇНИ
ІНСТИТУТ ПРОГРАМНИХ СИСТЕМ

ПРОБЛЕМИ ПРОГРАМУВАННЯ

науковий журнал

№ 1 січень-березень 2016

Заснований у березні 1999 р.

ЗМІСТ

Свідоцтво про державну реєстрацію КВ № 7490 від 01.07.2003

Науковий журнал "Проблеми програмування" занесений до переліку наукових фахових видань України, в яких можуть публікуватися основні результати дисертаційних робіт.

ISSN 1727-4907

NATIONAL ACADEMY OF SCIENCES OF UKRAINE
INSTITUTE OF SOFTWARE SYSTEMS

PROBLEMS IN PROGRAMMING

scientific journal

№ 1 **January – March** ***2016***

Founded in March, 1999

CONTENTS

УДК 004.4

Е.М. Лаврищева, О.А. Слабоспицкая, А.Ю. Стеняшин, А.Л. Колесник

ОБЪЕКТНО-КОМПОНЕНТНАЯ РАЗРАБОТКА ИЗМЕНЯЕМЫХ ПРОГРАММНЫХ СИСТЕМ

Объектно-компонентный метод (ОКМ) моделирования программных систем Лаврищевой – Грищенко развит согласованными моделями вариабельности систем и их вариантной конфигурационной сборки из компонентов, а также алгеброй операций изоморфного преобразования нерелевантных типов данных для этих компонентов. Введение в модель системы точек вариантности с их вариантами обеспечивает изменяемость систем и устойчивое взаимодействие их компонентов. Описана реализация формального аппарата в автоматизированном конфигураторе систем и его апробация в инструментально-технологическом комплексе ИПС НАНУ и экспериментальной фабрике программ КНУ им. Т. Шевченко.

Ключевые слова: объект, компонент, объектно-компонентный метод, модель вариабельности, артефакт, вариантная точка, готовый ресурс, управление моделями, конфигурационная сборка.

Введение

Большинство современных подходов к изменению сложной программной системы (ПС) все еще предполагают непосредственную корректировку ее кода. Она усложняет ПС, снижает ее качество и требует дополнительных затрат времени и ресурсов на устранение вносимых дефектов. Эти негативные последствия все более критичны в актуальных сегодня динамичных слабо формализованных предметных областях. Однако именно в них изменяемые ПС особенно востребованы из-за растущей нестабильности ожиданий потребителей и условий выполнения. Поэтому технологии разработки ПС, многократно изменяемых без доступа к коду, становятся актуальным вызовом программной инженерии.

Целый ряд подходов, предложенных в ответ на этот вызов, объединяет использование формализма Семейства ПС [1]. Это множество ПС с общим набором постоянных понятий и характеристик, а также поднаборами изменяемых характеристик отдельных ПС, названных вариантными. Изменения ПС, допустимые в их семействе, описываются моделью его вариабельности – пригодности ПС или артефакта к эффективному развитию, изменению, настройке или конфигурированию для использования в определенном контексте [1–3]. Стандартом де-факто служит модель характеристик (Feature Model). Характеристика – функция или показатель качества ПС, востребованный группой заинтересованных лиц, их требование либо ожидание.

Модель характеристик поддерживает проектирование изменяемых ПС как членов семейства. Однако их автоматизированную сборку из готовых ресурсов затрудняет отсутствие формализмов связывания ресурсов с характеристиками и композирования для их набора.

Альтернативный базис обеспечения изменяемости ПС предоставляет сборочное программирование [4, 5]. Заявленное в 1982 г. методом сборки больших систем из готовых модулей [6], сегодня оно интенсивно развивается в разнообразных формах: от сборочного конвейера (М. Фаулер, корпорация EPAM System) до фабрик индустриального производства программ (Дж. Гринфильд, И. Бей, Г. Ленц, технология AppFab) [7]. В Украине оно представлено, прежде всего, ОКМ моделирования ПС Лаврищевой – Грищенко [5, 8–12]. Преимущество ОКМ – это алгебры операций реинжиниринга компонентов повторного использования (КПИ) и сборки ПС из КПИ. Но ОКМ-разработке изменяемых ПС препятствует слабая предсказуемость характеристик такой сборки и ее реактивность (“with reuse”, а не “for reuse” [1]).

Цель работы – непротиворечивая интеграция технологий разработки семейств ПС, поддерживающих предсказуемые изменения наиболее эффективно,

и ОКМ для преодоления их ограничений. Статья обобщает результаты авторов в проекте ДР 0107U002205 ИПС НАН Украины под руководством доктора физико-математических наук, профессора Е.М. Лаврищевой.

Подходы к изменению систем

Конструирование линеек программных продуктов. На основании материалов [1] для развития ОКМ выбраны две технологии:

– конструирование линеек программных продуктов К. Пола (K. Pohl) [2];

– генерирующее программирование К. Чернецки (K. Czarnecki) [3].

Определяющая особенность первой технологии – представление процесса разработки семейства ПС взаимодействующими подпроцессами инженерии домена и приложений, которые координируются подпроцессами организационного и технического управления. Инженерия домена предназначена для определения обязательных и опциональных характеристик ПС в семействе (в виде модели характеристик) и создания готовых ресурсов их реализации (требований, архитектур, фрагментов кода и тестов, структур данных) вместе с правилами композирования ресурсов в ПС.

Модель характеристик формируется в подпроцессе задания границ семейства. Это иерархия опциональных и общих характеристик ПС с отношениями [2]:

– вариантного подчинения (подчиняющая характеристика реализуется в ПС только при реализации строгого подмножества подчиненных характеристик);

– импликации (реализация предпосылки влечет реализацию следствия);

– эквивалентности (характеристики реализуются в ПС одновременно);

– исключения (одновременная реализация характеристик недопустима).

Ресурсы создаются в подпроцессах анализа требований, реализации, проектирования и тестирования, образуя четырехуровневую платформу семейства.

В процессе инженерии приложений ресурсы платформы автоматизировано композируются, по заданным правилам, для реализации заданного набора обязательных и/или опциональных характеристик – подграфа модели характеристик.

В подпроцессах организационного и технического управления координируются операции инженерии домена и приложений, а также планируется и отслеживается создание и использование в ПС ресурсов платформы.

Генерирующее программирование изменяемых ПС. Вторая технология, интегрируемая с ОКМ, реализует парадигму разработки ПС и их семейств под девизом "от ручного труда к конвейерной сборке" с помощью специального формализма – генерирующей модели домена. В отличие от технологии К. Пола, для ее построения вводятся пространства *проблем* и *решений*, а также *база конфигурации* семейства ПС.

Пространство проблем содержит обязательные и опциональные понятия и характеристики ПС семейства. Для его представления широко используется модель характеристик.

В свою очередь, пространство решений объединяет объекты конвейерной сборки. Это готовые ресурсы реализации характеристик (каркасы, шаблоны, модули, КПИ, сервисы, артефакты разработки), представленные в современных языках программирования, предметно-ориентированных языках и языках описания интерфейсов. Наконец, элементами базы конфигурации являются механизмы описания, генерации и композирования ресурсов для реализации набора характеристик.

Генерирующая модель – отображение пространства проблемы в пространство решений, сопоставляющее допустимому набору обязательных и опциональных характеристик ПС конфигурационный файл. Он задает набор программных ресурсов для первичной реализации либо изменения характеристик ПС. Эти ресурсы автоматически порождаются (в случае трансформационной модели) либо выбираются (соответственно, при конфи-

гурационной модели) в пространстве решений.

Вместе с ресурсами конфигурационный файл описывает также и механизм(ы) их композирования из базы конфигурации. При этом формальное описание набора характеристик преобразуется в спецификацию программных ресурсов в языках программирования, задаваемых архитекторами семейства. При необходимости, для промежуточных преобразований могут использоваться релевантные предметно-ориентированные языки.

Технология сборки изменяемых программных систем

Формальный аппарат обеспечения изменяемости ПС. Сопоставительный анализ технологий конструирования линеек программных продуктов и генерирующего программирования, приведенный в предыдущем разделе, показывает критическую роль формализма связывания характеристик ПС с ресурсами их реализации для обеспечения изменяемости ПС. Именно его отсутствие требует ручного задания правил композирования ресурсов в ПС (в линейках продуктов) либо определения генерирующей модели (в генерирующем программировании).

Предлагается введение такого формализма за счет:

– определения модели вариантных характеристик семейства ПС, распространяющей модель характеристик на базовые артефакты процесса разработки (требования, архитектуру, программные ресурсы, тесты, структуры данных [13, 14]);

– выбора КПИ как ресурсов и конструктивного уточнения полученной модели с помощью ОКМ [14–16];

– введения интенсиональной объектно-компонентной модели семейства ПС, явно включающей полученное уточнение [15, 17].

Зафиксируем содержательные определения основных проявлений вариабельности [1–3], используемые далее.

Точка вариантности – представление артефакта процесса разработки ПС элемента поддерживаемого делового процесса, который может реализоваться несколькими способами.

Вариант для точки вариантности – способ реализации элемента делового процесса, который она описывает.

Зависимость (dependance) – отношение на множестве точек вариантности и вариантов, ограничивающее выбор вариантов для одних точек вариантности в зависимости от их выбора для других точек.

Ограничение (constraint) – зависимость, определенная только для точек вариантности. Типичные ограничения – отношения импликации/исключения.

Формализацию приведенных содержательных определений задает

Определение 1. Модель вариантных характеристик семейства ПС – это пара

$$M_{var} = (SV;\ AV); \qquad (1)$$

$$SV = \langle G_1; \langle G_t; TR_t \rangle, t=2,...,5; Con; Dep \rangle; \qquad (2)$$

$$AV(id_m) = \langle g_1; \langle \langle g_t, tr_t \rangle; t=2,...,m \rangle; \qquad (3)$$

$$\langle \langle p_t, tr_t \rangle; t=m+1,...,5 \rangle;\ cn;\ dp \rangle,$$

где SV – подмодель вариабельности в структуре семейства ПС;

AV – подмодель вариабельности в артефактах ПС;

$G_t = (F_t, L_t)$ в SV (2) – граф, вершины которого – идентификаторы артефактов типа t (требований, $t=1$; элементов архитектуры, $t=2$; программных ресурсов, $t=3$; тестов, $t=4$; структур данных, $t=5$), а дуги – бинарные отношения на F_t, обусловленные моделью характеристик;

TR_t – двусторонние связи между артефактами типов $t-1$ и t;

Con и Dep – предикаты на $\otimes_{t=1,...,5} F_t$, задающие ограничения и зависимости для артефактов;

g_t и p_t – подграфы G_t, описывающие артефакты, реализуемые артефактом типа m с идентификатором id_m при разра-

ботке ПС;

остальные элементы $AV(id_m)$ (3) – сужения соответствующих элементов *SV*.

Подмодель SV (2) включает вариантные точки артефактов всех типов для внесения и отслеживания изменений в них самих и в структуре создаваемых ПС семейства, определяя спектр согласованных изменений характеристик ПС в соответствии с требованиями и реализующих их артефактов в вариантных точках.

В свою очередь, модель AV (3) задает унифицированное представление артефакта разработки ПС и ее самой как "сквозного" вертикального фрагмента SV.

Дальнейшее уточнение M_{var} (1) с помощью ОКМ преобразует ее в объектно-компонентную модель семейства ПС.

Метод ОКМ основан на обобщении понятия объекта с помощью теории Фреге [5, 8–11] и использовании в компонентном методе создания ПС результатов объектного анализа предметной области. Метод предполагает логико-математическое моделирование задач предметной области с помощью четырех специальных графов ее объектов. Для их формирования и использования он включает алгебру объектного анализа, внешнюю и внутреннюю компонентную алгебры и алгебраическую систему операций преобразования неэквивалентных типов данных, передаваемых между разнородными объектами в структуре ПС [5, 9, 11].

Модель вариантных характеристик M_{var} последовательно уточняется и встраивается в *объектную подмодель вариабельности* семейства ПС на четырех уровнях проектирования метода ОКМ, выделенных в ОКМ. Ее вид устанавливает

Определение 2. Объектная подмодель вариабельности – четверка графов, детализирующих друг друга:

$$OM = \langle G_1; (G_2, TR_2);$$

$$(G_3, TR_3); (G_4, TR_4) \rangle, \qquad (4)$$

где G_1 – граф объектов предметной области на обобщающем уровне проектирования;

G_2 – представление модели характеристик на характеристическом уровне;

G_3 – архитектурно-компонентная модель семейства структурного уровня;

G_4 – интерфейсная модель взаимодействия КПИ на поведенческом уровне.

Связи трассируемости TR_t в OM (4) сопоставляют объектам функций моделируемых ПС (т. е. вершинам графа G_1) методы и данные (отображаемые вершинами графов G_2 и G_3), необходимые для взаимодействия этих объектов в составе ПС.

В поддержку сборки ПС, моделируемых с помощью OM (4), предложено дальнейшее преобразование OM в компонентную подмодель вариабельности семейства. Его суть – реализация методов объектов, представленных в OM, за счет КПИ с входными и выходными интерфейсами специального вида, описанными в [15, 16]. При этом терминальным объектам в G_3 и их интерфейсам в G_4 соответствует один и только один терминальный КПИ.

Определение 3. Компонентная подмодель вариабельности – пятерка

$$CM = \langle RC; In; ImC; Fim; D \rangle, \qquad (5)$$

где RC – терминальные КПИ для терминальных объектов OM (4);

In – интерфейсы КПИ, параметры которых содержат точки вариантности;

ImC – реализации терминальных КПИ в заданной среде;

$Fim(\cdot)$ – функции преобразования входных параметров терминальных КПИ;

D – структуры данных в сигнатурах интерфейсов терминальных КПИ.

Наконец, искомая модель фиксирует

Определение 4 [15–17]. Модель семейства изменяемых ПС – кортеж

$$M_{SF} = \langle M_{var}; (KP,\ PR); PC; M_{FM}; MK \rangle, \quad (6)$$

где M_{FM} – модель характеристик семейства ПС;

KP – готовые ресурсы семейства;

PR – предикат принадлежности KP;

PC – сборочный предикат, определяющий операции сборки ресурсов;

MK – модель их конфигурации.

Замена в кортеже M_{SF} (6) модели M_{var} ее объектно-компонентным уточнением (OM, CM) (4), (5) преобразует (6) в *объектно-компонентную* модель семейства изменяемых ПС, где характеристики отображаются объектами, а готовыми ресурсами служат КПИ.

Разработка изменяемых ПС в их семействе (6) осуществляется за счет конфигурирования ресурсов, прежде всего КПИ, согласно универсальной модели M_{var} (1)–(3) либо объектно-компонентной модели (OM, CM).

При этом выполняется ряд операций управления вариантами ПС в их семействе:

– выделение общих и вариантных характеристик ПС для заданной предметной области;

– построение модели характеристик M_{FM};

– преобразование M_{FM} в объектно-компонентную модель вариабельности семейства;

– формирование артефактов и ресурсов ПС с подбором готовых ресурсов из базы конфигурации (при ее наличии);

– планирование многократного использования ресурсов, частности КПИ, для (пере)сборки ПС с заданными характеристиками, представленными подграфом G_1 в модели OM (4);

– рефакторинг КПИ и ПС с использованием функционально эквивалентных КПИ для адаптации ПС к новым условиям выполнения и/или требованиям потребителей;

– вариантная сборка КПИ по модели CM (5).

Метод сборки и преобразования данных интерфейса. Метод сборки, интерфейс и готовые модули – базис сборочного программирования [4, 5, 11].

Метод сборки – это способ соединения разноязычных объектов в языках программирования, который базируется на теории спецификации и отображения типов и структур данных этих языков с помощью алгебраических систем, включающих типы данных и функции их эквивалентного преобразования.

Интерфейс (межмодульный и межъязыковый) выступает в качестве главной доминанты взаимодействующих компонентов и объектов в современных глобальных и сетевых средах.

Межмодульный интерфейс – модуль-посредник между двумя взаимодействующими объектами, который выполняет функции передачи и приема данных между ними.

Межъязыковый интерфейс – совокупность средств и методов взаимно однозначного преобразования структур и типов данных между языками программирования с помощью алгебраических систем, а также функций (и макроопределений) библиотеки интерфейса для обмена данными между разноязычными модулями. Библиотека интерфейсов для языков программирования операционной системы ЕС, разработанная В.Н. Грищенко, включала 64 интерфейсные функции и была передана в 52 организации СССР.

Развитие интерфейса для новых типов неструктурированных данных выполнено аспирантом ИПС НАНУ А.Ю. Стеняшиным [12].

Концепция интерфейса, как средства связи разных типов объектов в языках программирования, получила развитие в 90-х годах в языках Application Program Interface (API) и Interface Definition Language (IDL).

При практическом использовании интерфейс включает описание формальных параметров и оператор вызова (CALL, RPC, RMI и т. п.) со списком параметров и

их значениями. Значения параметров проверяются на соответствие базовым типам данных с помощью специальных аксиом и операций преобразования типов данных в классе языков программирования. Результат отображения – сгенерированные функции эквивалентного преобразования типов, которые записываются в интерфейсном модуле-посреднике связываемых объектов.

Как отметил в 1975 г. академик В.М. Глушков [6, 18], метод сборки из КПИ аналогичен конвейеру фабрики (например, фабрики Р. Форда) сборки автомобилей из готовых комплектующих и стыковочных деталей. В нем роль комплектующих выполняют терминальные КПИ и ресурсы различной степени сложности, а роль стыковки – их интерфейсы.

Алгебра сборки КПИ имеет вид:

$$\varphi = \{C, CE, \Omega\}, \qquad (7)$$

где C и CE – множества КПИ и компонентных сред [5];

Ω = {*incon, redev, link, makeaw, add, insert, redo*} – операции сборки КПИ и преобразования обмениваемых между ним данных.

Обработка этих данных производится с помощью примитивных функций генерации общих типов данных (general data types стандарта ISO/IEC 11404:2007) для фундаментальных типов данных языков программирования (и наоборот).

Выделены следующие операции сборки:

$incon(A, B, Int_A, Int_B)$ – организация взаимодействия КПИ A, B с интерфейсами Int_A, Int_B;

redev(A, B) – трансформация типов данных КПИ A, B;

$link_{PS}$ – сборка одноязычных КПИ с параметрами интерфейса в языке IDL;

$link_{SF}$ – сборка разноязычных КПИ с интерфейсами в IDL или API;

makeaw(AS , A) – удаление компонента A из системы AS ;

add(AS, A) – добавление компонента A к системе AS;

insert(A , AS) – вставка компонента A в систему AS ;

redo(x y, BD) – передача данных x , y в БД с соответствующим форматом.

Теория преобразования данных. В сборочном программирования разработана теория преобразования простых и структурных фундаментальных типов данных различных языков программирования [4, 6]. Она включает алгебраические системы для функций преобразования структурных типов к простым типам в множестве фундаментальных типов данных. При нерелевантности передаваемых данных, представленных в различных языках программирования, используются примитивные функции преобразования (например, типа *integer* к типу *character* и наоборот).

Эта теория развита для общих типов данных (general data types стандарта ISO/IEC 11404:2007), которые отображаются в типы данных современных языков программирования путем генерации для них соответствующих фундаментальных типов [12]. Генерация использует набор функций (процедур) в языке XML:

– преобразование типов данных для последовательности языков;

– формальное описание фундаментальных типов данных;

– представление данных общего типа в формате соответствующего фундаментального типа для обработки и верификации его схемы данных;

– отображение между общими и фундаментальными типами данных.

Для реализации процедур разработаны:

– библиотека функций преобразования общих типов данных (примитивных, агрегатных и генерированных) к фундаментальным типам данных (простым, структурным и сложным), необходимым в гетерогенной среде взаимодействия разноязычных КПИ и ПС;

– спецификация внешних типов данных компонентов, подсистем и систем в различных языках программирования;

– форматы данных новых модулей-посредников с операциями обращения к соответствующим процедурам из числа перечисленных для передачи данных, имеющих нерелевантный или перестроенный тип, между взаимодействующими компонентами.

Отдельные операции программно реализованы аспирантом ИПС НАНУ А.Ю Стеняшиным в разделе «Трансформация ТД» Инструментально-технологического комплекса ИПС НАНУ (http://sestudy.edu-ua.net).

Предложенная теория преобразования типов данных может использоваться для организации взаимодействия КПИ в облачных вычислениях.

Конфигурационная сборка изменяемых ПС из КПИ. Задачи конфигурационной сборки КПИ в ПС включают [14, 17, 19]:

1) трассировку требований к вариантам КПИ на уровнях архитектуры ПС;

2) порождение вариантов КПИ и ПС в точках вариантности M_{var} (1)–(3) или (OM, CM) (4), (5) в поддержку новых требований к ПС с учетом ограничений на качество и ресурсы;

3) управление конфигурированием ПС за счет выбора КПИ для них;

4) управление вариабельностью семейства ПС путем контроля полноты и непротиворечивости его состава для удовлетворения потребностей разработчиков и потребителей ПС за счет адекватной и своевременной актуализации вариантов КПИ и ПС.

Управление конфигурацией КПИ предполагает следующие операции для решения задач 1)–4):

– сборку КПИ по M_{var} или *CM*, т. е. реализацию операций $link_{PS}$ и $link_{SF}$ из (7);

– аудит уровня удовлетворительности вариабельности семейства ПС для их потребителей и разработчиков;

– аудит целостности конфигурации ПС на этапах их сопровождения и эксплуатации;

– планирование корректирующих действий по восстановлению удовлетворительного уровня вариабельности и/или целостности конфигурации ПС.

Для поддержки эффективного выполнения выделенных операций предлагается интегрировать процесс управления конфигурацией КПИ в процесс проактивного обоснованного управления вариабельностью семейства ПС [16, 20], показанный на рис. 1. Он представлен композицией четырех функций, выделенных как обобщения действий в известном цикле управления Э. Деминга (Plan, Do, Check, Act) в проанализированных индустриальных технологиях разработки семейств ПС [1–3].

Согласно рис. 1, эта композиция выполняется в единой информационной среде, структурированной на основании модели вариантных характеристик M_{var} или ее объектно-компонентного уточнения (OM, CM).

Предлагаются следующие целевые функции *F* управления вариабельностью:

– планирование состава ПС и КПИ с их вариантами для целевой предметной области (F_1);

– непосредственная разработка терминальных КПИ и их автоматизированная сборка в целевые ПС по моделям M_{var} и/или ($OM; CM$) (F_2);

– анализ соответствия состава ПС и КПИ потребностям потребителей с диагностикой неадекватности и выработкой корректирующих действий (F_3);

– актуализация моделей M_{var} и/или ($OM; CM$) либо текущего состава ПС и КПИ для устранения неадекватности (F_4).

Функции F_1 – F_4 дополнены сервисной функцией инициализации их информационной среды, в частности M_{var} и/или ($OM; CM$), и комплексной организационно-технологической подготовки.

Согласно рис. 1 стандартизированные представления КПИ (например, в языке WSDL) накапливаются в репозитории

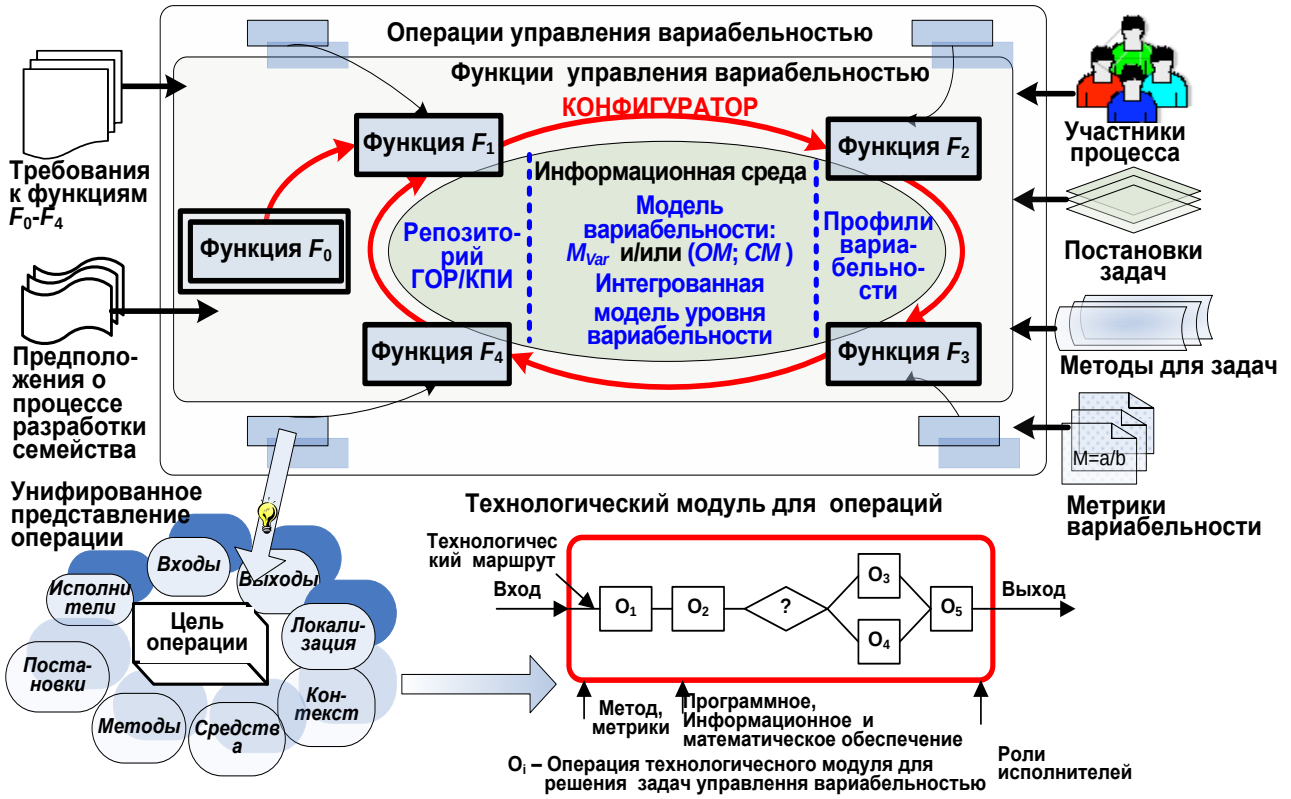

Рис. 1. Схема процесса управления вариабельностью семейства ПС

семейства изменяемых ПС. В подпроцессах реализации функции F_2 – их отбора, настройки и сборки в ПС – ключевую роль должно выполнять специализированное инструментальное средство – конфигуратор ПС. Именно конфигуратор должен обеспечивать объединение КПИ и их интерфейсов с вариантами артефактов ПС из репозитория в соответствии с моделями их вариабельности (AV) (3) либо ее объектно-компонентными уточнениями (OM, CM) (4), (5).

Реализация конфигуратора ПС

Макет конфигуратора и технологическая схема сборки КПИ с его помощью реализованы аспирантом ИПС НАНУ А.Л. Колесником в среде MS VS.Net и описаны в [14, 19].

Конфигуратор предоставляет заинтересованным лицам интерфейс для автоматизированного создания изменяемой ПС с заданными характеристиками из заранее разработанных КПИ. Его функциональные аналоги предлагаются в Интернет-магазинах автомобилей, ноутбуков и т. п., где покупатели могут заказать продукт с необходимыми им функциями.

Создаваемая ПС описывается в интерфейсе конфигуратора диаграммой характеристик – визуальным представлением графа G_1 из модели M_{var}, в котором перечисленные выше отношения между характеристиками обозначаются специальными пиктограммами, показанными на рис. 2. В среде конфигуратора элементы этой диаграммы реализуются средствами Windows Workflow Foundation в VS.Net и описываются предметно-ориентированным языком. Его словарь включает инструкции технологии, термины предметной области и артефакты процесса создания ПС.

Основной этап сборки КПИ конфигуратор – компиляция их исходного кода, преобразующая его в промежуточный код, или код выполнения в среде VS.Net. Специальная программа-посредник заменяет относительные адреса функций внешних библиотек их реальными адресами, используемыми при выполнении.

На рис. 3 показана схема работы конфигуратора. В качестве демонстраци-

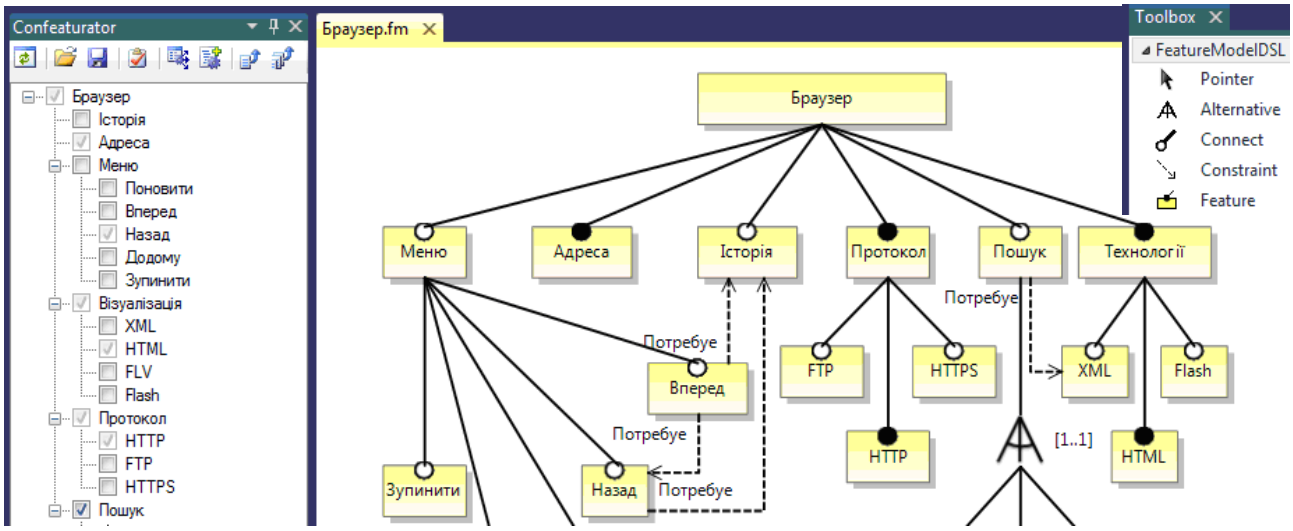

Рис. 2. Фрагмент диаграммы характеристик для семейства браузеров в конфигураторе

онного примера рассмотрена разработка семейства ПС для решения квадратного уравнения при различных значениях дискриминанта. Как показано на рис. 3, вначале разработчик получает задание на доработку/создание КПИ в среде VS.Net и помещает созданные КПИ в репозиторий семейства (стрелка 1 на рис. 3). В общем случае КПИ представлен парой файлов: ***.cs**, фиксирующих реализуемую бизнес-логику, и ***.xoml**, описывающих атомарные или составные объекты предметной области и задающих алгоритм выполнения бизнес-логики из ***.cs** файлов.

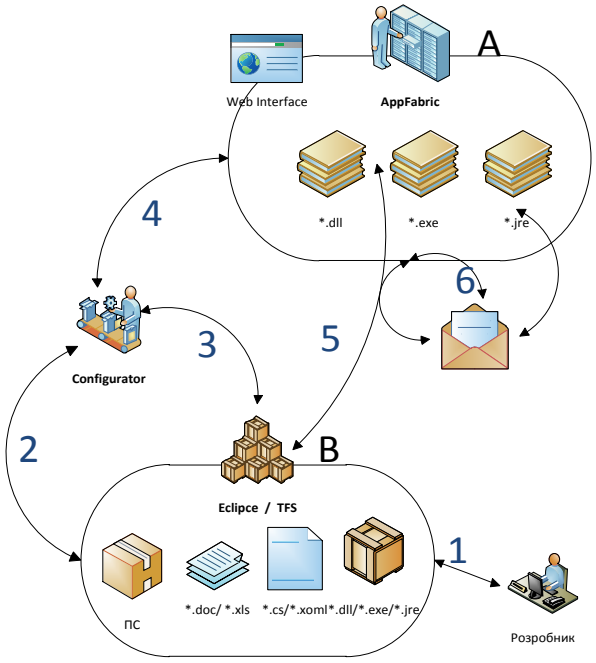

Рис. 3. Схема сборки КПИ в AppFab

Интерфейс конфигуратора обеспечивает связь с репозиторием и отображение в соответствующей экранной форме списка доступных КПИ (стрелка 2). Каждый элемент в окне дизайнера конфигуратора является атомарным КПИ, заранее разработанным и размещенным в репозитории. Под списком КПИ расположены детальные сведения о них из файлов ***.cs** (имя, родительский класс, описание, статус, привязка к конкретному методу).

Пользуясь графическим дизайнером, заинтересованные в ПС лица могут модифицировать или задать новую конфигурацию характеристик и построить соответствующую ПС из доступных КПИ.

Конфигурацию характеристик можно разместить на сервере или в репозитории семейства для последующего использования (стрелка 4 на рис. 3).

Класс КПИ представляет собой набор методов, выполняемых последовательно или параллельно согласно файлу ***.xoml**. Это позволяет манипулировать в среде конфигуратора теми КПИ, язык(и) реализации которых не поддерживаются в VS.Net, а также КПИ в форме сервисов.

В последнем случае для запуска программы AppFabric (блок A на рис. 3) следует задать команды: Пуск→ Программы →Internet Information Server (IIS). Выбирается нужный сервис для получения необходимой информации (адрес, связи, контракт) и занесения данных в код одного из методов файла ***.cs**.

Результатом является код:

```
1: private void codeActivi-
ty1_ExecuteCode_1
(object sender, EventArgs e)
2: {
3:  Uri address = new
Uri("http://localhost:63632/
/CountDiscriminant.
svc");//адреса
4:  WSHttpBinding binding =
= new WSHttpBinding();
//прив'язка
5:  EndpointAddress endpoint
= new EndpointAddress(address);
6:  ChannelFactory
<ICountDiscriminant>
factory = new
      ChannelFactory
<ICountDiscriminant>(binding,
endpoint); //контракт
7:  ICountDiscriminant
channel =
= factory.CreateChannel();
8:  D =
channel.GetDiscriminant
(a, b, c);  //D = b2 – 4ac:
9: }
```

Строки 3–7 обеспечивают создание специального объекта *channel* для отдаленного вызова, а строка 8 реализует его (стрелка 6 на рис. 3).

В среде конфигуратора можно запускать на выполнение алгоритм, загруженный в его редактор. Для этого нужно задать команды: Workflow → CompileWorkflow. Конфигуратор проанализирует и скомпилирует файлы ***.cs** и ***.xoml, сформирует библиотеку (***.dll**) и выведет сообщение о выполнении компиляции. Полученный файл загружается и извлекается из созданной библиотеки КПИ в оперативную память и запускает требуемый КПИ на выполнение.

Технологический комплекс сборки готовых ресурсов

Предложенная технология объектно-компонентной разработки изменяемых ПС с помощью CASE-инструментов (трансляторы для языков программирования, каркасы тестирования, генераторы и т. п.), а также интегрированных инструментальных средств (Eclipse, Protege, VS.Net, Corba, Java и т. п) реализована в Инструментально-технологическом комплексе (ИТК) ИПС НАНУ (http://sestudy.edu-ua.net) [21, 22].

В нем готовыми ресурсами являются КПИ. Они отображают функции и данные предметной области, представленные в модели M_{var} в виде функциональных объектов и объектов данных. Каждый КПИ специфицируется согласно соответствующим стандартам в языке WSDL, а его интерфейс – в языках IDL, API, SDIL и др. Это дает возможность собирать КПИ на единой основе, общей для всех видов разнородных ресурсов.

Технология разработки изменяемых ПС из готовых ресурсов в ИТК включает:

– проектирование ПС с использованием стандартного жизненного цикла;

– онтологическое проектирование доменов с заданием модели характеристик и архитектуры ПС из готовых компонентов;

– спецификацию разнородных программных ресурсов, прежде всего КПИ, в языках программирования, их реализацию и верификацию;

– отбор функционально готовых КПИ в репозитории;

– сборку разнородных КПИ и преобразование передаваемых между ними данных, нерелевантных по типу, формату, размеру и т. д.;

– изменение ранее созданных КПИ и ПС для адаптации под конкретные цели;

– описание специфики предметной области в предметно-ориентированных языках с использованием DSL TooLs VS.Net для получения исполняемого кода;

– тестирование КПИ и ПС, сбор данных для оценки качества ПС;

– сохранение результатов проектирования в репозитории КПИ;

– документирование КПИ.

В ИТК реализованы элементы основных современных парадигм программирования, необходимые для проектиро-

вания ПС из КПИ в различных предметных областях. В нем нашли отражение фундаментальные положения этих парадигм программирования, включая теорию взаимодействия и вариантности ПС, а также теорию моделирования и адаптации ПС, спроектированных за пределами ИТК. В среде ИТК представлены:

– технология изготовления ПС из КПИ на ряде технологических линиях;

– средства поддержки процессов жизненного цикла ПС по стандарту ISO/IEC 12207:2008 и оценки качества ПС согласно его рамочной модели в ISO/IEC 9126:2001;

– онтология вычислительной геометрии;

– линия обучения современным языкам программирования C#, Java и CASE-инструментам (Protégé, Eclipse, VS.Net, Java).

К технологическим линиям можно обращаться на Web-сайте ИТК. Для обучения дисциплине «Программная инженерия» можно воспользоваться электронным учебником: на сайте www.intuit.ru на русском языке; на сайте экспериментальной фабрики программ КНУ им. Т. Шевченко (programsfactory.univ.kiev.ua) украинском языке.

Выводы

Выявлены взаимодополнительные ограничения индустриальных технологий разработки семейств ПС и объектно-компонентного метода Лаврищевой – Грищенко для разработки изменяемых ПС: отсутствие формализма сборки программных ресурсов для характеристик ПС и, соответственно, слабая предсказуемость характеристик этой сборки.

Предложено преодоление ограничений за счет модели вариантных характеристик семейства ПС, распространяющей модель их характеристик на базовые артефакты процесса разработки. Для случая, когда ресурсами служат компоненты повторного использования, разработана объектно-компонентная модель вариабельности ПС, уточняющая модель вариантных характеристик, и модель семейства изменяемых ПС. Описана алгебра операций конфигурационной сборки компонентов по модели вариабельности и преобразования типов данных, обмениваемых между ними.

Предложена интеграция этих операций в процесс проактивного обоснованного управления вариабельностью семейства изменяемых ПС и технологическая схема последнего. Он представлен композицией функций планирования, реализации, контроля вариабельности и актуализации модели/состава семейства по его результатам. Функции реализуются в единой информационной среде, структурированной на основании модели вариантных характеристик или ее объектно-компонентного уточнения.

Проведена экспериментальная реализация конфигуратора компонентов в предложенном процессе. Описана апробация формального аппарата и конфигуратора в технологических линиях разработки изменяемых систем из компонентов, реализованных в ИТК ИПС НАН Украины.

1. *Product* Line Engineering [Электронный ресурс]. – Режим доступа: http://www.productlineengineering.com/. – Название с экрана.
2. *Pohl K., Bockle G., Linden F.J.* Software Product Line Engineering: Foundations, Principles and Techniques. – New York: Springer–Verlag, 2005. – 437 p.
3. *Чернецки К., Айзенекер У.* Порождающее программирование. Методы, инструменты, применение. – М.: ИД «Питер», 2005. – 730 с.
4. *Лаврищева Е.М.* Парадигмы программирования сборочного типа в программной инженерии. – Сб. трудов межд. конф. УкрПРОГ-2014. – С. 76–92.
5. *Лаврищева Е.М., Грищенко В.Н.* Сборочное программирование. Основы индустрии программных продуктов. – К.: Наук. думка, 2009. – 372 с.
6. *Лаврищева Е.М., Грищенко В.Н.* Связь разноязыковых модулей в ОС ЕС. – М.: Финансы и статистика, 1982. – 127 с.
7. *Lavrischeva K.M.* Theory and practice of software factories // In: Cybernetics and Sys-

tems Analysis. – 2011. – Vol. 47, N 6. – P. 961–972.
8. *Лаврищева Е.М., Грищенко В.Н.* Методы и средства компонентного программирования // Кибернетика и системный анализ. – 2003. – № 1. – С. 39–55.
9. *Грищенко В.Н.* Теоретические и прикладные аспекты компонентного программирования: автореф. дис. … док. физ.-мат. наук; Институт кибернетики им. В.М. Глушкова. – К., 2007. – 34 с.
10. *Лаврищева Е.М., Колесник А.Л., Стеняшин А.Ю.* Объектно-компонентное проектирование программных систем. Теоретические и прикладные вопросы // Вісник КНУ, серія фіз-мат. науки. – 2013. – № 4. – С. 150–162.
11. *Lavrischeva K., Stenyashin A., Kolesnyk A.* Object-Component Development of Application and Systems. Theory and Practice [Electronic resourse] // Journal of Software Engineering and Applications. – 2014. Mode of access: http://www.scirp.org/journal/jseaUSA.
12. *Лавріщева К.М., Стеняшин А.Ю.* Підхід щодо трансформації загальних типів даних стандарту ISO/IEC 11404 для використання в гетерогенних середовищах // 2nd International Conference on High Performance Computing-2012. – К.: КПІ, 2014. – С. 227–234.
13. *Лавріщева К.М., Слабоспицька О.О., Колесник А.Л., Коваль Г.І.* Теоретичні аспекти керування варіабельністю в сімействах програмних систем / // Вісник КНУ, серія фіз-мат. науки. – 2011. – № 1. – С. 151–158.
14. *Колесник А.Л.* Модели и методы разработки семейства вариантных программных систем: автореф. дис. … канд. физ.-мат. наук: КНУ им. Т.Г. Шевченко, 2013. – 22 с.
15. *Лавріщева К.М., Слабоспицька О.О.* Підхід до побудови об'єктно-компонентної моделі сімейства програмних продуктів / // Проблеми програмування. – 2013. – № 3. – С. 14–24.
16. *Slabospickaya O.* Feature Model of Software Product Line Enhancing to Enable Product Adaptability [in Ukrainian] // In: Bulletin of University of Kiev. Series: Physics & Mathematics, special issue, Kiev. – 2014. – P. 151–158.
17. *Лавріщева К.М., Колесник А.Л.* Концептуальні моделі розподілених компонентних систем // Проблеми програмування. – 2013. – № 1. – С. 21–33.
18. *Lavrischeva K., Aronov A., Dzubenko A.* Programs Factory – A conception of Knowledge Reprezentation of Scientifical Artifacts From Standpoint of Software Engineering / // Comp. and Inf. Sci., Canadian Center of Sci. and Edu. – 2013. – P. 21–28.
19. *Колесник* А.Л. Підходи до конфігурування компонентів повторного використання // Проблеми програмування. – 2011. – № 4. – С. 57–66.
20. *Kolesnyk A., Slabospitskaya O.* Tested Approach for Variability Management Enhancing in Software Product Line // In: Ermolayev V., Mayr H.C., Nikitchenko M. et al. (eds.): Proc. 8-th Int. Conf. ICTERI 2012, Kherson, Ukraine, June 6–10, 2012. – Vol. 848. – P. 125–133. [Электронный ресурс]. – Режим доступа: ceur-ws.org/Vol-848/ICTERI-2012-CEUR-WS-paper-31-p–155–162.pdf. – Название с экрана.
21. *Лаврищева Е.М.* Software Engineering компьютерных систем. Парадигмы, Технологии, CASE-средства программирования. – Киев: Наук. думка, 2014. – 284 с.
22. *Лавріщева К.М., Зінькович В.М., Колесник А.Л.* Інструментально-технологічний комплекс розробки и навчання прийомам виробництва програмних систем. Свідоцтво про реєстрацію авторського права на твір № 45292 від 27.08.2012 // Державна служба інтелектуальної власності України. – К.; 2012.

References

1. *Product* Line Engineering [Electronic resourse]. – Mode of access: http://www.product lineengineering.com/.
2. *Pohl K., Bockle G., Linden F.J.* Software Product Line Engineering: Foundations, Principles and Techniques. – New York: Springer–Verlag, 2005. – 437 p.
3. *Czarnecki K., Eisenecker U.* Generative Programming: Methods, Tools, and Applications. – Addison-Wesley, Reading, MA, USA, 2000 – 864 p.
4. *Lavrischeva E.M.* Paradigms of programming assembling type in software engeneering // Problems in Programming, 2014. – N 2–3. – P. 121–132. (in Russian).
5. *Lavrischeva E.M., Grischenko V.N.* Assembly Programming. Basics of Software Industry. – Kyiv: Naukova Dumka, 2009 (2nd ed.). – 372 p. (in Russian).

6. *Lavrischeva E.M., Grischenko V.N.* Interconnection of Multilingual Modules in OS ES. – M.: Finansy i Statystika, 1982. – 127 p.
7. *Lavrischeva K. M.* Theory and practice of software factories // In: Cybernetics and Systems Analysis, 2011. – Vol. 47. – N 6. – P. 961–972.
8. *Lavrishcheva E.M. Grishchenko V.N.* Methods and Tools of Component Programming // Cybernetics and Systems Analysis. – 2003. – Vol. 39. – № 1. – P. 33–45. (in Russian).
9. *Grischenko V.N.* Theoretical and Applied aspects of Component Programming: Ph. D. theses: spec. 01.05.03; V.M. Glushkov Institute of Cybernetics. – K., 2007. – 34 p. (in Russian).
10. *Lavrishcheva E., Stenyashin A., Kolesnyk A.* Object-Component Design. Theoretical and Applied Issues // Visn. Ser. Fiz.-Mat. Nayky, Kyiv St. Univ. im. Tarasa Shevchenka. Special Issue, 2013. – № 4. – C. 150–162. (in Russian).
11. *Lavrischeva K., Stenyashin A., Kolesnyk A.* Object-Component Development of Application and Systems. Theory and Practice [Electronic resourse] // Journal of Software Engineering and Applications, 2014. Mode of access: http://www.scirp.org/journal/jseaUSA.
12. *Lavrischeva K., Stenyashin A.* An approach for Standard ISO/IEC 11404 General Data Types transformation for use in heterogeneous environments // Second Int. Conf. on High Performance Computing-2012. – K.: KPI, 2012. – P. 227–234. (in Ukrainian).
13. *Lavrischeva K., Slabospitskaya O., Kolesnik A. at all.* The Theoretical View for Software Family Variability Management // Visn., Ser. Fiz.-Mat. Nayky, Kyiv Univ. im. Tarasa Shevchenka. – 2011. – N 1. – P. 45–53. (in Ukrainian).
14. *Kolesnik A.L.* Models and Methods for Variable Software Systems Familly Development: Ph. D. theses: Kyiv St. Univ. im. Tarasa Shevchenka. – 2013. – 22 p. (in Ukrainian).
15. *Lavrischeva K., Slabospickaya O.* An approach for Software Product Family Object-Component Model Elaborating // Problems in Programming. – 2013. – N 3. – P. 14–24. (in Ukrainian).
16. *Slabospickaya O.* Feature Model of Software Product Line Enhancing to Enable Product Adaptability [in Ukrainian] // In: Bulletin of University of Kiev. Series: Physics & Mathematics, special issue, Kiev. – 2014. – P. 151–158. (in Ukrainian).
17. *Lavrischeva K., Kolesnik A.* Conceptual Models for Distributed Software Systems // Problems in Programming, 2013. – N 1. – P. 21–33. (in Ukrainian).
18. *Lavrischeva K., Aronov A., Dzubenko A.* Programs Factory – A conception of Knowledge Reprezentation of Scientifical Artifacts From Standpoint of Software Engineering // Comp. and Inf. Sci., Canadian Center of Sci. and Edu. – 2013. – P. 21–28.
19. *Kolesnik A.* Approaches for Configuring Reusable Components // Problems in Programming. – 2011. – N 4. – P. 57–66. (in Ukrainian).
20. *Kolesnyk A., Slabospitskaya O.* Tested Approach for Variability Management Enhancing in Software Product Line // Proc. 8-th Int. Conf. ICTERI 2012, Kherson, Ukraine, June 6–10, 2012, CEUR-WS.org/Vol-848, urn:nbn:de:0074-848-8. – P. 125–133. [Electronic resource]. – Mode of access: ceur-ws.org/Vol-848/ICTERI-2012-CEUR-WS-paper-31-p–155–162.pdf.
21. *Lavrischeva E.M.* Software Engineering for Computer Systems. Paradigms, Technologies, CASE tools for Programming. – Kiev: Naukova Dumka, 2014.– 284 p. (in Russian).
22. *Lavrischeva K.M., Zinkovich V.M., Kolesnik A.L.* Instrumental-Technological Complex for Software Development and Production Scills Learning. A Certificate for authors' intellectual property N 45292 at 27.08.2012 // State Intellectual Property Service of Ukraine. – Kyiv, 2012.

Получено 06.10.2015

Об авторах:

Лаврищева Екатерина Михайловна,
доктор физико-математических наук,
профессор.
Количество публикаций в украинских изданиях – более 150.
Количество публикаций в иностранных индексированных изданиях – более 40,
http://orcid.org/0000-0002-1160-1077,

Слабоспицкая Ольга Александровна,
кандидат физико-математических наук,
старший научный сотрудник.
Количество публикаций в украинских изданиях – более 50,
Количество публикаций в иностранных индексированных изданиях – 5,
http://orcid.org/0000-0001-6556-0947,

Стеняшин Андрей Юрьевич,
аспирант.
Количество публикаций в украинских изданиях – более 10.
Количество публикаций в иностранных индексированных изданиях – 3.
http://orcid.org/0000-0001-7615-9024

Колесник Андрей Леонидович,
кандидат физико-математических наук, младший научный сотрудник.
Количество публикаций в украинских изданиях – более 10.
Количество публикаций в иностранных индексированных изданиях – 3.
http://orcid.org/0000-0001-1672-9201

Место работы авторов:

Московский физико-технический институт (государственный университет),
Россия, 141700, Московская область,
г. Долгопрудный,
Институтский переулок, д. 9.
Тел.: +7(495) 408 4554

Институт программных систем
НАН Украины,
03187, Киев-187,
Проспект Академика Глушкова, 40.
Тел. +38(044) 526 4286.

E-mail: lavryscheva@gmail.com,
ols.07@mail.ru,
andrey.stenyashin@gmail.com,
swabber@gmail.com

УДК 004.42:510.69

М.С. Нікітченко, С.С. Шкільняк

АЛГЕБРИ КВАЗІАРНИХ ТА БІ-КВАЗІАРНИХ РЕЛЯЦІЙ

Запропоновано поняття квазіарної реляції (відношення), введено операції над такими реляціями, описано алгебри квазіарних реляцій. Доведено ізоморфізм алгебри квазіарних реляцій та першопорядкової алгебри тотальних однозначних квазіарних предикатів. Побудовано алгебри бі-квазіарних реляцій, задані на множинах пар квазіарних реляцій. Визначено різні підкласи таких алгебр та досліджено їх зв'язки з алгебрами часткових однозначних, тотальних неоднозначних, часткових неоднозначних, монотонних, антитонних квазіарних предикатів.

Ключові слова: алгебра, логіка, ізоморфізм, реляція, квазіарний предикат.

Вступ

Поняття реляції (відношення) належить до найважливіших понять математики. Під *n*-арним відношенням зазвичай розуміють [1] множину кортежів довжини *n*. Водночас низка задач інформатики та програмування вимагають узагальнення цього поняття.

Скінченне *n*-арне відношення можна розглядати як таблицю, що має *n* стовпчиків. Рядок таблиці (*n*-ка базових значень даних) – це елемент відношення. При цьому в деяких випадках заповненими можуть бути не всі клітинки таблиці. Наприклад, якщо розглядати екзаменаційну відомість як таблицю, то не всі її клітинки заповнюються під час іспиту (зокрема, через неявку студента на іспит).

Формально таку частково заповнену таблицю можна задати наступним чином. Нехай V – множина атрибутів (предметних імен), A – множина предметних значень. Часткову функцію із V в A назвемо номінативною (іменною) множиною. Клас всіх таких множин позначаємо ${}^{V}A$. Довільну підмножину $R \subseteq {}^{V}A$ назвемо квазіарною реляцією. Тепер номінативна множина, що входить у реляцію R, може розглядатися як частково заповнений рядок таблиці.

Мета даної роботи – це побудова та вивчення алгебр квазіарних реляцій (відношень) та дослідження їх зв'язків із алгебрами квазіарних предикатів. Доведено ізоморфізм алгебри квазіарних відношень та першопорядкових алгебр тотальних однозначних квазіарних предикатів. Побудовано алгебри бі-квазіарних реляцій, задані на множинах пар квазіарних реляцій.

Визначено різні підкласи таких алгебр, встановлено їх зв'язки з алгебрами часткових однозначних, тотальних неоднозначних, часткових неоднозначних, монотонних, антитонних квазіарних предикатів.

Поняття, які тут не визначаються, тлумачимо в сенсі [2, 3]. Для полегшення читання наведемо необхідні для подальшого викладу визначення.

1. Іменні множини та квазіарні предикати

V-іменна множина (V-ІМ) над A – це однозначна функція вигляду $\delta : V \to A$. Подаємо V-ІМ у вигляді $[v_1 \mapsto a_1,\ldots,v_n \mapsto a_n,\ldots]$, де $v_i \in V$, $a_i \in A$, $v_i \neq v_j$ при $i \neq j$.

Клас всіх V-ІМ над A позначимо ${}^{V}A$.

Вводимо функцію $asn : {}^{V}A \to 2^{V}$ так:

$asn(d) = \{v \in V \mid v \mapsto a \in d$ для деякого $a \in A\}$.

Операцію $\|_{-x}$ видалення компоненти з іменем x та операцію ∇ накладки задамо таким чином:

$$d\|_{-x} = [v \mapsto a \in d \mid v \neq x];$$

$$\delta \nabla \eta = \eta \cup [v \mapsto a \in \delta \mid v \notin asn(\eta)].$$

Операцію $\mathrm{r}^{v_1,\ldots,v_n}_{x_1,\ldots,x_n} : {}^{V}A \to {}^{V}A$ реномінації задаємо так:

$$\mathrm{r}^{v_1,\ldots,v_n}_{x_1,\ldots,x_n}(d) = d \nabla [v_1 \mapsto d(x_1),\ldots,v_n \mapsto d(x_n)].$$

Замість $y_1,...,y_n$ пишемо також $\bar{y}$.

Операцію реномінації продовжуємо на множини ІМ:

$$\mathrm{r}^{\bar{v}}_{\bar{x}}(L) = \{\mathrm{r}^{\bar{v}}_{\bar{x}}(d) \mid d \in L\}, \text{ де } L \subseteq {}^{V}A.$$

Введемо відношення $=_{-x}$ рівності з точністю до компоненти з іменем x:

$$d_1 =_{-x} d_2, \text{ якщо } d_1\|_{-x} = d_2\|_{-x}.$$

Послідовне застосування двох операцій $\mathrm{r}^{\bar{v}}_{\bar{x}}$ (зовнішня) та $\mathrm{r}^{\bar{u}}_{\bar{y}}$ (внутрішня) можна подати у вигляді однієї операції реномінації, яку назвемо згорткою операцій $\mathrm{r}^{\bar{v}}_{\bar{x}}$ та $\mathrm{r}^{\bar{u}}_{\bar{y}}$ і позначатимемо $\mathrm{r}^{\bar{v}}_{\bar{x}} \bullet \mathrm{r}^{\bar{u}}_{\bar{y}}$.

Нехай маємо послідовне застосування операцій реномінації $\mathrm{r}^{v_1,...v_n,u_1,...,u_k}_{s_1,...,s_n,z_1,...,z_k}$ та $\mathrm{r}^{v_1,...,v_n,w_1,...,w_m}_{x_1,...,x_n,y_1,...,y_m}$, де $\{w_1,...,w_m\} \cap$ $\cap \{u_1,...,u_k\} = \varnothing$. Тоді для кожного $d \in {}^{V}A$ маємо:

$$\mathrm{r}^{v_1,...,v_n,w_1,...,w_m}_{x_1,...,x_n,y_1,...,y_m} \bullet \mathrm{r}^{v_1,...v_n,u_1,...,u_k}_{s_1,...,s_n,z_1,...,z_k}(d) =$$

$$= \mathrm{r}^{v_1,...,v_n,w_1,...,w_m}_{x_1,...,x_n,y_1,...,y_m}(\mathrm{r}^{v_1,...v_n,u_1,...,u_k}_{s_1,...,s_n,z_1,...,z_k}(d)) =$$

$$= \mathrm{r}^{v_1,...,v_n,w_1,...,w_m,u_1,...,u_k}_{a_1,...,a_n,b_1,...,b_m,z_1,...,z_k}(d),$$

де кожні a_i та b_j задаються так:

$$a_i = \begin{cases} x_i, \text{ якщо } x_i \notin \{v_1,...v_n,u_1,...,u_k\}, \\ s_l, \text{ якщо } x_i = v_l \text{ для деякого } v_l, \\ z_l, \text{ якщо } x_i = u_l \text{ для деякого } u_l; \end{cases}$$

$$b_j = \begin{cases} y_j, \text{ якщо } y_j \notin \{v_1,...v_n,u_1,...,u_k\}, \\ s_l, \text{ якщо } y_j = v_l \text{ для деякого } v_l, \\ z_l, \text{ якщо } y_j = u_l \text{ для деякого } u_l. \end{cases}$$

Під *V*-квазіарним предикатом на множині *A*, або *V-A*-квазіарним предикатом, розуміємо довільну часткову неоднозначну функцію вигляду $P : {}^{V}A \to$ $\to \{T, F\}$. Тут $\{T, F\}$ – множина істиннісних значень.

Для *V-A*-квазіарного предиката *P* задаємо області *істинності* та *хибності*:

$$T(P) = \{d \in {}^{V}A \mid T \in P(d)\};$$

$$F(P) = \{d \in {}^{V}A \mid F \in P(d)\}.$$

Ми трактуємо часткові неоднозначні квазіарні предикати як відношення між ${}^{V}A$ та множиною істиннісних значень $\{T, F\}$. Назвемо їх предикатами реляційного типу, або *R*-предикатами. Вони формалізують найпростіше уточнення поняття часткового неоднозначного предиката. Клас *V-A*-квазіарних *R*-предикатів позначимо PrR^{V}_{A}.

Ім'я $z \in V$ (строго) *неістотне* для предиката *P*, якщо для всіх $d_1, d_2 \in {}^{V}A$ таких, що $d_1 =_{-z} d_2$, маємо $P(d_1) = P(d_2)$.

V-A-квазіарний предикат *P*:

– однозначний, якщо $T(P) \cap F(P) = \varnothing$;

– тотальний, якщо $T(P) \cup F(P) = {}^{V}A$;

– всюди невизначений, якщо

$$T(P) = \varnothing \text{ та } F(P) = \varnothing;$$

– тотально насичений, якщо

$$T(P) = {}^{V}A \text{ та } F(P) = {}^{V}A.$$

Всюди невизначений предикат позначаємо як $\perp$, тотально насичений – як $\top$.

Повний образ предиката *P* на *d* позначаємо $P[d]$.

Предикат $P : {}^{V}A \to \{T, F\}$ *монотонний*, якщо $d \subseteq h \Rightarrow P[d] \subseteq P[h]$.

Для однозначних предикатів монотонність стає еквітонністю.

Однозначний предикат *P* *еквітонний*, якщо з умови $P(d)\downarrow$ та $d \subseteq d'$ випливає $P(d')\downarrow = P(d)$.

Для монотонних предикатів маємо: нехай $d \subseteq h$

$$d \in T(P) \Rightarrow h \in T(P) \text{ та } d \in F(P) \Rightarrow h \in F(P).$$

Предикат $P : {}^{V}A \to \{T, F\}$ *антитонний*, якщо $d \subseteq h \Rightarrow P[d] \supseteq P[h]$.

Для антитонних предикатів маємо: нехай $d \subseteq h$

$$h \in T(P) \Rightarrow d \in T(P) \text{ та } h \in F(P) \Rightarrow d \in F(P).$$

Частково впорядкована [4] множина *R* із відношенням порядку $\leq$ називається:

– замкненою вгору, якщо із $d \in R$ випливає: $h \in R$ для всіх *h* таких, що $d \leq h$;

– замкненою вниз, якщо із $d \in R$ випливає: $h \in R$ для всіх h таких, що $h \le d$.

У нас $\le$ – це відношення $\subseteq$.

Таким чином.

Твердження 1. Нехай *V-A*-квазіарний предикат *P* монотонний, тоді множини *T*(*P*) і *F*(*P*) замкнені вгору.

Твердження 2. Нехай *V-A*-квазіарний предикат *P* антитонний, тоді множини *T*(*P*) і *F*(*P*) замкнені вниз.

Часткові однозначні квазіарні предикати називатимемо *P*-предикатами, тотальні квазіарні предикати – *T*-предикатами, тотальні однозначні квазіарні предикати – *TS*-предикатами. Монотонні *R*-предикати, антитонні *R*-предикати, еквітонні *P*-предикати, антитонні *T*-предикати назвемо відповідно *RM*-предикатами, *RA*-предикатами, *PE*-предикатами, *TA*-предикатами.

Класи *V-A*-квазіарних *P*-предикатів, *T*-предикатів, *TS*-предикатів відповідно позначимо PrP_A^V, PrT_A^V, $PrTS_A^V$. Класи *V-A*-квазіарних *RM*-предикатів, *RA*-предикатів, *PE*-предикатів, *TA*-предикатів відповідно позначимо

$$PrRM_A^V, PrPE_A^V, PrTA_A^V, PrTA_A^V.$$

V-A-квазіарний предикат $\tilde{P}$ назвемо *дуальним* до *V-A*-квазіарного предиката *P*, якщо $T(\tilde{P}) = \sim F(P); F(\tilde{P}) = \sim T(P)$.

Тут $\sim$ – теоретико-множинна операція доповнення.

Прикладом пари взаємно дуальних предикатів є $\bot$ та $\top$.

Задамо відображення дуалізації $\delta : PrR_A^V \to PrR_A^V$ таким чином:

$$\delta(P) = \tilde{P} \text{ для кожного } P \in PrR_A^V.$$

Це означає:

$$T(\delta(P)) = \sim F(P); F(\delta(P)) = \sim T(P).$$

Відображення дуалізації інволютивне: $\delta(\delta(P)) = P$ для кожного $P \in PrR_A^V$.

Теорема 1. Маємо властивості:

Q – *P*-предикат $\Leftrightarrow$ $\tilde{Q}$ – *T*-предикат;

Q – *RM*-предикат $\Leftrightarrow$ $\tilde{Q}$ – *RA*-предикат;

Q – *PE*-предикат $\Leftrightarrow$ $\tilde{Q}$ – *TA*-предикат;

Q – *TS*-предикат $\Leftrightarrow$ $\tilde{Q}$ – *TS*-предикат.

Теорема 2. Маємо властивості:

$$\delta(PrP_A^V) = PrT_A^V,\ \delta(PrT_A^V) = PrP_A^V;$$

$$\delta(PrPE_A^V) = PrTA_A^V,\ \delta(PrTA_A^V) = PrPE_A^V;$$

$$\delta(PrRM_A^V) = PrRA_A^V,\ \delta(PrRA_A^V) = PrRM_A^V;$$

$$\delta(\{\bot\}) = \{\top\},\ \delta(\{\top\}) = \{\bot\};$$

$$\delta(PrR_A^V) = PrR_A^V,\ \delta(PrTS_A^V) = PrTS_A^V.$$

2. Композиційні алгебри квазіарних предикатів

На пропозиційному рівні композиції фактично працюють лише з виробленими предикатами істиннісними значеннями. Такі композиції називають логічними зв'язками. Основними логічними зв'язками є $\neg$ та $\vee$. Ми будемо використовувати $\neg$, $\vee$, &. Наведемо визначення цих зв'язок через області істинності та хибності відповідних предикатів.

Предикати $\neg(P)$, $\vee(P, Q)$, $\&(P, Q)$ традиційно позначаємо $\neg P$, $P \vee Q$, $P\&Q$. Вони задаються так:

$T(\neg P) = F(P)$;

$F(\neg P) = T(P)$;

$T(P \vee Q) = T(P) \cup T(Q)$;

$F(P \vee Q) = F(P) \cap F(Q)$;

$T(P\&Q) = T(P) \cap T(Q)$;

$F(P\&Q) = F(P) \cup F(Q)$.

Композиції $\neg$ та $\vee$ назвемо базовими пропозиційними композиціями.

Композиції &, $\to$, $\leftrightarrow$ є похідними, вони виражаються через $\neg$ та $\vee$:

$P\&Q = \neg(\neg P \vee \neg Q)$;

$P \to Q = \neg P \vee Q$;

$P \leftrightarrow Q = (P \to Q)\&(Q \to P)$.

На рівні чистих першопорядкових логік (кванторному рівні) до логічних зв'язок додаємо 1-арні параметричні компози-

ції *реномінації* $\mathrm{R}_{\bar{x}}^{\bar{v}}$ та *квантифікації* $\exists x$, $\forall x$.

Композиція $\mathrm{R}_{\bar{x}}^{\bar{v}}$ задається так.

Для кожного $d \in {}^{V}A$ маємо

$$\mathrm{R}_{\bar{x}}^{\bar{v}}(P)(d) = P(\mathrm{r}_{\bar{x}}^{\bar{v}}(d)).$$

Композицію $\mathrm{R}_{\bar{x}}^{\bar{v}}$ можна визначити через області істинності та хибності відповідного предиката:

$$T(\mathrm{R}_{\bar{x}}^{\bar{v}}(P)) = \{d \mid \mathrm{r}_{\bar{x}}^{\bar{v}}(d) \in T(P)\} =$$

$$= (\mathrm{r}_{\bar{x}}^{\bar{v}})^{-1}(T(P));$$

$$F(\mathrm{R}_{\bar{x}}^{\bar{v}}(P)) = \{d \mid \mathrm{r}_{\bar{x}}^{\bar{v}}(d) \in F(P)\} =$$

$$= (\mathrm{r}_{\bar{x}}^{\bar{v}})^{-1}(F(P)).$$

Композиції $\exists x$ і $\forall x$ задамо через області істинності та хибності предикатів $\exists xP$ і $\forall xP$:

$T(\exists xP) = \{d \in {}^{V}A \mid T \in P[d\nabla x \mapsto a]$ для деякого $a \in A\}$;

$F(\exists xP) = \{d \in {}^{V}A \mid F \in P[d\nabla x \mapsto a]$ для всіх $a \in A\}$;

$T(\forall xP) = \{d \in {}^{V}A \mid T \in P[d\nabla x \mapsto a]$ для всіх $a \in A\}$;

$F(\forall xP) = \{d \in {}^{V}A \mid F \in P[d\nabla x \mapsto a]$ для деякого $a \in A\}$.

Композиції $\neg$, $\vee$, $\mathrm{R}_{\bar{x}}^{\bar{v}}$, $\exists x$ – це базові композиції логік кванторного рівня.

Композиція $\forall x$ є похідною:

$\forall xP = \neg \exists x \neg P$.

Результатом послідовного виконання двох композицій $\mathrm{R}_{\bar{y}}^{\bar{w}}$ (застосовується першою) та $\mathrm{R}_{\bar{x}}^{\bar{v}}$ (застосовується другою) є їх згортка – композиція реномінації $\mathrm{R}_{\bar{x}}^{\bar{v}} \circ \mathrm{R}_{\bar{y}}^{\bar{w}}$. Вона визначається так: для кожного $d \in {}^{V}A$

$$\mathrm{R}_{\bar{x}}^{\bar{v}} \circ \mathrm{R}_{\bar{y}}^{\bar{w}}(P)(d) = P(\mathrm{r}_{\bar{y}}^{\bar{w}} \bullet \mathrm{r}_{\bar{x}}^{\bar{v}}(d)).$$

Основні властивості композицій реномінації:

– $\mathrm{R}_{z,\bar{x}}^{z,\bar{v}}(P) = \mathrm{R}_{\bar{x}}^{\bar{v}}(P)$;

– $\mathrm{R}_{\bar{x}}^{\bar{v}}(\neg P) = \neg \mathrm{R}_{\bar{x}}^{\bar{v}}(P)$;

– $\mathrm{R}_{\bar{x}}^{\bar{v}}(P \vee Q) = \mathrm{R}_{\bar{x}}^{\bar{v}}(P) \vee \mathrm{R}_{\bar{x}}^{\bar{v}}(Q)$;

– $\mathrm{R}_{\bar{x}}^{\bar{v}}(P \,\&\, Q) = \mathrm{R}_{\bar{x}}^{\bar{v}}(P) \,\&\, \mathrm{R}_{\bar{x}}^{\bar{v}}(Q)$;

– $\mathrm{R}_{y,\bar{x}}^{z,\bar{v}}(P) = \mathrm{R}_{\bar{x}}^{\bar{v}}(P)$ за такої умови: $z \in V$ неістотне для P.

Традиційні властивості $\exists x$ та $\forall x$:

– $\exists x \exists y P = \exists y \exists x P$; $\forall x \forall y P = \forall y \forall x P$;

– $\neg \exists x P = \forall x \neg P$; $\neg \forall x P = \exists x \neg P$;

– $\exists x \exists x P = \exists x P$; $\exists x \forall x P = \forall x P$;

– $\forall x \exists x P = \exists x P$; $\forall x \forall x P = \forall x P$;

– $\exists x P \vee \exists x Q = \exists x (P \vee Q)$;

– $\forall x P \,\&\, \forall x Q = \forall x (P \& Q)$.

Залучаючи до розгляду реномінації, маємо властивості:

– $\mathrm{R}_{\bar{v},y}^{\bar{u},x}(\exists xP) = \mathrm{R}_{\bar{v}}^{\bar{u}}(\exists xP)$;

зокрема: $\mathrm{R}_{y}^{x}(\exists xP) = \exists xP$;

– $\mathrm{R}_{\bar{v},y}^{\bar{u},x}(\forall xP) = \mathrm{R}_{\bar{v}}^{\bar{u}}(\forall xP)$;

зокрема: $\mathrm{R}_{y}^{x}(\forall xP) = \forall xP$;

– $\exists yP = \exists z \mathrm{R}_{z}^{y}(P)$, z неістотне для P;

– $\forall yP = \forall z \mathrm{R}_{z}^{y}(P)$, z неістотне для P;

– $\mathrm{R}_{\bar{x}}^{\bar{v}}(\exists yP) = \exists y \mathrm{R}_{\bar{x}}^{\bar{v}}(P)$, де $y \notin \{\bar{v}, \bar{x}\}$;

– $\mathrm{R}_{\bar{x}}^{\bar{v}}(\forall yP) = \forall y \mathrm{R}_{\bar{x}}^{\bar{v}}(P)$, де $y \notin \{\bar{v}, \bar{x}\}$;

– $\mathrm{R}_{\bar{x}}^{\bar{v}}(\exists yP) = \exists z \mathrm{R}_{\bar{x}}^{\bar{v}} \circ_{z}^{y}(P)$, де z неістотне для P, $z \notin \{\bar{v}, \bar{x}\}$;

– $\mathrm{R}_{\bar{x}}^{\bar{v}}(\forall yP) = \forall z \mathrm{R}_{\bar{x}}^{\bar{v}} \circ_{z}^{y}(P)$, де z неістотне для P, $z \notin \{\bar{v}, \bar{x}\}$.

Теорема 3. Композиції $\neg$, $\vee$, $\&$, $\mathrm{R}_{\bar{x}}^{\bar{v}}$, $\exists x$, $\forall x$ зберігають однозначність, тотальність, монотонність, антитонність квазіарних предикатів.

Наслідок 1. Класи P-предикатів, T-предикатів, TS-предикатів, RM-предикатів, RA-предикатів, PE-предикатів, TA-предикатів замкнені відносно композицій $\neg$, $\vee$, $\&$, $\mathrm{R}_{\bar{x}}^{\bar{v}}$, $\exists x$, $\forall x$.

Алгебру $(Pr_A^V, \{\neg, \vee, \mathrm{R}_{\bar{x}}^{\bar{v}}, \exists x\})$, де Pr_A^V – певний клас *V-A*-квазіарних предикатів, назвемо чистою першопорядковою композиційною предикатною алгеброю (алгеброю квазіарних предикатів).

Необхідна умова, щоб клас квазіарних предикатів Pr_A^V утворив алгебру – замкненість щодо операцій алгебри, тобто замкненість класу Pr_A^V щодо композицій $\neg, \vee, \mathrm{R}_{\bar{x}}^{\bar{v}}, \exists x$.

Надалі будемо розглядати композиційні предикатні алгебри із розширеною множиною базових композицій $CQ = \{\neg, \vee, \&, \mathrm{R}_{\bar{x}}^{\bar{v}}, \exists x, \forall x\}$.

В загальному випадку отримуємо алгебру $QR_A^V = (PrR_A^V, CQ)$ *V-A*-квазіарних *R*-предикатів.

Згідно наслідку 1, в алгебрі QR_A^V можна виділити такі підалгебри:

– алгебра *P*-предикатів
$QP_A^V = (PrP_A^V, CQ)$,

– алгебра *T*-предикатів
$QT_A^V = (PrT_A^V, CQ)$,

– алгебра *RM*-предикатів
$QRM_A^V = (PrRM_A^V, CQ)$,

– алгебра *RA*-предикатів
$QRA_A^V = (PrRA_A^V, CQ)$,

– алгебра *PE*-предикатів
$QPE_A^V = (PrPE_A^V, CQ)$,

– алгебра *TA*-предикатів
$QTA_A^V = (PrTA_A^V, CQ)$,

– алгебра *TS*-предикатів
$QTS_A^V = (PrTS_A^V, CQ)$.

Можна також виділити сингулярні підалгебри з 1-елементним носієм

$$\bot_{V\text{-}A} = (\{\bot\}, CQ),\ \top_{V\text{-}A} = (\{\top\}, CQ).$$

Предикатні алгебри (Pr_1, CQ) та (Pr_2, CQ) дуальні, якщо $\delta(\mathrm{Pr}_1) = \mathrm{Pr}_2$.

Тут δ – відображення дуалізації. Зрозуміло, що тоді $\delta(\mathrm{Pr}_2) = \mathrm{Pr}_1$.

Визначені вище предикатні алгебри утворюють такі дуальні пари:

QP_A^V та QT_A^V, QPE_A^V та QTA_A^V, QRM_A^V та QRA_A^V, $\bot_{V\text{-}A}$ та $\top_{V\text{-}A}$.

Алгебри QR_A^V та QTS_A^V є автодуальними.

3. Квазіарні реляції

Під квазіарною реляцією будемо розуміти [5] довільну $L \subseteq {}^V A$.

Квазіарні реляції можна трактувати як області істинності тотальних однозначних квазіарних предикатів. Дуальне трактування квазіарних реляцій – це області хибності таких предикатів.

Для квазіарних реляцій як множин ІМ вводимо традиційні операції: об'єднання $\cup$, перетин $\cap$, доповнення $\sim$.

При трактуванні квазіарних реляцій як областей істинності тотальних однозначних квазіарних предикатів композиціям $\neg$, $\vee$, & для предикатів відповідають операції $\sim$, $\cup$, $\cap$ для областей істинності відповідних предикатів. При дуальному трактуванні квазіарних реляцій як областей хибності зазначених предикатів композиціям $\neg$, $\vee$, & для предикатів відповідають операції $\sim$, $\cap$, $\cup$ для їх областей хибності.

Для квазіарних реляцій вводимо також спеціальні номінативні операції реномінації та квантифікації.

Операція реномінації $\rho_{\bar{x}}^{\bar{v}}$ індукована відповідною операцією реномінації ІМ $\mathrm{r}_{\bar{x}}^{\bar{v}}$:

$$\rho_{\bar{x}}^{\bar{v}}(L) = \{d \in {}^V A \mid \mathrm{r}_{\bar{x}}^{\bar{v}}(d) \in L\} = (\mathrm{r}_{\bar{x}}^{\bar{v}})^{-1}(L).$$

Операції квантифікації $\mathcal{E}x$ та $\mathcal{A}x$ індуковані відповідними композиціями квазіарних предикатів $\exists x$ та $\forall x$. Задамо їх так:

$\mathcal{E}x(L) = \{ d \in {}^V A \mid d\ \nabla x \mapsto a \in L$ для деякого $a \in A\}$;

$\mathcal{A}x(L) = \{ d \in {}^V A \mid d\ \nabla x \mapsto a \in L$ для всіх $a \in A\}$.

При трактуванні *L* як області істинності деякого предиката *P* квазіреляції $\mathcal{E}x(L)$ та $\mathcal{A}x(L)$ трактуємо як області істинності предикатів $\exists xP$ та $\forall xP$. При дуальному трактуванні *L* як області хибності предиката *P*

трактуємо $\mathcal{E}x(L)$ та $\mathcal{A}x(L)$ як області хибності предикатів $\forall xP$ та $\exists xP$.

Ім'я $z \in V$ *неістотне* для квазіарної реляції L, якщо для всіх $d_1,\ d_2 \in {}^{V}A$ таких, що $d_1 =_{-z} d_2$, маємо: $d_1 \in L \Leftrightarrow d_2 \in L$.

Послідовне застосування двох операцій реномінації можна подати у вигляді однієї, яку назвемо їх згорткою.

Згортка операцій $\rho_{\bar{y}}^{\bar{w}}$ (застосовується першою) та $\rho_{\bar{x}}^{\bar{v}}$ (застосовується другою) – це операція реномінації, яку позначаємо $\rho_{\bar{y}}^{\bar{w}} \circ_{\bar{x}}^{\bar{v}}$. Ця операція визначається так:

$$\rho_{\bar{x}}^{\bar{v}}(\rho_{\bar{y}}^{\bar{w}}(L)) = \rho_{\bar{y}}^{\bar{w}} \circ_{\bar{x}}^{\bar{v}}(L) = (\mathrm{r}_{\bar{y}}^{\bar{w}} \bullet_{\bar{x}}^{\bar{v}})^{-1}(L).$$

В класі квазіарних реляцій можна виділити підкласи замкнених вгору квазіарних реляцій та замкнених вниз квазіарних реляцій.

Твердження 3. Нехай V-A-квазіарний предикат P монотонний, тоді множини $T(P)$ і $F(P)$ замкнені вгору.

Твердження 4. Нехай V-A-квазіарний предикат P антитонний, тоді множини $T(P)$ і $F(P)$ замкнені вниз.

Розглянемо властивості квазіарних реляцій.

Для операцій $\sim$, $\cup$, $\cap$ маємо традиційні властивості булевої алгебри множин.

1. Комутативність $\cup$ та $\cap$:

$L \cup M = M \cup L$;

$L \cap M = M \cap L$.

2. Асоціативність $\cup$ та $\cap$:

$(L \cup M) \cup R = L \cup (M \cup R)$;

$(L \cap M) \cap R = L \cap (M \cap R)$.

3. Дистрибутивність $\cup$ відносно $\cap$ та $\cap$ відносно $\cup$:

$(L \cup M) \cap R = (L \cap R) \cup (M \cap R)$;

$(L \cap M) \cup R = (L \cup R) \cap (M \cup R)$.

4. Зняття подвійного заперечення:

$\sim\sim L = L$.

5. Ідемпотентність $\cup$ та $\cap$:

$L = L \cup L$;

$L = L \cap L$.

6. Закони де Моргана:

$\sim(L \cup M) = (\sim M) \cap (\sim L)$;

$\sim(L \cap M) = (\sim M) \cup (\sim L)$.

Можна вважати базовими операції $\sim$ та $\cup$, тоді операція $\cap$ є похідною:

$L \cap M = \sim((\sim M) \cup (\sim L))$.

Властивості номінативних операцій реномінації та квантифікації для квазіарних реляцій індуковані відповідними властивостями квазіарних предикатів.

Для операції реномінації маємо:

– $\rho_{z,\bar{x}}^{z,\bar{v}}(L) = \rho_{\bar{x}}^{\bar{v}}(L)$;

– $\rho_{\bar{x}}^{\bar{v}}(\sim L) = \sim \rho_{\bar{x}}^{\bar{v}}(L)$;

– $\rho_{\bar{x}}^{\bar{v}}(L \cup M) = \rho_{\bar{x}}^{\bar{v}}(L) \cup \rho_{\bar{x}}^{\bar{v}}(M)$;

– $\rho_{\bar{x}}^{\bar{v}}(L \cap M) = \rho_{\bar{x}}^{\bar{v}}(L) \cap \rho_{\bar{x}}^{\bar{v}}(M)$;

– $\rho_{y,\bar{x}}^{z,\bar{v}}(L) = \rho_{\bar{x}}^{\bar{v}}(L)$ за умови, що $z \in V$ неістотне для L.

Для операцій квантифікації маємо:

– $\mathcal{E}x(\mathcal{E}y(L)) = \mathcal{E}y(\mathcal{E}x(L))$;

– $\mathcal{A}x(\mathcal{A}y(L)) = \mathcal{A}y(\mathcal{A}x(L))$;

– $\sim(\mathcal{E}x(L)) = \mathcal{A}x(\sim(L))$;

– $\sim(\mathcal{A}x(L)) = \mathcal{E}x(\sim(L))$;

– $\mathcal{E}x(\mathcal{E}x(L)) = \mathcal{E}x(L)$;

– $\mathcal{E}x(\mathcal{A}x(L)) = \mathcal{A}x(L)$;

– $\mathcal{A}x(\mathcal{E}x(L)) = \mathcal{E}x(L)$;

– $\mathcal{A}x(\mathcal{A}x(L)) = \mathcal{A}x(L)$;

– $\mathcal{E}x(L) \cup \mathcal{E}x(M) = \mathcal{E}x(L \cup M)$;

– $\mathcal{A}x(L) \cap \mathcal{A}x(M) = \mathcal{A}x(L \cap M)$.

Можна вважати базовою операцію $\mathcal{E}x$, тоді операція $\mathcal{A}x$ є похідною:

$$\mathcal{A}x(L) = \sim \mathcal{E}x(\sim(L)).$$

Для операції реномінації маємо:

– $\mathcal{E}y(L) = \mathcal{E}y(\rho_z^y(L))$,

якщо z неістотне для L;

– $\mathcal{A}y(L) = \mathcal{A}y(\rho_z^y(L))$,

якщо z неістотне для L;

– $\rho^{\bar{u},x}_{\bar{v},y}(\mathcal{E}x(L)) = \rho^{\bar{u}}_{\bar{v}}(\mathcal{E}x(L))$;

зокрема: $\rho^{x}_{y}(\mathcal{E}x(L)) = \mathcal{E}x(L)$;

– $\rho^{\bar{u},x}_{\bar{v},y}(\mathcal{A}x(L)) = \rho^{\bar{u}}_{\bar{v}}(\mathcal{A}x(L))$;

зокрема: $\rho^{x}_{y}(\mathcal{E}x(L)) = \mathcal{E}x(L)$;

– $\rho^{\bar{v}}_{\bar{x}}(\mathcal{E}y(L)) = \mathcal{E}y(\rho^{\bar{v}}_{\bar{x}}(L))$,

якщо $y \notin \{\bar{v}, \bar{x}\}$;

– $\rho^{\bar{v}}_{\bar{x}}(\mathcal{A}y(L)) = \mathcal{A}y(\rho^{\bar{v}}_{\bar{x}}(L))$,

якщо $y \notin \{\bar{v}, \bar{x}\}$;

– $\rho^{\bar{v}}_{\bar{x}}(\mathcal{E}y(L)) = \mathcal{E}z(\rho^{\bar{v}}_{\bar{x}} \circ^{y}_{z}(L))$,

якщо z неістотне для L, $z \notin \{\bar{v}, \bar{x}\}$;

– $\rho^{\bar{v}}_{\bar{x}}(\mathcal{A}y(L)) = \mathcal{A}z(\rho^{\bar{v}}_{\bar{x}} \circ^{y}_{z}(L))$,

якщо z неістотне для L, $z \notin \{\bar{v}, \bar{x}\}$.

4. Алгебри квазіарних реляцій

Носієм алгебри квазіарних реляцій є множина квазіарних реляцій, а множина базових операцій – це $\{\sim, \cup, \cap, \rho^{\bar{v}}_{\bar{x}}, \mathcal{E}x, \mathcal{A}x\}$, яку позначимо O_{QR}. Зауважимо, що можна брати мінімальну множину базових операцій $\{\sim, \cup, \rho^{\bar{v}}_{\bar{x}}, \mathcal{E}x\}$, кожна з яких незалежна від інших, проте для зручності та виразності використовуємо саме O_{QR}.

Алгебра квазіарних реляцій – це об'єкт $A_{QR} = (2^{{}^{V}A}; O_{QR})$.

Теорема 4. Алгебра A_{QR} ізоморфна алгебрі *TS*-предикатів $QTS^{V}_{A} = (PrTS^{V}_{A}, CQ)$.

Можна задати два природних ізоморфізми алгебри QTS^{V}_{A} на алгебру A_{QR}:

$\varphi_T : PrTS^{V}_{A} \to 2^{{}^{V}A}$, його задаємо умовою $\varphi_T(P) = T(P)$;

$\varphi_F : PrTS^{V}_{A} \to 2^{{}^{V}A}$, його задаємо умовою $\varphi_F(P) = F(P)$.

Відображення φ_T зіставляє кожному предикату його область істинності, а відображення φ_F – його область хибності.

Зауважимо, що для тотального однозначного предиката P його області істинності та хибності пов'язані так:

$$T(P) = \sim F(P); F(P) = \sim T(P).$$

Для φ_T виконуються умови збереження значення базових операцій:

$\varphi_T(\neg P) = T(\neg P) = F(P) = \sim T(P) = \sim \varphi_T(P)$,

$\varphi_T(P \vee Q) = T(P \vee Q) = T(P) \cup T(Q) =$

$= \varphi_T(P) \cup \varphi_T(Q)$,

$\varphi_T(P \& Q) = T(P \& Q) = T(P) \cap T(Q) =$

$= \varphi_T(P) \cap \varphi_T(Q)$,

$\varphi_T(\mathrm{R}^{\bar{v}}_{\bar{x}}(P)) = T(\mathrm{R}^{\bar{v}}_{\bar{x}}(P)) = (\mathrm{r}^{\bar{v}}_{\bar{x}})^{-1}(T(P)) =$

$= \rho^{\bar{v}}_{\bar{x}}(T(P)) = \rho^{\bar{v}}_{\bar{x}}(\varphi_T(P))$,

$\varphi_T(\exists x P) = T(\exists x P) = \mathcal{E}x(T(P)) = \mathcal{E}x(\varphi_T(P))$,

$\varphi_T(\forall x P) = T(\forall x P) = \mathcal{A}x(T(P)) = \mathcal{A}x(\varphi_T(P))$.

Для φ_F теж виконуються умови збереження значення базових операцій:

$\varphi_F(\neg P) = F(\neg P) = T(P) = \sim F(P) = \sim \varphi_F(P)$,

$\varphi_F(P \vee Q) = F(P \vee Q) = F(P) \cap F(Q) =$

$= \varphi_F(P) \cap \varphi_F(Q)$,

$\varphi_F(P \& Q) = F(P \& Q) = F(P) \cup F(Q) =$

$= \varphi_F(P) \cup \varphi_F(Q)$,

$\varphi_F(\mathrm{R}^{\bar{v}}_{\bar{x}}(P)) = F(\mathrm{R}^{\bar{v}}_{\bar{x}}(P)) = (\mathrm{r}^{\bar{v}}_{\bar{x}})^{-1}(F(P)) =$

$= \rho^{\bar{v}}_{\bar{x}}(F(P)) = \rho^{\bar{v}}_{\bar{x}}(\varphi_F(P))$,

$\varphi_F(\exists x P) = F(\exists x P) = \mathcal{A}x(F(P)) = \mathcal{A}x(\varphi_F(P))$,

$\varphi_F(\forall x P) = F(\forall x P) = \mathcal{E}x(F(P)) = \mathcal{E}x(\varphi_F(P))$.

Таким чином, φ_T та φ_F – ізоморфізми алгебри QTS^{V}_{A} на алгебру A_{QR}.

Дія композицій $\neg$, $\vee$, &, $\mathrm{R}^{\bar{v}}_{\bar{x}}$, $\exists x$, $\forall x$ на предикати при відображенні φ_T відповідає дії операцій $\sim, \cup, \cap, \rho^{\bar{v}}_{\bar{x}}$, $\mathcal{E}x$, $\mathcal{A}x$ на квазіарні реляції – області їх істинності.

При відображенні φ_F дія композицій $\neg$, $\vee$, &, $\mathrm{R}^{\bar{v}}_{\bar{x}}$, $\exists x$, $\forall x$ на предикати відповідає дії операцій $\sim, \cap, \cup, \rho^{\bar{v}}_{\bar{x}}$, $\mathcal{A}x$, $\mathcal{E}x$, на квазіарні реляції – області їх хибності.

Для опису алгебр квазіарних реляцій природно використовувати першопорядкову мову із таким алфавітом:

– множина сигнатурних символів $Cs = \{\neg, \vee, \&, R_{\bar{x}}^{\bar{v}}, \exists x, \forall x\}$,

– множина Rs символів реляцій,

– множина V предметних імен, в якій виділена нескінченна підмножина $U \subseteq V$ тотально неістотних імен.

Предметні імена $x \in V$ позначають елементи множини базових даних A, сигнатурні символи позначають відповідні операції над реляціями, символи Rs позначають (виділяють) базові реляції в множині квазіарних реляцій.

Формули мови описують побудову складніших реляцій із базових. Дамо індуктивне визначення множини Fr формул:

– $Rs \subseteq Fr$; формули $p \in Rs$ атомарні;

– нехай $\Phi, \Psi \in Fr$, тоді

$$\neg\Phi, \vee\Phi\Psi, \&\Phi\Psi, R_{\bar{x}}^{\bar{v}}\Phi, \exists x\Phi, \forall x\Phi \in Fr.$$

Інтерпретуємо мову на алгебрах квазіарних реляцій.

Задамо стандартну інтерпретацію, коли квазіарні реляції трактуються як області істинності тотальних однозначних квазіарних предикатів. При такій інтерпретації сигнатурні символи $\neg, \vee, \&$, $R_{\bar{x}}^{\bar{v}}$, $\exists x$, $\forall x$ відповідно інтерпретуються як операції $\sim, \cup, \cap, \rho_{\bar{x}}^{\bar{v}}, \mathcal{E}x, \mathcal{A}x$.

Для позначення базових реляцій задаємо тотальне однозначне відображення $I_{RT} : Rs \to 2^{V^A}$. Далі продовжимо його до відображення $I_{RT} : Fr \to 2^{V^A}$:

– $I_{RT}(\neg\Phi) = \sim I_{RT}(\Phi)$,

– $I_{RT}(\vee\Phi\Psi) = I_{RT}(\Phi) \cup I_{RT}(\Psi)$,

– $I_{RT}(\&\Phi\Psi) = I_{RT}(\Phi) \cap I_{RT}(\Psi)$,

– $I_{RT}(R_{\bar{x}}^{\bar{v}}(\Phi)) = \rho_{\bar{x}}^{\bar{v}}(I_{RT}(\Phi))$,

– $I_{RT}(\exists x\Phi) = \mathcal{E}x(I_{RT}(\Phi))$,

– $I_{RT}(\forall x\Phi) = \mathcal{A}x(I_{RT}(\Phi))$.

Задамо тепер дуальну інтерпретацію, коли квазіарні реляції трактуються як області хибності тотальних однозначних квазіарних предикатів. При дуальній інтерпретації сигнатурні символи $\neg, \vee, \&$, $R_{\bar{x}}^{\bar{v}}$, $\exists x$, $\forall x$ інтерпретуються відповідно як операції $\sim, \cap, \cup, \rho_{\bar{x}}^{\bar{v}}, \mathcal{A}x, \mathcal{E}x$.

Для позначення базових реляцій задамо тотальне однозначне $I_{RF} : Rs \to 2^{V^A}$.

Продовжимо його до $I_{RF} : Fr \to 2^{V^A}$:

– $I_{RF}(\neg\Phi) = \sim I_{RF}(\Phi)$,

– $I_{RF}(\vee\Phi\Psi) = I_{RF}(\Phi) \cap I_{RF}(\Psi)$,

– $I_{RF}(\&\Phi\Psi) = I_{RF}(\Phi) \cup I_{RF}(\Psi)$,

– $I_{RF}(R_{\bar{x}}^{\bar{v}}(\Phi)) = \rho_{\bar{x}}^{\bar{v}}(I_{RF}(\Phi))$,

– $I_{RF}(\exists x\Phi) = \mathcal{A}x(I_{RF}(\Phi))$,

– $I_{RF}(\forall x\Phi) = \mathcal{E}x(I_{RF}(\Phi))$.

Таким чином, клас тотальних однозначних квазіарних предикатів можна описати як за допомогою композиційних предикатних алгебр, так і за допомогою алгебр квазіарних реляцій. Останнє можна робити двома способами: трактуючи реляції як області істинності предикатів і трактуючи їх як області хибності предикатів.

5. Алгебри бі-квазіарних реляцій

Для задання тотального однозначного квазіарного предиката необхідно вказавати його область істинності або його область хибності, при цьому області істинності та хибності пов'язані за допомогою операції заперечення. При переході до нетотальних чи неоднозначних предикатів ця залежність зникає. Для задання нетотального чи неоднозначного квазіарного предиката необхідно вказувати як область його істинності, так і область хибності.

Дія композицій $\neg$, $\vee$, $\&$, $R_{\bar{x}}^{\bar{v}}$, $\exists x$, $\forall x$ на квазіарні предикати відповідає дії операцій $\sim, \cup, \cap, \rho_{\bar{x}}^{\bar{v}}, \mathcal{E}x, \mathcal{A}x$ на області їх істинності та дії операцій $\sim, \cap, \cup, \rho_{\bar{x}}^{\bar{v}}, \mathcal{A}x, \mathcal{E}x$ на області їх хибності. Таким чином, дія композиції на квазіарний предикат рівносильна дії відповідної операції та дуальної до неї до двох квазіарних реляцій – області

істинності та області хибності.

Побудуємо алгебри, визначені на множинах пар квазіарних реляцій. Назвемо їх алгебрами бі-квазіарних реляцій.

Алгебри бі-квазіарних реляцій мають вигляд $A_{BQR} = (2^{{}^{V}A} \times 2^{{}^{V}A}; O_{BQR})$, де $O_{BQR} = \{\neg_B, \vee_B, \&_B, R^{\bar{v}}_{\bar{x}\,B}, \exists x_B, \forall x_B\}$ є множиною базових операцій.

Операції $\neg_B, \vee_B, \&_B, R^{\bar{v}}_{\bar{x}\,B}, \exists x_B, \forall x_B$ діють на парах квазіарних реляцій так:

– як операції $\sim, \cup, \cap, \rho^{\bar{v}}_{\bar{x}}, \mathcal{E}x, \mathcal{A}x$ на першій компоненті пари (стандартно, як операції на областях істинності);

– як операції $\sim, \cap, \cup, \rho^{\bar{v}}_{\bar{x}}, \mathcal{A}x, \mathcal{E}x$ на другій компоненті пари (дуально, як операції на областях хибності).

Таким чином, на парах квазіарних реляцій визначаємо:

$$\neg_B(L, M) = (M, L);$$

$$(L, M) \vee_B (R, S) = (L \cup R, M \cap S);$$

$$(L, M) \&_B (R, S) = (L \cap R, M \cup S);$$

$$R^{\bar{v}}_{\bar{x}\,B}(L, M) = (\rho^{\bar{v}}_{\bar{x}}(L), \rho^{\bar{v}}_{\bar{x}}(M));$$

$$\exists_B x\,(L, M) = (\mathcal{E}x(L), \mathcal{A}x(M));$$

$$\forall_B x\,(L, M) = (\mathcal{A}x(L), \mathcal{E}x(M)).$$

Бі-квазіарна реляція $(L, M) \subseteq {}^{V}A \times {}^{V}A$:

– однозначна, якщо $L \cap M = \varnothing$;

– тотальна, якщо $L \cup M = {}^{V}A$;

– однозначна, якщо $L \cap M = \varnothing$;

– замкнена вгору, якщо L та M замкнені вгору;

– замкнена вниз, якщо L та M замкнені вниз.

Згідно наслідку 1, класи *R*-предикатів, *P*-предикатів, *T*-предикатів, *TS*-предикатів, *RM*-предикатів, *RA*-предикатів, *PE*-предикатів, *TA*-предикатів замкнені відносно композицій $\neg$, $\vee$, &, $\mathrm{R}^{\bar{v}}_{\bar{x}}$, $\exists x$, $\forall x$.

Звідси отримуємо.

Теорема 5. Класи однозначних, тотальних, замкнених вгору, замкнених вниз бі-квазіарних реляцій замкнені відносно операцій $\neg_B, \vee_B, \&_B, R^{\bar{v}}_{\bar{x}\,B}, \exists x_B, \forall x_B$.

Таким чином виділяємо наступні класи бі-квазіарних реляцій:

– *SR* – клас однозначних,

– *TR* – клас тотальних,

– *STR* – клас однозначних тотальних,

– *MR* – клас замкнених вгору,

– *AR* – клас замкнених вниз,

– *SER* – клас замкнених вгору однозначних,

– *TAR* – клас замкнених вниз тотальних бі-квазіарних реляцій.

Ми отримуємо наступні підалгебри алгебри A_{BQR}:

$A_{SBQR} = (SR; O_{BQR})$ – алгебра однозначних бі-квазіарних реляцій;

$A_{TBQR} = (TR; O_{BQR})$ – алгебра тотальних бі-квазіарних реляцій;

$A_{STQR} = (STR; O_{BQR})$ – алгебра однозначних тотальних бі-квазіарних реляцій;

$A_{MBQR} = (MR; O_{BQR})$ – алгебра замкнених вгору бі-квазіарних реляцій;

$A_{ABQR} = (AR; O_{BQR})$ – алгебра замкнених вниз бі-квазіарних реляцій;

$A_{SEBQR} = (SER; O_{BQR})$ – алгебра замкнених вгору однозначних бі-квазіарних реляцій;

$A_{TABQR} = (TAR; O_{BQR})$ – алгебра замкнених вниз тотальних бі-квазіарних реляцій.

У випадку *TS*-предикатів області їх істинності та хибності пов'язані за допомогою операції заперечення. Тому всі елементи *STR* мають вигляд $(L, \sim L)$.

Для бі-квазіарних реляцій відображення дуалізації $\delta : 2^{{}^{V}A} \times 2^{{}^{V}A} \to 2^{{}^{V}A} \times 2^{{}^{V}A}$ задамо так: $\delta(L, M) = (\sim M, \sim L)$.

Теорема 6. Відображення дуалізації:

1) є автоморфізмом алгебри A_{BQR};

2) є ізоморфізмом алгебр A_{SBQR} та A_{TBQR};

3) є ізоморфізмом алгебр A_{MBQR} та A_{ABQR};

4) є ізоморфізмом алгебр A_{SEBQR} та A_{TABQR};

5) є тотожним автоморфізмом алгебри A_{STQR}.

Твердження п. 5 очевидне.

Доведемо п. 1. Покажемо, що для δ виконуються умови збереження значення базових операцій. Маємо:

$$\delta(\neg_B(L, M)) = \delta(M, L) = (\sim L, \sim M) =$$
$$= \neg_B(\sim M, \sim L) = \neg_B(\delta(L, M));$$

$$\delta((L, M) \vee_B (R, S)) = \delta(L \cup R, M \cap S) =$$
$$= (\sim(M \cap S), \sim(L \cup R)) = (\sim M \cup \sim S, \sim L \cap \sim R) =$$
$$= (\sim M, \sim L) \vee_B (\sim S, \sim R) = \delta(L, M) \vee_B \delta(R, S);$$

$$\delta((L, M) \&_B (R, S)) = \delta(L \cap R, M \cup S) =$$
$$= (\sim(M \cup S), \sim(L \cap R)) = (\sim M \cap \sim S, \sim L \cup \sim R) =$$
$$= (\sim M, \sim L) \&_B (\sim S, \sim R) = \delta(L, M) \&_B \delta(R, S);$$

$$\delta(R^{\bar{v}}_{\bar{x}\,B}(L, M)) = \delta(\rho^{\bar{v}}_{\bar{x}}(L), \rho^{\bar{v}}_{\bar{x}}(M)) =$$
$$= (\sim \rho^{\bar{v}}_{\bar{x}}(M), \sim \rho^{\bar{v}}_{\bar{x}}(L)) = (\rho^{\bar{v}}_{\bar{x}}(\sim M), \rho^{\bar{v}}_{\bar{x}}(\sim L)) =$$
$$= R^{\bar{v}}_{\bar{x}\,B}(\sim M, \sim L) = R^{\bar{v}}_{\bar{x}\,B}(\delta(L, M));$$

$$\delta(\exists_B x\,(L, M)) = \delta(\mathcal{E}x(L), \mathcal{A}x(M)) =$$
$$= (\sim \mathcal{A}x(M), \sim \mathcal{E}x(L)) = (\mathcal{E}x(\sim M), \mathcal{A}x(\sim L)) =$$
$$= \exists_B x\,(\sim M, \sim L) = \exists_B x\,(\delta(L, M));$$

$$\delta(\forall_B x\,(L, M)) = \delta(\mathcal{A}x(L), \mathcal{E}x(M)) =$$
$$= (\sim \mathcal{E}x(M), \sim \mathcal{A}x(L)) = (\mathcal{A}x(\sim M), \mathcal{E}x(\sim L)) =$$
$$= \forall_B x\,(\sim M, \sim L) = \forall_B x\,(\delta(L, M)).$$

Таким чином, δ – автоморфізм алгебри A_{BQR}.

Подібним чином доводимо пп.2–4.

Теорема 7. 1) алгебри A_{BQR} і QR^V_A ізоморфні;

2) алгебри A_{SBQR}, A_{TBQR}, QP^V_A, QT^V_A ізоморфні;

3) алгебри A_{MBQR}, A_{ABQR}, QRM^V_A, QRA^V_A ізоморфні;

4) алгебри A_{SEBQR}, A_{TABQR}, QPE^V_A, QTA^V_A ізоморфні;

5) алгебри A_{STQR} і QTS^V_A ізоморфні.

Твердження п. 5 очевидне.

Доведемо п. 1. Для цього задамо відображення ізоморфізму φ алгебри QR^V_A на алгебру A_{BQR}.

Відображення $\varphi : PrR^V_A \to 2^{V_A} \times 2^{V_A}$ задаємо наступною умовою:

$$\varphi(P) = (T(P), F(P)).$$

Таке відображення φ зіставляє кожному предикату його область істинності та область хибності.

Покажемо, що для φ виконуються умови збереження значення базових операцій:

$$\varphi(\neg P) = (T(\neg P), F(\neg P)) =$$
$$= (F(P), T(P)) = \neg_B(T(P), F(P)) = \neg_B(\varphi(P));$$

$$\varphi(P \vee Q) = (T(P \vee Q), F(P \vee Q)) =$$
$$= (T(P) \cup T(Q), F(P) \cap F(Q)) =$$
$$= (T(P), F(P)) \vee_B (T(Q), F(Q)) = \varphi(P) \vee_B \varphi(Q);$$

$$\varphi(P \& Q) = (T(P \& Q), F(P \& Q)) =$$
$$= (T(P) \cap T(Q), F(P) \cup F(Q)) =$$
$$= (T(P), F(P)) \&_B (T(Q), F(Q)) =$$
$$= \varphi(P) \&_B \varphi(Q);$$

$$\varphi(\mathrm{R}^{\bar{v}}_{\bar{x}}(P)) = (T(\mathrm{R}^{\bar{v}}_{\bar{x}}(P)), F(\mathrm{R}^{\bar{v}}_{\bar{x}}(P))) =$$
$$= ((\mathrm{r}^{\bar{v}}_{\bar{x}})^{-1}(T(P)), (\mathrm{r}^{\bar{v}}_{\bar{x}})^{-1}(F(P))) =$$
$$= (\rho^{\bar{v}}_{\bar{x}}(T(P)), \rho^{\bar{v}}_{\bar{x}}(F(P))) = \rho^{\bar{v}}_{\bar{x}}(T(P), F(P)) =$$
$$= R^{\bar{v}}_{\bar{x}\,B}(T(P), F(P)) = R^{\bar{v}}_{\bar{x}\,B}(\varphi(P));$$

$$\varphi(\exists x P) = (T(\exists x P), F(\exists x P)) =$$
$$= (\mathcal{E}x(T(P)), \mathcal{A}x(F(P))) = \exists_B x\,(T(P), F(P)) =$$
$$= \exists_B x\,(\varphi(P));$$

$$\varphi(\forall x P) = (T(\forall x P), F(\forall x P)) =$$
$$= (\mathcal{A}x(T(P)), \mathcal{E}x(F(P))) = \forall_B x\,(T(P), F(P)) =$$
$$= \forall_B x\,(\varphi(P)).$$

Таким чином, φ – ізоморфізм алгеб-

ри QR_A^V на алгебру A_{BQR}.

Подібним чином доводимо пп. 2–4.

Для опису алгебр бі-квазіарних реляцій використовуємо описану вище першопорядкову мову із множиною сигнатурних символів $Cs = \{\neg, \vee, \&, R_{\bar{x}}^{\bar{v}}, \exists x, \forall x\}$.

Інтерпретуємо цю мову на алгебрах бі-квазіарних реляцій таким чином.

Кожну бі-квазіарну реляцію трактуємо як пару множин – область істинності та область хибності певного квазіарного предиката. Сигнатурні символи $\neg, \vee, \&$, $R_{\bar{x}}^{\bar{v}}, \exists x, \forall x$ інтерпретуються відповідно як операції $\neg_B, \vee_B, \&_B, R_{\bar{x}\,B}^{\bar{v}}, \exists x_B, \forall x_B$ на множині бі-квазіарних реляцій.

Для позначення базових бі-реляцій задаємо тотальне однозначне відображення $I_{BR} : Rs \to 2^{V_A} \times 2^{V_A}$. Таке I_{BR} далі продовжимо до $I_{BR} : Fr \to 2^{V_A} \times 2^{V_A}$:

– $I_{BR}(\neg\Phi) = \neg_B(I_{BR}(\Phi))$,

– $I_{BR}(\vee\Phi\Psi) = I_{BR}(\Phi) \vee_B I_{BR}(\Psi)$,

– $I_{BR}(\&\Phi\Psi) = I_{BR}(\Phi) \&_B I_{BR}(\Psi)$,

– $I_{BR}(R_{\bar{x}}^{\bar{v}}(\Phi)) = R_{\bar{x}\,B}^{\bar{v}}(I_{BR}(\Phi))$,

– $I_{BR}(\exists x\Phi) = \exists_B x\,(I_{BR}(\Phi))$,

– $I_{BR}(\forall x\Phi) = \forall_B x\,(I_{BR}(\Phi))$.

Таким чином, класи квазіарних предикатів можна описати як за допомогою композиційних предикатних алгебр, так і за допомогою алгебр бі-квазіарних реляцій. Кожна бі-квазіарна реляція визначає область істинності та область хибності квазіарного предиката.

Висновки

В роботі побудовано та досліджено низку алгебр квазіарних реляцій (відношень), розглянуто їх зв'язки із алгебрами квазіарних предикатів. На множині всіх квазіарних реляцій природним чином задаються булеві операції об'єднання, перетину, доповнення, а також спеціальні номінативні операції реномінації та квантифікації. Встановлено ізоморфізм алгебри квазіарних реляцій та першопорядкової алгебри тотальних однозначних квазіарних предикатів.

Побудовано алгебри бі-квазіарних реляцій, задані на множинах пар квазіарних реляцій. Визначено та досліджено різні підкласи таких алгебр. Встановлено ізоморфізми алгебри бі-квазіарних реляцій і алгебри квазіарних предикатів; ізоморфізми алгебр однозначних, тотальних, тотальних однозначних бі-квазіарних реляцій та алгебр часткових однозначних, тотальних, тотальних однозначних квазіарних предикатів; ізоморфізми алгебр замкнених вгору, замкнених вниз, замкнених вгору однозначних, замкнених вниз тотальних бі-квазіарних реляцій та алгебр монотонних, антитонних, однозначних еквітонних, тотальних антитонних квазіарних предикатів.

1. *Глушков В.М., Цейтлин Г.Е., Ющенко Е.Л.* Алгебра, языки, программирование. – К.: Наукова думка, 1974. – 328 с.
2. *Нікітченко М.С., Шкільняк С.С.* Математична логіка та теорія алгоритмів. – К.: ВПЦ Київський університет, 2008. – 528 с.
3. *Нікітченко М.С., Шкільняк С.С.* Прикладна логіка. – К.: ВПЦ Київський університет, 2013. – 278 с.
4. *Birkhoff G.* Lattice theory. – Amer. Math. Soc., 1967. – 418 p.
5. *Нікітченко М.С., Шкільняк С.С.* Алгебри квазіарних відношень // Theoretical and Applied Aspects of Program Systems Development (TAAPSD'2014): 11th international conference: proceeding. – K., 2014. – С. 174–181.

References

1. *Glushkov V., Ceytlin G., Yuschenko E.* (1974). Algebras, languages, programming. – Kyiv: Naukova dumka (in Russian).
2. *Nikitchenko M., Shkilniak S.* (2008). Mathematical logic and theory of algorithms. – Kyiv: VPC Kyivskyi Universytet (in Ukrainian).
3. *Nikitchenko M. and Shkilniak S.* (2013). Applied logic. – Kyiv: VPC Kyivskyi Universytet (in Ukrainian).

4. *Birkhoff G.* (1967) Lattice theory. Amer. Math. Soc., 1967.
5. *Nikitchenko M. and Shkilniak S.* (2014). Algebras of quasiary relations. In Theoretical and Applied Aspects of Program Systems Development (TAAPSD'2014): 11th international conference: proceeding. – Kyiv. P. 174–181 (in Ukrainian).

Одержано 09.11.2015

Про авторів:

Нікітченко Микола Степанович,
доктор фізико-математичних наук,
професор, завідувач кафедри теорії
та технології програмування.
Кількість наукових публікацій –
понад 200, у тому числі у фахових
виданнях – 100.
Кількість наукових публікацій в
іноземних виданнях – 30.
Індекс Гірша – 8 (6 з 2010).
http://orcid.org/0000-0002-4078-1062.

Шкільняк Степан Степанович,
доктор фізико-математичних наук,
професор, професор кафедри теорії
та технології програмування.
Кількість наукових публікацій –
понад 200,
у тому числі у фахових виданнях – 90.
Кількість наукових публікацій в
іноземних виданнях – 14.
Індекс Гірша – 4 (3 з 2010).
http://orcid.org/0000-0001-8624-5778.

***Місце роботи авторів*:**

Київський національний університет
імені Тараса Шевченка,
01601, Київ, вул. Володимирська, 60.
Тел.: (044) 259 0519,
(044) 522 0640 (д).
E-mail: ttp@unicyb.kiev.ua

УДК 004.42:510.69

О.С. Шкільняк

ВІДНОШЕННЯ ЛОГІЧНОГО НАСЛІДКУ В ЛОГІКАХ КВАЗІАРНИХ ПРЕДИКАТІВ

Вивчаються відношення логічного наслідку в логіках тотальних однозначних, часткових однозначних, тотальних неоднозначних та часткових неоднозначних предикатів. Поряд із розглянутими раніше відношеннями типів *T, F, TF, IR, DI*, для логік квазіарних предикатів запропоновано і досліджено відношення типів $T\vee F$ та *C*. Описано властивості відношень логічного наслідку. Наведено приклади, які засвідчують відмінності розглянутих відношень. Показана нетранзитивність відношень типів $T\vee F$ та *C*, можливість моделювання відношень типу *C* за допомогою відношень типу *TF*. Встановлено співвідношення між різними відношеннями логічного наслідку.

Ключові слова: логіка, предикат, семантика, логічний наслідок.

Вступ

Розвиток інформаційних технологій зумовлює розширення сфери застосування математичної логіки. Створено багато різноманітних логічних систем, які успішно використовуються в інформатиці й програмуванні. Ці системи зазвичай базуються на класичній логіці предикатів. Водночас принципові обмеження класичної логіки спонукають необхідність розробки нових, програмно-орієнтованих логічних формалізмів. Такими є композиційно-номінативні логіки (КНЛ), збудовані на базі спільного для логіки й програмування композиційно-номінативного підходу.

КНЛ вивчались, зокрема, в [1–4].

Фундаментальним поняттям логіки є поняття логічного наслідку. Широке використання в програмуванні часткових відображень, які можуть бути неоднозначними, робить актуальною проблему дослідження логік із нетрадиційними семантиками та відношень логічного наслідку для цих логік. Такі відношення є семантичною основою побудови числень секвенційного типу. Для пропозиційної логіки нестандартні семантики та відношення логічного наслідку розглянуто в [5]. Для чистих першопорядкових КНЛ (ЧКНЛ) такі відношення вивчались в [2–4].

Мета даної роботи – це дослідження відношень логічного наслідку для КНЛ тотальних однозначних, часткових однозначних, тотальних неоднозначних та часткових неоднозначних предикатів. Поряд із розглянутими в [2–4] відношеннями типів *T, F, TF, Cl* (в нових позначеннях *IR*), *Cm* (в нових позначеннях *DI*), для КНЛ пропонуються і вивчаються відношення типів $T\vee F$ та *C*. Описано властивості таких відношень, наведено низку прикладів, які засвідчують відмінності одних відношень від інших. Показана нетранзитивність відношень типів $T\vee F$ і *C*, можливість моделювання ${}^{R}|=_{C}$ за допомогою ${}^{R}|=_{TF}$. Встановлено співвідношення між різними відношеннями логічного наслідку.

Невизначені в даній роботі поняття тлумачимо в сенсі [1, 2].

1. Квазіарні предикати, їх різновиди. Алгебри предикатів

Під *V-A*-квазіарним предикатом будемо розуміти довільну часткову неоднозначну функцію вигляду $P : {}^{V}A \to \{T, F\}$.

Тут ${}^{V}A$ – клас *V-A*-іменних множин, $\{T, F\}$ – множина істиннісних значень.

V-A-іменна множина (*V-A*-ІМ) – це часткова однозначна функція $d : V \to A$. Трактуємо *V* і *A* як множини предметних імен (змінних) і предметних значень.

Позначимо через $P(d)$ множину тих значень, які *P* може прийняти на $d \in {}^{V}A$. Маємо $P(d) \subseteq \{T, F\}$, тому $P(d)$ може бути одним із значень: $\{\varnothing\}$, $\{T\}$, $\{F\}$, $\{T, F\}$.

Області істинності та хибності предиката $P : {}^{V}A \to \{T, F\}$ – це множини

$$T(P) = \{d \in {}^{V}A \mid T \in P(d)\},$$

$$F(P) = \{d \in {}^{V}A \mid F \in P(d)\}.$$

Ми трактуємо часткові неоднозначні квазіарні предикати як відношення між ${}^{V}A$ та $\{T, F\}$. Їх називають [2] предикатами реляційного типу, або *R*-предикатами.

V-A-квазіарний предикат *P*:

– однозначний, якщо $T(P) \cap F(P) = \varnothing$;

– тотальний, якщо $T(P) \cup F(P) = {}^{V}A$.

Часткові однозначні предикати назвемо *P*-предикатами, тотальні – *T*-предикатами, тотальні однозначні – *TS*-предикатами. Класи часткових неоднозначних, часткових однозначних, тотальних, тотальних однозначних *V-A*-квазіарних предикатів позначимо PrR_A^V, PrP_A^V, PrT_A^V, $PrTS_A^V$.

V-A-квазіарний предикат *P*:

– неспростовний (частково істинний), якщо $F(P) = \varnothing$;

– виконуваний, якщо $T(P) \neq \varnothing$;

– тотально істинний, якщо $T(P) = {}^{V}A$;

– тотально хибний, якщо $F(P) = {}^{V}A$;

– тотожно істинний, якщо

$T(P) = {}^{V}A$ та $F(P) = \varnothing$;

– тотожно хибний, якщо

$T(P) = \varnothing$ та $F(P) = {}^{V}A$;

– всюди невизначений, якщо

$T(P) = \varnothing$ та $F(P) = \varnothing$;

– тотально насичений (повне бінарне відношення), якщо $T(P) = {}^{V}A$ та $F(P) = {}^{V}A$.

Кожний неспростовний та кожний невиконуваний предикат є однозначними.

Всюди невизначений *V-A*-квазіарний предикат позначимо як $\bot_A^V$, тотожно істинний – як T_A^V, тотожно хибний – як F_A^V, тотально насичений – як $\top_A^V$. Якщо *V* та *A* маються на увазі, предикати $\bot_A^V$, T_A^V, F_A^V, $\top_A^V$ позначаємо як $\bot$, T, F, $\top$.

V-A-квазіарний предикат $\tilde{P}$ дуальний до *V-A*-квазіарного предиката *P*, якщо

$$T(\tilde{P}) = \overline{F(P)};\ F(\tilde{P}) = \overline{T(P)}.$$

Із визначень випливає:

Q – *P*-предикат $\Leftrightarrow$ $\tilde{Q}$ – *T*-предикат;

Q – *T*-предикат $\Leftrightarrow$ $\tilde{Q}$ – *P*-предикат;

якщо Q – *TS*-предикат, то $\tilde{Q} = Q$.

Задамо відображення дуалізації $\delta : PrR_A^V \to PrR_A^V$ наступним чином:

$$\delta(P) = \tilde{P} \text{ для кожного } P \in PrR_A^V.$$

Для класів предикатів маємо:

$$\delta(PrP_A^V) = PrT_A^V,\ \delta(PrT_A^V) = PrP_A^V;$$

$$\delta(PrR_A^V) = PrR_A^V,\ \delta(PrTS_A^V) = PrTS_A^V;$$

$$\delta(\{\bot_A^V\}) = \{\top_A^V\},\ \delta(\{\top_A^V\}) = \{\bot_A^V\}.$$

Розглянемо тепер композиції квазіарних предикатів.

На пропозиційному рівні композиції працюють лише з істиннісними значеннями, які вироблені предикатами. Традиційна їх назва – логічні зв'язки. Основними є 1-арна композиція заперечення $\neg$ та 2-арні композиції диз'юнкція $\vee$, кон'юнкція &, імплікація $\to$, еквіваленція $\leftrightarrow$.

Пропозиційні композиції задамо через області істинності й хибності відповідних предикатів. Предикати $\neg(P)$, $\vee(P, Q)$, $\to(P, Q)$, $\&(P, Q)$, $\leftrightarrow(P, Q)$ традиційно позначаємо як $\neg P$, $P \vee Q$, $P \to Q$, $P \& Q$, $P \leftrightarrow Q$.

Предикати $\neg P$ та $P \vee Q$ задамо так:

$T(\neg P) = F(P);\ F(\neg P) = T(P);$

$T(P \vee Q) = T(P) \cup T(Q);\ F(P \vee Q) = (P) \cap F(Q).$

$\neg$ та $\vee$ – це базові пропозиційні композиції. Композиції $\to$, &, $\leftrightarrow$ є похідними, вони виражаються через $\neg$ та $\vee$:

$$P \to Q = \neg P \vee Q;$$

$$P \& Q = \neg(\neg P \vee \neg Q);$$

$$P \leftrightarrow Q = (P \to Q) \& (Q \to P).$$

На рівні ЧКНЛ до пропозиційних композицій додаємо композиції реномінації та квантифікації. Спочатку опишемо необхідні для цього операції над *V-A*-ІМ.

Операцію ∇ накладки задаємо так:

$$\delta \nabla \eta = \eta \cup [v \mapsto a \in \delta \mid v \notin asn(\eta)].$$

Тут $asn(d) = \{v \in V \mid v \mapsto a \in d\}$.

Параметризовану за множиною пар імен операцію $\mathrm{r}_{x_1,\ldots,x_n}^{v_1,\ldots,v_n} : {}^{V}A \to {}^{V}A$ реномінації задаємо так:

$\mathrm{r}^{v_1,...,v_n}_{x_1,...,x_n}(d) = d\nabla[v_1 \mapsto d(x_1),...,v_n \mapsto d(x_n)]$.

Замість $y_1,...,y_n$ далі пишемо $\bar{y}$.

Задамо композицію реномінації $\mathrm{R}^{\bar{v}}_{\bar{x}}$: $\mathrm{R}^{\bar{v}}_{\bar{x}}(P)(d) = P(\mathrm{r}^{\bar{v}}_{\bar{x}}(d))$ для кожного $d \in {}^{V}A$.

Параметричні композиції квантифікації $\exists x$ і $\forall x$ можна задати через області істинності та хибності відповідних предикатів. При цьому композиція $\forall x$ є похідною.

Дамо визначення предиката $\exists xP$:

$T(\exists xP) = \{d \in {}^{V}A \mid T \in P[d\nabla x \mapsto a]$ для деякого $a \in A\}$;

$F(\exists xP) = \{d \in {}^{V}A \mid F \in P[d\nabla x \mapsto a]$ для всіх $a \in A\}$.

Композицію $\forall x$ задаємо умовою

$$\forall xP = \neg\exists x\neg P.$$

Композиції $\neg$, $\vee$, $\mathrm{R}^{\bar{v}}_{\bar{x}}$, $\exists x$ називають базовими композиціями ЧКНЛ.

Властивості композицій квазіарних предикатів описано в [1–4].

Зокрема, $\neg$, $\vee, \mathrm{R}^{\bar{v}}_{\bar{x}}, \exists x$ зберігають однозначність і тотальність квазіарних предикатів. Звідси випливає, що класи *P*-предикатів, *T*-предикатів, *TS*-предикатів замкнені відносно композицій $\neg$, $\vee, \mathrm{R}^{\bar{v}}_{\bar{x}}, \exists x$.

Для предикатів $\bot$ та $\top$ маємо:

$\neg\bot = \bot$, $\bot \vee \bot = \bot$, $\mathrm{R}^{\bar{v}}_{\bar{x}}(\bot) = \bot$, $\exists x(\bot) = \bot$,
$\neg\top = \top$; $\top \vee \top = \top$; $\mathrm{R}^{\bar{v}}_{\bar{x}}(\top) = \top$; $\exists x(\top) = \top$.

Отже, 1-елементні множини $\{\bot\}$ та $\{\top\}$ замкнені відносно $\neg$, $\vee, \mathrm{R}^{\bar{v}}_{\bar{x}}, \exists x$.

Алгебра $QR^V_A = (PrR^V_A, CQ)$, де $CQ = \{\neg, \vee, \mathrm{R}^{\bar{v}}_{\bar{x}}, \exists x\}$, називається чистою першопорядковою композиційною алгеброю квазіарних предикатів. Можна виділити наступні підалгебри алгебри QR^V_A:

$QP^V_A = (PrP^V_A, CQ)$ – алгебра *P*-предикатів,

$QT^V_A = (PrT^V_A, CQ)$ – алгебра *T*-предикатів,

$QTS^V_A = (PrTS^V_A, CQ)$ – алгебра *TS*-предикатів.

Виділяємо сингулярні підалгебри $\bot_{V\text{-}A} = (\{\bot^V_A\}, CQ)$ та $\top_{V\text{-}A} = (\{\top^V_A\}, CQ)$.

Нехай δ – відображення дуалізації.

Алгебри (Pr_1, CQ) та (Pr_2, CQ) дуальні, якщо $\delta(\mathrm{Pr}_1) = \mathrm{Pr}_2$ та $\delta(\mathrm{Pr}_2) = \mathrm{Pr}_1$.

Маємо пару дуальних алгебр QP^V_A та QT^V_A, алгебри QR^V_A та QTS^V_A автодуальні. Дуальними є $\bot_{V\text{-}A}$ та $\top_{V\text{-}A}$.

2. Мови та семантичні моделі

Семантичними моделями ЧКНЛ є [1, 2] чисті першопорядкові композиційні системи квазіарних предикатів. Вони мають вигляд (A, Pr, CQ).

Композиційна система (A, Pr, CQ) задає алгебру даних (A, Pr) та композиційну алгебру предикатів (Pr, CQ). Терми композиційної алгебри трактуємо як формули мови ЧКНЛ.

Алфавіт мови ЧКНЛ:

– множина $Cs = \{\neg, \vee, R^{\bar{v}}_{\bar{x}}, \exists x\}$ символів базових композицій;

– множина *Ps* предикатних символів;

– множина *V* предметних імен (змінних), в якій виділена множина $U \subseteq V$ тотально неістотних [2] імен.

Четвірку $\Sigma = (V, U, Cs, Ps)$ назвемо розширеною сигнатурою мови.

Дамо індуктивне визначення множини *Fr* формул:

– $Ps \subseteq Fr$; формули $p \in Ps$ – атомарні;

– $\Phi, \Psi \in Fr \Rightarrow \neg\Phi, \vee\Phi\Psi, R^{\bar{v}}_{\bar{x}}\Phi, \exists x\Phi \in Fr$.

Для зручності далі пишемо скорочення формул (див. [1, 2]), користуючись символами похідних композицій та інфіксною формою запису для $\vee$, &, $\to$, $\leftrightarrow$.

Інтерпретуємо мову на композиційних системах вигляду $\mathrm{CS} = (A, \mathrm{Pr}, CQ)$. Імена $x \in V$ позначають елементи множини базових даних *A*, символи композицій – композиції із *CQ*. Символи *Ps* позначають базові предикати в множині Pr. Для опису

цього позначення задамо тотальне однозначне відображення $I : Ps \to \Pr$. Відображення інтерпретації формул $I : Fr \to \Pr$ задамо як розширення відображення $I : Ps \to \Pr$ згідно побудови формул із простіших за допомогою символів Cs:

– $I(\neg\Phi) = \neg(I(\Phi))$,

– $I(\vee\Phi\Psi) = \vee(I(\Phi), I(\Psi))$,

– $I(R^{\bar{V}}_{\bar{x}}(\Phi)) = \mathrm{R}^{\bar{V}}_{\bar{x}}(I(\Phi))$;

– $I(\exists x\Phi) = \exists x(I(\Phi))$.

Трійку $\boldsymbol{J} = (\mathrm{CS}, \Sigma, I)$ називатимемо інтерпретацією мови ЧКНЛ сигнатури Σ. Скорочено інтерпретації позначаємо (A, I).

Предикат $\boldsymbol{J}(\Phi)$ – значення формули Φ при інтерпретації $\boldsymbol{J}$ – позначимо Φ_J.

Виділення підалгебр квазіарних предикатів виділяє відповідні класи інтерпретацій. Маємо загальний клас R-інтерпретацій та підкласи P-інтерпретацій, T-інтерпретацій, TS-інтерпретацій. Такі класи інтерпретацій назвемо семантиками, будемо їх позначати відповідно $\mathcal{R}$, $\mathcal{P}$, $\mathcal{T}$, $\mathcal{TS}$.

Отже, далі можна говорити про R-семантику, T-семантику, P-семантику, TS-семантику логік квазіарних предикатів.

Логіки R-предикатів, T-предикатів, P-предикатів, TS-предикатів назвемо логіками з R-семантикою, T-семантикою, P-семантикою, TS-семантикою відповідно.

Семантики $\mathcal{P}$, $\mathcal{T}$, $\mathcal{R}$ в [2] названо неокласичною, пересиченою, загальною.

Для семантик $\mathcal{R}$, $\mathcal{P}$, $\mathcal{T}$, $\mathcal{TS}$ маємо:

$\mathcal{TS} \subset \mathcal{P} \subset \mathcal{R}$, $\mathcal{TS} \subset \mathcal{T} \subset \mathcal{R}$.

Відображення дуалізації δ продовжимо на класи інтерпретацій.

Інтерпретація $\delta(\boldsymbol{J}) = (A,\ I_\delta)$ дуальна до інтерпретації $\boldsymbol{J} = (A, I)$, якщо для кожного $\Phi \in Ps$ маємо $T(\Phi_{\delta(J)}) = \overline{F(\Phi_J)}$ та $F(\Phi_{\delta(J)}) = \overline{T(\Phi_J)}$. Тоді $\boldsymbol{J}$ дуальна до $\delta(\boldsymbol{J})$: $T(\Phi_J) = \overline{F(\Phi_{\delta(J)})}$ та $F(\Phi_J) = \overline{T(\Phi_{\delta(J)})}$.

Виділення дуальних пар алгебр квазіарних предикатів індукує виділення дуальних пар класів інтерпретацій. Зокрема:

$\delta(\mathcal{P}) = \mathcal{T}$, $\delta(\mathcal{T}) = \mathcal{P}$, $\delta(\mathcal{R}) = \mathcal{R}$, $\delta(\mathcal{TS}) = \mathcal{TS}$.

Отже, P-семантика і T-семантика дуальні, а R-семантика і TS-семантика автодуальні.

Нехай інтерпретації $\boldsymbol{J}$ та $\boldsymbol{\vartheta}$ дуальні. Тоді (див. [2]) для всіх $\Phi \in Fr$ маємо:

$$T(\Phi_J) = \overline{F(\Phi_\vartheta)} \text{ та } F(\Phi_\vartheta) = \overline{T(\Phi_J)}.$$

Для кожної $\boldsymbol{J} \in \mathcal{P}$ можна побудувати систему тотальних розширень $\boldsymbol{M} \in \mathcal{TS}$. Це означає: для кожного $p \in Ps$ маємо $T(p_J) \subseteq T(p_M) = {}^{V}A$ та $F(p_J) \subseteq F(p_M) = {}^{V}A$. Згідно теореми про розширення [1] тоді $T(\Phi_J) \subseteq T(\Phi_M) = {}^{V}A$ та $F(\Phi_J) \subseteq F(\Phi_M) = {}^{V}A$ для кожної $\Phi \in Fr$.

Нехай $\mathcal{J}$ – клас інтерпретацій.

Формула Φ неспростовна при інтерпретації $\boldsymbol{J}$, або $\boldsymbol{J}$-неспростовна (позн. $\boldsymbol{J} \models \Phi$), якщо предикат Φ_J – неспростовний.

Формула Φ неспростовна в $\mathcal{J}$ (позн. ${}^{\mathcal{J}}\models \Phi$), якщо $\boldsymbol{J} \models \Phi$ для кожної $\boldsymbol{J} \in \mathcal{J}$.

Формула Φ виконувана при інтерпретації $\boldsymbol{J}$, або $\boldsymbol{J}$-виконувана, якщо Φ_J – виконуваний предикат.

Формула Φ виконувана в $\mathcal{J}$, якщо Φ $\boldsymbol{J}$-виконувана при деякій $\boldsymbol{J} \in \mathcal{J}$.

Кожна формула буде виконуваною в $\mathcal{T}$ та в $\mathcal{R}$ (беремо інтерпретацію, задану сингулярною алгеброю $\top_{V\text{-}A}$).

Отже, поняття виконуваної формули змістовне лише для P-інтерпретацій.

Формула Φ тотально істинна при інтерпретації $\boldsymbol{J}$ (позн. $\boldsymbol{J} \models\!\equiv \Phi$), якщо Φ_J – тотально істинний предикат.

Формула Φ тотально істинна в $\mathcal{J}$ (позн. ${}^{\mathcal{J}}\models\!\equiv \Phi$), якщо $\boldsymbol{J} \models\!\equiv \Phi$ для кожної $\boldsymbol{J} \in \mathcal{J}$.

Формула Φ тотожно істинна при інтерпретації $\boldsymbol{J}$, якщо $\Phi_J = \mathrm{T}$.

Формула Φ тотожно істинна в $\mathcal{J}$, якщо Φ тотожно істинна при кожній $\boldsymbol{J} \in \mathcal{J}$.

Подібним чином даємо визначення:

– тотально хибної при інтерпретації $\boldsymbol{J}$ та тотально хибної в $\mathcal{J}$ формули;

– тотожно хибної при інтерпретації $\boldsymbol{J}$ та тотожно хибної в $\mathcal{J}$ формули.

Теорема 1. 1) для R-семантики класи неспростовних, тотально істинних і тотожно істинних формул порожні;

2) для P-семантики класи тотально істинних і тотожно істинних формул порожні;

3) для T-семантики класи неспростовних і тотожно істинних формул порожні;

4) для TS-семантики класи неспростовних, тотально істинних і тотожно істинних формул збігаються.

Відповідні твердження теореми 1 можна сформулювати для класів тотально хибних та тотожно хибних формул.

Твердження 1. $^{P}|=\Phi \Leftrightarrow {}^{T}|\equiv\Phi$.

Φ_J буде неспростовним при $\boldsymbol{J}\in\mathcal{P} \Leftrightarrow$ $\Leftrightarrow \Phi_\vartheta$ тотально істинний при дуальній $\vartheta\in\mathcal{T}$.

Твердження 2. $^{P}|=\Phi \Leftrightarrow \neg\Phi$ невиконувана в $\mathcal{P}$.

Справді, $^{P}|=\Phi \Leftrightarrow F(\Phi_J)=\varnothing$ для кожної $\boldsymbol{J}\in\mathcal{P} \Leftrightarrow T(\neg\Phi_J)=\varnothing$ для кожної $\boldsymbol{J}\in\mathcal{P} \Leftrightarrow \neg\Phi$ невиконувана.

Теорема 2. $^{P}|=\Phi \Leftrightarrow \Phi$ тотожно істинна в $\mathcal{TS}$.

Доводимо $\Rightarrow$. Якщо Φ не є тотожно істинною в $\mathcal{TS}$, то $F(\Phi_J)\neq\varnothing$ для деякої $\boldsymbol{J}\in\mathcal{TS}$, тому $^{P}|\neq\Phi$ в силу $\mathcal{TS}\subset\mathcal{P}$.

Доводимо $\Leftarrow$. Якщо $^{P}|\neq\Phi$, то маємо $F(\Phi_J)\neq\varnothing$ для деякої $\boldsymbol{J}\in\mathcal{P}$. Візьмемо для $\boldsymbol{J}$ систему тотальних розширень $\boldsymbol{M}\in\mathcal{TS}$. Тоді $F(\Phi_J)\subseteq F(\Phi_M)$, звідки $F(\Phi_M)\neq\varnothing$, тому Φ не є тотожно істинною в $\mathcal{TS}$.

Наслідок 1. Φ тотожно істинна в $\mathcal{TS} \Leftrightarrow {}^{P}|=\Phi \Leftrightarrow {}^{T}|\equiv\Phi$.

Поняття тавтології для ЧКНЛ вводимо традиційним чином (див. [1]).

Формула пропозиційно нерозкладна, якщо вона атомарна або має вигляд $\exists x\Phi$ чи $R\frac{\bar{v}}{\bar{x}}\Phi$. Нехай Fr_0 – множина пропозиційно нерозкладних формул. Істиннісна оцінка мови – це тотальне відображення $\tau: Fr_0 \to \{T, F\}$. Таке τ продовжуємо [1] до відображення $\tau: Fr \to \{T, F\}$ згідно дії композицій $\neg$ та $\vee$ на предикати.

Формула Φ *тавтологія*, якщо $\tau(\Phi) = T$ для кожної істиннісної оцінки τ.

Теорема 3. Φ тавтологія $\Rightarrow$ Φ тотожно істинна в $\mathcal{TS}$.

Нехай Φ утворена із пропозиційно нерозкладних формул $\varphi_1, \ldots, \varphi_n$. Припустимо, що Φ не тотожно істинна в $\mathcal{TS}$, тоді $\Phi(d) = F$ для деяких $\boldsymbol{J} = (A, I) \in \mathcal{TS}$ та $d \in {}^{V}A$. Візьмемо істиннісну оцінку τ: $\tau(\varphi_i) = \varphi_{iJ}(d)$. Тоді $\tau(\Phi) = F$, тому Φ не тавтологія.

Наслідок 2. Пропозиційна формула Φ є тавтологія $\Leftrightarrow$ Φ тотожно істинна в $\mathcal{TS} \Leftrightarrow$ Φ неспростовна в $\mathcal{P} \Leftrightarrow$ Φ тотально істинна в $\mathcal{T}$.

3. Відношення логічного наслідку

На основі різних співвідношень між областями істинності та хибності предикатів можна ввести низку відношень на множині формул мови ЧКНЛ.

Спочатку вводимо (див. [2]) відношення наслідку для двох формул при інтерпретації на фіксованій інтерпретації $\boldsymbol{J}$.

1. Істиннісний, або T-наслідок $_J|=_T$:

$$\Phi\ {}_J|=_T \Psi \Leftrightarrow T(\Phi_J)\subseteq T(\Psi_J).$$

2. Хибнісний, або F-наслідок $_J|=_F$:

$$\Phi\ {}_J|=_F \Psi \Leftrightarrow F(\Psi_J)\subseteq F(\Phi_J).$$

3. Сильний, або TF-наслідок $_J|=_{TF}$:

$$\Phi\ {}_J|=_{TF} \Psi \Leftrightarrow T(\Phi_J)\subseteq T(\Psi_J) \text{ та } F(\Psi_J)\subseteq F(\Phi_J).$$

4. Неспростовнісний, або IR-наслідок $_J|=_{IR}$: $\Phi\ {}_J|=_{IR} \Psi \Leftrightarrow T(\Phi_J)\cap F(\Psi_J)=\varnothing$.

5. Дуальний до IR, або DI-наслідок $_J|=_{DI}$: $\Phi\ {}_J|=_{DI} \Psi \Leftrightarrow F(\Phi_J)\cup T(\Psi_J)={}^{V}A$.

До цих "природних" відношень наслідку, досліджених в [2–4], додамо ще два.

6. C-наслідок $_J|=_C$:

$$\Phi\ {}_J|=_C \Psi \Leftrightarrow T(\Phi_J)\cap F(\Psi_J)\subseteq F(\Phi_J)\cup T(\Psi_J).$$

7. $T\vee F$-наслідок $_J|=_{T\vee F}$:

$$\Phi\ {}_J|=_{T\vee F} \Psi \Leftrightarrow T(\Phi_J)\subseteq T(\Psi_J) \text{ або } F(\Psi_J)\subseteq F(\Phi_J).$$

C-наслідок для пропозиційної логіки розглянуто в [5]. Дослідити $T\vee F$-наслідок запропонував М.С. Нікітченко.

Відповідні відношення логічного наслідку в семантиці α визначаємо за схемою:

$\Phi\ {}^{\alpha}|{=}_{*}\Psi$, якщо $\Phi\ {}_{J}|{=}_{*}\Psi$ для кожної $\boldsymbol{J}\in\alpha$.

Твердження 3. Усі визначені вище відношення рефлексивні.

Твердження 4. Маємо:

${}_{J}|{=}_{IR}\subseteq{}_{J}|{=}_{C}$, ${}_{J}|{=}_{DI}\subseteq{}_{J}|{=}_{C}$;

${}_{J}|{=}_{TF}\subseteq{}_{J}|{=}_{T}\subseteq{}_{J}|{=}_{T\vee F}$, ${}_{J}|{=}_{TF}\subseteq{}_{J}|{=}_{F}\subseteq{}_{J}|{=}_{T\vee F}$.

У випадках класичної логіки та логіки *TS*-предикатів маємо $T(\Phi_J)=\overline{F(\Phi_J)}$ та $F(\Phi_J)=\overline{T(\Phi_J)}$. Для цих логік усі наведені відношення логічного наслідку втрачають відмінності, вони збігаються і фактично стають єдиним логічним наслідком. Для логіки *TS*-предикатів таке відношення логічного наслідку позначимо ${}^{TS}|{=}$. Отже:

Теорема 4. ${}^{TS}|{=}_{TF}={}^{TS}|{=}_{T}={}^{TS}|{=}_{F}=$ $={}^{TS}|{=}_{IR}={}^{TS}|{=}_{DI}={}^{TS}|{=}_{C}={}^{TS}|{=}_{T\vee F}={}^{TS}|{=}$.

Твердження 5. ${}^{P}|{=}_{DI}={}^{T}|{=}_{IR}=\varnothing$.

Візьмемо *P*-інтерпретацію, задану алгеброю $\bot_{V\text{-}A}$, на ній усі формули інтерпретуються як $\bot$. Звідси маємо ${}^{P}|{=}_{DI}=\varnothing$.

Візьмемо *T*-інтерпретацію, задану алгеброю $\top_{V\text{-}A}$, на ній усі формули інтерпретуються як $\top$. Звідси маємо ${}^{T}|{=}_{IR}=\varnothing$.

Наслідок 3. ${}^{R}|{=}_{IR}={}^{R}|{=}_{DI}=\varnothing$.

У випадку $\boldsymbol{J}\in\mathcal{P}$ маємо

${}_{J}|{=}_{T}\subseteq{}_{J}|{=}_{IR}$ та ${}_{J}|{=}_{F}\subseteq{}_{J}|{=}_{IR}$.

Справді, для $\Phi,\Psi\in Fr$ із умов $T(\Phi_J)\subseteq T(\Psi_J)$ та $T(\Psi_J)\cap F(\Psi_J)=\varnothing$ маємо $T(\Phi_J)\cap F(\Psi_J)=\varnothing$; із умов $F(\Psi_J)\subseteq F(\Phi_J)$ та $T(\Phi_J)\cap F(\Phi_J)=\varnothing$ маємо $T(\Phi_J)\cap F(\Psi_J)=\varnothing$.

У випадку $\boldsymbol{J}\in\mathcal{T}$ маємо

${}_{J}|{=}_{T}\subseteq{}_{J}|{=}_{DI}$ та ${}_{J}|{=}_{F}\subseteq{}_{J}|{=}_{DI}$.

Справді, для $\Phi,\Psi\in Fr$ із умов $F(\Phi_J)\cup T(\Phi_J)={}^{V}A$ та $T(\Phi_J)\subseteq T(\Psi_J)$ маємо $F(\Phi_J)\cup T(\Psi_J)={}^{V}A$; із $F(\Psi_J)\cup T(\Psi_J)={}^{V}A$ та $F(\Psi_J)\subseteq F(\Phi_J)$ маємо $F(\Phi_J)\cup T(\Psi_J)={}^{V}A$.

Таким чином, маємо.

Теорема 5. ${}^{P}|{=}_{T}\subseteq{}^{P}|{=}_{IR}$, ${}^{P}|{=}_{F}\subseteq{}^{P}|{=}_{IR}$; ${}^{T}|{=}_{T}\subseteq{}^{T}|{=}_{DI}$, ${}^{T}|{=}_{F}\subseteq{}^{T}|{=}_{DI}$.

Наслідок 4. ${}^{P}|{=}_{*}\subseteq{}^{TS}|{=}$ та ${}^{T}|{=}_{*}\subseteq{}^{TS}|{=}$.

Справді, $\mathcal{TS}\subset\mathcal{P}$ та $\mathcal{TS}\subset\mathcal{T}$.

Теорема 6. Маємо ${}^{P}|{=}_{IR}={}^{TS}|{=}$.

Згідно наслідку 4 маємо ${}^{P}|{=}_{IR}\subseteq{}^{TS}|{=}$. Покажемо ${}^{TS}|{=}\subseteq{}^{P}|{=}_{IR}$.

Нехай супротивне: ${}^{TS}|{=}\not\subseteq{}^{P}|{=}_{IR}$. Тоді для деяких $\Phi,\Psi\in Fr$ та $\boldsymbol{J}\in\mathcal{P}$ маємо $\Phi\ {}^{TS}|{=}\Psi$ та $\Phi\ {}_{J}|{\neq}_{IR}\Psi$. Останнє означає, що $T(\Phi_J)\cap F(\Psi_J)\neq\varnothing$. Візьмемо для $\boldsymbol{J}$ систему тотальних розширень $\boldsymbol{M}\in\mathcal{TS}$. Тоді $T(\Phi_J)\subseteq T(\Phi_M)$ та $F(\Psi_J)\subseteq F(\Psi_M)$, звідки $T(\Phi_M)\cap F(\Psi_M)\neq\varnothing$, тому $\Phi\ {}_{M}|{\neq}_{IR}\Psi$. Звідси випливає $\Phi\ {}^{TS}|{\neq}\Psi$ – суперечність.

Теорема 7. Нехай інтерпретації $\boldsymbol{A}$ та $\boldsymbol{B}$ дуальні. Тоді маємо:

1) $\Phi\ {}_{A}|{=}_{T}\Psi\Leftrightarrow\Phi\ {}_{B}|{=}_{F}\Psi$ та

$\Phi\ {}_{A}|{=}_{F}\Psi\Leftrightarrow\Phi\ {}_{B}|{=}_{T}\Psi$;

2) $\Phi\ {}_{A}|{=}_{TF}\Psi\Leftrightarrow\Phi\ {}_{B}|{=}_{TF}\Psi$;

3) $\Phi\ {}_{A}|{=}_{T\vee F}\Psi\Leftrightarrow\Phi\ {}_{B}|{=}_{T\vee F}\Psi$;

4) $\Phi\ {}_{A}|{=}_{IR}\Psi\Leftrightarrow\Phi\ {}_{B}|{=}_{DI}\Psi$ та

$\Phi\ {}_{A}|{=}_{DI}\Psi\Leftrightarrow\Phi\ {}_{B}|{=}_{IR}\Psi$;

5) $\Phi\ {}_{A}|{=}_{C}\Psi\Leftrightarrow\Phi\ {}_{B}|{=}_{C}\Psi$.

Зауважимо, що пп. 1, 2, 4 теореми доведено в [2]. Тому доведемо пп. 3, 5.

Доводимо п. 3. $\Phi\ {}_{A}|{=}_{T\vee F}\Psi\ \Leftrightarrow$ $\Phi\ {}_{A}|{=}_{T}\Psi$ або $\Phi\ {}_{A}|{=}_{F}\Psi\ \Leftrightarrow$ (за п. 1) $\Phi\ {}_{B}|{=}_{F}\Psi$ або $\Phi\ {}_{B}|{=}_{T}\Psi\Leftrightarrow\Phi\ {}_{B}|{=}_{T\vee F}\Psi$.

Доводимо п. 5. Маємо $\Phi\ {}_{A}|{=}_{C}\Psi\ \Leftrightarrow$ $\Leftrightarrow T(\Phi_A)\cap F(\Psi_A)\subseteq F(\Phi_A)\cup T(\Psi_A)\Leftrightarrow$

$\Leftrightarrow\overline{F(\Phi_B)}\cap\overline{T(\Psi_B)}\subseteq\overline{T(\Phi_B)}\cup\overline{F(\Psi_B)}\Leftrightarrow$

$\Leftrightarrow\overline{F(\Phi_B)\cup T(\Psi_B)}\subseteq\overline{T(\Phi_B)\cap F(\Psi_B)}\ \Leftrightarrow$

$T(\Phi_B)\cap F(\Psi_B)\subseteq F(\Phi_B)\cup T(\Psi_B)\Leftrightarrow\Phi\ {}_{B}|{=}_{C}\Psi$.

Теорема 8. Маємо (див. також [2]):

$$\Phi\ {}^{R}|{=}_{T}\Psi\Leftrightarrow\Phi\ {}^{R}|{=}_{F}\Psi\Leftrightarrow\Phi\ {}^{R}|{=}_{TF}\Psi.$$

Покажемо: $\Phi\ |{=}_{T}\Psi\Rightarrow\Phi\ |{=}_{TF}\Psi$.

Нехай супротивне: $\Phi\ |{=}_{T}\Psi$, проте $\Phi\ |{\neq}_{TF}\Psi$. Тоді $T(\Phi_J)\subseteq T(\Psi_J)$ для кожної і $\boldsymbol{J}$, проте $\Phi\ {}_{A}|{\neq}_{TF}\Psi$ для деякої $\boldsymbol{A}$. Тоді неправильно $F(\Psi_A)\subseteq F(\Phi_A)$, тому для дуальної $\boldsymbol{B}$

неправильно $T(\Phi_B) \subseteq T(\Psi_B)$, а це суперечить $\Phi \models_T \Psi$.

Аналогічно $\Phi \models_F \Psi \Rightarrow \Phi \models_{TF} \Psi$.

Із теорем 4, 6–8 маємо:

Наслідок 5. ${}^{R}\!\models_T = {}^{R}\!\models_F = {}^{R}\!\models_{TF}$; ${}^{P}\!\models_{IR} = {}^{T}\!\models_{DI} = {}^{P}\!\models_C = {}^{T}\!\models_C = {}^{TS}\!\models$; ${}^{P}\!\models_T = {}^{T}\!\models_F$; ${}^{P}\!\models_F = {}^{T}\!\models_T$; ${}^{P}\!\models_{TF} = {}^{T}\!\models_{TF}$; ${}^{P}\!\models_{T\vee F} = {}^{T}\!\models_{T\vee F}$.

Отже, із перелічених вище відношень різними можуть бути лише такі:

${}^{P}\!\models_{IR}$, ${}^{P}\!\models_T$, ${}^{P}\!\models_F$, ${}^{P}\!\models_{TF}$, ${}^{P}\!\models_{T\vee F}$, ${}^{R}\!\models_C$, ${}^{R}\!\models_{TF}$.

Твердження 6. Відношення ${}_J\!\models_T$, ${}_J\!\models_F$, ${}_J\!\models_{TF}$ та відношення ${}^{P}\!\models_T$, ${}^{P}\!\models_F$, ${}^{P}\!\models_{TF}$, ${}^{R}\!\models_{TF}$ є транзитивними.

Складніше із транзитивністю ${}_J\!\models_{IR}$, ${}_J\!\models_{DI}$, ${}_J\!\models_C$, ${}_J\!\models_{T\vee F}$ та ${}^{P}\!\models_{IR}$, ${}^{P}\!\models_{T\vee F}$, ${}^{R}\!\models_C$.

Приклад 1. Нехай $\boldsymbol{A} \in \mathcal{P}$, $p, q, s \in Ps$. Задамо предикат p_A як T, q_A – як $\perp$, s_A – як F. Тоді $p_A \models_{IR} q$ та $q_A \models_{IR} s$, проте $p_A \not\models_{IR} s$.

Отже, відношення ${}_A\!\models_{IR}$ нетранзитивне. Звідси для дуальної інтерпретації $\boldsymbol{B}$ маємо нетранзитивність ${}_B\!\models_{DI}$. Проте для відношення ${}^{P}\!\models_{IR}$ ситуація нормалізується.

Теорема 9. ${}^{P}\!\models_{IR}$ транзитивне.

Нехай маємо супротивне: $\Phi \, {}^{P}\!\models_{IR} \Psi$, $\Psi \, {}^{P}\!\models_{IR} \Xi$, проте $\Phi \, {}^{P}\!\not\models_{IR} \Xi$. Тоді для деякої $\boldsymbol{A} \in \mathcal{P}$, маємо $T(\Phi_A) \cap F(\Xi_A) \neq \varnothing$. Візьмемо для $\boldsymbol{A}$ деяку інтерпретацію тотальних розширень $\boldsymbol{M}$, тоді $T(\Phi_A) \subseteq T(\Phi_M)$ та $F(\Xi_A) \subseteq F(\Xi_M)$, звідки $T(\Phi_M) \cap F(\Xi_M) \neq \varnothing$. Предикати Φ_M, Ψ_M, Ξ_M тотальні, звідки отримуємо

$T(\Phi_M) \cup F(\Phi_M) = T(\Psi_M) \cup F(\Psi_M) =$
$= T(\Xi_M) \cup F(\Xi_M) = {}^{V}\!A$, тоді умови $\Phi \, {}^{P}\!\models_{IR} \Psi$ та $\Psi \, {}^{P}\!\models_{IR} \Xi$ дають $T(\Phi_M) \subseteq T(\Psi_M)$ та $T(\Psi_M) \subseteq T(\Xi_M)$, звідки $T(\Phi_M) \subseteq T(\Xi_M)$, що суперечить $T(\Phi_M) \cap F(\Xi_M) \neq \varnothing$.

Приклад 2. Нехай $p, q \in Ps$, Φ – це формула $p \vee (q \& \neg q)$, Ψ – це $p \,\&\, (q \vee \neg q)$.

Для кожної $\boldsymbol{A} \in \mathcal{P}$ маємо:

$$T(\Phi_A) = T(p_A) \cup (T(q_A) \cap F(q_A)) = T(p_A),$$
$$F(\Phi_A) = F(p_A) \cap (T(q_A) \cup F(q_A)) \subseteq F(p_A),$$
$$T(\Psi_A) = T(p_A) \cap (T(q_A) \cup F(q_A)) \subseteq T(p_A),$$
$$F(\Psi_A) = F(p_A) \cup (T(q_A) \cap F(q_A)) = F(p_A).$$

Отже, $\Phi_A \models_T p$ та $p_A \models_F \Psi$ для кожного $\boldsymbol{A} \in \mathcal{P}$, тому $\Phi \, {}^{P}\!\models_{T\vee F} p$ та $p \, {}^{P}\!\models_{T\vee F} \Psi$.

Водночас маємо $p \, {}^{P}\!\models_{TF} \Phi$ та $\Psi \, {}^{P}\!\models_{TF} p$, тому $p \, {}^{P}\!\models_{T\vee F} \Phi$ та $\Psi \, {}^{P}\!\models_{T\vee F} p$.

Задамо інтерпретацію $\boldsymbol{B} \in \mathcal{P}$ таку:

$$T(\Psi_B) = T(p_B) \cap (T(q_B) \cup F(q_B)) \subset T(p_B),$$
$$F(\Phi_B) = F(p_B) \cap (T(q_B) \cup F(q_B)) \subset F(p_B).$$

Проте $T(\Phi_B) = T(p_B)$, $F(\Psi_B) = F(p_B)$, звідки $\Phi \, {}^{P}\!\models_T \Psi$ та $\Phi \, {}^{P}\!\not\models_F \Psi$, тому $\Phi \, {}^{P}\!\not\models_{T\vee F} \Psi$.

Отже, отримуємо наступну теорему.

Теорема 10. ${}^{P}\!\models_{T\vee F}$ нетранзитивне.

Для довільних формули φ та інтерпретації $\boldsymbol{J}$ далі позначаємо $T(\varphi_J) \cap F(\varphi_J)$ як $\varphi_{J\cap}$, а $T(\varphi_J) \cup F(\varphi_J)$ як $\varphi_{J\cup}$.

Приклад 3. Нехай $p, q \in Ps$, Φ – це формула $p \vee (q \& \neg q)$, Ψ – це формула $p \,\&\, (q \vee \neg q)$. Для довільної $\boldsymbol{A} \in \mathcal{R}$ маємо:

$$T(\Phi_A) = T(p_A) \cup q_{A\cap},\ F(\Phi_A) = F(p_A) \cap q_{A\cup},$$
$$T(\Psi_A) = T(p_A) \cap q_{A\cup},\ F(\Psi_A) = F(p_A) \cup q_{A\cap};$$
$$T(\Phi_A) \cap F(p_A) = (T(p_A) \cup q_{A\cap}) \cap F(p_A) =$$
$$= (T(p_A) \cap F(p_A)) \cup (F(p_A) \cap q_{A\cap}),$$
$$F(\Phi_A) \cup T(p_A) = (F(p_A) \cap q_{A\cup}) \cup T(p_A).$$

Проте $F(p_A) \cap q_{A\cap} \subseteq F(p_A) \cap q_{A\cup}$, $T(p_A) \cap F(p_A) \subseteq T(p_A)$, звідки $T(\Phi_A) \cap F(p_A) \subseteq$ $\subseteq F(\Phi_A) \cup T(p_A)$, тому $\Phi_A \models_C p$.

Тепер маємо $T(p_A) \cap F(\Psi_A) = T(p_A) \cap$ $\cap (F(p_A) \cup q_{A\cap}) = (T(p_A) \cap F(p_A)) \cup (T(p_A) \cap q_{A\cap})$,

$$F(p_A) \cup T(\Psi_A) = F(p_A) \cup (T(p_A) \cap q_{A\cup}).$$

Водночас $T(p_A) \cap q_{A\cap} \subseteq T(p_A) \cap q_{A\cup}$, $T(p_A) \cap F(p_A) \subseteq F(p_A)$, звідки $T(p_A) \cap F(\Psi_A) \subseteq$ $\subseteq F(p_A) \cup T(\Psi_A)$, тому $p_A \models_C \Psi$.

Таким чином, $\Phi \, {}^{R}\!\models_C p$ та $p \, {}^{R}\!\models_C \Psi$.

Проте $p \, {}^{R}\!\models_{TF} \Phi$ та $\Psi \, {}^{R}\!\models_{TF} p$, тому $p \, {}^{R}\!\models_C \Phi$ та $\Psi \, {}^{R}\!\models_C p$.

Задамо інтерпретацію $\boldsymbol{B} \in \mathcal{P}$ таку:

$T(p_B) = F(p_B) \neq \varnothing$, $T(q_B) = F(q_B)$, $T(p_B) \cap T(q_B) = \varnothing$. Тоді:

$$T(\Phi_B) \cap F(\Psi_B) = (T(p_B) \cup q_{B\cap}) \cap (F(p_B) \cup$$
$$\cup q_{B\cap}) = T(p_B) \cup q_{B\cap} \neq \varnothing,$$
$$F(\Phi_B) \cup T(\Psi_B) = (F(p_B) \cap q_{B\cup}) \cup (T(p_B) \cap q_{B\cup})$$
$$= \varnothing \cup \varnothing = \varnothing.$$

Отже, $\Phi\ {}_B|{\neq_C}\ \Psi$, тому $\Phi\ {}^R|{\neq_C}\ \Psi$.

Таким чином, отримуємо.

Теорема 11. ${}^R|{=_C}$ нетранзитивне.

Отже, відношення ${}^P|{=_{T\vee F}}$ та ${}^R|{=_C}$ мають не зовсім прийнятні властивості. Вони не задовольняють постулату Тарського про транзитивність логічного наслідку.

Наведемо визначення тавтологічного наслідку для пари формул (див. [1, 2]).

Ψ є тавтологічним наслідком Φ (позн. $\Phi\ {|{=_t}}\ \Psi$), якщо $\Phi\to\Psi$ – тавтологія.

Відношення ${|{=_t}}$ рефлексивне і транзитивне.

Для пропозиційних формул умова $\Phi\ {}^{TS}|{=}\ \Psi$ означає, що $\Phi\ {|{=_t}}\ \Psi$. Справді, $\Phi\ {}^{TS}|{=}\ \Psi \Leftrightarrow \Phi\to\Psi$ тотожно істинна в $\mathcal{TS} \Leftrightarrow$ (наслідок 2) $\Phi\to\Psi$ є тавтологія $\Leftrightarrow \Phi\ {|{=_t}}\ \Psi$.

Теорема 12. 1) для пропозиційних формул маємо:

$$\Phi\ {}^P|{=_{IR}}\ \Psi \Leftrightarrow \Phi\ {}^T|{=_{DI}}\ \Psi \Leftrightarrow \Phi\ {}^P|{=_C}\ \Psi \Leftrightarrow$$
$$\Leftrightarrow \Phi\ {}^T|{=_C}\ \Psi \Leftrightarrow \Phi\ {}^{TS}|{=}\ \Psi \Leftrightarrow \Phi\ {|{=_t}}\ \Psi;$$

2) у випадку класичної семантики пропозиційної логіки усі описані вище відношення збігаються.

Зауважимо, що в монографії О.Д. Смирнової ([5] стор. 190) неточно сказано, що відношення типу $[f]$ в релевантній семантиці (в наших термінах термінах відношення ${}^R|{=_C}$) формалізується класичною логікою, проте це не так: згідно прикладу 3 маємо $p\vee(q\&\neg q)\ {}^R|{\neq_C}\ p\ \&\ (q\vee\neg q)$, хоча $p\vee(q\&\neg q)\to p\ \&\ (q\vee\neg q)$ – тавтологія. Ще одна неточність: там же сказано, що відношення типу $[a]$, $[b]$, $[c]$ в релевантній семантиці (в наших термінах відношення ${}^R|{=_T}$, ${}^R|{=_F}$, ${}^R|{=_{TF}}$) не є еквівалентними, хоча (наслідок 5) ${}^R|{=_T} = {}^R|{=_F} = {}^R|{=_{TF}}$.

Відношення логічного наслідку ${}^P|{=_{IR}}$, ${}^P|{=_T}$, ${}^P|{=_F}$, ${}^P|{=_{TF}}$, ${}^R|{=_{TF}}$ індукують відповідні відношення логічної еквівалентності ${}^P{\sim_{IR}}$, ${}^P{\sim_T}$, ${}^P{\sim_F}$, ${}^P{\sim_{TF}}$, ${}^R{\sim_{TF}}$.

Визначаємо їх за такою схемою:

$$\Phi\ {}^\alpha{\sim_*}\ \Psi, \text{ якщо } \Phi\ {}^\alpha|{=_*}\ \Psi \text{ та } \Psi\ {|{=_*}}\ \Phi.$$

Подібним чином визначаємо відношення тавтологічної еквівалентності ${\sim_t}$:

$$\Phi\ {\sim_t}\ \Psi, \text{ якщо } \Phi\ {|{=_t}}\ \Psi \text{ та } \Psi\ {|{=_t}}\ \Phi.$$

Твердження 7. Відношення ${}^P{\sim_{IR}}$, ${}^P{\sim_T}$, ${}^P{\sim_F}$, ${}^P{\sim_{TF}}$, ${}^R{\sim_{TF}}$, ${\sim_t}$ рефлексивні, транзитивні та симетричні.

Можна визначити (див. [2]) відношення еквівалентності формул при інтерпретації $\boldsymbol{J}$ за наступною схемою:

$$\Phi\ {}_J{\sim_*}\ \Psi, \text{ якщо } \Phi\ {}_J|{=_*}\ \Psi \text{ та } \Psi\ {}_J|{=_*}\ \Phi.$$

Твердження 8. $\Phi\ {}^\alpha{\sim_*}\ \Psi \Leftrightarrow \Phi\ {}_J{\sim_*}\ \Psi$ для кожної $\boldsymbol{J}\in\boldsymbol{\alpha}$.

Для відношення ${}_J{\sim_{TF}}$ отримуємо:

$$\Phi\ {}_J{\sim_{TF}}\ \Psi \Leftrightarrow T(\Phi_J) = T(\Psi_J) \text{ та } F(\Phi_J) = F(\Psi_J).$$

Отже, $\Phi\ {}_J{\sim_{TF}}\ \Psi$ означає, що $\Phi_J = \Psi_J$.

Відношення ${\sim_T}$, ${}_J{\sim_F}$, ${}_J{\sim_{TF}}$, ${}_J{\sim_{TF}}$ транзитивні, проте ${}_J{\sim_{IR}}$ нетранзитивне:

Справді, нехай $\boldsymbol{A}\in\mathcal{P}$, $p, q, s\in Ps$. Задамо p_A як T, q_A – як $\perp$, s_A – як F. Тоді $p\ {}_A{\sim_{IR}}\ q$ та $q\ {}_A{\sim}\ s$, проте неправильно $p\ {}_A{\sim_{IR}}\ s$.

Твердження 9. Логічна зв'язка $\leftrightarrow$ узгоджується з відношенням ${}^P{\sim_{IR}}$:

$$\Phi\ {}^P{\sim_{IR}}\ \Psi \Leftrightarrow {}^P|{=}\ \Phi\leftrightarrow\Psi.$$

Твердження 10. 1) для пропозиційних формул умова $\Phi\ {}^\alpha{\sim_*}\ \Psi$, де ${}^\alpha{\sim_*}$ – одне з визначених вище відношень логічної еквівалентності, означає, що тоді $\Phi\ {\sim_t}\ \Psi$;

2) для класичної семантики пропозиційної логіки усі описані вище відношення логічної еквівалентності збігаються.

Введемо відношення ${}^P{\sim_{T\vee F}}$ та ${}^R{\sim_C}$.

Відношення ${}^P{\sim_{T\vee F}}$ визначаємо так:

$$\Phi\ {}^P{\sim_{T\vee F}}\ \Psi \Leftrightarrow \Phi\ {}^P|{=_{T\vee F}}\ \Psi \text{ та } \Psi\ {}^P|{=_{T\vee F}}\ \Phi.$$

Задамо відношення $\Phi\ {}_J{\sim_{T\vee F}}\ \Psi$ так:

$$\Phi\ {}_J{\sim_{T\vee F}}\ \Psi \Leftrightarrow \Phi\ {}_J|{=_{T\vee F}}\ \Psi \text{ та } \Psi\ {}_J|{=_{T\vee F}}\ \Phi.$$

Тоді $\Phi\ {}^P{\sim_{T\vee F}}\ \Psi \Leftrightarrow \Phi\ {}_J{\sim_{T\vee F}}\ \Psi$ для кожної $\boldsymbol{J}\in\mathcal{P}$.

Твердження 11. Відношення ${}_J{\sim_{T\vee F}}$ та ${}^P{\sim_{T\vee F}}$ нетранзитивні.

Для формул прикладу 2 для кожної $\boldsymbol{A}\in\mathcal{P}$ маємо $\Phi\ {}_A{\sim_{T\vee F}}\ p$ та $p\ {}_A{\sim_{T\vee F}}\ \Psi$, тому маємо $\Phi\ {}_A{\sim_{T\vee F}}\ p$ та $p\ {}_A{\sim_{T\vee F}}\ \Psi$. Проте для інтерпретації $\boldsymbol{B}$ прикладу 2 маємо $\Phi\ {}_B|{\neq_{T\vee F}}\ \Psi$ та $\Phi\ {}_B|{\neq_{T\vee F}}\ \Psi$, тому неправильно $\Phi\ {}_B{\sim_{T\vee F}}\ \Psi$ та

неправильно $\Phi\ {}^{P}\!\sim_{T\vee F}\Psi$.

Отже, ${}_{J}\!\sim_{T\vee F}$ та ${}^{P}\!\sim_{T\vee F}$ не є відношеннями еквівалентності.

Відношення ${}^{R}\!\sim_{C}$ визначимо так:

$$\Phi\ {}^{R}\!\sim_{C}\Psi \Leftrightarrow \Phi\ {}^{R}\!|\!=_{C}\Psi \text{ та } \Psi\ {}^{R}\!|\!=_{C}\Phi.$$

Задамо відношення $\Phi\ {}_{J}\!\sim_{C}\Psi$ так:

$$\Phi\ {}_{J}\!\sim_{C}\Psi \Leftrightarrow \Phi\ {}_{J}\!|\!=_{C}\Psi \text{ та } \Psi\ {}_{J}\!|\!=_{C}\Phi.$$

Тоді $\Phi\ {}^{R}\!\sim_{C}\Psi \Leftrightarrow \Phi\ {}_{J}\!\sim_{C}\Psi$ для кожної $\boldsymbol{J}\in\mathcal{R}$.

Твердження 12. Відношення ${}_{J}\!\sim_{C}$ та ${}^{R}\!\sim_{C}$ нетранзитивні.

Для формул прикладу 3 для кожної $\boldsymbol{A}\in\mathcal{R}$ маємо $\Phi\ {}_{A}\!\sim_{C} p$ та $p\ {}_{A}\!\sim_{C}\Psi$, тому маємо $\Phi\ {}_{A}\!\sim_{C} p$ та $p\ {}_{A}\!\sim_{C}\Psi$. Проте для інтерпретації $\boldsymbol{B}$ прикладу 3 маємо $\Phi\ {}_{B}\!|\!\neq_{C}\Psi$ та $\Phi\ {}_{B}\!|\!\neq_{C}\Psi$, тому неправильно $\Phi\ {}_{B}\!\sim_{C}\Psi$ та неправильно $\Phi\ {}^{R}\!\sim_{C}\Psi$.

Таким чином, ${}_{J}\!\sim_{C}$ та ${}^{R}\!\sim_{C}$ теж не є відношеннями еквівалентності.

4. Відношення логічного наслідку для множин формул

Нехай $\Sigma\subseteq Fr$, $\boldsymbol{J}$ – інтерпретація. Скорочено позначимо $\bigcap_{\Phi\in\Sigma} T(\Phi_J)$ як $T^{\wedge}(\Sigma_J)$, $\bigcap_{\Phi\in\Sigma} F(\Phi_J)$ як $F^{\wedge}(\Sigma_J)$, $\bigcup_{\Phi\in\Sigma} T(\Phi_J)$ як $T^{\vee}(\Sigma_J)$, $\bigcup_{\Phi\in\Sigma} F(\Phi_J)$ як $F^{\vee}(\Sigma_J)$.

Δ є *IR*-наслідком Γ при $\boldsymbol{J}$ (позн. $\Gamma\ {}_{J}\!|\!=_{IR}\Delta$), якщо $T^{\wedge}(\Gamma_J)\cap F^{\wedge}(\Delta_J)=\varnothing$.

Δ є *DI*-наслідком Γ при $\boldsymbol{J}$ (позн. $\Gamma\ {}_{J}\!|\!=_{DI}\Delta$), якщо $F^{\vee}(\Gamma_J)\cup T^{\vee}(\Delta_J)={}^{V}\!A$.

Δ є *T*-наслідком Γ при $\boldsymbol{J}$ (позн. $\Gamma\ {}_{J}\!|\!=_{T}\Delta$), якщо $T^{\wedge}(\Gamma_J)\subseteq T^{\vee}(\Delta_J)$.

Δ є *F*-наслідком Γ при $\boldsymbol{J}$ (позн. $\Gamma\ {}_{J}\!|\!=_{F}\Delta$), якщо $F^{\wedge}(\Delta_J)\subseteq F^{\vee}(\Gamma_J)$.

Δ є *TF*-наслідком Γ при $\boldsymbol{J}$ (позн. $\Gamma\ {}_{J}\!|\!=_{TF}\Delta$), якщо $\Gamma\ {}_{J}\!|\!=_{T}\Delta$ та $\Gamma\ {}_{J}\!|\!=_{F}\Delta$.

Відповідні відношення логічного наслідку в семантиці α визначаємо за схемою:

$\Gamma\ {}^{\alpha}\!|\!=_{*}\Delta$, якщо $\Gamma\ {}_{J}\!|\!=_{*}\Delta$ для кожної $\boldsymbol{J}\in\alpha$.

Δ є тавтологічним наслідком Γ (позн. $\Gamma\ |\!=_{t}\Delta$), якщо для кожної істиннісної оцінки $\tau: Fp\to\{T,F\}$ із умови $\tau(\Phi)=T$ для всіх $\Phi\in\Gamma$ маємо: $\tau(\Psi)=T$ для деякої $\Psi\in\Delta$.

Відношення наслідку та логічного наслідку для множин формул рефлексивні та нетранзитивні (див. [1, 2]).

Розглянуті в попередньому розділі відношення наслідку та логічного наслідку для формул є окремими випадками відповідних відношень для множин формул, їх позначення та властивості (окрім транзитивності) переносяться на випадок відповідних відношень для множин формул.

Відношення наслідку та логічного наслідку для логік квазіарних предикатів (окрім введених тут відношень типу $T\vee F$ та C) вивчались в [1–4]. Розглянемо основні властивості цих відношень.

Теорема 13. Нехай інтерпретації $\boldsymbol{J}$ та $\mathcal{J}$ дуальні. Тоді:

1) $\Gamma\ {}_{J}\!|\!=_{T}\Delta \Leftrightarrow \Gamma\ {}_{\mathcal{J}}\!|\!=_{F}\Delta$ та

$\Gamma\ {}_{J}\!|\!=_{F}\Delta \Leftrightarrow \Gamma\ {}_{\mathcal{J}}\!|\!=_{T}\Delta$;

2) $\Gamma\ {}_{J}\!|\!=_{TF}\Delta \Leftrightarrow \Gamma\ {}_{\mathcal{J}}\!|\!=_{TF}\Delta$;

3) $\Gamma\ {}_{J}\!|\!=_{IR}\Delta \Leftrightarrow \Gamma\ {}_{\mathcal{J}}\!|\!=_{DI}\Delta$ та

$\Gamma\ {}_{J}\!|\!=_{DI}\Delta \Leftrightarrow \Gamma\ {}_{\mathcal{J}}\!|\!=_{IR}\Delta$;

4) $\Gamma\ {}_{J}\!|\!=_{T\vee F}\Delta\Psi \Leftrightarrow \Gamma\ {}_{\mathcal{J}}\!|\!=_{T\vee F}\Delta$;

5) $\Gamma\ {}_{J}\!|\!=_{C}\Delta \Leftrightarrow \Gamma\ {}_{\mathcal{J}}\!|\!=_{C}\Delta$.

Серед розглянутих відношень логічного наслідку для множин формул різними можуть бути лише такі:

${}^{P}\!|\!=_{IR}$, ${}^{P}\!|\!=_{T}$, ${}^{P}\!|\!=_{F}$, ${}^{P}\!|\!=_{TF}$, ${}^{P}\!|\!=_{T\vee F}$, ${}^{R}\!|\!=_{C}$, ${}^{R}\!|\!=_{TF}$, $|\!=_{t}$.

Розглянемо наслідки, коли одна з множин формул порожня.

$\varnothing\ {}_{J}\!|\!=_{*}\Delta$ означає $\mathrm{T}\ {}_{J}\!|\!=_{*}\Delta$,

$\Gamma\ {}_{J}\!|\!=_{*}\varnothing$ означає $\Gamma\ {}_{J}\!|\!=_{*}\mathrm{F}$.

Звідси маємо наступне.

$\varnothing\ |\!=_{t}\Phi$ означає, що Φ – тавтологія;

$\Phi\ |\!=_{t}\varnothing$ означає: Φ – суперечність;

$\varnothing\ {}_{J}\!|\!=_{IR}\Phi$ та $\varnothing\ {}_{J}\!|\!=_{F}\Phi$ означають $F(\Phi_J)=\varnothing$, тобто $\boldsymbol{J}\!|\!=\Phi$;

$\varnothing\ {}_{J}\!|\!=_{DI}\Phi$ та $\varnothing\ {}_{J}\!|\!=_{T}\Phi$ означають $T(\Phi_J)={}^{V}\!A$, тобто $\boldsymbol{J}\!|\!\equiv\Phi$;

$\varnothing_J|=_{T\vee F}\Phi$ означає: $T(\Phi_J) = {}^V A$ або $F(\Phi_J) = \varnothing$, тобто $\boldsymbol{J}|\equiv\Phi$ або $\boldsymbol{J}|=\Phi$;

$\varnothing_J|=_C\Phi$ означає $F(\Phi_J)\subseteq T(\Phi_J)$;

$\varnothing_J|=_{TF}\Phi$ означає $T(\Phi_J) = {}^V A$ та $F(\Phi_J) = \varnothing$, тобто $\Phi_J = \mathrm{T}$;

$\Phi_J|=_{IR}\varnothing$ та $\Phi_J|=_T\varnothing$ означають $T(\Phi_J) = \varnothing$;

$\Phi_J|=_{DI}\varnothing$ та $\Phi_J|=_F\varnothing$ означають $F(\Phi_J) = {}^V A$;

$\Phi_J|=_{T\vee F}\varnothing$ означає: $T(\Phi_J) = \varnothing$ або $F(\Phi_J) = {}^V A$;

$\Phi_J|=_C\varnothing$ означає $T(\Phi_J)\subseteq F(\Phi_J)$;

$\Phi_J|=_{TF}\varnothing$ означає: $T(\Phi_J) = \varnothing$ та $F(\Phi_J) = {}^V A$.

Звідси отримуємо:

$\varnothing^P|=_{IR}\Phi$ та $\varnothing^P|=_F\Phi$ означають: для всіх $\boldsymbol{J}\in\mathcal{P}$ маємо $\varnothing_J|=_{IR}\Phi$, тобто ${}^P|=\Phi$;

$\varnothing^P|=_C\Phi$ теж означає ${}^P|=\Phi$;

$\varnothing^T|=_{DI}\Phi$ та $\varnothing^T|=_T\Phi$ означають: для всіх $\boldsymbol{J}\in\mathcal{T}$ маємо $\varnothing_J|=_{DI}\Phi$, тобто ${}^T|\equiv\Phi$;

$\varnothing^T|=_C\Phi$ теж означає ${}^T|\equiv\Phi$;

$\varnothing^P|=_{T\vee F}\Phi$ означає: $T(\Phi_J) = {}^V A$ або $F(\Phi_J) = \varnothing$, для всіх $\boldsymbol{J}\in\mathcal{P}$, звідки ${}^P|=\Phi$;

$\varnothing^T|=_{T\vee F}\Phi$ означає: $T(\Phi_J) = {}^V A$ або $F(\Phi_J) = \varnothing$, для всіх $\boldsymbol{J}\in\mathcal{T}$, звідки ${}^T|\equiv\Phi$;

не існує $\Phi\in Fr$ таких, що $\varnothing^R|=_C\Phi$ або $\varnothing^P|=_{TF}\Phi$ або $\varnothing^T|=_{TF}\Phi$ або $\varnothing^R|=_{TF}\Phi$;

$\Phi^P|=_{IR}\varnothing$, $\Phi^P|=_F\varnothing$, $\Phi^P|=_C\varnothing$ та $\varnothing^P|=_{T\vee F}\Phi$ означають ${}^P|=\neg\Phi$;

$\Phi^T|=_{DI}\varnothing$, $\Phi^T|=_T\varnothing$, $\Phi^T|=_C\varnothing$ та $\varnothing^T|=_{T\vee F}\Phi$ означають ${}^T|\equiv\neg\Phi$;

не існує $\Phi\in Fr$ таких, що $\Phi^R|=_C\varnothing$ або $\Phi^P|=_{TF}\varnothing$ або $\Phi^T|=_{TF}\varnothing$ або $\Phi^R|=_{TF}\varnothing$.

Враховуючи наслідок 1, отримуємо.

Теорема 14. $\varnothing^P|=_C\Phi \Leftrightarrow \varnothing^P|=_{IR}\Phi \Leftrightarrow \varnothing^P|=_F\Phi \Leftrightarrow \varnothing^T|=_C\Phi \Leftrightarrow \varnothing^T|=_{DI}\Phi \Leftrightarrow \Leftrightarrow \varnothing^T|=_T\Phi \Leftrightarrow {}^P|=\Phi \Leftrightarrow {}^T|\equiv\Phi$;

$\Phi^P|=_C\varnothing \Leftrightarrow \Phi^P|=_{IR}\varnothing \Leftrightarrow \Phi^P|=_F\varnothing \Leftrightarrow \Leftrightarrow \Phi^T|=_C\varnothing \Leftrightarrow \Phi^T|=_{DI}\varnothing \Leftrightarrow \Phi^T|=_T\varnothing \Leftrightarrow \Leftrightarrow {}^P|=\neg\Phi \Leftrightarrow {}^T|\equiv\neg\Phi$.

Властивості контрапозиції:

1) $\Phi^P|=_{IR}\Psi \Leftrightarrow \neg\Psi^P|=_{IR}\neg\Phi$;

2) $\Phi^P|=_{TF}\Psi \Leftrightarrow \neg\Psi^P|=_{TF}\neg\Phi$ та $\Phi^R|=_{TF}\Psi \Leftrightarrow \neg\Psi^R|=_{TF}\neg\Phi$;

3) $\Phi^P|=_{T\vee F}\Psi \Leftrightarrow \neg\Psi^P|=_{T\vee F}\neg\Phi$;

4) $\Phi^R|=_C\Psi \Leftrightarrow \neg\Psi^R|=_C\neg\Phi$;

5) $\Phi^P|=_T\Psi \Leftrightarrow \neg\Psi^P|=_F\neg\Phi$ та $\Phi^P|=_F\Psi \Leftrightarrow \neg\Psi^P|=_T\neg\Phi$.

Водночас неправильні такі твердження:

1) $\Phi^P|=_T\Psi \Rightarrow \neg\Psi^P|=_T\neg\Phi$;

2) $\Phi^P|=_F\Psi \Rightarrow \neg\Psi^P|=_F\neg\Phi$.

Справді, візьмемо $p, q, s\in Ps$ та інтерпретацію $\boldsymbol{J}$: $F(q_J)\nsubseteq T(p_J)\cup F(p_J)$, $T(q_J)\nsubseteq T(s_J)\cup F(s_J)$. Тоді $\neg p\&p^P|=_T q$ та $\neg q_J|\neq_T\neg p\vee p$, $q^P|=_F\neg s\vee s$ та $\neg s\&s_J|\neq_F\neg q$.

Розглянемо можливість перенесення формули з лівої частини логічного наслідку в праву і навпаки.

Теорема 15. Маємо:

1) $\Gamma^P|=_{IR}\Delta,\Phi \Leftrightarrow \neg\Phi,\Gamma^P|=_{IR}\Delta$ та $\Gamma^P|=_{IR}\Delta,\neg\Phi \Leftrightarrow \Phi,\Gamma^P|=_{IR}\Delta$;

2) $\Gamma^R|=_C\Delta,\Phi \Leftrightarrow \neg\Phi,\Gamma^R|=_C\Delta$ та $\Gamma^R|=_C\Delta,\neg\Phi \Leftrightarrow \Phi,\Gamma^R|=_C\Delta$;

3) $\Gamma^P|=_T\Delta,\Phi \Rightarrow \neg\Phi,\Gamma^P|=_T\Delta$ та $\Gamma^P|=_T\Delta,\neg\Phi \Rightarrow \Phi,\Gamma^P|=_T\Delta$;

4) $\Phi,\Gamma^P|=_F\Delta \Rightarrow \Gamma^P|=_F\Delta,\neg\Phi$ та $\neg\Phi,\Gamma^P|=_F\Delta \Rightarrow \Gamma^P|=_F\Delta,\Phi$.

Теорема 16. Можливо:

1) $\neg\Phi,\Gamma^P|=_T\Delta$ та $\Gamma^P|\neq_T\Delta,\Phi$; $\Phi,\Gamma^P|=_T\Delta$ та $\Gamma^P|\neq_T\Delta,\neg\Phi$;

2) $\Gamma^P|=_F\Delta,\neg\Phi$ та $\Phi,\Gamma^P|\neq_F\Delta$; $\Gamma^P|=_F\Delta,\Phi$ та $\neg\Phi,\Gamma^P|\neq_F\Delta$;

3) $\neg\Phi,\Gamma^P|=_{TF}\Delta$ та $\Gamma^P|\neq_{TF}\Delta,\Phi$; $\Gamma^P|=_{TF}\Delta,\neg\Phi$ та $\Phi,\Gamma^P|\neq_{TF}\Delta$;

4) $\neg\Phi,\Gamma^R|=_{TF}\Delta$ та $\Gamma^R|\neq_{TF}\Delta,\Phi$; $\Gamma^R|=_{TF}\Delta,\neg\Phi$ та $\Phi,\Gamma^R|\neq_{TF}\Delta$;

5) $\neg\Phi,\Gamma^P|=_{T\vee F}\Delta$ та $\Gamma^P|\neq_{T\vee F}\Delta,\Phi$; $\Gamma^P|=_{T\vee F}\Delta,\neg\Phi$ та $\Phi,\Gamma^P|\neq_{T\vee F}\Delta$.

Доводимо п. 1. Візьмемо $\boldsymbol{J}$ таку: $T(\Phi_J)=F(\Phi_J)=\varnothing$, $T(\Gamma_J)\neq\varnothing$. Маємо $\neg\Phi,\Gamma\,{}^{P}|=_T\neg\Phi$, однак $\Gamma_J|\neq_T\neg\Phi,\Phi$; маємо $\Phi,\Gamma\,{}^{P}|=_T\Phi$, але $\Gamma\,{}^{P}|\neq_T\Phi,\neg\Phi$.

Доводимо п. 2. Візьмемо $\boldsymbol{J}$ таку: $T(\Phi_J)=F(\Phi_J)=\varnothing$, $F(\Delta_J)\neq\varnothing$. Маємо $\Delta\vee\neg\Phi\,{}^{P}|=_F\Delta,\neg\Phi$, проте $\Phi,\Delta\vee\neg\Phi\,{}^{P}|\neq_F\Delta$; маємо $\Delta\vee\Phi\,{}^{P}|=_F\Delta,\Phi$, але $\neg\Phi,\Delta\vee\Phi\,{}^{P}|\neq_F\Delta$.

Доводимо п. 4. Завжди маємо $\neg\Phi,\Gamma\,{}^{R}|=_{TF}\neg\Phi$, однак $\Gamma\,{}^{R}|\neq_T\neg\Phi,\Phi$, якщо взяти $\boldsymbol{J}$ таку: $T(\Phi_J)=F(\Phi_J)=\varnothing$, $T(\Gamma_J)\neq\varnothing$.

Маємо $\Delta\vee\neg\Phi\,{}^{R}|=_{TF}\Delta,\neg\Phi$, проте $\Phi,\Delta\vee\neg\Phi\,{}^{R}|\neq_F\Delta$, якщо взяти $\boldsymbol{J}$ таку: $T(\Phi_J)=F(\Phi_J)=\varnothing$, $F(\Delta_J)\neq\varnothing$.

Аналогічно доводимо п. 3.

Доводимо п. 5. Нехай $p,q\in Ps$, Φ – це формула $p\vee(q\&\neg q)$, Ψ – це $p\,\&\,(q\vee\neg q)$.

Тоді маємо $\neg\Psi,\Phi\,{}^{P}|=_T\Psi$:

$T(\neg\Psi\,\&\,\Phi)=F(\Psi)\cap T(\Phi)=$

$=F(p\,\&\,(q\vee\neg q))\cap T(p\vee(q\&\neg q))=$

$=F(p)\cap T(p)=\varnothing$.

Далі маємо $\Phi\,{}^{P}|=_F\Psi,\neg\Phi$:

$F(\Psi\vee\neg\Phi)=F(\Psi)\cap T(\Phi)=\varnothing$.

Звідси маємо $\neg\Psi,\Phi\,{}^{P}|=_{T\vee F}\Psi$ та $\Phi\,{}^{P}|=_{T\vee F}\Psi,\neg\Phi$, але (приклад 3) $\Phi\,{}^{P}|\neq_{T\vee F}\Psi$.

Отже, для ${}^{P}|=_{IR}$ та ${}^{R}|=_C$ можна робити перенесення формули з лівої частини логічного наслідку в праву і навпаки; а для ${}^{P}|=_T$, ${}^{P}|=_F$, ${}^{P}|=_{TF}$, ${}^{P}|=_{T\vee F}$, ${}^{R}|=_{TF}$ – не можна.

Теорема 17. Маємо:

1) $\neg\Phi,\Phi,\Gamma\,{}^{P}|=_T\Delta$, проте $\neg\Phi,\Phi,\Gamma\,{}^{P}|\neq_F\Delta$;

2) $\Gamma\,{}^{P}|=_F\Delta,\neg\Psi,\Psi$, але $\Gamma\,{}^{P}|\neq_T\Delta,\neg\Psi,\Psi$;

3) $\neg\Phi,\Phi,\Gamma\,{}^{P}|=_{TF}\Delta,\neg\Psi,\Psi$;

4) $\neg\Phi,\Phi,\Gamma\,{}^{R}|\neq_{TF}\Delta,\neg\Psi,\Psi$.

Для п. 1 візьмемо Φ,Ψ та інтерпретацію $\boldsymbol{J}$: $T(\Phi_J)=F(\Phi_J)=\varnothing$ та $F(\Psi_J)\neq\varnothing$. Тоді $\neg\Phi,\Phi_J|\neq_F\Psi$, тому $\neg\Phi,\Phi\,{}^{P}|\neq_F\Psi$.

Для п. 2 візьмемо Φ,Ψ та інтерпретацію $\boldsymbol{J}$: $T(\Psi_J)=F(\Psi_J)=\varnothing$ та $T(\Phi_J)\neq\varnothing$. Тоді $\Phi_J|\neq_T\neg\Psi,\Psi$, тому $\Phi\,{}^{P}|\neq_T\neg\Psi,\Psi$.

Доведемо п. 3. Для кожної $\boldsymbol{J}\in\mathcal{P}$ маємо $T(\Phi_J)\cap F(\Phi_J)=\varnothing$ та $F(\Psi_J)\cap T(\Psi_J)=\varnothing$, звідки $\neg\Phi,\Phi,\Gamma\,{}^{P}|=_T\Delta,\neg\Psi,\Psi$ та $\neg\Phi,\Phi,\Gamma\,{}^{P}|=_F\Delta,\neg\Psi,\Psi$, тому отримуємо $\neg\Phi,\Phi,\Gamma\,{}^{P}|=_{TF}\Delta,\neg\Psi,\Psi$.

Для п. 4 візьмемо Φ,Ψ та інтерпретацію $\boldsymbol{J}$ таку: $T(\Phi_J)=F(\Phi_J)={}^{V}A$ та $T(\Psi_J)=F(\Psi_J)=\varnothing$. Тоді $\Phi\&\neg\Phi_J|\neq_{TF}\Psi\vee\neg\Psi$, тому $\neg\Phi,\Phi\,{}^{R}|\neq_{TF}\neg\Psi,\Psi$.

Відношення ${}^{R}|=_C$ зводиться до ${}^{R}|=_{TF}$.

Для $\Sigma\subseteq Fr$ введемо позначення:

$\neg\Sigma=\{\neg\Phi\,|\,\Phi\in\Sigma\}$.

Теорема 18. Маємо

$$\Gamma\,{}^{R}|=_C\Delta\Leftrightarrow\neg\Delta,\Gamma\,{}^{R}|=_{TF}\Delta,\neg\Gamma.$$

Для $\boldsymbol{J}\in\mathcal{R}$ та множини $\neg\Sigma$ маємо:

$$T^{\wedge}(\neg\Sigma_J)=\bigcap_{\Phi\in\Sigma}T(\neg\Phi_J)=\bigcap_{\Phi\in\Sigma}F(\Phi_J)=F^{\wedge}(\Sigma_J),$$

$$T^{\vee}(\neg\Sigma_J)=\bigcup_{\Phi\in\Sigma}T(\neg\Phi_J)=\bigcup_{\Phi\in\Sigma}F(\Phi_J)=F^{\vee}(\Sigma_J),$$

$$F^{\wedge}(\neg\Sigma_J)=\bigcap_{\Phi\in\Sigma}F(\neg\Phi_J)=\bigcap_{\Phi\in\Sigma}T(\Phi_J)=T^{\wedge}(\Sigma_J),$$

$$F^{\vee}(\neg\Sigma_J)=\bigcup_{\Phi\in\Sigma}F(\neg\Phi_J)=\bigcup_{\Phi\in\Sigma}T(\Phi_J)=T^{\vee}(\Sigma_J).$$

Для кожної $\boldsymbol{J}\in\mathcal{R}$ тоді маємо:

$$\Gamma_J|=_C\Delta\Leftrightarrow T^{\wedge}(\Gamma_J)\cap F^{\wedge}(\Delta_J)\subseteq F^{\vee}(\Gamma_J)\cup T^{\vee}(\Delta_J)$$

$$\Leftrightarrow\quad\Leftrightarrow T^{\wedge}(\Gamma_J)\cap T^{\wedge}(\neg\Delta_J)\subseteq T^{\vee}(\neg\Gamma_J)\cup T^{\vee}(\Delta_J)\quad\Leftrightarrow$$

$$\Leftrightarrow F^{\wedge}(\neg\Gamma_J)\cap F^{\wedge}(\Delta_J)\subseteq F^{\vee}(\Gamma_J)\cup F^{\vee}(\neg\Delta_J)\qquad\Leftrightarrow$$

$$\Leftrightarrow\neg\Delta,\Gamma_J|=_T\Delta,\neg\Gamma\quad\Leftrightarrow\quad\neg\Delta,\Gamma_J|=_F\Delta,\neg\Gamma\quad\Leftrightarrow$$

$$\Leftrightarrow\neg\Delta,\Gamma_J|=_{TF}\Delta,\neg\Gamma.$$

Звідси отримуємо

$$\Gamma\,{}^{R}|=_C\Delta\Leftrightarrow\neg\Delta,\Gamma\,{}^{R}|=_{TF}\Delta,\neg\Gamma.$$

Зокрема, для $\Phi,\Psi\in Fr$ маємо:

$$\Phi\,{}^{R}|=_C\Psi\Leftrightarrow\neg\Psi,\Phi\,{}^{R}|=_{TF}\Psi,\neg\Phi.$$

Приклад 4. Маємо

$\Phi\&(P\vee\neg P)\vee(Q\&\neg Q)\,{}^{R}|=_C\Phi\&\neg S\vee S$ та

$\Phi\&(P\vee\neg P)\vee(Q\&\neg Q)\,{}^{P}|\neq_{T\vee F}\Phi\&\neg S\vee S$.

Формули $\Phi\&(P\vee\neg P)\vee(Q\&\neg Q)$ та $\Phi\&\neg S\vee S$ позначимо α та β.

Для кожної $\boldsymbol{J}\in\mathcal{R}$ маємо:

$$T(\alpha_J)=(T(\Phi_J)\cap P_{J\cup})\cup Q_{J\cap};$$

$F(\alpha_J) = (F(\Phi_J) \cup P_{J\cap}) \cap Q_{J\cup} =$

$= (F(\Phi_J) \cap Q_{J\cup}) \cup (P_{J\cap} \cap Q_{J\cup})$;

$T(\beta_J) = (T(\Phi_J) \cap F(S_J)) \cup T(S_J)$;

$F(\beta_J) = (F(\Phi_J) \cup T(S_J)) \cap F(S_J)$.

Тоді $T(\alpha_J) \cap F(\beta_J) = ((T(\Phi_J) \cap P_{J\cup}) \cup Q_{J\cap}) \cap$
$\cap (F(\Phi_J) \cup T(S_J)) \cap F(S_J) =$
$= (T(\Phi_J) \cap P_{J\cup} \cap F(\Phi_J) \cap F(S_J)) \cup$
$\cup (T(\Phi_J) \cap P_{J\cup} \cap S_{J\cap}) \cup (Q_{J\cap} \cap F(\Phi_J) \cap F(S_J)) \cup$
$\cup (Q_{J\cap} \cap S_{J\cap}) \subseteq (T(\Phi_J) \cap F(S_J)) \cup T(S_J) \cup$
$\cup (F(\Phi_J) \cap Q_{J\cup}) \cup (P_{J\cap} \cap Q_{J\cup}) = T(\beta_J) \cup F(\alpha_J)$,
тобто $\alpha_J |=_C \beta$. Отже, $\alpha^R |=_C \beta$.

Візьмемо $\boldsymbol{A} \in \mathcal{P}$ таку: $P_{A\cup} \cap Q_{A\cup} = \varnothing$;
$(P_{A\cup} \cup Q_{A\cup}) \cap S_{A\cup} = \varnothing$; $T(\alpha_A) \neq \varnothing$;
$F(\beta_A) \neq \varnothing$ (при цьому $P_{A\cap} = Q_{A\cap} = \varnothing$).
Маємо $T(\alpha_A) = T(\Phi_A) \cap P_{A\cup} \subseteq P_{A\cup}$,
$F(\alpha_A) = F(\Phi_A) \cap Q_{A\cup} \subseteq Q_{A\cup}$;
$T(\beta_A) = (T(\Phi_A) \cap F(S_A)) \cup T(S_A) \subseteq S_{A\cup}$;
$F(\beta_A) = (F(\Phi_A) \cup T(S_A)) \cap F(S_A) \subseteq S_{A\cup}$.

Згідно $(P_{A\cup} \cup Q_{A\cup}) \cap S_{A\cup} = \varnothing$ тоді $T(\alpha_A) \not\subseteq T(\beta_A)$ та $F(\beta_A) \not\subseteq F(\alpha_A)$, звідки $\alpha_A |\neq_T \beta$ та $\alpha_A |\neq_F \beta$. Отже, $\alpha^P |\neq_{T\vee F} \beta$.

Приклад 5. Маємо

$\Phi \vee (P\&\neg P) \vee (Q\&\neg Q)^P |=_T \Phi$,

$\Phi \vee (P\&\neg P) \vee (Q\&\neg Q)^P |\neq_F \Phi$ та

$\Phi \vee (P\&\neg P) \vee (Q\&\neg Q)^R |\neq_C \Phi$.

$\Phi \vee (P\&\neg P) \vee (Q\&\neg Q)$ позначимо α.

Для кожної $\boldsymbol{J} \in \mathcal{P}$ маємо $T(\alpha_J) =$
$= T(\Phi_J) \cup P_{J\cap} \cup P_{J\cap} = T(\Phi_J)$, звідки $\alpha_J{}^P |=_T \Phi$.

Візьмемо $\boldsymbol{B} \in \mathcal{P}$ таку: $F(\Phi_B) \neq \varnothing$,
$P_{B\cup} = Q_{B\cup} = \varnothing$. Тоді маємо $F(\alpha_B) =$
$= F(\Phi_B) \cap P_{B\cup} \cap Q_{B\cup} = \varnothing$. Отже,
$F(\Phi_B) \not\subseteq F(\alpha_B)$, тому $\alpha_B |\neq_F \Phi$ та $\alpha^P |\neq_F \Phi$.

Візьмемо $\boldsymbol{A} \in \mathcal{R}$ таку:

$P_{A\cap} = P_{A\cup}$, $Q_{A\cap} = Q_{A\cup}$, $P_{A\cup} \cap Q_{A\cup} = \varnothing$;
$T(\Phi_A) \neq \varnothing$, $F(\Phi_A) \neq \varnothing$, $T(\Phi_A) \cap F(\Phi_A) = \varnothing$;
тоді $F(\alpha_A) = F(\Phi_A) \cap P_{A\cup} \cap Q_{A\cup} = \varnothing$, звідки отримуємо $F(\alpha_A) \cup T(\Phi_A) = T(\Phi_A)$.
Тепер $T(\alpha_A) \cap F(\Phi_A) \not\subseteq T(\Phi_A) = F(\alpha_A) \cup T(\Phi_A)$, звідки $\alpha_A |\neq_C \Phi$, тому $\alpha^R |\neq_C \Phi$.

Приклад 6. Маємо

$\Phi^P |=_F \Phi \,\&\, (P \vee \neg P) \,\&\, (Q \vee \neg Q)$,

$\Phi^P |\neq_T \Phi \,\&\, (P \vee \neg P) \,\&\, (Q \vee \neg Q)$ та

$\Phi^R |\neq_C \Phi \,\&\, (P \vee \neg P) \,\&\, (Q \vee \neg Q)$.

Показується аналогічно прикладу 5.

Приклад 7. Маємо

$\Phi\&\neg\Phi \vee P\&\neg P^P |=_{TF} (\Phi \vee \neg\Phi) \,\&\, (P \vee \neg P)$ та

$\Phi\&\neg\Phi \vee P\&\neg P^R |\neq_C (\Phi \vee \neg\Phi) \,\&\, (P \vee \neg P)$.

Формули $\Phi\&\neg\Phi \vee P\&\neg P$ та $(\Phi \vee \neg\Phi) \,\&\, (P \vee \neg P)$ позначимо α та β.

Для кожної інтерпретації $\boldsymbol{J}$ маємо:

$T(\alpha_J) = \Phi_{J\cap} \cup P_{J\cap}$, $F(\alpha_J) = \Phi_{J\cup} \cap P_{J\cup}$,

$T(\beta_J) = \Phi_{J\cup} \cap P_{J\cup}$, $F(\beta_J) = \Phi_{J\cap} \cup P_{J\cap}$.

Звідси $T(\alpha_J) = F(\beta_J)$, $F(\alpha_J) = T(\beta_J)$.

Тепер для кожної $\boldsymbol{B} \in \mathcal{P}$ маємо $T(\alpha_B) = \varnothing$ та $F(\beta_B) = \varnothing$, тому $\alpha_B |=_T \beta$ та $\alpha_B |=_F \beta$. Звідси $\alpha^P |=_{TF} \beta$.

Візьмемо $\boldsymbol{A} \in \mathcal{R}$ таку:

$\Phi_{A\cap} \neq \varnothing$, $P_{A\cap} \neq \varnothing$, $\Phi_{A\cup} \cap P_{A\cup} = \varnothing$.

Тоді $T(\alpha_A) \cap F(\beta_A) = \Phi_{A\cap} \cup P_{A\cap} \neq \varnothing$, $F(\alpha_A) \cup T(\beta_A) = \Phi_{A\cup} \cap P_{A\cup} = \varnothing$. Звідси $\alpha_A |\neq_C \beta$, тому $\alpha^R |\neq_C \beta$.

Зведемо результати щодо наявності відповідного наслідку для випадків, розглянутих в теоремі 15 та прикладах 2–7, в таблицю. В усіх наведених прикладах маємо $^P|=_{IR}$ та не маємо $^R|=_{TF}$.

Введемо такі скорочені позначення.

LC_1: $\neg\Phi, \Phi, \Gamma |= \Delta$;

LC_2: $\Gamma |= \neg\Psi, \Psi, \Delta$;

LC_3: $\neg\Phi, \Phi, \Gamma |= \neg\Psi, \Psi, \Delta$;

LC_4: $\Phi \vee (Q\&\neg Q) |= \Phi$;

LC_5: $\Phi |= \Phi \,\&\, (Q \vee \neg Q)$;

LC_6: $\Phi \vee (Q\&\neg Q) |= \Phi \,\&\, (Q \vee \neg Q)$;

LC_7: $\Phi \vee (P\&\neg P) \vee (Q\&\neg Q) |= \Phi$;

LC_8: $\Phi |= \Phi \,\&\, (P \vee \neg P) \,\&\, (Q \vee \neg Q)$;

LC_9: $\Phi\&\neg\Phi \vee P\&\neg P |= (\Phi \vee \neg\Phi) \,\&\, (P \vee \neg P)$;

LC_{10}: $\Phi\&(P \vee \neg P) \vee (Q\&\neg Q) |= \Phi\&\neg S \vee S$.

Таблиця. Наявність логічного наслідку

	$^{P}\models_{T\vee F}$	$^{P}\models_{T}$	$^{P}\models_{F}$	$^{P}\models_{TF}$	$^{R}\models_{C}$
LC_1	+	+	–	–	+
LC_2	+	–	+	–	+
LC_3	+	+	+	+	+
LC_4	+	+	–	–	+
LC_5	+	–	+	–	+
LC_6	–	–	–	–	–
LC_7	+	+	–	–	–
LC_8	+	–	+	–	–
LC_9	+	+	+	+	–
LC_{10}	–	–	–	–	+

Отже, отримуємо наступну теорему.

Теорема 19. Маємо такі співвідношення між розглянутими відношеннями:

$^{P}\models_{TF}\subset{}^{P}\models_{T}\subset{}^{P}\models_{T\vee F}$, $^{P}\models_{TF}\subset{}^{P}\models_{F}\subset{}^{P}\models_{T\vee F}$, $^{P}\models_{T\vee F}\subset{}^{P}\models_{IR}$;

$^{R}\models_{TF}\subset{}^{P}\models_{TF}$, $^{R}\models_{TF}\subset{}^{R}\models_{C}\subset{}^{P}\models_{IR}$;

$^{P}\models_{T}\not\subset{}^{P}\models_{F}$, $^{P}\models_{F}\not\subset{}^{P}\models_{T}$, $^{P}\models_{TF}\not\subset{}^{R}\models_{C}$, $^{R}\models_{C}\not\subset{}^{P}\models_{T\vee F}$;

$\models_{t}\subset{}^{P}\models_{IR}$; $^{P}\models_{T\vee F}\not\subset\models_{t}$, $^{R}\models_{C}\not\subset\models_{t}$.

Властивості відношень логічного наслідку для множин формул є семантичною основою побудови секвенційних числень для відповідних класів логік квазіарних предикатів. Такі числення збудовано для відношень $^{P}\models_{IR}$, $^{P}\models_{T}$, $^{P}\models_{F}$, $^{P}\models_{TF}$, $^{R}\models_{TF}$.

Розглянемо особливості побудови секвенційних числень для $^{P}\models_{T\vee F}$ та $^{R}\models_{C}$.

Відмінності відношень логічного наслідку проявляються вже на пропозиційному рівні, тому розглянемо лише пропозиційні властивості декомпозиції формул.

Властивості, пов'язані з реноменаціями і кванторами, досліджено в [2–4].

Для відношень $^{P}\models_{IR}$, $^{P}\models_{T}$, $^{P}\models_{F}$, $^{P}\models_{TF}$ та $^{R}\models_{TF}$ маємо (див. [2–4]) такі властивості (тут $\models$ – одне з цих відношень):

$\neg\neg_{\mathrm{L}}$) $\neg\neg\Phi,\Gamma\models\Delta\Leftrightarrow\Phi,\Gamma\models\Delta$;

$\neg\neg_{\mathrm{R}}$) $\Gamma\models\Delta,\neg\neg\Phi\Leftrightarrow\Gamma\models\Delta,\Phi$;

$\vee_{\mathrm{L}}$) $\Phi\vee\Psi,\Gamma\models\Delta\Leftrightarrow\Phi,\Gamma\models\Delta$ та $\Psi,\Gamma\models\Delta$;

$\vee_{\mathrm{R}}$) $\Gamma\models\Delta,\Phi\vee\Psi\Leftrightarrow\Gamma\models\Delta,\Phi,\Psi$;

$\neg\vee_{\mathrm{L}}$) $\neg(\Phi\vee\Psi),\Gamma\models\Delta\Leftrightarrow\neg\Phi,\neg\Psi,\Gamma\models\Delta$;

$\neg\vee_{\mathrm{R}}$) $\Gamma\models\Delta,\neg(\Phi\vee\Psi)\Leftrightarrow\Gamma\models\Delta,\neg\Phi$ та $\Gamma\models\Delta,\neg\Psi$.

Згідно теореми 15, для $^{P}\models_{IR}$ та $^{R}\models_{C}$ додатково справджуються:

$\neg_{\mathrm{L}}$) $\neg\Phi,\Gamma\,{}^{P}\models_{IR}\Delta\Leftrightarrow\Gamma\,{}^{P}\models_{IR}\Delta,\Phi$;

$\neg_{\mathrm{R}}$) $\Gamma\,{}^{P}\models_{IR}\Delta,\neg\Phi\Leftrightarrow\Phi,\Gamma\,{}^{P}\models_{IR}\Delta$.

Згідно теореми 16, для $^{P}\models_{T}$, $^{P}\models_{F}$, $^{P}\models_{TF}$ та $^{R}\models_{TF}$ ці властивості неправильні.

Розглянемо властивості, які гарантують наявність логічного наслідку.

Для кожного з наслідків маємо:

C) $\Phi,\Gamma\models\Delta,\Phi$.

Додатково гарантують наявність відповідного логічного наслідку такі властивості (їх виділення зайве для відношень $^{P}\models_{IR}$ та $^{R}\models_{C}$ в силу властивостей $\neg_{\mathrm{L}}$ та $\neg_{\mathrm{R}}$):

CL) $\Phi,\neg\Phi,\Gamma\,{}^{P}\models_{T}\Delta$;

CR) $\Gamma\,{}^{P}\models_{F}\Delta,\Phi,\neg\Phi$;

CLR) $\Phi,\neg\Phi,\Gamma\,{}^{P}\models_{TF}\Delta,\Psi,\neg\Psi$;

CL$\vee$*R*) $\Phi,\neg\Phi,\Gamma\,{}^{P}\models_{T\vee F}\Delta$ або $\Gamma\,{}^{P}\models_{T\vee F}\Delta,\Phi,\neg\Phi$.

З'ясуємо, чи правильні властивості декомпозиції формул для $^{P}\models_{T\vee F}$ та $^{R}\models_{C}$.

Для всіх наслідків властивості $\neg\neg_{\mathrm{L}}$, $\neg\neg_{\mathrm{R}}$, $\vee_{\mathrm{R}}$, $\neg\vee_{\mathrm{L}}$ випливають із визначень.

Теорема 20. Для $^{P}\models_{T\vee F}$ неправильними є властивості $\vee_{\mathrm{L}}$ та $\neg\vee_{\mathrm{R}}$.

Для кожної $\boldsymbol{J}\in\mathcal{P}$ маємо:

$\Phi\vee\Psi,\Gamma_J\models_{T\vee F}\Delta\Leftrightarrow$

$(T(\Phi_J)\cup T(\Psi_J))\cap T(\Gamma_J)\subseteq T(\Delta_J)$ або

$F(\Delta_J)\subseteq(F(\Phi_J)\cap F(\Psi_J))\cup F(\Gamma_J)\Leftrightarrow$

$(T(\Phi_J)\cap T(\Gamma_J))\cup(T(\Psi_J)\cap T(\Gamma_J))\subseteq T(\Delta_J)$ або

$F(\Delta_J)\subseteq(F(\Phi_J)\cup F(\Gamma_J))\cap(F(\Psi_J)\cup F(\Gamma_J))\Leftrightarrow$

$(T(\Phi_J)\cap T(\Gamma_J)\subseteq T(\Delta_J)$ та $T(\Psi_J)\cap T(\Gamma_J)\subseteq T(\Delta_J))$ або $(F(\Delta_J)\subseteq(F(\Phi_J)\cup F(\Gamma_J))$ та $F(\Delta_J)\subseteq F(\Psi_J)\cup F(\Gamma_J))\Leftrightarrow(\Phi,\Gamma_J\models_T\Delta$ та $\Psi,\Gamma_J\models_T\Delta)$ або $(\Phi,\Gamma\models_F\Delta$ та $\Psi,\Gamma\models_F\Delta)\Rightarrow$ $(\Phi,\Gamma_J\models_{T\vee F}\Delta$ та $\Psi,\Gamma_J\models_{T\vee F}\Delta)$ або

$(\Phi, \Gamma \models_{T\vee F} \Delta$ та $\Psi, \Gamma \models_{T\vee F} \Delta) \Leftrightarrow \Phi, \Gamma_J \models_{T\vee F} \Delta$ та $\Psi, \Gamma_J \models_{T\vee F} \Delta$.

Отже, $\Phi\vee\Psi, \Gamma \overset{P}{\models}_{T\vee F} \Delta \Rightarrow$ $\Rightarrow \Phi, \Gamma \overset{P}{\models}_{T\vee F} \Delta$ та $\Psi, \Gamma \overset{P}{\models}_{T\vee F} \Delta$.

Подібним чином показуємо $\Gamma \overset{P}{\models}_{T\vee F} \Delta, \neg(\Phi\vee\Psi) \Rightarrow \Gamma \overset{P}{\models}_{T\vee F} \Delta, \neg\Phi$ та $\Gamma \overset{P}{\models}_{T\vee F} \Delta, \neg\Psi$.

Покажемо: з умови $\Phi, \Gamma \overset{P}{\models}_{T\vee F} \Delta$ та $\Psi, \Gamma \overset{P}{\models}_{T\vee F} \Delta$ не випливає $\Phi\vee\Psi, \Gamma \overset{P}{\models}_{T\vee F} \Delta$.

Формули $S\,\&\,(P\vee\neg P)$, $S\,\&\,(Q\vee\neg Q)$, $S\,\&\,(Q\vee\neg Q)\vee(P\&\neg P)$ позначимо α, β, γ.

Покажемо: $\alpha \overset{P}{\models}_F \beta$ та $\gamma \overset{P}{\models}_T \beta$, тому $\alpha \overset{P}{\models}_{T\vee F} \beta$ та $\gamma \overset{P}{\models}_{T\vee F} \beta$, проте $\alpha\vee\gamma \overset{P}{\not\models}_{T\vee F} \beta$.

Для кожної $\boldsymbol{J}\in\mathcal{P}$ маємо:

$T(\alpha_J) = T(S_J) \cap P_{J\cup}$, $F(\alpha_J) = F(S_J)$;

$T(\beta_J) = T(S_J) \cap Q_{J\cup}$, $F(\beta_J) = F(S_J)$;

$T(\gamma_J) = T(S_J) \cap Q_{J\cup}$, $F(\gamma_J) = F(S_J) \cap Q_{J\cup}$;

$T(\alpha\vee\gamma_J) = T(\alpha_J)\cup T(\gamma_J) = T(S_J) \cap (P_{J\cup} \cup Q_{J\cup})$;

$F(\alpha\vee\gamma_J) = F(\alpha_J)\cap F(\gamma_J) = F(S_J) \cap Q_{J\cup}$.

Звідси $F(\alpha_J) = F(\beta_J)$ та $T(\gamma_J) = T(\beta_J)$, що дає $\alpha \overset{P}{\models}_F \beta$ та $\gamma \overset{P}{\models}_T \beta$.

Візьмемо $\boldsymbol{A}\in\mathcal{P}$ таку: $P_{A\cup} \cap Q_{A\cup} = \varnothing$, $T(S_A) \neq \varnothing$, $F(S_A) \neq \varnothing$, $P_{A\cup} \neq \varnothing$, $Q_{A\cup} \neq \varnothing$, $T(S_A)\cap Q_{A\cup} \neq \varnothing$, $F(S_A)\cap P_{A\cup} \neq \varnothing$.

Тоді $T(\alpha\vee\gamma_A) = T(S_A) \cap (P_{A\cup} \cup Q_{A\cup}) \supset$ $\supset T(S_A) \cap Q_{A\cup} = T(\beta_A)$, тому $T(\gamma_A) \not\subseteq T(\beta_A)$; $F(\beta_A) = F(S_A) \supset F(S_A) \cap Q_{J\cup} = T(\alpha\vee\gamma_J)$, отже $F(\beta_A) \not\subseteq F(\gamma_A)$. Звідси $\alpha\vee\gamma_A \not\models_T \beta$, $\alpha\vee\gamma_A \not\models_F \beta$, що дає $\alpha\vee\gamma_A \not\models_{T\vee F} \beta$, тому й $\alpha\vee\gamma \overset{P}{\not\models}_{T\vee F} \beta$.

Для інтерпретації $\boldsymbol{A}$ також маємо $\alpha_A \not\models_T \beta$ та $\gamma_A \not\models_F \beta$, звідки $\alpha \overset{P}{\not\models}_T \beta$ та $\gamma \overset{P}{\not\models}_F \beta$.

Подібним чином показуємо:

$\neg\beta \overset{P}{\models}_F \neg\alpha$ та $\neg\beta \overset{P}{\models}_T \neg\gamma$, водночас $\neg\beta_A \not\models_{T\vee F} \neg(\alpha\vee\gamma)$, тому $\neg\beta \overset{P}{\not\models}_{T\vee F} \neg(\alpha\vee\gamma)$.

Отже, із умови $\Gamma \overset{P}{\models}_{T\vee F} \Delta, \neg\Phi$ та $\Gamma \overset{P}{\models}_{T\vee F} \Delta, \Psi$ не випливає $\Gamma \overset{P}{\models}_{T\vee F} \Delta, \neg(\Phi\vee\Psi)$.

Твердження 13. Для $\overset{R}{\models}_C$ неправильними є властивості $\vee_L$ та $\neg\vee_R$.

Маємо $P\vee(Q\&\neg Q)\vee(S\&\neg S) \overset{R}{\not\models}_C P$ за прикладом 5. Проте $P\vee(Q\&\neg Q) \overset{R}{\models}_C P$.

Справді, для кожної $\boldsymbol{J}\in\mathcal{R}$ маємо:

$T(P\vee(Q\&\neg Q)_J)\cap F(P_J) =$

$= (T(P_J)\cup(F(Q_J)\cap T(Q_J)))\cap F(P_J) =$

$= (T(P_J)\cap F(P_J))\cup(F(Q_J)\cap T(Q_J)\cap F(P_J)) \subseteq$

$\subseteq T(P_J)\cup((F(Q_J)\cup T(Q_J))\cap F(P_J))$.

Аналогічно $P\vee(S\&\neg S) \overset{R}{\models}_C P$.

Таким чином, на додаток до того, що $\overset{P}{\models}_{T\vee F}$ та $\overset{R}{\models}_C$ нетранзитивні, для них стають неправильними деякі властивості декомпозиції формул, що не дає змоги безпосередньо будувати для них секвенційні числення. Проте для відношення $\overset{R}{\models}_C$ таку неприємність можливо обійти зведенням $\overset{R}{\models}_C$ до $\overset{R}{\models}_{TF}$. Справді, за теоремою 18 маємо: $\Gamma \overset{R}{\models}_C \Delta \Leftrightarrow \neg\Delta, \Gamma \overset{R}{\models}_{TF} \Delta, \neg\Gamma$. Це означає, що немає необхідності будувати секвенційне числення для формалізації $\overset{R}{\models}_C$, адже $\Gamma \overset{R}{\models}_C \Delta \Leftrightarrow$ секвенція $\vdash\neg\Delta, \vdash\Gamma, \dashv\Delta, \dashv\neg\Gamma$ вивідна в численні для $\overset{R}{\models}_{TF}$.

Висновки

В роботі досліджено відношення логічного наслідку в логіках тотальних однозначних, часткових однозначних, тотальних неоднозначних та часткових неоднозначних квазіарних предикатів. Поряд із розглянутими раніше відношеннями типів *T*, *F*, *TF*, *IR* і *DI*, для логік квазіарних предикатів тут запропоновано відношення типів $T\vee F$ та *C*. Описано властивості відношень логічного наслідку для формул та множин формул. Розглянуто окремі випадки відношень, коли одна із множин формул порожня. Наведено приклади, які засвідчують відмінності одних відношень від інших. Показано, що відношення $\overset{P}{\models}_{T\vee F}$, та $\overset{R}{\models}_C$ нетранзитивні, для $\overset{R}{\models}_C$ та $\overset{P}{\models}_{T\vee F}$ неправильні деякі властивості декомпозиції формул, проте $\overset{R}{\models}_C$ моделюється за допомогою $\overset{R}{\models}_{TF}$. Встановлено співвідношення між різними відношеннями логічного наслідку.

1. *Нікітченко М.С., Шкільняк С.С.* Математична логіка та теорія алгоритмів. – К.: ВПЦ Київський університет, 2008. – 528 с.

2. *Нікітченко М.С., Шкільняк С.С.* Прикладна логіка. – К.: ВПЦ Київський університет, 2013. – 278 с.
3. *Нікітченко М.С., Шкільняк О.С., Шкільняк С.С.* Першопорядкові композиційно-номінативні логіки із узагальненими реномінаціями // Проблеми програмування. – 2014. – № 2–3 – С. 17–28.
4. *Nikitchenko M., Shkilniak S.* Semantic Properties of Logics of Quasiary Predicates // Workshop on Foundations of Informatics: Proceedings FOI-2015. – Chisinau, Moldova. – P. 180–197.
5. *Смирнова Е.Д.* Логика и философия. – М.: РОССПЕН, 1996. – 304 с.

References

1. *Nikitchenko M. Shkilniak S.* (2008). Mathematical logic and theory of algorithms. – Kyiv: VPC Kyivskyi Universytet (in Ukrainian).
2. *Nikitchenko M. and Shkilniak S.* (2013). Applied logic. Kyiv: VPC Kyivskyi Universytet (in Ukrainian).
3. *Nikitchenko M., Shkilniak O. and Shkilniak S.* (2014). First-order composition-nominative logics with generalized renominations // In Problems in Progamming. N 2–3, P. 17–28 (in Ukrainian).
4. Nikitchenko M. and Shkilniak S. (2015). Semantic Properties of Logics of Quasiary Predicates // In Workshop on Foundations of Informatics: Proceedings FOI-2015. – Chisinau, Moldova, P. 180–197.
5. *Smirnova E.* (1996). Logic and Philosophy. Moscow: ROSSPEN (in Russian).

Одержано 12.11.2015

Про автора:

Шкільняк Оксана Степаніна,
кандидат фізико-математичних наук,
доцент, доцент кафедри інформаційних систем.
Кількість наукових публікацій в українських виданнях – 76, у тому числі у фахових виданнях – 28.
Кількість наукових публікацій в іноземних виданнях – 8.
http://orcid.org/0000-0003-4139-2525.

***Місце роботи автора*:**

Київський національний університет імені Тараса Шевченка,
01601, Київ, вул. Володимирська, 60.
Тел.: (044) 2590511 (роб.),
(067) 8879978.
E-mail: me.oksana@gmail.com

УДК 007.3

А.М. Глибовець

АЛГОРИТМ ПОШУКУ ЗВ'ЯЗКІВ І ЗАЛЕЖНОСТЕЙ МІЖ ДАНИМИ WEB-СТОРІНОК

Методи видобування й аналізу даних – відносно нова і перспективна галузь комп'ютерних наук, що знайшла своє застосування в системах інформаційному пошуку. У роботі запропоновано алгоритм пошуку зв'язків і залежностей у колекціях Web-сторінок. Алгоритм не передбачає пошуку релевантних ресурсів. Цю функцію виконує пошукова система. Вона також робить очищення, інтеграцію та вибір даних. Особливістю алгоритму є використання вже існуючого сховища даних (пошукова система або сховище даних), мовна незалежність і простота реалізації.

Ключові слова: пошукова система, PatternRecognition, алгоритм пошуку, зв'язки і залежності.

Вступ

Задачі і методи видобування і розпізнавання Web даних (PatternRecognition) лежать на межі проблематики баз даних і штучного інтелекту. Останнім часом гостро постала потреба у добуванні лише корисної інформації і знань. З'явилася окрема галузь видобування даних Webmining, яка використовує методи DataMining для виявлення і пошуку залежностей у WWW на основі деякого «осмислення» даних [1, 2]. Традиційно виділяють чотири етапи аналізу Web даних (далі – даних): вхідний, попередньої обробки, моделювання, аналізу моделі. З'явилися такі категорії Web-Mining як аналіз використання Web ресурсів (WebUsageMining), видобування Web-структур (WebStructureMining), видобування Web-контенту (WebContent-Mining) [3].

У цій роботі, найбільшу увагу ми приділимо саме *видобуванню* Web-*контенту*. Надалі префікс Web опустимо, розуміючи його присутність із контексту видобування контенту – процес видобування знань із вмісту документів (Web-сторінок). Дані сторінок представляються у вигляді текстової, аудіо, відео інформації, або у деякому структурованому вигляді, наприклад, таблиці чи списку. Оскільки більшість такої інформації є текстовою, для її обробки варто використовувати методи *інтелектуального аналізу тексту* (IAT, TextMining або KnowledgeDiscoveryinText) [4]. Останній включає структуризацію тексту (парсинг, стеммінг), виявлення текстових паттернів, аналіз і представлення кінцевої інформації.

Ключовими задачами IAT є кластеризація текстів, обробка змін у колекціях текстів і пошук, які вирішуються на основі розпізнавання іменованих елементів (сутностей, власних назв, імен), пошуку зв'язків об'єкта, виділення термінології (знаходження ключових слів), автореферування (виділення з тексту змістовної чи оціночної інформації).

Більшість Web-документів представляють текстову інформацію у HTML форматі. Цей формат має багато спеціальних символів розмітки, за допомогою яких можна ідентифікувати корисну інформацію. Проте, навіть спеціальна розмітка Web-сторінки мало впливає на їх структурованість. Звичайний текстовий документ складається з абзаців чи параграфів, тоді як Web-сторінка складається з різних елементів розмітки таких як навігаційна панель, меню, таблиці, заголовки. Тому, стандартні методи інтелектуального аналізу тексту важко застосувати у процесі аналізу даних Web-сторінок.

Здійснюючи запит, користувач зазвичай отримує відповідь у вигляді списку документів, які пошукова система вважає релеватними відповідно до отриманого запиту. Важливим фактором при визначенні релевантності документу є кількість гіперпосилань на даний документ з інших документів, відвідуваність сторінки, кількість раніше здійснених запитів

(схожих з даними за тими чи іншими ознаками), які були здійснені і в яких даний документ був визначений як релевантний. Різна інформація про об'єкт пошуку може міститися в різних документах різної релевантності і для уточнення деякого факту доводиться переглянути велику кількість документів для пошуку взаємозв'язку між інформацією про об'єкт що міститься в кожному з цих документів. Постає задача ефективного пошуку таких взаємозв'язків. Сучасні пошукові системи суттєво просунулись у питаннях визначення релевантності документів, проте вони не мають засобів аналізу інформації для розпізнавання зазначених перехресних зв'язків.

Опишемо запропонований нами алгоритм пошуку зв'язків і залежностей (АПЗЗ) у даних Web-сторінок. Отримавши на вхід запит про деякий об'єкт, пошукова система з АПЗЗ надає на виході окрім інформації про об'єкт і інформацію про його зв'язки з іншими об'єктами.

Основна частина

Визначення основних понять. Запит (далі позначатимемо «Зп») – набір слів (формалізованих або заданих природньою мовою), які описують об'єкт пошуку.

Експерт (користувач) – особа, яка здійснює пошук. Вона зазвичай має певні знання (уявлення) про об'єкт пошуку або його частину.

Неважливі слова (далі позначатимемо «Нс») – слова, словосполучення, терміни і символи, які не дають корисної інформації про об'єкт пошуку. До таких слів можна віднести усі знаки пунктуації, займенники, більшість дієслів, службові частини мови, спеціальні символи розмітки (HTML, XML тощо).

Ключові слова (далі позначатимемо «Кс») – слова і терміни, які несуть смислове навантаження і описують об'єкт пошуку.

Опис об'єкта пошуку здійснюється у запиті експертом, який здійснює пошук. Тоді множина ключових слів є найбільшою підмножиною слів з множини слів запиту, яка не містить неважливих слів:

$$Кс = Зп \setminus Нс.$$

Стоп-слова (далі позначатимемо «Сс») – слова та терміни, що визначаються експертом як такі, що не повністю відповідають його запиту і при будь-яких можливих зв'язках з запитом зменшують релевантність документу, у якому зустрічається стоп-слово, або експерту неважливі зв'язки об'єкта пошуку із словом позначеним як стоп-слово.

Ресурс – документ, що містить деяку текстову інформацію. Ресурсом будемо називати будь-який текстовий документ, Web-сторінку (чи навіть Web-портал).

Пошукова система (ПС) – система, що здійснює пошук ресурсів згідно заданого запиту експерта без їхнього аналізу та пошуку зв'язків.

Сучасні Web-сторінки – це елементи деякого Web-порталу чи Web-застосунку, які несуть велику кількість інформації і оперують дуже об'ємними динамічними і статичними HTML документами. Оскільки HTML представляє інформацію в ієрархічному вигляді і кожен елемент структурно належить якомусь тегові, можна зробити висновок. Інформація, яка має сенс, а також несе деякі знання розміщується в межах одного тегу або в деякій неперервній частині документу HTML розмітки. Тобто, важлива інформація про об'єкт пошуку міститься у безпосередній близькості від ключового слова, за яким здійснювався пошук. Тому можна стверджувати, що для знаходження зв'язків об'єкта пошуку з іншими об'єктами чи явищами, необхідно проаналізувати об'єкти, явища і терміни, які знаходяться в деякому околі від ключових слів пошуку у документі HTML. Зазвичай такі об'єкти несуть додаткову інформацію чи приховані знання про об'єкт пошуку і можуть бути використані в уточнюючих запитах до сховища даних (у нашому випадку пошукової системи), для виявлення наступних зв'язків та прихованих знань.

Підготовча робота

АПЗЗ не передбачає здійснення пошуку релевантних ресурсів. Цим займа-

ється ПС. Вона виконує функції очищення, інтеграції і вибору даних. Робота алгоритму починається з отримання запиту від експерта. Запит може бути заданий як природною мовою, так і іншими, більш формалізованими способами. Система реалізація алгоритму надсилає аналогічний (або нормалізований) запит до ПС, для отримання списку ресурсів, які позначаються ПС як релевантні документи. У відповідь система отримує від ПС список релевантних ресурсів ($D=\{T_1,T_2,...,T_n\}$), серед яких і буде здійснюватись подальший пошук зв'язків і залежностей.

Зрозуміло, що якість роботи алгоритму прямо пропорційно залежить від якості видачі ПС.

Для будь-якого алгоритму обробки текстової інформації важливим є процес нормалізації тексту. Він складається із трьох кроків: розбиття тексту на лексеми та побудова множини усіх лексем; стематизації кожної лексеми; вилучення з отриманої множини усіх неважливих слів (Нс). Після нормалізації отримується множина термінів для подальшого аналізу.

Наступним кроком є аналіз кожного ресурсу для пошуку взаємопов'язаних з об'єктом пошуку понять у визначеному околі.

Побудова списку важливих слів здійснюється наступним чином:

– нехай S_{all} – список пар <термін, кількість> усіх важливих термінів, порожній на початку аналізу;

– для кожного ресурсу T із D, будується нормалізований список термінів S;

– для кожного із термінів запиту, знаходимо його індекс і у списку S;

– для кожного індексу i, будуємо його окіл ($i-n,i+n$), де n – деяка наперед задана константа;

– для кожного індексу з околу ($i-n,i+n$) з множини S, додаємо термін, що знаходиться за цим індексом до множини S_{all} таким чином, що якщо термін вже існує у множині S_{all} збільшуємо значення кількості на одиницю, якщо ні – додаємо пару (термін, 1).

По закінченню S_{all} міститиме терміни, які зустрічаються з ключовими словами найчастіше у деякому околі, що вказує на взаємозв'язок між об'єктом пошуку та знайденими термінами.

Оскільки список S_{all} містить пари (термін, кількість) його можна впорядкувати за кількістю. Чим частіше зустрічається термін у такій множині, тим сильнішим є його зв'язок з об'єктом пошуку.

Як ми зазначали раніше, пошуком релевантних даних займається ПС. Використання списку термінів S_{all}, для побудови уточнюючих запитів , які можуть мати зв'язок з об'єктом пошуку, може мати якісний вплив на покращення релевантності і виявлення нових зв'язків.

Уточнюючі запити – це повторно здійснений запит до ПС, у якому змінився набір ключових слів. Після першої ітерації роботи алгоритму, у розпорядженні експерта постане список термінів, пов'язаних із запитом користувача. Серед таких термів з високою ймовірністю будуть такі, що матимуть якісний вплив на результати пошуку за використанням цих термінів у новому запиті у комбінації з попередньо вказаними ключовими словами. Після уточнюючого запиту і здійснення аналізу ресурсів за тією ж процедурою, що і на першій ітерації, список важливих термінів поповниться новими елементами, які мали зв'язок з об'єктом пошуку. Ітеративний процес можна продовжувати як завгодно довго, в залежності від потреб експерта. Чим більше ітерацій буде здійснено, тим глибші зв'язки вдасться виявити на кожній наступній ітерації.

Опис алгоритму

Формалізований опис алгоритму відображає наступна процедура:

```
procedure FindRelationsAlgo
begin
        define Q; # Query
        define S_all; # Results set
        define res_list;
        define res_amount = 0;
```

```
        define continue = TRUE;
        input(Q);
        S_all = empty_set();
        res_list = empty_list();
        do
            Q = normalize(Q);
            res_list = query_for_resources(Q);
            res_amount = length(res_list);

            for resource in res_list do

    clean(resource);
                define S ;
                define index_list;
                define around_set;
                define N;
                N = 20;
                S = build_term_set(resource);
                index_list = query_index_list( S , Q);
                around_set = build_around_set(S, index_list, N);
                S _all = union_of_sets(S_all, around_set);
            end for
            sort( S _all);
            present_results(S_all);

            input(continue);
            if continue == TRUE
                define k_words;
                input(k_words);
                Q = union_of_sets(Q, k_words);
            end if
        while continue == TRUE;

        return S _all;
    end
    end procedure FindRelationsAlgo
```

Опишемо основні складові алгоритму.

Зрозуміло, що Q – запит експерта (користувача), за яким здійснюється пошук.

Функція ***normalize*** (Q), здійснює нормалізацію запиту користувача, виділення ключових слів, усунення неважливих слів і за необхідності (в залежності від пошукової системи) кожен термін запиту лематизується і стемінгується. На вихід функція видає список ключових слів запиту.

Функція ***query_for_resources*** (Q) здійснює комунікацію зі сховищем даних (пошуковою системою). Отриманий на вхід список ключових слів передається ПС. Після обробки запиту пошуковою системою, функція отримує у список релевантних ресурсів. Зауважимо, що цей список може представлятися у різних форматах (найпоширенішим є формат HTML). Тому наша функція повинна коректно опрацювати отриманий результат і подати на вихід лише змістовний список ресурсів.

Процедура ***clean*** (*resource*) здійснює початкову обробку ресурсу, представленому у деякому форматі (найпоширеніший – HTML). Тому попередньо ресурс слід привести до змістовного вигляду. У процедурі виділяється смислова (текстова) частина ресурсу, видаляються символи розмітки, знаки пунктуації, неважливі слова.

Функція ***build_term_set*** (*resource*) із отриманого на вхід тексту ресурсу подає на вихід список термінів та слів. Важливим є порядок термінів у списку. Слова в отриманому списку проіндексовані і розташовані в такому ж порядку, в якому вони зустрічаються в оригіналі ресурсу.

Функція ***query_index_list*** (S, Q) отримавши на вхід два списки термінів подає на вихід список індексів входження кожного терміну з Q у S.

Функція ***build_around_set*** (S, *index_list*, N) отримавши на вхід S, список індексів та значення радіусу околу N, подає на вихід список термінів (слів), які знаходяться в околі N кожного індексу із списку *index_list.*

Реалізація функції ***build_around_set*** (*S , index_list, N*) на псевдокоді виглядатиме так:

```
function build_around_set ( S , index_list, N )
begin
    define res_list;
    for index in index_list do
        define i = index - N ;
        if i < 0
            i = 0;
        end if
        while i < index + N AND i < length(index_list) do
            if i != index

add_to_list(res_list, S [i]);
            end if
        end while
    end for
    return res_list;
end
end function build_around_set
```

Звернемо увагу на значення аргументу *N* . Цей параметр ініціалізовано константою і не змінюється в процесі аналізу. Від підбору аргументу *N* залежить розмір результативного списку. Надто мале *N* призведе до того, що кількість термінів, які потраплять у результативну множину буде малою, і з високою ймовірністю, якість проведеного аналізу буде невисокою, оскільки велика кількість зв'язаних термінів проігнорується, не потрапивши в результативний список. З іншого боку, використання надто великого радіусу околу значно збільшить часові затрати роботи алгоритму, але результати суттєво не зміняться. Результати тестування показали, що оптимальними є значення з проміжку 20–35.

Функція ***union_of_sets*** *(S _all, around_set)* виконує роль об'єднання результату.

Для тестування алгоритму побудовано програмну систему у вигляді клієнт-серверного Web застосунку (Web-сайту), серверна частина якого відповідала за аналіз і роботу самого алгоритму, а користувачу надавалася Web-сторінка з отриманими даними. Інтерфейсом доступу та пошуку релевантних даних служила пошукова система Google. Мовою програмування виступав *Python*.

Існує багато OpenSource бібліотек та інструментів розроблених мовою Python, що реалізують додаткові функції для роботи з текстом, HTTP запитами, мережею, зображеннями, інтерфейсом користувача. Однією з таких бібліотек є бібліотека *Grab*. Основними функціями цієї бібліотеки є підготовка мережевого запиту (cookies, http-заголовки), відправлення запитів, отримання відповідей сервера та їх попередня обробка, робота з DOM-деревом отриманої у відповідь сторінки [5]. Ми використали Grab для роботи з контентом Web сторінки, а саме для отримання текстової інформації із сторінки та очищенням її від HTML тегів.

Для реалізації Web-серверу ми скористалися найпоширенішим Python Framework для розробки Web-систем Django, дистрибутив якого містить також і Web-сервер [6].

Представлення отриманих результатів – важлива складова будь-якої системи видобування даних. У нашому випадку, отримані результати представлені у вигляді HTML сторінок.

Результати тестування

Реалізована система показала хороші результати роботи.

Середня кількість понять, що є загальними і не являють собою приховані знання становить приблизно 30 – 40 % , а обсяг даних, що були прихованими до обробки запиту системою і були представлені у таблиці результатів відповідно становили 70 – 50 % з усіх отриманих даних в таблиці результатів.

Важливо зазначити, що якість роботи системи залежить від загальної кількості даних про об'єкт пошуку, що містяться у сховищі даних. Так при побудові профайлів користувачів, які є публічними особами кількість термінів, що викривають нові знання про об'єкт серед усіх отриманих результатів висока (понад 60 %), а для осіб, які не є публічними, а значить кількість інформації у сховищі даних про та-

ких осіб менша, результати дещо нижчі (40 – 50 %).

Як бачимо, система успішно здійснює пошук зв'язків і залежностей. Після здійснення користувачем додаткових запитів, у відповідь система надає нові зв'язки та нову інформацію про об'єкт пошуку.

Висновки

Описаний алгоритм дозволяє здійснювати ефективний пошук зв'язків між об'єктом пошуку і даними Web-сторінок. Найсуттєвішою перевагою алгоритму є використання вже існуючого сховища даних. АПЗЗ використовує як сховище даних будь-яку існуючу ПС (або сховище даних), до яких існує доступ. Це дозволяє скоротити часові затрати і уникнути дублювання розробки уже існуючих систем. Другим привабливим фактором алгоритму є мовна незалежність. АПЗЗ дуже мало залежить від мови, що використовується користувачем. Навіть якщо не здійснювати обробку природної мови у процесі виконання алгоритму, результати роботи практично не змінюються. Часто обробку природних мов здійснює сама пошукова система. На кінець маємо відмітити і простоту реалізації.

Недоліками алгоритму є націленість на роботу з Web-сторінками. Деякою мірою алгоритм залежить від розміщення даних у ресурсі.

Суттєвою є і залежність від роботи пошукових систем. Алгоритм залежить як від якості пошуку, так і від швидкодії. Низька релевантність пошуку спричинить низьку якість результатів. Багато пошукових систем не дозволяють здійснювати велику кількість автоматичних запитів, що, за дуже складних аналізів може призвести до неправильного результату.

Особливості реалізації соціальних мереж та великих порталів, на яких зберігається корисна інформація, зокрема про їхніх користувачів, часто не дозволяють здійснити аналіз АПЗЗ. Часто у відповідь на запит до таких порталів отримується закодована сторінка (наприклад, JavaScript код), який не піддається аналізу. З іншого боку, кожна з великих соціальних мереж надає спеціальні інструменти для роботи з даними, що розміщені у мережі. Використання таких інструментів може якісно вплинути на роботу алгоритму, особливо при пошуку зв'язків деякої особи з іншими об'єктами, особами і даними.

Шляхами удосконалення алгоритму бачиться покращення роботи з соціальними мережами і побудова дерева залежностей на кожному кроці взаємодії із ПС. Побудова деякого дерева, на кожному рівні якого зберігатимуться нові знайдені зв'язки допоможе відповісти на питання: як саме, або за рахунок чого об'єкт має зв'язок з іншим.

Програмна реалізація алгоритму показала хороші результати роботи. Окрім посилань на сторінки користувача, який є об'єктом запиту, у соціальних мережах користувач системи отримує таблицю із зв'язками (профайл) об'єкта пошуку з іншими об'єктами. Середнє відношення корисних даних у результативній таблиці складає 55 – 65 %.

1. *Глибовець М.М., Глибовець А.М., Поляков М.В.* Інтелектуальні мережі // Навчальний посібник, Дніпропетровськ, Нова ідеологія, 2014. – 464 с.
2. *Глибовець М.М., Жигмановський А.А., Заболотний Р.І., Захоженко П.О.* Веб сервіси оброблення документів // Національний університет "Києво-Могилянська академія". – К.: НаУКМА, 2012. – 212 с.
3. *Глибовец А.Н., Глибовец Н.Н., Покопцев Д.Е., Сидоренко М.О.* Структурированные данные и семантическая паутина: технологии Wiki // Проблеми програмування. – 2013. – № 1. – С. 45–67.
4. *Петренко А.І.* Grid і інтелектуальна обробка даних [Електронний ресурс]. – Режим доступу: http://datamining.netallted.cad.kiev.ua/downloads/DataMining.pdf
5. *grab* 0.6.29 : Python Package Index [Електронний ресурс] – Режим доступу: https://pypi.python.org/pypi/grab/0.6.29 – 2015 р.
6. *Django*: The Web framework for perfectionists with deadlines [Елекронний ресурс] – Режим доступу: https://www.djangoproject.com/

References

1. *Glybovets M.M., Glybovets A.M., Poliakov M.V.* Intellectual networks, Dnipropetrovsk, newideology. – 2014. – 464 p.
2. *Glybovets M.M., Jigmanovskiy A.A., Zabolotniy R.I., Zahojenko P.O.* Webservices for documents processing. – K.: National university of "Kyiv-Mohyla academy". 2012. – 212 p.
3. *Glybovets M.M Glybovets A.M., Pokoptsev D.E., Sidorenko M.O.* Structured data and thesemantic web // Problems of programming. – 2013. – N 1. – P. 45–67.
4. *Petrenko A.I.* Grid and intelligent data processing [online] Available from: http://netallted.cad.kiev.ua/downloads/DataMining.pdf. [accesed: 2008]. – 2008.
5. *grab* 0.6.29 Python Package Index. [online] Available from: https://pypi.python.org/pypi/grab/0.6.29. [accesed © 1990–2015]. – 2015.
6. *Django* The Web framework for perfectionists with deadlines. [online] Available from: https://www.djangoproject.com/. [accesed: © 2005–2015]. – 2015.

Одержано 16.12.2015

Про автора:

Глибовець Андрій Миколайович,
кандидат фізико-математичних наук,
доцент кафедри мережних технологій
Кількість наукових публікацій в українських виданнях – 28.
Індекс Гірша – 3.
http://orcid.org/0000-0003-4282-481X

Місце роботи автора:

Національний університет
«Києво-Могилянська академія»,
04655, Київ, вул. Г. Сковороди 2.
Тел.: (044) 463 6985.
E-mail: andriy@glybovets.com.ua

УДК 681.3

І.Ю. Гришанова

АНАЛІТИЧНИЙ ОГЛЯД МЕТОДІВ І ЗАСОБІВ ІНФОРМАЦІЙНОГО ПОШУКУ В SEMANTIC WEB

В статті надаються та аналізуються методи і засоби інформаційного пошуку в середовищі Semantic Web. Надаються базові поняття інформаційного пошуку, задачі, моделі та класифікація систем інформаційного пошуку за різними ознаками. Наводяться приклади існуючих сучасних пошукових систем, а також надається перелік ознак 3-х поколінь пошукових систем. Запропонована модель інформаційного пошуку в новому середовищі Semantic Web та Web речей розширює класифікацію пошукових систем та модель пошуку з урахуванням можливості пошуку нових об`єктів, доступних по інтернету, та використання знань, що подані в Semantic Web.

Ключові слова: інформаційний пошук, семантичний пошук, пошукові системи, Semantic Web.

Вступ

Філософське і історичне визначення інформаційного пошуку. Важливість персоніфікації у процесі пошуку

Information retrieval (IR) is finding material (usually documents) of an unstructured nature (usually text) that satisfies an information need from within large collections (usually stored on computers) [1].

Інформаційний пошук є процес знаходження матеріалу (зазвичай документів) неструктурованої природи (частіше текстів), які задовольняють інформаційній потребі, у великих колекціях (зазвичай, які зберігаються на комп`ютерах).

Класичне поняття інформаційного пошуку (IR – *information retrieval*, ІП) базується на задоволенні потреби користувачів у пошуку інформації, тобто інформаційної потреби (information need). Класичне визначення інформаційного пошуку базується на підставі того факту, що користувач спонукається інформаційною потребою.

В найбільш загальному сенсі під ***інформаційною потребою*** розуміється необхідність в інформації, яка потребує задоволення і зазвичай виражена в інформаційному запиті. Наприклад, планування поїздки формує інформаційну потребу вивчити розклад руху поїздів та іншого транспорту. Такий процес може бути виконаний різним чином – за допомогою телефону, безпосередньо в касах, в агенстві з продажу квитків, або за допомогою пошуку та сайту в Інтернеті. Однак незалежно від форм задоволення інформаційної потреби, сама по собі вона залишається невідмінною.

Необхідно зазначити, що коли необхідний маршрут обрано та білети вже придбані, ця інформація втрачає свою цінність для користувача, при цьому вона залишається цінною для інших потенційних споживачів. Така властивість повної втрати цінності інформації (її споживацької вартості) для певного споживача в певний момент, є важливою особливістю інформаційної потреби, що суттєво відрізняє її від інших видів потреб людини. Одна й та сама інформація знов може стати предметом споживання в випадках, якщо вона буде надана іншому споживачеві, або якщо перед тим самим споживачем знов стане така сама задача, або якщо запас знань споживача зросте, що дозволить йому побачити в цій інформації нові аспекти.

Таким чином, інформаційні потреби мають суто індивідуальний (персональний) характер. Вони залежать не тільки від особливостей задач, що вирішуються, але й від психологічних, освітніх та інших особистих відмінностей особи, що приймає рішення.

Зазвичай виділяють два основних типи інформаційних потреб:

- поточні, які зумовлені притаманною людині допитливістю і які виражаються в його прагненні бути в курсі усьо-

го, що відбувається в світі;

- конкретні (спеціальні), які виражаються в прагненні отримати інформацію, необхідну для вирішення конкретної задачі – дослідницької, професійної, управлінської тощо [2].

Основна мета задачі інформаційного пошуку – допомогти користувачу знайти інформацію, яка йому необхідна. Процес інформаційного пошуку в загальному вигляді включає в себе послідовність операцій, які направлені на збір, обробку і надання необхідної інформації зацікавленим особам. Процес інформаційного пошуку складається з наступних етапів:

- визначення (уточнення) інформаційної потреби і формулювання інформаційного запиту;
- визначення сукупності можливих інформаційних джерел;
- вилучення інформації з виявлених інформаційних джерел;
- ознайомлення з отриманою інформацією і оцінювання результатів пошуку.

Базовими поняттями оцінювання ефективності пошуку є **релевантність** та **пертинентність.**

Вирішальною умовою ефективного задоволення інформаційної потреби є чітке усвідомлення і чітке вираження того, яка інформація насправді потрібна споживачеві для вирішення поставленого перед ним завдання. Без цього важко розраховувати на отримання релевантного та пертинентного результату.

З моменту виникнення у людини інформаційної потреби, він починає оцінювати всю інформацію, що надходить до нього, під кутом зору цієї потреби, розділяючи цю інформацію на релевантну і нерелевантну. Іншими словами, інформаційна потреба виникає у людини при постановці перед нею якогось завдання. Людина обмірковує цю задачу, в результаті чого у нього в мозку складається образ задачі, або її модель. Цей образ і служить еталоном, з яким порівнюється вся подальша інформація, що надходить. Якщо інформація має відношення до еталону, вона вважається доречною. Все, що не має відношення до еталону – вважається **нерелевантною** інформацією.

Під впливом міркувань над сутністю поставленої задачі та вмістом релевантної інформації, що накопичується, уява людини про цю задачу може уточнюватися та змінюватися. Психологи називають такий процес зростанням стану поінформованості про завдання.

Коли людиною накопичено необхідну кількість інформації і виконано деякий міркувальний процес, вона знаходить рішення задачі. Після цього вся інформація, що пов’язана з рішенням задачі, переміщується в зону архівного зберігання. Таким чином, інформаційна потреба може бути охарактеризована як усвідомлена потреба в інформації, яка необхідна для вирішення поставленої задачі за розробленим планом.

Можливо припустити, що процес вирішення будь-якої наукової задачі починається з прийняття будь-яких передумов і припущень, які в подальшому піддаються коригуванню і зміні. Під образом чи моделлю завдання слід розуміти гіпотезу, яка є важливим засобом організації наукового пошуку.

Вчення про психологічні установки дозволяє пояснити поняття пертинентності, яке є одним з ключових понять теорії інформаційного пошуку. Під **пертинентністю** розуміється відповідність знайдених документів або відомостей справжній інформаційній потребі вченого або спеціаліста, яку він нерідко сам може ясно не усвідомлювати.

З запропонованої інтерпретації сутності інформаційної потреби та механізму її задоволення випливає, що віднесення інформації, що надходить до людини, до категорії релевантної чи нерелевантної, повністю визначається тим, який образ поставленої задачі склався у даної людини. Сам цей образ залежить, принаймні, від трьох наступних факторів:

- інформації, яка вже накопичена людиною в її пам’яті;
- обраного шляху рішення задачі;
- темпів і проміжних результатів рішення.

Ще раз необхідно зазначити, що образ завдання під впливом інформації, що надходить, та проміжних результатів рішення цієї задачі, уточнюється або навіть змінюється. У зв'язку з цим змінюються і ознаки, за якими розпізнається і відбирається релевантна інформація. Тому для адекватного інформаційного обслуговування фахівців необхідно, щоб процес пошуку був не тільки індивідуальним, але й включав у себе постійний зворотній зв'язок для своєчасного урахування змін у його інформаційній потребі.

Базові поняття інформаційного пошуку

Основним засобом передачі інформації у часі і просторі є документ. ***Документ*** визначається як засіб закріплення будь-яким чином на спеціальному матеріалі будь-якої (деякої) інформації про факти, події, явища об'єктивної дійсності і розумової діяльності людини [3]. Документи мають різну форму подання. В автоматизованих інформаційно-пошукових системах це текстова інформація на природній мові. В повсякденному житті – це може бути друкована стаття, книга тощо. В Інтернет це може бути рисунок, відео-ролик або сайт.

З точки зору теорії інформації ***документ*** – це змістовно закінчена одиниця інформації, яка представлена на якій-небудь природній мові, що ідентифікується унікальним чином.

Поняття інформаційного пошуку вперше запровадив в інформатиці американський математик Келвин Муерс в 1947 році. ІП називається деяка послідовність операцій, яка виконується з метою знаходження документів, які містять певну інформацію (з подальшою видачею цих документів або їх копій), або з метою видачі фактичних даних, які надають відповіді на задані питання.

Спонукальним приводом інформаційного пошуку, як було зазначено вище, є інформаційна потреба, яка виражена у формі інформаційного запиту. Об'єктами інформаційного пошуку можуть бути документи, відомості про їх наявність та/або місцезнаходження, фактографічна інформація.

Інформаційний запит представляє собою інформаційну потребу, яка сформульована на природній мові. Результат «перекладу» інформаційного запиту на інформаційно-пошукову мову (ІПМ) називають ***пошуковим образом запиту*** (ПОЗ). Синтаксис і семантика ІПМ визначається структурою і наповненням документів та загальними задачами системи.

Інформаційний пошук розрізняють наступним чином:

- в залежності від мети – адресний пошук (формально-механічний) та семантичний (тематичний);
- в залежності від об'єкта пошуку – документний та фактографічний;
- в залежності від ступеня використання технічних засобів – ручний або автоматизований;
- в залежності від функціональної ролі – домінуючі/другорядні, центральні /периферичні, сталі/ситуаційні потреби.

Усі види інформаційного пошуку перетинаються, тому що цілі та об'єкти часто взаємопов'язані. Наприклад, документний і фактографічний види пошуку можуть бути як адресними, так і семантичними.

Інформаційний пошук здійснюється за допомогою інформаційно-пошукових систем. ***Інформаційно-пошукова система*** (ІПС) – це комплекс пов'язаних між собою окремих частин, який призначений для виявлення в будь-якій множині елементів інформації, які відповідають заданому інформаційному запиту. Масив елементів інформації, в якому виконується інформаційний пошук, називається *пошуковим масивом*.

Інформаційно-пошукові системи розділяються на *документальні* та *фактографічні*. Документальні ІПС у відповідь на запит видають оригінали, копії або адреси місцезнаходження документів, що містять потрібну інформацію. Підклас документальних ІПС, які видають лише бібліографічні описи документів, що знайдені, іноді називаються бібліографічними ІПС.

На відміну від документальних ІПС фактографічні пошукові системи призначені для видачі безпосередньо необхідної інформації (наприклад, температури кипіння якоїсь рідини, температури води в морі біля конкретного населеного пункту; структурних або молекулярних формул хімічних сполук, що мають певні властивості тощо).

Принципової відмінності між документальними і фактографічними ІПС немає. Головною ознакою, що поєднує документальні і фактографічні ІПС до одного загального класу є те, що на запити вони можуть видавати таку й тільки таку інформацію, яка була раніше в них введена.

Кожна документальна ІПС (як ручна, так і автоматизована), містить наступні частини:

- ІПМ;
- правила перекладу текстів документу і запитів з природної мови на ІПМ;
- формальні правила (алгоритми) пошуку;
- технічні засоби, які реалізують алгоритми пошуку;
- масив (множина) документів (або їх адрес), які записані на якихось носіях інформації (в сучасних пошукових системах Інтернету – база індексу).

Інформаційний пошук здійснюється за певними правилами, які визначають стратегію пошуку, тобто способи досягнення оптимального результату. Стратегія інформаційного пошуку залежить від типу пошукової задачі, критеріїв видачі і характеру діалогу між споживачами інформації і ІПС.

В загальному вигляді процедура інформаційного пошуку складається з чотирьох етапів:

- уточнення інформаційної потреби і формулювання запиту;
- визначення сукупності інформаційних масивів;
- вилучення інформації з інформаційних масивів;
- ознайомлення користувача з отриманою інформацією і оцінювання результатів пошуку.

Найбільш загальний вигляд алгоритму пошуку, що проводиться незалежно від форми носіїв і ступеня автоматизації, показаний на рис. 1.

Рис. 1. Загальний вигляд алгоритму пошуку

Постановка пошукової проблеми. На цьому етапі користувач формулює точне визначення і фіксує те, що буде шукати і в якій області знань (предметній області – ПрО). Таким чином множина пошуку звужується визначеними межами.

Створення тезаурусу проблеми. На цьому етапі користувач створює (складає) перелік слів, які найбільш повно відображають ПрО або проблему, що була визначена. Як рекомендують спеціалісти з бібліографічного пошуку, цей перелік повинен мати приблизно 10–15 слів.

В залежності від поставленого завдання тезаурус може бути складений на декількох мовах, для пошуку серед вітчизняних та зарубіжних джерел інформації. Робота над тезаурусом ведеться весь час, і в процесі виявлення нових термінів вони

тут же додаються до тезаурусу. Найбільш прийнятною є структура тезаурусу у вигляді семантичних зрізів. У цьому випадку для кожного основного терміну окремо будується таблиця для супутних та шумових слів. Шумових слів у джерелі бути не повинно. Тобто користувач отримує пакет таблиць, які можна окремо розширювати і модифікувати в ході пошуку.

Відбір джерел інформації для пошуку. Джерела інформації (масив) обираються виходячи з характеру проблеми (тобто де найбільш доступні та повно надані джерела) та можливостей користувача (доступ до Інтернету, бібліотеки тощо).

Виконання пошуку засобами, які притаманні джерелу інформації. На цьому етапі користувач з тезаурусу складає пошукові запити і реалізує їх методами пошуку, які специфічні для даного ресурсу. В бібліотеці – це пошук в каталогах, якщо інформацією володіють люди або організації – пошук та звернення до них, у мережі Інтернет – використовуються пошукові машини та каталоги, телеконференції та списки розсилки, сайти та інше. Як формат, так і семантика запитів варіюється в залежності від предметної області та використовуваного інформаційного ресурсу.

Як рекомендують спеціалісти з бібліографічного пошуку, запити необхідно складати таким чином, щоб область пошуку була максимально конкретизована та звужена. Необхідно віддавати перевагу декільком вузьким запитам ніж одному, але розширеному. В загальному випадку для кожного основного поняття з тезаурусу готується окремий пакет запитів. Після чого проводиться пробне виконання запитів – для уточнення та доповнення тезаурусу, в тому числі для відсікання шумової інформації.

Оцінювання отриманих результатів пошуку. В результаті пошуку користувач отримує результативну множину документів, які надалі необхідно проаналізувати і вирішити наскільки повно вони покривають поставлену пошукову проблему.

Перелік ресурсів, отриманих у результаті запиту, рекомендується обробляти в два етапи. На першому етапі відсікаються вочевидь нерелевантні джерела і знову ж таки проводиться семантичний аналіз з метою уточнення тезаурусу та модифікації подальших запитів. На другому етапі обробки користувач послідовно вивчає кожен з знайдених ресурсів для безпосереднього аналізу інформації, що знаходиться в них. У процесі аналізу отриманої інформації, її треба:

- оцінити (за ступенем вірогідності, важливості, таємності, пов'язаності між собою, можливості використання);
- інтерпретувати (в світлі інших даних і глибинної інтуїції), виявивши її місце в загальній мозаїці фактів;
- визначити, чи потрібна (і яка) додаткова інформація;
- ефективно використати (врахувати у своїх планах, передати кому слід, притримати до потрібного моменту).

Прийняття рішення про продовження (закінчення) пошуку. Якщо, оцінюючи результати пошуку, користувач прийшов до висновку, що необхідна інформація знайдена вся, тоді пошук можна припиняти – подальші пошуки будуть зайвою тратою дорогоцінного часу. У зворотній ситуації (неповні відомості) користувачеві доведеться приймати рішення про те, на якому з етапів була допущена помилка, і спробувати виправити її, після чого повторити процес пошуку з цього місця заново. В цьому випадку можливі три варіанти: неправильно складений тезаурус проблеми, неправильно обране інформаційне джерело або користувач скористався недоцільними методами пошуку (наприклад, виконував пошук суто наукової інформації – статті за допомогою загально використовуваного пошукового Інтернет-сервісу). Такі ітерації необхідно повторювати, поки не буде досягнуто позитивного результату. При цьому існує стовідсотково методологічна проблема – при ефективному пошуку завжди стоять два суперечливих завдання: збільшення охоплення з метою отримання максимальної кількості значимої інформації та зменшення охоплення з метою мінімального обсягу шумової інформації. І най-

складніше, як завжди, знайти золоту середину [4].

Найбільш ефективним методом пошуку документів, які містять наукову інформацію є вивчення (прочитання) кожного окремого документу. Зрозуміло, що такий спосіб є практично неможливим, оскільки кількість документів, як правило, буває занадто великим, щоб всі їх можна було прочитати при кожному інформаційному запиті. Тому доводиться використовувати інший, менш ефективний метод, при якому ІП здійснюється не за самими текстами документів (умістом), а за короткими характеристиками змісту або певними зовнішніми ознаками документів. Для цього кожен документ забезпечується ***пошуковим образом документа*** (ПОД) – характеристикою, в якій стисло виражається основний зміст документу. Як було зазначено вище, інформаційний запит також має бути сформульований у вигляді такої ж короткої характеристики – ПОЗ. Завдяки цьому процедура ІП зводиться до зіставлення ПОД з заданим ПОЗ. Якщо ПОД з необхідною і достатньою мірою збігається з ПОЗ, вважається, що цей документ відповідає на інформаційний запит. Таке зіставлення виправдане лише тоді, коли пошуковий образ і пошуковий запит формулюються в термінах однієї мови, та ще такого, в якому кожна фраза допускає одне й тільки одне тлумачення.

ПОД містить загальний опис змісту документа. Тому такий метод не може забезпечити знаходження в бібліотеці всіх документів, які містять потрібну інформацію. Крім того, в масиві знайдених документів можуть бути такі, що фактично не відповідають даному інформаційному запиту. Такі документи створюють “пошуковий шум”.

Важливо пам’ятати, що інформація, яка міститься в наукових документах, об’єктивно підпорядковується закону розсіювання. Повнота і точність пошуку являють собою конкуруючі показники: підвищення одного з них веде до зниження іншого. При збільшенні повноти пошуку, ми неминуче зменшуємо його точність і, навпаки, збільшуючи точність пошуку, зменшуємо його повноту.

Ефективність інформаційного пошуку визначають показники, які характеризують знаходження релевантних документів. Вони підрозділяються на семантичні (*точність* та *повнота пошуку, коефіцієнт інформаційного шуму, коефіцієнт втрат* тощо) та техніко-економічні (оперативність пошуку, вартість та трудоємність пошуку).

Відповідність знайдених у процесі інформаційного пошуку знань або даних інформаційній потребі користувача (в особовому випадку – інформаційному запиту) називається ***пертинентністю***. Змістовна відповідність відображуваного результату його запиту за формальними (синтаксичними, морфологічними) ознаками називається ***релевантністю***.

З проблемою інформаційного пошуку першими зіткнулися бібліотекарі. Для того, щоб читачі могли знаходити в фондах бібліотеки документи, які їх цікавлять, в ній створювалися різні каталоги та вказівники. В одній з найбільших бібліотек давнини – в Александрійській бібліотеці – в 47 р. до н. е. нараховувалось біля 700 тис. томів (свитків папірусу). Складений Калімахом каталог до фондів цієї бібліотеки (приблизно в 250 р. до н. е.) мав обсяг 120 томів. Як основні елементи книгоопису в цьому каталозі використовувалися ім’я автора та назва (заголовок) твору. Якщо твір не мав назви, то Калімах приводив його початкові рядки.

Простішим ПОД є його заголовок. Спираючись на заголовок книги або статті читач у більшості випадків може судити про те, чи представляє для нього інтерес ця книга або стаття і чи варто з нею ознайомитися досконало.

Анотацію та реферат документу також можна вважати його пошуковими образами. Із збільшенням обсягу реферативних журналів кількість анотацій та рефератів, що містяться в них, стало настільки великим, що реферативні журнали довелося забезпечувати додатковим довідковим апаратом – системою покажчиків, які значно полегшують для читачів рішення інформаційно-пошукових задач. Таким чином, реферативні журнали, а також реферативні журнали з системою по-

кажчиків – це найпростіші документальні ІПС, розраховані на індивідуальне використання.

Існує три основних **типи інформаційно-пошукових задач**:

- ретроспективний інформаційний пошук, тобто пошук вже існуючих документів (всіх або частини), які містять відомості про певне питання;
- термінове сповіщення окремих спеціалістів (абонентів) про публікації, які мають для них потенційний інтерес. Даний тип інформаційного пошуку називається виборчим (адресним) розподілом інформації. Він виконується за постійними інформаційними запитами (так званими "профілями інтересів"), які формуються самими споживачами. Це окремий випадок інформаційного пошуку;
- пошук імен спеціалістів, які володіють інформацією з певного питання.

З розвитком Semantick Web та Web речей, цей перелік розширюється можливістю пошуку інформаційних об'єктів, які доступні за допомогою Інтернету.

2. Класична модель інформаційного пошуку

Базова стандартна модель, яка використовується в більшості книг з інформаційного пошуку виглядає, як показано на рис. 2 [5].

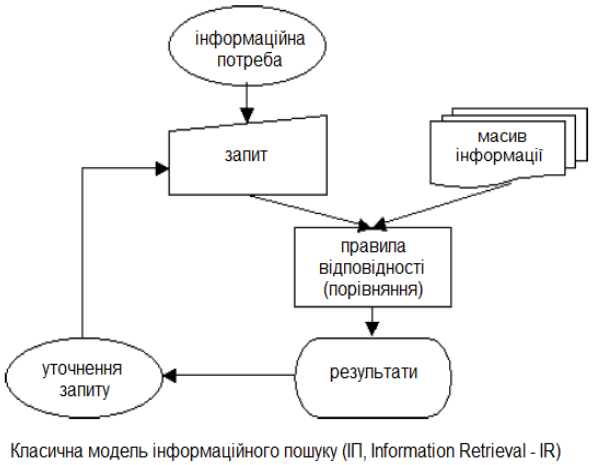

Рис. 2. Класична модель інформаційного пошуку

Як було зазначено вище, користувач, спонуканий інформаційною потребою, складає запит на деякій мові запитів. Запит посилається системі, яка вибирає з колекції документів (масив інформації) такі документи, що відповідають запиту згідно з визначеними правилами відповідності. Процес уточнення запиту може використовуватися для створення нових запитів та/або для очищення результатів.

Процес пошуку базується на використанні визначеної моделі пошуку. Модель пошуку характеризується наступними параметрами:

– форма подання документів і запитів;

– критерій змістовної відповідності;

– методи ранжування результатів запитів;

– механізм зворотнього зв'язку для оцінювання релевантності документів.

Наведемо стисло класичні моделі інформаційного пошуку:

- булева модель;
- ймовірнісна модель;
- векторна модель;
- дескрипторна модель та моделі, базовані на класифікаторах.

Булева модель. В цій моделі документ подається за допомогою набору термінів, які зберігаються в індексі. Кожен термін представлений як булева змінна. Документ (ПОД) подається як поєднання термінів. Вагові коефіцієнти не вводяться. Запит (ПОЗ) формується як довільний булевський вираз, що складається з термінів, пов'язаних логічними операціями (AND, OR, NOT). Мірою відповідності є значення статусу виборки (TRUE або FALSE). Така модель проста в реалізації і використовується в багатьох документальних ІПС. Ефективність пошуку невисока і неможливо ранжування документів за релевантністю.

Ймовірнісна модель. В основі ймовірнісних моделей лежить принцип його ранжування (Probabilistic Ranking Principle, PRP). Цей принцип заключається в наступному – найбільш загальна ефективність пошуку досягається у випадку, коли результативні документи ранжують-

ся за убуванням ймовірності їх релевантності запиту. Спочатку для кожного документу оцінюється ймовірність того, що він релевантний запиту, а потім за цими оцінками виконується ранжування документів.

Для отримання таких оцінок існують різні способи, а також додаткові допущення та гіпотези, які створені на основі апріорних відомостей про документи колекції. Відповідно до цього існує багато реалізацій ймовірнісної моделі пошуку. Наприклад, така оцінка може бути обчислена у відповідності з теоремою Байєса за деякою функцією ймовірностей входження термів даного документу в релевантні та нерелевантні документи. Використовуючи навчальну вибірку (навчальний масив даних) обчислюється ймовірність входження заданого терму в релевантні та нерелевантні документи [6].

Просторово-векторна модель (Vector Space Model) запропонована Солтоном в 1975 році, але на даний час має велике поширення. Векторні моделі, на відміну від булевих, дозволяють ранжувати результативну множину документів запиту. Документи (та запити до них) представляють собою набір векторів у n-мірному просторі [7]. Простір містить n базисних нормалізованих векторів, де n – загальна кількість різних термів в усіх документах. Значення компонентів вектора визначає вага терму (терміну). Показник відповідності (релевантності) визначається як оцінка кореляції між векторами. Така кореляція може бути скалярним добутком (множенням) вектора запиту на вектор документу [8]. Документи ранжують за спаданням скалярних добутків.

Дескрипторна модель є найпростішою моделлю пошуку. В ній документ задається у вигляді набору, асоційованих з ним зовнішніх атрибутів. У простих системах дескрипторного пошуку подання документу описується сукупністю слів або фраз лексики предметної області (ПрО), які характеризують зміст документа. Ці слова і словосполучення називаються дескрипторами. Індексування документу в таких системах реалізується призначенням для нього сукупності дескрипторів. При цьому дескриптори можуть приписуватися документу як на підставі його змісту, так і на підставі його назви. Такі два процеси називаються відповідно індексуванням документу за змістом та індексуванням за назвою [9]. В деяких дескриптивних системах індексування документів здійснюється вручну експертами ПрО, в інших воно виконується автоматично.

Дескрипторні системи можна віднести до класу систем, орієнтованих на бібліографічний пошук або пошук у каталозі.

Моделі, базовані на класифікаторах – є однією з різновидів найпростіших моделей пошуку. Документ у цій моделі, як і в дескриптивних системах, подається у вигляді сукупності асоційованих з ним атрибутів. Атрибутами є ідентифікатори класів, до яких відноситься даний документ. Класи формують ієрархічну структуру класифікатора. Запит може бути представлений двома способами:

– простий варіант, коли запитом є ідентифікатор будь-якого класу з заданого класифікатора. Критерій релевантності документу запиту – клас документу збігається з класом, поданим у запиті, або є його підкласом;

– складний варіант – в запиті можна вказати кілька класів класифікатора. Критерій релевантності документу запиту – клас документу збігається з будь-яким із зазначених у запиті класом, або є його підкласом.

Моделі, базовані на класифікаторах, близькі до булевських моделей.

Необхідно зазначити, що класичні моделі розглядають незалежність слів (термів). Для подання документів та запитів застосовується одразу декілька моделей.

Ефективність пошуку (інформаційно-пошукових систем) аналізується і регулюється перш за все за рівнем релевантності й пертинентності в частині вдосконалення організації запитів користувачів, пошуку за параметрами, за рахунок кластеризації, пошуку за подобою, ранжуванням відгуків, використання «сюжетних підходів», всебічного використання семантичних методів (у тому числі із застосу-

ванням автоматичного групування документів за класифікатором, автоматичним визначенням раніше незаданих або слабо структурованих документів, ранжування документів за змістовою релевантністю, автоматичного аналізу та змістовного перетворення запитів, виявлення семантично подібних документів на зразок порівнянню з еталоном, наприклад, з використанням матриці Александера).

3. Типи пошуку

Інформаційний пошук можна розділити на наступні види:

- ***повнотекстовий пошук*** – при цьому здійснюється пошук в усьому змісті документу. Прикладами повнотекстового пошуку є більшість пошукових систем Інтернету, як Yandex, Google тощо. Зазвичай, для прискорення пошуку повнотекстовий пошук використовує попередньо створені індекси (індексну базу);
- ***пошук за метаданими*** – це пошук за деякими атрибутами документу, які підтримуються системою. Наприклад, назва документу, дата створення, розмір, автор тощо. Прикладом пошуку за реквізитами є діалог пошуку в файловій системі (наприклад, в ОС MS Windows). Цей пошук зазвичай використовує дескриптивну модель пошуку;
- ***пошук зображення*** – це пошук за вмістом зображення. Пошукова система зазвичай використовує алгоритми штучного інтелекту – порівняння за зразком та пошуку за подібністю;
- ***пошук музики*** – аналогічно пошуку зображення, виконує пошук за зразком у колекції музичних даних;
- ***пошук інформаційних об'єктів*** здійснюється в середовищі Web речей; виконує комбінований пошук інформаційних об`єктів, що доступні в Інтернет, з використанням мета-описів цих об'єктів та з урахуванням типу об'єкта.

4. Класифікація видів пошуку

Адресний пошук. Процес пошуку документів здійснюється за суто формальними ознаками, які вказані у запиті. Для здійснення такого типу пошуку необхідні наступні умови:

– наявність у документі точної адреси;

– забезпечення суворого порядку розташування документів у запам'ятовуючому пристрої або в сховищі системи.

Адресами документів можуть бути адреси Web-серверів та Web-сторінки, елементи бібліографічного запису, адреси зберігання документів у сховищі.

Документальний пошук. Процес пошуку здійснюється в сховищі інформаційно-пошукової системи первинних документів або в базі даних вторинних документів, що відповідають запиту користувача.

Існує два різновиди документального пошуку:

– бібліотечний, який спрямований на знаходження первинних документів;

– бібліографічний, який спрямований на знаходження відомостей про документи, які подані в вигляді бібліографічних записів.

Фактографічний пошук. Процес пошуку полягає у пошуку фактів, які відповідають інформаційному запиту. До фактографічних даних відносяться відомості, які добуті з первинних або вторинних документів, або які отримані безпосередньо з джерел їх виникнення.

Розрізняють два підвиди фактографічного пошуку:

– документально-фактографічний, який полягає у пошуку в документах фрагментів тексту, які містять факти;

– фактологічний (опис фактів), який припускає створення нових фактографічних описів у процесі пошуку шляхом логічної обробки знайденої фактографічної інформації.

Семантичний пошук. Цей пошук полягає у пошуку документів за їх змістом. Для здійснення такого типу пошуку необхідні наступні умови:

– переклад змісту документів і запитів з природної мови на інформаційно-пошукову мову для створення пошукових образів документу і запиту;

– створення пошукового опису, в якому вказується додаткова умова пошуку.

Принципова різниця між адресним та семантичним пошуками полягає у тому, що при адресному пошуку документ розглядається як об'єкт з точки зору форми, а при семантичному пошуку – з точки зору змісту. При семантичному пошуку знаходиться множина документів без зазначення адрес. Це є принциповою відмінністю каталогів і картотек. Бібліотека – це є зібрання бібліографічних записів без вказування адрес.

5. Інформаційний пошук у Web-середовищі

Поява та розвиток Інтернету сприяли розширенню поняття пошуку та появі більш специфічного поняття Web-пошуку. Оскільки в контексті Web фактори взаємодії людини з комп'ютером та когнітивні аспекти грають найважливішу роль, корисно деталізувати цю модель, як показано на рис. 3.

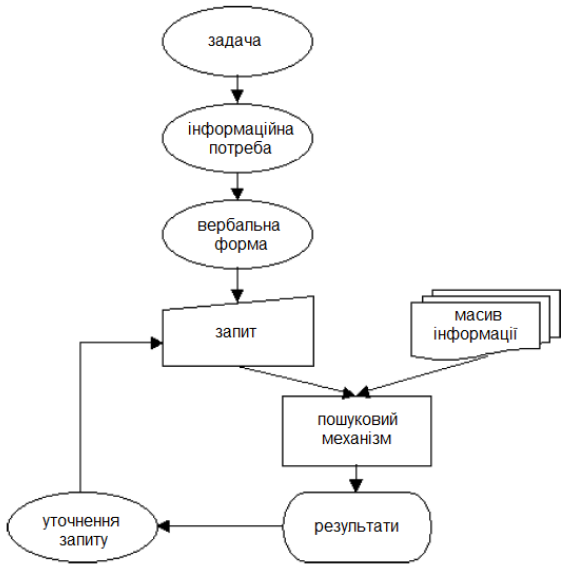

Рис. 3. Класична модель інформаційного пошуку, поширена на Інтернет-мережу (Web)

Як було зазначено раніше, інформаційна потреба асоціюється з (викликається) деякою задачею. Ця потреба вербалізується (найбільш часто це виконується ментально та не дуже чітко) та транслюється в запит, що надається пошуковому механізму. Цей процес висвітлення та створення запиту з інформаційної потреби, в контексті Web здобув велику увагу: в статті Хольстера та Струбе [10] вказується на тому, що досвідчені користувачі та новачки конструюють запити по-різному. Навarро – Пьєтро та ін. [11] вивели когнітивну модель для Web-пошуку, Мурамату та Прат [12] дослідили ментальну модель користувачів пошукових механізмів тощо. Також у [13] необхідно зауважити, що всі ці дослідження базуються на припущенні, що Web-пошуковці мотивовані (спонукувані) інформаційною потребою.

5.1. Таксономія Web-пошуку. В контексті Web, вираз "потреба спонукає запит" часто не є інформативним. У 2002 році автор [14] класифікував запити у відповідності до їх намірів на три наступних класи:

- ***навігаційні запити***. Такі запити мають на меті негайний намір побачити певний сайт;
- ***інформаційні запити***. Вони виражають намір отримати деяку інформацію, яка вважається існуючою на одній або більше Web-сторінках;
- ***транзакційні запити***. Ці запити виражають намір виконати якусь Web-опосередковану діяльність – покупку в Інтернет-магазині, завантаження файлів тощо.

Навігаційні запити. Метою таких запитів є дістатися певного сайту, який користувач має на увазі. Це визначено тим, що користувач можливо відвідував цей сайт у минулому, або він припускає, що такий сайт існує. Наприклад:

Запит	Можливий результат
compaq	Http://www.compaq.com
Фуршет	http://www.furshet.ua/
Газета по-киевски	http://mycityua.com

Цей тип пошуку іноді вважається, як пошук "загальновідомого предмету" в класичному ІП. Прикладом такого пошуку

стало завдання "Пошук домашньої Web-сторінки", яке регулярно проводиться при тестуванні пошукових систем при конференції з текстового пошуку (Text Retrieval Conference).

Навігаційні запити зазвичай мають тільки один правильний результат.

Транзакційні запити. Мета таких запитів полягає у тому, щоб досягти місця (сайту), де можливо провести подальшу взаємодію (транзакція) для досягнення певної мети. До основних категорій для таких запитів можна віднести здійснення покупок, пошук різних Web-опосередкованих сервісів, завантаження різного типу файлів (зображень, пісень і т. д.), доступ до деяких баз даних (наприклад, типу Yellow Pages), пошук серверів (наприклад, для ігор) і т. д.

Результати таких запитів з точки зору класичного ІП дуже важко оцінити. Все, що можливо – це бінарне значення оцінки, скажімо, відповідно чи не відповідно. Проте найбільш важливі для користувачів зовнішні чинники (наприклад, ціна товару, швидкість обслуговування, якість і таке інше), як правило, в загальних пошукових системах недоступні.

Інформаційні запити. Метою таких запитів є знайти інформацію, яка припускається існує у Webі в статичній формі. В подальшому взаємодій ніяких не передбачається, за винятком читання. Під статичною формою мається на увазі, що цільовий документ не створюється як відповідь на запит користувача. Ця різниця дещо розмита, оскільки змішування результатів, що характерно для третього покоління пошукових систем, можливо, призведе до використання динамічних сторінок.

В будь-якому випадку, інформаційні запити – найбільш приближені до класичного поняття інформаційного пошуку (IR), і тому вони далі будуть розглянуті детальніше.

На відміну від звичайного пошуку, більшість інформаційних запитів, що здійснюються в Інтернеті, семантично є надзвичайно широкими, наприклад, "автомобілі" або "Сан-Франциско", водночас як деякі можуть бути вузькими, наприклад "normocytic anemia" або "метрична система". Досліди інформаційних запитів, проведені в [14] відзначають, що майже 15 % усіх пошуків за бажану мету вважають гарну колекцію посилань за заданою темою, ніж один добрий документ.

Експериментальні результати дослідження типів запитів надані в таблиці.

Таблиця. Класифікація запитів користувачів

Type of query	User Survey	Query Log Analysis
Navigational	24.5 %	20 %
Informational	?? (estimated 39 %)	48 %
Transactional	> 22 % (estimated 36 %)	30 %

Пошукові системи необхідні для вирішення всіх трьох типів запитів, хоча кожен тип задовольняється досить різними результати. Розуміння цієї таксономії має важливе значення для успішного розвитку Web-пошуку. Сучасні пошукові системи добре вирішують інформаційні та навігаційні запити, але транзакційні запити задовольняються лише опосередковано. Шлях підвищення ефективності пошуку лежить в удосконаленні семантичного аналізу (тобто розуміння того, про що запит) та змішування різних зовнішніх баз даних.

5.2. Визначення пошуку в Web-середовищі. В зв'язку з появою Web, поняття пошуку в середовищі Інтернету набуло іншого змісту. Поняття пошукової системи стало більш широким та глибшим. Наведемо декілька новітніх визначень поняття пошукової системи (Search Engine), що прийняті нині в західній науковій літературі.

Пошукова система – це комп'ютерна програма, яка отримує (retrieves) файли або документи, або дані з бази даних або з комп'ютерної мережі (зокрема, з Інтернету) [15].

Пошукова система – це комп'ютерна програма, яка знаходить (finds) інформацію в Інтернеті шляхом пошуку слів,

які були введені (як запит – уточнення автора) [16].

Пошукова система – це комп'ютерне програмне забезпечення для пошуку даних (з текстів або баз даних) для отримання конкретної інформації, а також: сайт у Web-мережі, який використовує програмне забезпечення для пошуку ключових слів на інших сайтів [17].

В контексті Web з огляду на тезу, що „потреба спонукає запит", у клас поняття пошукових систем почали включати системи „запитання-відповідь" (answer engine), які дуже часто є фактографічними ПС. Але деякі системи для отримання результату пошуку вже починають використовувати процедури логічного виводу.

Зважаючи на вищесказане, пошукова система, в контексті Web, використовує спеціалізоване програмне забезпечення, яке має на вході від користувача пошуковий/і термін/и і на виході надає список Web-сторінок, які вважаються найбільш релевантними. Більшість пошукових систем мають величезні бази даних мільярдів Web-сторінок. Розрізняють два типи Web-пошукових систем: пошукові системи, базовані на кроулінгу та каталоги.

Пошукові системи, базовані на кроулінгу (Crawler-based). Такі системи створюють свої списки Web-сторінок автоматично. Вони "сканують" (crawl) Інтернет за допомогою робота-"павука" (spider, програма, яка відвідує Web-сторінки, читає їх і слідує далі за посиланнями, знайденими на Web-сторінці), і повертають користувачу результати пошуку, які ранжовані у порядку важливості. Павук повторно відвідує Web-сторінки кожні кілька місяців для найчастішого оновлення своєї індексної бази відповідно до внесених на Web-сторінки змін. Головна перевага пошукових систем, базованих на кроулінгу, полягає у тому, що будь-які зміни, які внесені до Web-сторінки, будуть впливати на його базу і відповідно – результати пошуку. Таким чином, актуальність змісту Web-сторінок збігається з ключовими словами, що використовуються для пошуку.

Каталоги, що створені людиною (human based directory), залежать від людей, які його створили та поповнюють. Вони виконують пошук за ключовими словами в коротких описах Web-сторінок, представлених Web-майстрами та спеціалістами, що рецензують та перевіряють каталог. Разом з цим, Web-сторінки переглядаються людиною і розміщуються в відповідну ієрархію категорій. Таким чином, зміни, внесені до Web-сторінки, на відміну від скануючих пошукових систем, не будуть мати ніякого впливу на збережений у каталозі опис. Отже, хоча на Web-сторінці і міститься відповідна інформація, яка відповідає запиту, але вона не буде відображена в списку результатів пошуку доки Web-майстер не змінить опис Web-сторінки. Саме з цієї причини один з найперших та найбільших каталог, сформований людиною Yahoo! перетворено у більш популярну пошукову систему на базі сканеру. Таким чином утворюються комбіновані пошукові системи. Оскільки каталоги містять інформацію, перевірену людиною, ця інформація використовується для фільтрування та ранжування результатів пошуку.

Окрім зазначеного вище, розрізняють наступні **типи пошукових механізмів**:

- пошукові системи;
- Web-каталоги;
- віртуальні бібліотеки;
- мета-пошукові механізми.

Пошукові системи (Search Engines) є найбільш широким класом ІПС та найбільш популярним і загальновживаним. Вони характеризуються наступними властивостями:

- мають базу даних Web-сторінок;
- пошук здійснюють за ключовими словами;
- мають скануючого робота.

Яскравим прикладом такої системи є пошукова система Google.

Web-каталоги (Web Directories). Як було вказано вище, вони:

- мають колекцію Web-ресурсів;
- організовані за тематичними категоріями в ієрархію;

- організація в категорії та інше проводиться вручну.

Приклад такого каталогу – загальновідомий каталог Yahoo.

Віртуальні бібліотеки (Virtual Libraries). Такі бібліотеки характеризуються наступними ознаками:

- мають колекцію Web-джерел;
- оцінюються фахівцями з предметної області;
- слабо автоматизовані, живляться людськими ресурсами.

Приклад типової бібліотеки – бібліотечний індекс Інтернету – Librarians Index to the Internet www.lii.org.

Мета-пошукові механізми (Meta-Search Tools). З назви видно, що такі механізми використовують ресурси інших пошукових систем, а результати фільтрують та ранжують згідно своїх заданих правил. Такі системи характеризуються:

- не мають власної бази даних;
- вони здійснюють запити до інших пошукових механізмів, розташованих у Web;
- мають дуже поганий дизайн і можуть тільки змінювати порядок ранжування результатів.

Класичний приклад такої системи є MetaCrawler.com. Такі системи користуються попитом, оскільки вони повертають більш короткий список посилань, що психологічно більш прийнятно для людини.

5.3. Еволюція пошукових систем інтернет. У зв'язку з таксономією, наведеною вище, в 2002 році в [14] було визначено три етапи (генерації) розвитку Web-пошукових систем.

Перше покоління пошукових систем використовувало в основному інформацію, яка знаходилась безпосередньо на Web-сторінках (текст і форматування), ці пошукові системи дуже близькі до класичних ПС. Такі системи виконують в основному тільки інформаційні запити. Типовими прикладами таких систем в 1995–1997 роках були загальновідомі AltaVista, Excite, Webcrawler і т. д. Ранжування сайтів відбувалося тільки за рахунок контенту сторінок.

Важливі фактори, які враховувалися при ранжуванні, включали щільність ключових слів на Web-сторінці, назву, і місцезнаходження цих ключових слів у цьому документі. Також ПС першого покоління для обчислення релевантності враховували мета-тегі, використання ключових слів в імені домену, а також в URL-адресі (докладніше – див. [29]).

Основні спам-фільтри робили перевірку на наявність ключових слів у тексті, представлених на сторінці тим самим кольором, що і фон документу, тобто невидимих людському зору. На той час з'явилися перші портали, в наслідок чого результати пошуку перетворилися у величезні рекламні щити та перевантажені інформацією жовті сторінки.

Друге покоління пошукових систем (початок появи 1998–1999 рр.) характеризується використанням інформації, яка існує поза Web-сторінкою – Web-специфічних даних таких, як аналіз посилань (link analysis), тексту якорів (anchor-text) та відстеження даних, що передаються з http-запитом (click-through data). Таким чином вони стали брати до уваги структуру Web-мережі.

Друге покоління більш щільно пов'язано з семантикою запитів, яка береться з аналізу даних, що подані у Webі поза сторінки. Деякі з основних компонентів, які вони використовують є відстеження кліків (tracking clicks), репутація сторінки (page reputation), індекс популярності (link popularity), темпоральні спостереження (temporal tracking, кількість часу, що проводять відвідувачі на сторінці), та якість посилань (link quality). Пізніше, ПС другого покоління почали використовувати вектори термів (term vectors) [18], аналіз статистики відвідування (stats analysis), кеш-дані (cache data) і контекст. Як аналіз контексту розглядається пошук на сторінці пар ключових слів, які складаються з двох слів. Це дозволяє краще виконати віднесення сторінки до певної категорії.

Першою системою, яка почала використовувати аналіз посилань між сторі-

нками як один з основних факторів ранжирування, стала система Google (PageRank). ПС DirectHit стала першою, хто побудував ранжування на аналізі даних, що передаються під час http-запиту. В даний час всі основні системи використовують всі ці типи даних. Використання Google PageRank та метод відстеження кліків DirectHit та тривалості візиту, підвищило ефективність пошуку.

Пошукові системи другого покоління підтримують як інформаційні, так і навігаційні запити. Аналіз посилань та текст якорів мають вирішальне значення для навігаційних запитів.

Третє покоління пошукових систем. На даний час третє покоління пошукових систем знаходиться в стані зародження та початкового розвитку. Ці пошукові системи є спробою поєднати дані з різних джерел для досягнення головної мети – видачі результату, що відповідає потребі користувача. Наприклад, на запит „Ялта", ПС має надавати пряме посилання на сторінку бронювання готелів у Ялті, сервер мап з мапою міста, на сервер погоди з інформацією про погоду і т. д. Таким чином, третє покоління – це покоління пошукових систем, які виходять за рамки обмежень фіксованої бази даних за допомогою семантичного аналізу, визначення контексту пошуку, вибору динамічної бази даних і т. д. Завдання полягає у тому, щоб забезпечити інформаційні, навігаційні і транзакційні запити.

Третє покоління пошукових технологій покликані об'єднати масштабованість існуючих Інтернет-пошукових систем з новими та удосконаленими моделями пошуку релевантності; вони починають враховувати вподобання користувача, співробітництво, колективний інтелект, багатий досвід користувачів, та багато інших спеціалізованих можливостей, які роблять інформацію більш значимою, а пошук – більш продуктивним.

Пошукові системи третього покоління додають до бази даних векторів термів похідні слова (word stemming) і тезаурус, що надає допомогу у здійсненні пошуку за контекстом [19]. Автоматичне визначення ключових пар також допомагає автоматичній категоризації сторінки, визначенню де користувач хоче провести покупку, а де – здійснити пошук, що має видати абсолютно різні результати пошуку на основі контексту або намірів користувача.

Технології третього покоління збагачені картами Web, які є корисними для фільтрації – видалення дублікатів сайтів, а також багатьох самостійних сторінок, які привертають трафік на всього лише декілька ключових слів. Це означає, що сторінки типу дорвеєв (doorways), гейтвеєв (gateways), вхідних (entry, splash) – спеціально створені спам-сторінки для цільової розкрутки сайту на визначені позиції ключових слів, незабаром будуть відфільтровані.

Вони також будуть витягувати як можна більше даних про індивідуальні пошукові звички користувача. Всі основні пошукові системи планують створення персональних профілів та агентів, які будуть накопичувати знання про користувача протягом певного періоду часу та використовувати їх виходячи з минулих пошукових звичок.

Поява Семантичного Web (докладніше див. [20]) надало нові можливості і ще більше диференціювало поняття інформаційного пошуку. Семантичний Web надав можливість використовувати існуючу семантичну інформацію – подану за допомогою семантичної розмітки, використовуючи семантичні зв'язки, виконуючі різні операції виведення на семантичних даних, а також порівняння семантичної інформації. Змінюється і алгоритм ранжування результуючих документів – вводиться поняття семантичного ранжування документів. Змінюється алгоритм пошуку, він стає дедалі розподіленим, змінюються методи задання пошукового запиту. Поява різних типів поданої у Web інформації (різної модальності – мультимедійної інформації, відео, аудіо тощо) потребує використання інших підходів. Існуюче розділення пошуку за типом інформації – пошук відео, пошук картинок, тощо (Google, Яндекс) – дуже стиснено і неінформативне. Існує синергетична потреба – виконання пошуку в різних

типах інформації та подальше змішування результатів.

Поява нового явища – Web речей (Web Of Things), який містить не тільки звичні документи, але й електронні пристрої та інші побутові речі, які підключені до Інтернету і можуть керуватися і знаходитися віддалено, також потребує врахування таких нових типів інформаційних об'єктів.

Таким чином, пошукові системи 3-го покоління виходять за рамки класичного (традиційного) поняття пошуку в зв'язку з появою нових типів інформації та нових вимог, що ставлять користувачі перед пошуковими системами.

В західній літературі з`явився термін Search 2.0, який асоціюється з третім поколінням, але має більш чіткі обриси і більш орієнтовано на бізнес-аудиторію [21]. У Webі вже існує десяток проектів, які вважаються проектами Search 2.0 – Swicki (http://www.swicki.com/), Rollyo (http://www.rollyo.com/),
Clusty (http://www.clusty.com/),
Wink (http://www.wink.com/),
Lexxe (http://www.lexxe.com/), тощо.

5.4. Приклади технологічних рішень пошукових систем третього покоління. З розвитком нових технологій та стандартів, паралельно з науковими дослідженнями, та спираючись на них, компанії бізнес-сектору прагматично розвивають нове покоління пошукових систем – «розумних» ПС, "smarter" search engines. Наведемо приклади таких технологічних рішень пошукових систем, які інтелектуалізують процес пошуку за рахунок:

- структурування та представлення (подання) даних, отриманих з Інтернету;
- реалізації семантичної фільтрації за якістю;
- організації пошуку серед структурованих даних в Інтернеті;
- пошуку в режимі реального часу в Інтернеті;
- пошуку в «глибинному» Web ('deep web') [22].

Структурування та подання даних

Wolfram Alpha (Система обчислювання знань, Computational Knowledge Engine, http://www.wolframalpha.com/, 2009). Цей амбіційний проект стартував 5 березня 2009 року. Автор цього Web-сервісу – британський фізик Стівен Вольфрам (Stephen Wolfram), голова компанії Wolfram Research, розробник широко відомої у наукових колах програми Mathematica.

На відміну від традиційних пошукових систем, які обмежуються тим, що за запитом користувача видають список посилань на сайти, які мають відповідати запиту, сервіс Wolfram Alpha самостійно аналізує запити користувача і представляє йому зведену релевантну інформацію.

З огляду на прийняту класифікацію ця система є системою „питання-відповідь". Автор позиціонує систему не як пошуковий сервіс (search engine), а як Computational Knowledge Engine («систему обчислювання знання»).

Ця система об'єднує обчислювальні потужності Mathematica з інструментами, які експліцитно оперують з усіма типами даних з тим щоб надати точну відповідь на запитання, яке сформульоване в природньомовній формі, в будь-яких можливих предметних областях [23]. Оскільки ця система є бізнес-застосуванням, докладного опису її функціювання у вільному доступі не має.

Спочатку Wolfram Alpha працював у закритому (тестовому) режимі, а з 18 травня 2009 р. Web-сервіс відкритий для всіх бажаючих. За час закритого тестування було оброблено близько 23 млн. запитів, а за перший тиждень після відкриття – близько 100 млн. На сьогоднішній день Wolfram Alpha є безкоштовним Web-сервісом.

Предметні області, які обробляються в системі – математика, фізика, хімія, астрономія, статистика та дані статистичного аналізу, дати та час, географія, погода, здоров'я та медицина, культура та медіа, музика та освіта, люди та історія, фінанси, лінгвістика і досягнення високих технологій, спорт тощо.

Можливості системи [24]:

- переведення одиниці виміру з однієї системи в іншу;
- якщо задати хімічну формулу, система видасть основну інформацію про цю речовину / хімічний елемент;
- якщо ввести в рядок пошуку 1 apple + 1 orange, – система видасть кількість калорій, протеїнів, вітамінів, відсутність / наявність холестерину і т. д.;
- якщо ввести назву міста, то система видає інформацію про те, де воно знаходиться, кількість жителів, схематичне розташування на карті, поточний час, поточну температуру, вологість, швидкість вітру, стан хмарності, висоту над рівнем моря, найближчі міста (з відстанню до них і з кількістю мешканців у цих містах). Натиснувши на посилання „Show coordinates", можна дізнатися координати міста. Натиснувши на посилання „Satellite image", система завантажить знімки міста з супутника (буде завантажений сайт "Карти Google");
- система виконує різні обчислення: якщо ввести в рядок пошуку, наприклад, $ 999 + 15 %, Wolfram Alpha зробить необхідні обчислення;
- система надає інформацію про будь-який сайт. Якщо ввести в рядок пошуку URL сайту, система видасть детальну інформацію: хто є хостинг-провайдером, де він розташований, кількість переглядів і кількість візитів за добу, site rank, найменування і розмір титульної сторінки, кількість вихідних посилань, кількість «зображень»;
- система може проводити не тільки найпростіші обчислення, але й вирішувати різні рівняння: якщо ввести, наприклад, x ^ 3 sin (x), система видасть рішення у вигляді графіка та в аналітичному вигляді;
- обробка музики, якщо ввести в рядок пошуку, наприклад, C Eb GC, то система надасть вичерпну інформацію про ці музичні ноти;
- обробка імен, якщо ввести два різних імені, наприклад, Vera, Natasha, в результаті система видає статистичні дані, що свідчать про те, як часто використовуються ці імена;
- обробка фінансової інформації: система може надавати інформацію про економічний стан (наприклад, про наявність акціонерного капіталу, вартості однієї акції і т. д.) двох компаній, назви яких вводяться у пошуковий рядок з пробілом між назвами;
- обробка часової інформації: якщо ввести дату в форматі, наприклад, august 28, 1959, то система видасть, який це був день тижня, можна буде підрахувати, скільки часу (років, місяців, тижнів, днів) пройшло з цієї дати, хто з відомих людей народився в цей день, які свята припадають на цей день.

Для того, щоб дізнатися джерела інформації, які використовував Wolfram Alpha, унизу, під знайденої інформацією знаходиться кнопка „Sources".

Всю інформацію, яку згенерував («навольфраміл» – сленг) Wolfram Alpha, можна зберегти у вигляді PDF-файлу.

Нажаль, система обробляє тільки англомовні запити.

Google Squared

Google Squared – цей експериментальний пошуковий механізм було заявлено 3 червня 2009 р. На відміну від класичних – «традиційних» пошукових систем, Google Squared не видає на запит користувача сторінку зі списком посилань на Web-ресурси, що відповідають запиту. Як результати пошуку користувачу виводиться зведена таблиця з інформацією з запросу. Тобто Google Squared, як і сервіс Wolfram Alpha, самостійно аналізує (намагається аналізувати) запити користувача і надає йому зведену релевантну інформацію.

В офіційному блозі пошукового гіганта сказано так: «...Squared Google не шукає Web-сторінки за вашим запитом…він автоматично вибирає і організовує факти зі всього Інтернету» [25].

Як і Wolfram Alpha, сервіс Google Squared не підтримує українську та російську мови.

Порівняльне тестування Google Squared та Wolfram Alpha, наведене авто-

ром у червні 2009 р. в [26] показує, що аналітичні характеристики і можливості системи Google Squared на даний час явно поступаються Wolfram Alpha.

Google Squared був експериментальним проектом, в якому корпорація Google проводила тестування функціоналу роботи пошукової системи з урахуванням структурованої інформації та початком інтелектуальної обробки знань. На даний час проект закрито.

Sensebot

SenseBot (http://www.sensebot.net/, 2008 р.) заявлена як семантична пошукова система, яка на пошуковий запит генерує текстові анотації (резюме), складені з Web-сторінок, які відносяться до теми пошукового запиту. Ця система для вилучення змісту з Web-сторінок і представлення його користувачеві узгодженим чином використовує інтелектуальну обробку текстів (text mining) і мультидокументну сумарізацію (multidocument summarization). Разом з результатами система видає „семантичну хмару" концептів ("Semantic Cloud" of concepts), що дозволяє направити увагу та керувати результатами.

Оскільки SenseBot є семантичною пошуковою системою, це означає, що вона намагається зрозуміти семантику отриманих у результаті сторінок. Вона використовує, як було зазначено вище, інтелектуальну обробку текстів для розбору Web-сторінок і визначення їх основних семантичних концептів.

На верхньому рівні, система отримує джерела, які видаються пошуковою системою як результат. Після цього система виконує інтелектуальну обробку тексту, отриманого з кожного джерела, вилучаючи ключові концепти. Подібності між джерелами оцінюються і ті, що семантично знаходяться далеко від запиту або не зв`язані з загальною масою знайдених джерел, відкидаються. Концептам присвоюється вага, а також для концептів, які представлені у запиті, задаються преференційні значення. Після чого виконується відповідно до запатентованого алгоритму мультидокументна сумаризація – збір підсумкового документу, складеного з текстів резюме, які згенеровані зі знайдених документів. Таким чином, на запит користувача фактичними результатами Web-пошуку є резюме, згенероване зі знайдених документів.

Найкращі результати можуть бути досягнуті на множині текстових документів, які по суті знаходяться близько до заданої теми. Найкраща область застосування цієї системи, як зазначає її розробник, є вертикальні пошукові системи і портали – фінансові, медичні, правові, бібліотеки і т. д. Що стосується загального Web-пошуку, деяка кількість "шуму" неминуча, навіть для тих джерел, що знаходяться на перших сторінках результатів, які вважаються найбільш релевантними [27].

Реалізація семантичної фільтрації інформації за якістю

Hakia

Цей відомий проект (http://www.hakia.com/) засновано в 2004 р. Для роботи системи була розроблена альтернативна інфраструктура, яка використовує алгоритм SemanticRank, який використовує онтологічну семантику, обчислювальну лінгвістику та нечітку логіку. На час, коли система була в відкритому доступі, вона охоплювала тільки предметну область з медицини та здоров'я. Заявлялося, що семантична технологія Hakia забезпечує новий досвід пошуку, який орієнтований на якість, а не популярність. Для проведення подальшого дослідження в галузі ІП досить корисними є основні 3 критерії, яким одночасно мають задовольняти якісні результати:

– якісні результати надходять з заслуговуючих довіри Web-сайтів, рекомендованих бібліотекарами або довіреними особами;

– якісні результати представляють собою найбільш свіжу наявну інформацію;

– якісні результати залишаються абсолютно релевантними до запиту.

Проект був відкритий для користування до квітня 2014 р. На даний час його повністю закрили і надалі використовують

для закритих комерційних рішень з підтримки Web-сайтів з обмеженою ПрО.

Організація пошуку серед структурованих даних у Webі

SWSE

На даний час вже існує багато даних, які відповідають запропонованим стандартам Семантичного Weby (наприклад RDF та OWL). Вже існує багато малих вертикальних словників і онтологій, які все більше використовуються різними спільнотами для вирішення своїх конкретних задач: користувачі Weby публікують описи своїх профілів з використанням формату FOAF (Friend of a Friend), провайдери новин транслюють добірку новин у вигляді RSS (RDF Site Summary), зображення анотуються з використанням різноманітних RDF-словників тощо.

SWSE (http://swse.deri.org/) представляє собою сервіс, який постійно вивчає та індексує Семантичний Web (Semantic Web) і забезпечує легкий у використанні інтерфейс, за допомогою якого користувачі можуть знайти дані, які вони шукають.

SWSE індексує триплети RDF або OWL, знайдені в Web, і надає послугу з пошуку серед цих триплетів.

На даний час проект закритий для зовнішнього використання і інтегрований у загальні проекти консорціуму W3C.

Swoogle

Swoogle (http://swoogle.umbc.edu/) –пошукова система, створена спеціально в рамках розвитку Семантичного Web. Кроулери Swoogle сканують Web з метою пошуку спеціального класу Web-документів, які називаються семантичними Web-документами, тобто які написані мовами RDF або OWL. Ця пошукова система індексує знайдені семантичні документи і зберігає їх, поступово формуючи онтологічну базу знань, та виконує пошук серед RDF-триплетів, видаючи в результатах пошуку посилання на джерела, які їх містять та фрагменти відповідних онтологій. Пошук здійснюється за ключовими словами та з використанням додаткових онтологічних конструкцій – обмежень.

Аналогічні функції пропонують і пошукові системи WatsOn, Semanticweb-search, Sindice (http://sindice.com/), Falcons.

Пошук у Web у режимі реального часу

Topsy

Пошукова система **Topsy** (http://www.topsy.com/) у режимі реального часу сканує інформацію, яка постійно генерується користувачами соціальних мереж Twitter, Digg, тощо. Якщо повідомлення містить посилання на Web-сторінку, то в разі, якщо за алгоритмом системи таке посилання буде вважатися важливим, воно буде проіндексоване. Таке індексування пошукова система проводить у режимі реального часу – поява нового посилання на сервісі одразу викликає процес індексування. Алгоритм визначення важливості посилань враховує багато умов, одними з головних є авторитетність джерела інформації та рівень довіри (trust).

Кінцевим результатом роботи пошукової системи є пошуковий досвід, який дозволяє користувачам знаходити свіжий, найбільш соціально значущий контент у реальному часі у Web. Результати пошуку індексуються в залежності від їх актуальності та популярності. Окрім текстової інформації, система індексує фото та відео, а також інформацію з соціальних мереж (твіти, пости тощо).

Scoopler

Scoopler – один з найкращих стартапів, який запропонував виконання пошуку в режимі реального часу. Аналогічно пошуковій системі Topsy, робот цієї ПС збирає і організовує контент актуального типу – новини, фотографії та відеоматеріали значних подій, а також посилання на найгарячіші нотатки поточного дня. Джерелами контенту, який індексується, є постійні оновлення, що поступають з сервісів Twitter, Flickr, Digg, Delicious тощо. На даний час система викуплена корпорацією Google і використовується для пошуку в соціальній мережі Google+.

Пошук в «глибинному» Web ('deep web')

DeepDyve

DeepDyve(http://www.deepdyve.com/) – пошуково-„дослідницька" система, яка використовує власні (комерційні) технології пошуку та індексування, що дозволяють відбирати багатий, релевантний контент з тисяч журналів, мільйонів документів і мільярдів незадіяних Web-сторінок глибинного Web. Дослідники, студенти, технічні спеціалісти, бізнес-користувачі, а також споживачі іншої інформації, можуть отримати доступ до багатої інформації, що зберігається в „глибинному Web" – інформації, яка складає переважну більшість в Інтернеті, але не індексується традиційними пошуковими системами. Пошуково-дослідницька система DeepDyve відчиняє шлях до цього поглибленого професійного контенту і повертає результати, які не навантажені інформацією з оглядових (реферативних) сайтів та іншою нерелевантною інформацією.

Система використовує запатентований алгоритм KeyPhrase ™, який застосовує метод індексації, отриманий при дослідженнях в області геноміки. Алгоритм шукає збіг патернів і символи за спеціальною метрикою. Система знаходить відповідність документів там, де традиційні пошукові системи нічого не знаходять. Тому ця система ідеально підходить для пошуку складних даних, що містяться в глибинному Web.

Також існує багато пошукових систем, що виконують пошук у глибинному Web, які спеціалізуються на конкретній предметній області та містять перевірені і рецензовані спеціалістами статті. Такі ПС, як правило, мають вузько спрямовані репозиторії, що надає реальну перевагу для цілеспрямованого пошуку дослідника в певній ПрО.

До таких спеціалізованих порталів можна віднести Mednar (www.mednar.com) – портал з глибинного пошуку в галузі медицини, Biznar (www.biznar.com) – пошук в бізнес-галузі, Worldwidescience (www.worldwidescience.org) – глобальний науковий портал, Science.gov (www.science.gov) – науковий портал уряду США, Scitopia (www.scitopia.org) – пошукова система наукової інформації і патентів, Nutrition.gov (www.nutrition.gov) – портал, який містить інформацію про здоров'я. Більшість порталів глибинного Web підтримують механізми кластеризації за темами.

Висновки

Однією з причин підвищеного інтересу до проекту Semantic Web є надія на поліпшення пошуку в Web. Дослідження з цієї проблеми ведуться в різних напрямках і дають різноманітні результати у вигляді різних пошукових систем. Такі системи, як Swoogle, дозволяють лише виконувати пошук онтологій за ключовими словами. Але такий сервіс є дуже корисним для розробників семантичних систем і онтологій, хоча він і не розрахований на звичайного користувача. [28]. Джерелами інформації у них служать набори RDF-даних, включаючи дані, що пов'язані в рамках проекту Linked Open Data і мікроформати.

Можна відзначити й інші пошукові системи Semantic Web, багато з яких знаходяться на стадії бета-тестування, тому оцінити їх можливості поки важко. Деякі системи йдуть шляхом „углиблення у Web", інші – більш прискіпливо розвивають алгоритми інтелектуального аналізу та використовують різноманітні джерела інформації про документи, які знаходяться „поза-документом" у Web. Розвиток технологій інформаційного пошуку призвів до інтенсивного використання мета-інформаційно-пошукових систем, багато-агентних інформаційно-пошукових систем, систем, побудованих на реалізації онтологічних, мовних та управлінських угод і їм подібних. Більшість пошукових систем йдуть шляхом розвитку персоналізації пошуку, тобто розпізнання та задоволення потреб користувача.

Традиційні пошукові системи стають все більш точними та об'ємними, однак вони не можуть перевершити інтелект людини. Вони можуть лише порівнювати слова, а не зміст ідеї, яка обговорюється ними. Нові технології пошукових систем 3-го покоління ще знаходяться в стадії формування, але вже нині вони дають по-

зитивні результати. Новий пошук може допомогти зробити пошук більш значущим, суб'єктивним і прив'язаним до задач (task-based), що стоять перед користувачем. Таким чином, розвиток пошукових систем йде в напрямку задоволення потреб окремого користувача, з його перевагами, характером, звичками, поведінкою, рівнем підготовки і знань тощо.

1. *Christopher D. Manning, Prabhakar Raghavan, Hinrich Schütze*. An Introduction to Information Retrieval, Online edition (c) 2009 Cambridge UP, Draft of April 1, 2009,Website: http://www.informationretrieval.org
2. *Черний Ю.Ю.* Школа наукової інформації. Інформаційні потреби. Основи інформаційного пошуку, http://www.bogoslov.ru/text/321597.html
3. *Захаров В.П.* Информационно-поисковые системы. Учебно-методическое пособие, Санкт-Петербург, 2005.
4. *Медведь В.Н.* Методы поиска информации, http://northedu.ru/content/view/115/159/
5. *Van Rijsbergen C.J*. Information Retrieval. London: Butterworths, 1979. Available at http://www.dcs.gla.ac.uk/Keith/Preface.html
6. *Шарапов Р.В., Шарапова Е.В., Саратовцева О.А*. Модели информационного поиска.
7. *Некрестьянов И.С.* Тематико-ориентированные методы информационного поиска: Дис. … канд. техн. наук. – Санкт-Петербургский государственный университет. – СПб, 2000. – 88 с.
8. *Дубинский А.Г.* Некоторые вопросы применения векторной модели представления документов в информационном поиске // Управляющие системы и машины. – 2001. – № 4. – С. 77–83.
9. *Когаловский М.Р.* Перспективные технологии информационных систем. – М.: ДМК Пресс; М.: Компания АйТи, 2003. – 288 с.
10. *Holscher C. and Strube G.* Web search behaviour of Internet experts and Newbies. Proceedings of WWW9. 2000. Available at http://www9.org/w9cdrom/81/81.html.
11. *Navarro-Prieto R., Scaife M. & Rogers Y*. Cognitive Strategies in Web Searching. Proceedings of the 5th Conference on Human Factors & the Web, 1999. Available at http://zing.ncsl.nist.gov/hfweb/proceedings/navarro-prieto/index.html.
12. *Muramatu J. and Pratt W.* Transparent queries: Investigating Users' Mental Models of Search Engines, Proceedings of SIGIR 2001.
13. *Choo C. W., Detlor B., and Turnbull D*. Information Seeking on the Web – An integrated model of browsing and searching. Proceedings of the Annual Meeting of the American Society for Information Science (ASIS), 1999. Available at http://choo.fis.utoronto.ca/fis/respub/aisis99/
14. *Broder A*. A taxonomy of web search, IBM Research, ACM SIGIR Forum archive. – 2002. – Vol. 36 , Issue 2. – P. 3–10.
15. *Лексична* база англійської мови WordNet, http://wordnet.princeton.edu/perl/webwn
16. *Онлайн* словник http://dictionary.cambridge.org/
17. *Онлайн* словник http://www.merriam-webster.com/
18. *Rodnessey J*. New Search Engines: The Next Generation of Google Competition, 2009, http://webupon.com/search-engines/new-search-engines-the-next-generation-of-google-competition/
19. *Nobles R*. The Future Of Search Engine Optimizing, http://www.searchengineworkshops.com/articles/se-optimization-future.html
20. *Андон Ф.И., Гришанова И.Ю., Резниченко В.А.* Semantic Web как новая модель информационного пространства интернет // Проблеми програмування. – 2008. – № 2–3. – С. 417–430.
21. *Ezzy E*., Search 2.0 vs Traditional Search, 2006, http://www.readwriteweb.com/archives/search_20_vs_tr.php
22. *McLoughlin S*. Searching on the web; the new breed of search engines, 2009, http://relativemusings.blogspot.com/2009/05/searching-on-web-new-breed-of-smarter.html
23. *Wolfram S*. Wolfram Alpha – computational knowledge engine, 2009 http://base-technology.blogspot.com/2009/03/wolfram-alpha-computational-knowledge.html
24. *Сидоров В*. Wolfram Alpha – Computational Knowledge Engine, или Как сложить яблоко с апельсином?, блог, 2009, http://netler.ru/pc/wolfram.htm
25. *Official* Google Blog: Square your search results with Google Squared, http://googleblog.blogspot.com/2009/06/square-your-search-results-with-google.html
26. *Сидоров В*. Google Squared: как успех Wolfram Alpha взбудоражил Google и что из этого вышло?.., блог, 2009, http://netler.ru/pc/google-squared.htm

27. *Soubbotin D.* Summarization, the Answer to Web Search: Interview with Dmitri Soubbotin of SenseBot, Search Engine Journal, 2007, http://www.searchenginejournal.com/summarization-the-answer-to-web-search-interview-with-dmitri-soubbotin-of-sensebot/6094/
28. *Левшин Д.* Web, часть третья // Открытые системы. – 2008. – № 2. http://cio.ru/text/print/302/8165094.html
29. *Рогушина Ю.В., Гришанова І.Ю.* Разработка принципов представления электронных изданий, обеспечивающих корректную индексацию поисковыми системами Интернет // Проблеми програмування. – 2004. – № 4. – С. 39–47.

References

1. *Christopher D. Manning, Prabhakar Raghavan, Hinrich Schütze* An Introduction to Information Retrieval, Online edition (c)2009 Cambridge UP, Draft of April 1, 2009, Website: http://www.informationretrieval.org
2. *Cherniy Y.Y.* School of scientific information. Information needs. Basics of information retrieval, http://www.bogoslov.ru/text/321597.html
3. *Zacharov V.P.* Informational retrieval systems, Learning manual, St. Petersburg, 2005
4. *Medvedj V.N.* Methods of information retrieval, http://northedu.ru/content/view/115/159/
5. *Van Rijsbergen C.J.* Information Retrieval. London: Butterworths, 1979. Available at http://www.dcs.gla.ac.uk/Keith/Preface.html
6. *Sharapov P.B.,Sharapova E.V., Saratovceva O.A.*, Models of information retirieval.
7. *Nekrestyanov I.S.* Topic – oriented methods of information retrieval: The Thesis of Ph.D.: 05.13.11 / Saint-Petersburg State University – St.Pt., 2000. – 88 p.
8. *Dubinskyi A.G.* Some questions of the use of the vector model for the document`s presentation in the information retrieval // Control Systems and Computers. – 2001. – N 4. – P. 77–83.
9. *Kogalovskyi M.R.* Prospective technologies of information systems. – M.: DMK Press; Moscow: IT Company, 2003. – 288 p.
10. *Holscher C. and Strube G.* Web search behaviour of Internet experts and Newbies. Proceedings of WWW9. 2000. Available at http://www9.org/w9cdrom/81/81.html.
11. *Navarro-Prieto R., Scaife M. & Rogers Y.* Cognitive Strategies in Web Searching. Proceedings of the 5th Conference on Human Factors & the Web, 1999. Available at http://zing.ncsl.nist.gov/hfweb/proceedings/navarro-prieto/index.html.
12. *Muramatu J. and Pratt W.* Transparent queries: Investigating Users' Mental Models of Search Engines, Proceedings of SIGIR 2001.
13. *Choo C. W., Detlor B., and Turnbull D.* Information Seeking on the Web – An integrated model of browsing and searching. Proceedings of the Annual Meeting of the American Society for Information Science (ASIS), 1999. Available at http://choo.fis.utoronto.ca/fis/respub/aisis99/
14. *Broder A.* A taxonomy of web search, IBM Research, ACM SIGIR Forum archive. – 2002. – Vol. 36 , Issue 2. – P. 3–10.
15. *Lexical* base of English language WordNet, http://wordnet.princeton.edu/perl/webwn
16. *Online* vocabulary http://dictionary.cambridge.org/
17. *Online* vocabulary http://www.merriam-webster.com/
18. *Rodnessey J.* New Search Engines: The Next Generation of Google Competition, 2009, http://webupon.com/search-engines/new-search-engines-the-next-generation-of-google-competition/
19. *Nobles R.* The Future Of Search Engine Optimizing, http://www.searchengineworkshops.com/articles/se-optimization-future.html
20. *Andon P.I., Grishanova I.J., Reznichenko V.A.* Semantic Web as a new model of the information space of the Internet // Problems in Programming. – 2008. – N 2–3, P. 417–430.
21. *Ezzy E.* Search 2.0 vs Traditional Search, 2006, http://www.readwriteweb.com/archives/search_20_vs_tr.php
22. *McLoughlin S.* Searching on the web; the new breed of search engines, 2009, http://relativemusings.blogspot.com/2009/05/searching-on-web-new-breed-of-smarter.html
23. *Wolfram S.* Wolfram Alpha – computational knowledge engine, 2009 http://basetechnology.blogspot.com/2009/03/wolfram-alpha-computational-knowledge.html
24. *Sidorov V.* Wolfram Alpha – Computational Knowledge Engine, or How To Add Apple with Orange?, blog, 2009, http://netler.ru/pc/wolfram.htm
25. *Official* Google Blog: Square your search results with Google Squared, http://googleblog.blogspot.com/2009/06/square-your-search-results-with-google.html

26. *Sidorov V.* Google Squared: How the Success of Wolfram Alpha Stirred up Google and What Happened?.., blog, 2009, http://netler.ru/pc/google-squared.htm
27. *Soubbotin D.* Summarization, the Answer to Web Search: Interview with Dmitri Soubbotin of SenseBot, Search Engine Journal, 2007, http://www.searchenginejournal.com/summarization-the-answer-to-web-search-interview-with-dmitri-soubbotin-of-sensebot/6094/
28. *Levshin D.* Web, part 3, "Open systems". – 2008. – N 2. http://cio.ru/text/print/302/8165094.html
29. *Rogushina J.V., Grishanova I.Y.* Development of the Principles of Electronic Publications, Providing the Correct Indexing of Internet Search Engines // Problems in Programming – 2004, N 4. – P. 39–47.

Одержано 08.12.2015

Про автора:

Гришанова Ірина Юріївна,
науковий співробітник,
Кількість наукових публікації в українських виданнях – 15.
http://orcid.org/0000-0003-4999-6294.

Місце роботи автора:

Інститут програмних систем
НАН України,
03181, Київ-187,
Проспект Академіка Глушкова, 40.
E-mail: i26031966@gmail.com

УДК 681.3

Ю.В. Рогушина

ВИКОРИСТАННЯ ОНТОЛОГІЙ ДЛЯ ПЕРСОНІФІКОВАНОГО ПОШУКУ ЗНАНЬ У ПРИРОДНОМОВНИХ ТЕКСТІВ

Запропонований у роботі підхід до персоніфікації пошуку інформаційних ресурсів та інформаційних об'єктів, що базується на побудові та використанні тезаурусу задачі користувача, дозволяє використовувати знання щодо предметної області пошуку та структури інформаційних об'єктів, представлені за допомогою відповідних онтологій. Наведені визначення семантичного пошуку, його суб'єктів та компоненті дозволяють більш чітко формулювати проблеми, пов'язані з пошуком інформації у відкритому середовищі Web. Програмна реалізація запропонованого підходу підтверджує ефективність його практичного використання.

Ключові слова: семантичний пошук, інформаційний об'єкт, онтологія, тезаурус задачі.

Вступ

В процесі розвитку суспільства з великою швидкістю збільшуються обсяги інформації, що обробляється, ускладнюється її структура та методи обробки.

Для сучасного етапу розвитку інформаційних технологій (ІТ) характерні наступні тенденції: 1) все більше інформаційних систем (ІС) стають інтелектуальними та використовують знання; 2) переважна частина ІС працює у відкритому середовищі (Web, локальні та корпоративні мережі, хмари тощо) і орієнтовані на отримання відомостей із зовнішніх інформаційних ресурсів (ІР), не залежних від їх розробників; 3) все більше поширення отримують різноманітні мобільні пристрої для обробки інформації, що відповідають специфіці користувачів.

З цього випливає важливість проблеми пошуку знань та використання знань у пошуку даних, а також персоніфікація такого пошуку.

Актуальність проблеми пошуку

Значна частина знань, накопичених у результаті розвитку людського суспільства і різних предметних областей, міститься в документах у вигляді природномовного тексту. У Web представлення також велика кількість мультимедійних документів, і з ростом поширення різних підключених до Інтернет пристроїв їхній обсяг зростає значно швидше, але в них міститься набагато менше корисної інформації (наприклад, різні фотографії і відеофайли, як правило, цікаві лише тим, хто їх знімав).

У структурованому вигляді (онтології, метаописи, семантична розмітка і т. п.) представлено набагато менше відомостей. Якщо ж порівнювати зусилля, необхідні для витягу знань (змісту), то природномовні ресурси обробляти значно легше, ніж мультимедійні, а сама форма представлення забезпечує первинний пошук релевантних проблемі ресурсів (наприклад, пошук за ключовими словами у тексті) з подальшою семантизацією [1, 2]. Крім того, при розпізнаванні мультимедійних ІР спочатку значна частина інформації перетворюється на природномовний текст.

Це обумовлює важливість розвитку методів аналізу природномовних документів, що включають пошук релевантних предметній області документів, розпізнавання їхнього змісту і поповнення відповідних онтологій [3–5]. Це потребує створення засобів отримання, збереження, пошуку та використання знань з урахуванням таких властивостей середовища Web, як динамічність та гетерогенність. Крім того, необхідно обрати засоби інтероперабельного представлення знань, які мають достатні функціональні можливості та надають можливість для їх повторного використання та обробки як комп'ютерними програмами, так і людьми.

Для цього виникає потреба у розробці моделі інформаційного середовища сучасного Web; моделі користувача, що відображає його інформаційні потреби, предметну область, яка його цікавить, та проблеми, які він прагне вирішити; і моделі інформаційних ресурсів, що має відображати не тільки їх формальні властивості, але й семантику.

Процеси глобальної інформатизації орієнтовані на побудову та інтероперабельне використання міждисциплінарних знань. Але ефективне використання знань потребує розвитку відповідних засобів їх знаходження та подання. Це вимагає розвитку інженерії знань і засобів менеджменту знань.

Відносно новим напрямком у цій сфері є онтологічний інжиніринг, що забезпечує повторне й інтероперабельне застосування накопичених у суспільстві знань [6–8]. Онтології використовуються в системах обробки знань для їхнього структурування й інтеграції [9]. Тому актуальні питання автоматизованого створення і поповнення онтологій на основі гетерогенних і динамічних ресурсів Web, їхньої інтеграції і співставлення, а також створення методів логічного виведення на них [10].

Системи семантичного пошуку

Семантичний пошук – це метод інформаційного пошуку, у якому релевантність документа запиту визначається семантично (за близькістю змісту), а не синтаксично (приміром, за частотою використання ключових слів у документі).

Можна розглядати семантичний пошук як розвиток традиційного інформаційного пошуку, в якому з метою підвищення пертинентності пошуку (тобто для більш ефективного задоволення інформаційних потреб користувача) використовується обробка знань, що стосуються як самого користувача та його інформаційних потреб (персоніфікація пошуку), так і про інформаційні ресурси, серед яких здійснюється пошукова процедура. Системи семантичного пошуку – це певна інтелектуальна надбудова над традиційними інформаційно-пошуковими системами (ІПС) – як загального призначення, так і спеціалізованими.

Система семантичного пошуку (ССП) – це інформаційна система, що забезпечує пошук та розпізнавання інформаційних об’єктів (ІО) різних типів з використанням знань для співставлення запиту з наявними інформаційними ресурсами на семантичному рівні.

Відкрита ССП – це ССП, в якій використовуються не тільки внутрішні, але й зовнішні бази знань, структура та контент яких не залежать від розробника ССП.

Інформаційний об’єкт – модель об’єкта предметної області (ПрО) в інформаційному просторі, яка визначає структуру, атрибути, обмеження цілісності і, можливо, поводження цього об’єкта.

Результатом семантичного пошуку може бути як здобуття інформації щодо ІО, неявно присутньої у певному ІР (як текстовому, так і мультимедійному), так і надання користувачеві відомостей про наявні ІР у певному порядку та певній формі, що відповідають персональним потребам саме цього користувача.

Те, які саме знання використовуються, як вони представлені і як вони обробляються, залежить як від специфіки розроблювальної ІПС, так і від концепції, обраної її розроблювачами, але в загальному випадку результати семантичного пошуку – множина з n елементів є функцією від запиту користувача, індексу ІПС та знань, що містяться в базі знань (БЗ) ІПС:

$$I_s = \{i_j,\, j = \overline{1,\, n}\} = f\left(z, DB_{inc}, KB_{inc}\right).$$

Якщо мова йде про семантичний пошук у Web, то слід враховувати, що при цьому в Web можуть знаходитися не тільки інформаційні об’єкти, серед яких здійснюється пошук, але і зовнішні бази знань, що використовуються для пошуку. Тому при створенні таких систем варто враховувати, що такі зовнішні БЗ можуть змінювати контент, структуру і приступність незалежно від розроблювачів ІПС. Тому результати пошуку в Web є функцією ще й від вмісту зовнішніх БЗ:

$$I_{web\_s} = \left\{ i_j, j = \overline{1,n} \right\} = f\left(\begin{array}{l} z, DB_{іпс}, KB_{іпс}, \\ \left\{ KB_{web\_k}, k = \overline{1.m} \right\} \end{array} \right).$$

При рішенні задачі семантичного пошуку, пов'язаного з розпізнаванням набору складних ІО, виникає ряд різних видів проблем, для опису яких необхідно визначити використовувані при цьому терміни, зокрема, визначити, які відомості є результатом такого пошуку, які – його умовами. Тому потрібна класифікація ІО та пов'язаних з ними пошукових ситуацій.

У найбільш простому варіанті інформаційного пошуку на вході пошукова система отримує набір ключових слів, а на виході дає набір посилань на документи.

Значно складніше вирішити пошукову задачу, для якої вхідними даними служить опис складної проблеми, у якій описана взаємодія множини складно структурованих ІО, а на виході необхідно отримати посилання на екземпляри ІО, що задовольняють поставленим умовам.

Для сучасних семантичних Web-застосувань характерні ІО зі складною структурою, які пов'язані з певними об'єктами реального або віртуального світу (програмні агенти, Web-сервіси, семантично розмічені ресурси, елементи Web of Things, онтологічні описи тощо). Оцінка складності структури атомарних елементів залежить від їхньої кількості і кількості зв'язків між ними.

Приміром, ІПС, що впорядковує знайдені за запитом користувача документи, враховуючи його персональні властивості, і використовує для цього інформацію з історії взаємодії з користувачем, менш інтелектуальна, ніж ІПС, що впорядковує знайдені за запитом користувача документи, використовуючи для формалізації сфери інтересів користувача онтологію відповідної ПрО, та виконує семантичну розмітку знайдених документів термінами з онтології ПрО [11].

Як показує аналіз публікацій, один з перспективних підходів до завдання контексту пошуку ґрунтується на онтологіях, що містять перелік основних термінів, зв'язки між ними і правила виведення (так, у проекті Semantic Web, спрямованому на аналіз семантики ІР, саме онтологічний підхід є основою для подання знань про різні ПрО).

Онтологія ІО O_{IO} – онтологічна структура, що містить клас ІО $t_{IO} \in T_{IO}$ і його підкласи, які описують різні підмножини ІО, а також класи T_{Prop}, необхідні для опису властивостей різних ІО.

$$O_{IO} = \left\langle T_{IO} \cup T_{Prop}, R, A \right\rangle.$$

Для того, щоб користувач міг охарактеризувати ІО, який його цікавить, йому потрібно послатися на клас довільної формально описаної онтології.

Таким чином, ІО можна розглядати як клас певної онтології, який має набір характеристик, які описують його структуру і можливі зв'язки з іншими класами й екземплярами класів. Для більш точного визначення проблеми доцільно ввести кілька визначень

Екземпляр ІО – екземпляр якого-небудь підкласу ІО відповідної онтології, який можна однозначно ідентифікувати і який має власне ім'я.

Ситуація – непорожня множина ІО та екземплярів ІО одного чи різних класів, така, що для будь-якого ІО з цієї множини існує зв'язок хоча б з одним іншим ІО з цієї множини.

Якщо в ситуації використовуються ІО, описані за допомогою різних онтологій, то необхідно (явно чи за допомогою засобів автоматизованого зіставлення онтологій) встановити зв'язки між цими онтологіями (чи хоча б між тими ІО і класами, що описують властивості тих ІО, що фігурують у ситуації).

Схема ситуації – ситуація, у якій не використовуються екземпляри ІО.

Неприпустима схема ситуації – схема ситуації, всі умови якої не можуть бути виконані при жодному наборі екземплярів ІО.

Схема є *неприпустимою*, якщо в ній містяться суперечні умови:

$$f_0(a_1,...,a_n), f_1(a_1,...,a_n),...,$$
$$f_m(a_1,...,a_n), a_i \in t_i \subseteq T_{IO},$$

і з деякої їхньої підмножини можна вивести заперечення однієї з цих умов, тобто з

$$f_1(a_1,...,a_n),...,f_m(a_1,...,a_n),$$
$$a_i \in t_i \subseteq T_{IO}$$

логічно виводиться $\neg f_0(a_1,...,a_n)$.

Унікальна ситуація – ситуація, всі умови якої можуть бути виконані лише при єдиному наборі екземплярів ІО. Приклад такої ситуації – пошук книги, посилання на яку міститься у відеофільмі.

Конкретизована ситуація – ситуація, в описі якої, крім класів ІО, описано хоча б один конкретний екземпляр ІО. Приклад такої ситуації – знайти всі організації, в яких працювали особи, які проживали в одному будинку з особою X, що має ідентифікаційний номер Y.

Персональна ситуація – ситуація, в якій використовується екземпляр класу "користувач" онтології семантичного пошуку, який характеризує ту особу, що задає цю ситуацію. Персональні ситуації можуть мати стандартизований опис, в якому відомості щодо користувача є параметрами.

Цей варіант задачі досить поширений, коли користувач намагається знайти якісь відомості саме для себе – приміром, посилання на власні публікації, можливість свого працевлаштування в певній організації, рейтинг своєї спеціальності тощо. Кожна персональна ситуація є конкретизованою через використання конкретного екземпляра класу "користувач". Використання персональних ситуацій дозволяє розробляти типові запити, в яких певна частина інформації не вводиться користувачем вручну, а імпортується з його профілю. Приміром, можна задавати замість запиту "знайти в публікаціях автора А всі посилання на публікації автору В" значно простіший для користувача запит "знайти в публікаціях автора А всі посилання на мої публікації", для якого перелік "мої публікації" має будуватися автоматично та оновлюватися за рахунок пошуку у Web-ресурсах.

Ситуація *задовольняє схемі*, якщо для всіх ІО й їх екземплярів виконуються умови, що входять до складу схеми.

Пошук *нездійсненний*, якщо його умовою є неприпустима схема ситуації.

Пошук *виконуваний*, якщо його умови можуть бути виконані (навіть якщо не виявлена така комбінація ІО, що задовольняє цим умовам).

Пошук *тривіальний*, якщо його результатом є унікальна ситуація.

В інших термінах можна вважати схему ситуації пошуковим запитом, а множину ситуацій – його результатом.

Постановка задачі

Щоб підвищити ефективність семантичного пошуку, необхідно забезпечити його *персоніфікацію*, тобто використання знань про інформаційні потреби, сферу інтересів та здатність до сприйняття інформації окремих користувачів. Тому виникає потреба у розробці формалізованої моделі інформаційних потреб користувача, у засобах її поповнення інформацією та у методах її співставлення з наявними інформаційними ресурсами.

Використання онтологій для персоніфікованого аналізу природномовних текстів

Щоб використовувати онтологічні знання в процесі семантичного пошуку, потрібно забезпечити як механізми автоматизованого створення онтологічних моделей предметних областей та інформаційних потреб, так і методи їх співставлення. Пропонується в якості такого механізму використовувати тезаурус задачі, який відображає поточні інформаційні потреби користувача на основі онтології ПрО, обраної користувачем.

У загальному випадку тезаурус – це словник основних понять мови, що позначаються окремими словами чи словосполученнями, з визначеними семантичними зв'язками між ними [12]. Тезаурус можна розглядати як окремий випадок онтології [13]. Лексика тезауруса включає

множину слів і/чи множину фраз [14–16]. Види підтримуваних семантичних зв'язків між ними можуть бути залежними чи незалежними від конкретної ПрО. Звичайно такі зв'язки визначають синоніми, омоніми, антоніми понять мови, підтримують між ними відношення виду «ціле – частина», «рід – вид», «використовується для», «працює в» тощо. Надалі в ССП будемо розуміти під тезаурусом задачі наступне:

Тезаурус задачі – це множина термінів ПрО, необхідних для опису та вирішення задачі, для якої користувач намагається за допомогою ССП знайти певну інформацію. Для кожного з них може бути визначена їх вага, що дозволяє охарактеризувати важливість та пертинентність терміну для поточної задачі, та онтологія, з якої імпортовано відповідний термін.

Для кожного тезаурусу задачі існує хоча б одна онтологія ПрО, на якій він базується. В такому тезаурусі онтологічні зв'язки між термінами не відображаються явно, проте вони використовуються в процесі побудови тезауруса задачі за онтологією ПрО. Приміром, можна побудувати тезаурус, який містить визначену підмножину термінів $T_0 \subseteq X$ та терміни, пов'язані з ними відношенням $r \in R$.

Тезаурус задачі є персоніфікованим, тобто для рішення однієї й тої ж задачі різні користувачі можуть використовувати тезауруси, які значно різняться один від одного. Це залежить не тільки від тих природних мов, на які розраховує користувач, і не тільки від використаних онтологій, але й від індивідуальних переконань та переваг користувача в обраній ПрО.

Слід відмітити, що побудова тезауруса задачі є відносно складною та трудомісткою, тому доцільно виконувати цю операцію тільки в тих випадках, коли задача, що вирішується, відноситься до сфери постійних та складних інформаційних потреб користувача, а пошук відомостей для задоволення відповідної інформаційної потреби має враховувати багато умов та обмежень. Приміром, до таких задач може віднести пошук нової наукової літератури або інструментальних засобів з певного питання, що має відокремити визначені в тезаурусі напрямки досліджень.

Недоцільно будувати тезаурус задачі для одноразових запитів у сфері, де користувач не є експертом і тому не може сам враховувати достатню кількість зовнішніх знань. В такому випадку зусилля з побудови тезауруса будуть більшими від отриманого ефекту [17].

Інформацію, що міститься у тезаурусі задачі, можна поділити на дві частини – операційну та службову. Операційна інформація безпосередньо використовується у семантичному пошуку за допомогою такого тезауруса, а службова інформація описує шляхи побудови тезауруса, його інформаційні джерела, та може використовуватися для подальших операцій з цим тезаурусом (приміром, якщо відомо, з якої онтології експортовано певний термін, то можна експортувати й його підкласи або екземпляри).

Простий тезаурус задачі – це тезаурус, який базується на термінах однієї онтології ПрО.

Складений тезаурус задачі – це тезаурус, який базується на термінах двох або більш онтологій ПрО.

Складений тезаурус може бути побудований як поєднання двох або більше простих тезаурусів. Слід відмітити, що складений тезаурус задачі може містити терміни з однаковими іменами, отримані з різних онтологій, які не будуть еквівалентними.

Формальна модель простого тезауруса задачі $Th = \langle T, R_{\text{Th}}, O \rangle$, де T – множина термінів, $T \subseteq X$ а $R_{\text{Th}} \subseteq R$ – множина відношень між цими термінами, що використовувалися для побудови тезауруса. Множини T й R скінчені.

Формальна модель складеного тезауруса задачі

$$\text{Th} = < \text{T} = \left\{ \left(x_{i_j} \in X_i, O_i \right), i = \overline{1,n}, j = \overline{1,m_i} \right\},$$

$$\text{R}_{\text{Th}} = \bigcup_{i=1}^{n} \text{R}_{\text{Th}\,i}, O_1, ..., O_n >,$$

де T – множина пар термінів онтологій та посилань на відповідну онтологію, R_{Th} –

об'єднання множин відношень між термінами онтологій, що використовувалися для побудови тезауруса, та перелік всіх онтологій, що застосовуються для його побудови.

Перехід від онтологій до тезаурусів дозволяє значно спростити структуру знань, що обробляються, забезпечуючи прийнятну для практичної реалізації швидкість оброблення. Але використання оригінальних онтологій ПрО як основи для побудови тезауруса забезпечує доступність всіх наявних знань щодо ПрО, з яких користувач (вручну або автоматизовано) може обрати саме ту частку, що безпосередньо пов'язана з конкретною задачею.

Крім того, тезаурус задачі можна розглядати як спрощену та персоналізовану онтологію, що характеризує термінологічну основу поточної задачі користувача.

Природномовний опис задачі може містити:

– набір слів та словосполучень, які користувач вважає важливими для задачі (якщо користувач здатний самостійно побудувати весь такий набір, то можна вважати проблему побудови тезауруса вирішеною, але у більшості випадків цей набір треба поповнювати та фільтрувати);

– опис ІО, що входять до складу ситуації, що має стати результатом пошуку;

– природномовний текст, що характеризує цю ситуацію (постановку задачі, опис проекту; технічне завдання тощо);

– набір природномовних текстів, який користувач вважає пертинентними задачі (з цієї інформації потрібно здобути інформацію щодо того, які саме поняття ПрО є значущими для поточної задачі).

Таким чином, для побудови тезауруса задачі потрібно вирішити наступну задачу – побудувати метод, який забезпечує співставлення термінів онтології з природномовним текстом.

Повністю автоматизувати цю задачу неможливо, тому що в процесі її вирішення потрібно використовувати неформалізовані знання та переконання користувача щодо того, що саме його цікавить і може бути корисним для його проблеми. Але можливо спростити цю роботу, частково автоматизувавши пошук потенційно цікавих фрагментів.

Алгоритм побудови тезауруса задачі

Вхідними даними для побудови тезауруса задачі є природномовний опис задачі, онтологія ПрО, до якої користувач відносить свою задачу. Якщо результатом пошуку має стати не документ, а ситуація, то доцільно також використовувати онтологію ІО, в якій містяться відомості щодо властивостей, структури, елементів та екземплярів такого ІО. У випадку обробки конкретизованої ситуації наявність онтології ІО дозволяє чітко структурувати умови користувача [18].

Якщо обробляється персональна ситуація, то для надання знань щодо користувача може використовуватися внутрішня онтологія ССП та її клас "користувач". Це позбавляє користувача від введення відомостей, вже відомих ССП щодо його індивідуальних властивостей. При цьому можуть оброблятися як формальні характеристики, такі як вік, місце проживання, так і семантичні, такі як сфера інтересів або компетентність в певній ПрО.

У деяких складних випадках для опису ІО можуть використовувати ся не одна, а кілька онтологій.

Якщо якась одна онтологія не формалізує ПрО задовільно до задачі користувача, то можна використовувати сукупність онтологій, вважаючи їх незалежними, а їх множини термінів – такими, що не перетинаються (якщо дві онтології містять терміни з однаковими назвами, то ці терміни вважаються різними). Крім того, в онтології (чи множині онтологій) не всі елементи є корисними для конкретної задачі.

Eman 1. Відбір множини пертинентних онтологій.

Цей відбір виконується користувачем на основі його знань та переконань. Як правило, вибирати потрібно серед тих онтологій, які запропоновані в ССП. Це забезпечує не тільки якість онтологій для відображення знань ПрО, але й їх придатність для обробки засобами системи. У

більш загальному випадку можна шукати онтології у різноманітних репозиторіях, де властивості онтологій та їх домени охарактеризовані на семантичному рівні [19, 20].

При цьому можуть використовуватися як онтології ПрО пошуку, так і онтології ІО. Приміром, якщо користувач прагне знайти відомості щодо Web-сервісу для індуктивного здобуття знань з даних та їх візуалізації, то він може відібрати онтологію ПрО “Data Mining” та онтологію ІО “Web-сервіси”.

Для цього необхідно виконати аналіз пошукової ситуації, визначити, які ІО в ній використовуються та до яких ПрО належать умови пошуку.

Eman 2. Відбір у множині онтологій термінів, пертинентних задачі.

Такий відбір може виконуватися користувачем безпосередньо або будуватися за певними правилами (приміром, підкласи обраного класу чи його екземпляри, поняття на визначеній семантичній відстані від обраного поняття). За різними онтологіями можна побудувати кілька простих тезаурусів задачі а потім поєднати їх у складений тезаурус.

На цьому етапі створюється початковий варіант тезауруса задачі, який надалі потрібно вдосконалювати та поповнювати лінгвістичною інформацією, потрібною для співставлення з природномовними текстами.

Eman 3. Розробка лінгвістичної БЗ (ЛБЗ) тезауруса задачі, яка має містити фрагменти природномовного тексту, що відповідають його термінам.

Спочатку для кожного терміну тезауруса задачі до ЛБЗ додаються всі словоформи цього терміну, що відповідають різним відмінкам та множині слова або словосполучення, що використовується як його ім’я в обраній природній мові. Для цього можуть використовуватися словозміни (флексії), тобто системні засоби утворення різних форм того самого слова відповідно до його синтаксичних пов’язань з іншими словами в реченні або словосполученні без зміни його лексичного значення. Для цього в українській мові, приміром, можуть використовуватися закінчення змінних морфем-афіксів і постфіксів, зміни основи слова для слів з внутрішньою флексією (“стіл”–“столи”), зміни суфікса (“швикий”–“швидший”) або зміни префікса (“більший”–“найбільший”) та сполучення флексійної форми слова з прийменником (“атрибут”–“з атрибутом”) [20].

Словозміна застосовується при відмінюванні іменника, прикметника, займенника й числівника за відмінами, а в них відмінками, числами й родами та дієслова за відмінами, способами, часами, особами, числами, родами (в минулому часі й умовному способі), видами й станами тощо.

Побудувати такі конструкції на основі відповідних знань щодо правил словозміни досить просто для окремих слів, а для словосполучень потребує додаткового лінгвістичного аналізу, метою якого є визначення головного слова, що змінюється, та поділу пов’язаних з ним слів на ті, що змінюються, та ті, що залишаються незмінними за правилами природної мови. Наприклад, у словосполученні “онтологічна модель предметної області” основним є друге слово, змінюваним – перше, а незмінюваними – третє та четверте.

Інший метод здобуття словоформ для термінів тезауруса задачі базується на використанні Wiki-ресурсів [21, 22]. Такі ресурси можуть не тільки допомогти у формуванні онтології ПрО, що відповідає потребам користувача [23], але й стати джерелом лінгвістичної інформації. Якщо вдається знайти сторінку Вікіпедії, що відповідає певному терміну тезауруса, то з коду сторінок, що на неї посилаються можна імпортувати конструкції-посилання, пов’язані з цим терміном: [[ім’я сторінки| словоформа терміну, доречна в тексті]]. Приміром, [[технічна інформатика| технічній інформатиці]], [[Глушков_В_М |Віктора Михайловича Глушкова]]. Додаткову інформацію можна здобувати і з категоризації сторінок. Крім того, для семантичних Wiki-ресурсів можна аналізувати семантичні властивості термінів.

На наступному кроці до основного слова чи словосполучення, що визначає термін тезауруса, додаються його синоні-

мічні варіанти, актуальні для визначеної ПрО. Приміром, для словосполучення "онтологічна модель предметної області" – це словосполучення "онтологічний опис ПрО" та "онтологія домену". Слід відмітити, що для побудови синонімічних виразів використовуються не тільки відомості щодо природної мови, але й знання ПрО (приміром, експортовані з онтології ПрО відношення "є еквівалентним класом").

Ще один крок – переклад словосполучень, що відповідають термінам тезауруса, на інші мови, які знає користувач. Наприклад, для словосполучення "онтологічна модель предметної області" – це словосполучення "domain ontology" англійською мовою та "онтологическая модель предметной области" – російською. До отриманих перекладів застосовуються аналогічні операції побудови словоформ, що відповідають правилам обраних мов.

Наступний крок – варіанти порядку слів у словосполученні та його елементи, які припустимі в рамках кожної з обраних природних мов. Приміром, екземпляр певного класу "Іванов Олександр" може описуватися також як "Олександр Іванов" або "Іванов О.".

Остання операція цього етапу – користувач може вручну додати або видалити певні словосполучення, які він вважає відповідними термінам тезауруса. Після цього тезаурус задачі зберігається та може використовуватися для семантичного аналізу довільних природномовних текстів.

Eman 4. Анотування тезауруса задачі.

На цьому етапі доцільно створити опис тезауруса задачі, який описує як його формальні властивості (кількість термінів, оброблювані природні мови), так і семантичні характеристики – ПрО (через посилання на відповідні онтології), призначення, проблеми, для рішення яких він може застосовуватися. Це забезпечує повторне використання знань, що відображені в такому тезаурусі.

Eman 5. Вдосконалення тезауруса задачі.

У багатьох випадках користувач може застосовувати для власних цілей раніше створені тезауруси задач – як розроблені ним самим, так і розроблені іншими користувачами (якщо ті надають свої тезауруси у відкритий доступ).

Це доцільно у тих випадках, коли нова задача користувача є дещо зміненим варіантом задачі, що вирішувалася раніше (в такому випадку можна вручну відредагувати тезаурус, додавши чи видаливши кілька термінів). Якщо ж задача є узагальненням або уточненням попередніх задач, то доцільно застосовувати теоретико-множинні операції над тезаурусами, такі як перетин, об'єднання та доповнення. Приміром, до тезауруса ПрО "Онтологічний аналіз" в одному випадку можна додати тезаурус "Логічне виведення", а в іншому – "Візуалізація знань". Такий підхід значно зменшує час на модифікацію тезаурусів задач, але потребує створення досить докладних анотацій створюваних тезаурусів.

Eman 6. Визначення ваги термінів тезауруса задачі.

Після того, як формування множини термінів, що входять до складу тезауруса задачі, закінчується, користувачеві необхідно вказати, яку вагу для поточної задачі має кожен термін [24].

У найпростішому випадку (за замовчанням) можна вважати, що всі терміни мають однакову вагу, що дорівнює одиниці. Але на практиці зазвичай користувачеві зрозуміло, що деякі терміни значно важливіші за інші. Крім того, існують ситуації, коли наявність певного терміну в ІР свідчить про його низьку релевантність для задачі (хоча сам термін може бути важливим для ПрО в цілому). Приміром, користувачеві потрібно знайти мови подання онтологій, які не базуються на XML. Тоді термін "XML" буде присутній в онтології задачі, але з негативною вагою.

Такий тезаурус задачі надалі можна використовувати для того, щоб знаходити ІР, що найбільш пертинентні задачі користувача (ці ІР можуть бути знайдені довільною зовнішньою інформаційно-пошуковою системою (ІПС) за набором ключових слів, які вводить користувач для

попереднього опису своєї інформаційної потреби, або ж набір ІР може запропонувати сам користувач – як з зовнішніх джерел, таких як тематична бібліотека або сайт, так і з внутрішніх архівів).

Надалі саме з цих ІР слід здобувати знання, що будуть корисні для побудови пошукової ситуації, але методи здобуття таких знань знаходяться поза розглядом даної роботи.

Використовувати тезаурус задачі можна двома способами: 1) для кожного з аналізованих ІР будувати тезаурус, а потім порівнювати його з тезаурусом задачі, або 2) безпосередньо порівнювати тезаурус задачі з контентом кожного ІР, використовуючи для цього співставлення знання, накопичені у ЛБЗ.

Перший підхід доцільно використовувати, якщо ІР з відносно невеликої фіксованої множини багаторазово аналізуються на пертинентність потребі користувача в фіксованій ПрО та для однієї або подібних задач. Приміром, це можна використовувати для індексації та швидкого пошуку у власних архівах або у власній електронній бібліотеці. Тоді можна побудувати тезауруси всіх ІР для певної ПрО і використовувати їх багаторазово.

Надалі будемо розуміти під *зваженим тезаурусом задачі* спрощений варіант складеного тезауруса, в якому інформація щодо походження кожного терміну не прив'язується до певної онтології.

$$Th_w = < X, W, \{\cup O_z\} >,$$

таке, що $x_i \in X$, $i = \overline{1,n}$ – термін тезауруса задачі, w_i – вага цього терміну в тезаурусі задачі, а $\{\cup O_z\}$ – множина онтологій, які використовувалися при побудові цього тезауруса. X є скінченою непорожньою множиною.

Кожному такому тезаурусу задачі відповідає ЛБЗ L така, що для кожного $x_i \in X$ існує $L_i = \{l_{i_j}, j = \overline{1, m_i}\}$ – непорожня скінчена множина фрагментів природномовних текстів, що відповідає терміну тезауруса задачі $x_i \in X$. Ці множини не перетинаються:

$$\forall i \neq k \; L_i \cap L_k = \varnothing.$$

Якщо ж інформація за ключовими словами імпортується зовнішньою ІРС з інформаційного простору Web, то через динамічність цього середовища в будь-якому разі потрібно кожного разу заново аналізувати контент кожного ІР, і тому більш придатним є другий підхід, який детальніше розглянутий далі.

Використання тезауруса задачі для фільтрації ІР

Алгоритм персоніфікованої семантичної фільтрації ІР за допомогою тезауруса задачі складається з наступних кроків:

Користувач вводить запит, ідентифікуючи свою інформаційну потребу за допомогою набору ключових слів. Потрібно відмітити, що на цей момент система семантичного пошуку (ССП) має вже певні відомості про цього користувача, отримані в процесі взаємодії з ним та відображені в його моделі [25, 26]. Приміром, для цього може використовуватися онтологічна модель, що персоніфікує взаємодію користувача з різними інтелектуальними застосуваннями [27, 28].

Запит передається до зовнішнього пошукового механізму, який відбирає з набору ІР ті, що містять введені ключові слова (у виродженому випадку набір ключових слів – порожня множина, і подальшому аналізу підлягає весь набір приступних ІР). Результати виконання запиту – n посилань на ІР і їхні короткі описи

$$I = \{< Ref_r, D_r >\}, \; r = \overline{1,p},$$

де Ref_r – ідентифікатор (приміром, http-адреса для ІР, знайдених в Web) відповідного ІР, а d_r – інформація про цей ІР, що зовнішня ІПС надає користувачу у відповідь на запит.

Якщо множина I не порожня, тобто вдалося знайти хоча б один ІР ($p \geq 1$), то потрібно встановити порядок, в якому пропонувати користувачу відомості про знайдений ІР. Тоді для всіх ІР з цієї множини Ref_r, $r = \overline{1,p}$ потрібно виконати

наступну процедуру – спробувати знайти в них фрагменти тексту з множини L, що відповідають кожному з термінів тезауруса задачі. Можна проводити швидкий аналіз – пошук лише в d_r або повний аналіз – пошук в повному контенті ІР. Перший варіант значно швидший, але другий дає значно більш релевантні результати.

В результаті цього аналізу для кожного r-го ІР формується вектор співставлення

$$a_r = < a_{r_1} ..., a_{r_n} >$$

такий, що a_{r_i}, $i = \overline{1,n}$ – кількість співставлень у тексті r-го ІР з і-м терміном тезауруса задачі.

Співставлення виконується з використанням ЛБЗ: якщо l_{i_j}, що співвідноситься з терміном тезаурусу задачі $x_i \in X$, входить до складу r-го ІР, тоді a_{r_i} збільшується на 1 (у спрощеному варіанті порівняння елементів ЛДЗ виконується з контентом короткого опису відповідного ІР).

Значущість кожного ІР для задачі оцінюється як функція від цього вектора та вектора ваги кожного терміну.

Для більшості задач може використовуватися наступна формула:

$$f_r = \sum_{i=1}^{n} a_{r_i} * w_i . \qquad (1)$$

Якщо потрібно порівнювати оцінки ІР для різних тезаурусів задачі (приміром, для оцінки досліджування ПрО та як її відображення у різних наборах ІР – електронних бібліотеках, сайтах тощо), то виникає необхідність у використанні замість (1) нормованої оцінки. Але таку нормовану оцінку можна використовувати тільки в тому випадку, якщо не застосовуються негативні ваги термінів тезауруса задачі.

$$fn_r = \sum_{i=1}^{n} \frac{a_{r_i} * w_i}{\sum_{i=1}^{n} a_{r_i} * \sum_{i=1}^{n} w_{r_i}} . \qquad (2)$$

Значення оцінки (2) завжди буде знаходитися в діапазоні між 0 та 1, в той час як значення оцінки (1) може приймати довільне значення, як позитивне, так і негативне.

Отримані оцінки використовуються для перевпорядкування знайдених ІР: користувач отримує у першу чергу ІР з більш високими коефіцієнтами відповідності поточній задачі користувача.

Програмна реалізація

Вищезапропонований метод побудови тезауруса задачі застосовується в системі семантичного пошуку "МАІПС" [29].

Ця система має забезпечити виконання складних багаторазових запитів у спеціалізованих ПрО, пов'язаних з професійними або науковими інтересами користувачів. Запити таких користувачів можуть повторюватися від сеансу до сеансу або змінюватися, але залишатися у рамках певної ПрО пошуку, в якій користувачі є експертами. Система надає користувачу ті результати пошуку, що відносяться до предметних областей, які його цікавлять і відповідають його інформаційним потребам.

"МАІПС" дозволяє зберігати і повторно виконувати запити, зберігати формальний опис області інтересів користувача у вигляді тезаурусів задачі та онтологій ПрО. Отримані результати перевпорядковуються з урахуванням цих знань, а також персоніфікованого індексу легкості читання природномовних ІР [30, 31].

"МАІПС" не замінює собою ІПС. Вона є посередником між користувачем та існуючими засобами пошуку. Її призначення – зробити звертання користувача до ІР більш ефективним, зручним та швидким.

Крім того, у "МАІПС" при профілюванні користувачів використовується специфічний для природномовних ІР критерій оцінювання – складність тексту для розуміння. Особливістю системи є використання оригінального знання-орієнтованого алгоритму, що дозволяє визначити складність розуміння тексту для конкретного користувача (для того, щоб

формалізувати рівень обізнаності користувача в певних ПрО, використовуються тезауруси тих предметних областей, що цікавлять користувачів).

Наукова новизна "МАІПС" полягає в інтегрованому використанні онтологічного подання знань, агентної парадигми та технологій Semantic Web для пошуку інформації на семантичному рівні.

Основні технології та методи, інтегровані в "МАІПС":

- застосування технологій Semantic Web [32]: використання OWL [33] для інтероперабельного представлення онтологій та тезаурусів, що описують ПрО;
- реалізація теоретико-множинних операцій над тезаурусам;
- методи генерації тезаурусів за природномовними текстами;
- використання технологій Web 2.0 [34] (хмар тегів – для візуалізації пошукових тезаурусів; соціальних сервісів – для взаємодії між користувачами);
- оригінальні алгоритми впорядкування інформаційних ресурсів, знайдених системою, з урахуванням ваги онтологічних термінів;
- використання критеріїв оцінки читабельності тексту для пошуку інформації, що відповідає персональним потребам користувача;
- використання методів індуктивного виведення для узагальнення досвіду роботи "МАІПС";
- застосування мультиагентного підходу до створення моделі інтелектуальної інформаційно-пошукової системи та представлення компонентів системи як інтелектуальних BDI-агентів для формалізації поведінки системи в цілому [35];
- використання парадигми інтелектуальних Web-сервісів для опису функцій агентів системи, що дозволяє їх інтероперабельне багаторазове використання [36].

Основою "МАІПС" є технології Semantic Web, зокрема, мова представлення онтологій OWL і засоби його обробки. Для представлення знань щодо того, що цікавить користувача, використовуються онтології ПрО та базовані на них тезауруси задач. При цьому тезаурус будується користувачем за відповідною онтологією самостійно, а онтологія обирається з набору онтологій, запропонованих на сайті розробниками системи.

В процесі розвитку "МАІПС" виникла потреба в підключенні репозиторію онтологій, щоб користувачі могли повторно використовувати знання ПрО, доступні в Web. При цьому пошук може здійснюватися не тільки за ключовими словами, а і за іншими важливими властивостями онтологій – обсяг, розробники, кількість та типи відношень, базові DL, діалекти мов подання тощо. Тому надалі представляється доцільним реалізувати в "МАІПС" засоби взаємодії з репозиторіями онтологій, що підтримують пошук потрібної користувачу онтології, виявлення схожих на обрану користувачем онтологій, а також зіставлення побудованого користувачем тезауруса з іншими онтологіями і тезаурусами.

"МАІПС" базується на онтологічній моделі, що описує семантику взаємодії користувачів і ресурсів "МАІПС" в інформаційному просторі Web. Ця модель також може застосовуватися у побудові тезауруса задачі, якщо пошукова ситуація є персональною (тобто стосується саме даного користувача) [37].

Отримавши у відповідь від зовнішньої ІПС набір інформаційних ресурсів, "МАІПС" намагається здобути з них потрібні користувачеві відомості. У найпростішому випадку, якщо потрібний користувачеві ІО є документом (можливо, певного типу), система перевпорядковує отримані посилання на ІР з урахуванням персональних особливостей користувача та збережених у БД системи відомостей про ці ІР. У більш складних випадках з ІР здобуваються відомості про властивості атрибутів шуканого ІО. Приміром, якщо користувачеві був потрібен Web-сервіс з певними властивостями, то він отримає опис вхідних і вихідних даних наявних Web-сервісів, що відповідають його умовам та опис їх роботи.

"МАІПС" реалізована як серверне Інтернет-застосування мовою PHP версії

5.0. Для збереження внутрішніх даних використовується XML (надалі планується використання СУБД MySQL). Онтології зберігаються у форматах RTF і OWL, тезауруси – у форматі XML.

При редагуванні вже створеного тезауруса можна вводити вагу різних термінів, що позначають їх важливість для пошуку (як позитивну, так і негативну), цілі числа від – 9 до + 9. Ця інформація дозволяє відображати тезаурус у вигляді хмари тегів (червоним кольором відмічені терміни з негативною вагою, синім – з позитивною, розмір шрифта відображає числові значення ваги) (рис. 1).

Рис. 1. Створення тезауруса задачі в "МАІПС"

Результати пошуку, отримані від зовнішньої ІПС (наприклад, від Google) перевпорядковуються за допомогою критерію (1). Крім цього, "МАІПС" пропонує користувачеві адресу та анотацію кожного текстового ІР та оцінки легкості його читання (рис. 2), обчислені з урахуванням термінів тезауруса задачі – слова та словосполучення, пов'язані з термінами цього тезауруса, не вважаються складними для користувача.

Слід зазначити, що для представлення знань у "МАІПС" використовуються онтології, які можна поділити на дві окремі групи – *внутрішні* і *зовнішні*. Внутрішні онтології створюються безпосередньо розроблювачами "МАІПС" і можуть поповнюватися в процесі взаємодії "МАІПС" із користувачами. Основною особливістю такі онтології є те, що розроблювачам цілком відома їхня структура і вміст, тому можна прогнозувати кінцівку обчислень.

Рис. 2. Впорядкування результатів пошуку в системі "МАІПС"

Зовнішні онтології дозволяють інтегрувати в "МАІПС" динамічні і розподілені знання, доступ до яких забезпечує Web. Пошук таких онтологій може здійснюватися в різноманітних репозиторіях, чи ж вони можуть бути сформовані в процесі роботи користувача з іншими інтелектуальними додатками. У загальному випадку інформація про складність структури таких онтологій, про те, на яких дескриптивних логіках вони базуються, наскільки повними є такі знання, відсутня, і це не дозволяє прогнозувати час роботи алгоритмів для глибокого аналізу й обробки таких онтологій. Тому в "МАІПС" для обробки зовнішніх онтологій застосовуються спрощені алгоритми, що використовують тільки найбільш прості властивості онтологій (наприклад, обробляються тільки відношення "клас-підклас"). Для більшості задач інформаційного пошуку цього досить, але для пошуку сукупностей складних ІО – наприклад, для дослідження і композиції семантичних Web-сервісів чи формування мультиагентних систем – потрібно використовувати більш складні структурні зв'язки. Саме для таких ситуацій до складу "МАІПС" входять онтології складних ІО.

Висновки

Використання онтологічного аналізу для інтелектуалізації пошукових процедур забезпечує створення семантичної надбудови над традиційними інформаційно-пошуковими системами та дозволяє використовувати знання щодо індивідуальних інформаційних потреб користувачів.

Запропонований у роботі підхід орієнтований на користувачів з постійними та складними інформаційними інтересами, такими як науково-дослідницька діяльність у певній сфері. Це припускає здатність таких користувачів до аналізу відповідної предметної області, обізнаності в її основних поняттях та зв'язках між ними та їх потребу в створенні складних багаторазових запитів.

Для персоніфікації пошуку інформаційних об'єктів та інформаційних ресурсів, які містять відомості, потрібні користувачам для розв'язання поточних задач, запропоновано будувати та застосовувати тезауруси таких задач. Це дозволяє використовувати знання щодо предметної області пошуку та структури інформаційних об'єктів, які мають самі користувачі або які отримуються з відповідних онтологій. Визначення термінів, пов'язаних з семантичним пошуком, дозволяє більш чітко формулювати проблеми, пов'язані з пошуком інформації у відкритому середовищі Web.

Програмна реалізація запропонованого підходу підтверджує ефективність його практичного використання.

1. *Amerland D.* Google Semantic Search: Search Engine Optimization (SEO) Techniques That Gets Your Company More Traffic, Increases Brand Impact and Amplifies Your Online Presence. – Que Publishing. – 2013. – 230 p.
2. *Lawrence S.* Context in the Web Search. – http://citeser.nj.nec.com/ lawrence00context.html.
3. *Berry M.W.* Survey of text mining // Computing Reviews 45.9, 2004. – 244 p.
4. *Andon P., Deretsky V.* Approach to Automatic Creation of Ontology from Documents for Improving Existent Information Retrieval // Proc.of 2-nd Balkan Conference in Informatics (BCI'2005) November 17–19, 2005. – P. 236–241.
5. *Cimiano P.* Ontology Learning and Population from Text: Algorithms, Evaluation and Applications. Ontology Learning and Population from Text: Algorithms, Evaluation and Applications. – Springer-Verlag New York, Inc. Secaucus, NJ, USA, 2006. – 347 p.
6. *Fensel D., Harmelen F., Horrocks I., McGuinness D., Patel-Schneider P.* OIL: An Ontology Infrastructure for the Semantic Web. – http://www.cs.man.ac.uk/ %7Ehorrocks/Publications/download/2001/ IEEE-IS01.pdf.
7. *Gruber T.R.* What is an Ontology? – http://www-ksl.stanford.edu/kst/what-is-an-ontology.html.
8. *Uschold M., Grüninger M.* Ontologies: Principles, Methods and Applications // Knowledge Engineering Review 11(2), 1996. – P. 93–155.
9. *Guarino N.* Formal Ontology in Information Systems // Formal Ontology in Information Systems. Proceedings of FOIS'98, Trento, Italy, 1998. – P. 3–15.
10. *Боргест Н.М., Коровин М.Д.* Онтологии: современное состояние, стандарты, средства поддержки. Уч. пособие. СРАУ. – Самара, 2013. – 84 с.
11. *Клещев А.С., Артемьева И.Л.* Математические модели онтологий предметных областей. Часть 1. Существующие подходы к определению понятия «онтология» // Научно-техническая информация, серия 2. – 2001. – С. 20–27.
12. *ISO* 25964-1:2011, Thesauri and interoperability with other vocabularies. Part 1: Thesauri for information retrieval. – Geneva: International Organization for Standards, 2011.
13. *Нариньяни А.С.* Кентавр по имени ТЕОН: Тезаурус + Онтология. – http://www.artint.ru/articles/narin/teon.htm.
14. *Браславский П.И., Гольдштейн С.Л., Ткаченко Т.Я.* Тезаурус как средство описания систем знаний // Информационные процессы та системы. – 1997. – № 11, Серия 2. – С. 16–22.
15. *Величко В., Волошин П., Свитла С.* Автоматизированное создание тезауруса терминов предметной области для локальных поисковых систем. – www.foibg.com/ibs_ isc /ibs-15/ibs-15.pdf.
16. *Добров Б.В., Иванов В.В., Лукашевич Н.В., Соловьев В.Д.* Онтологии и тезаурусы: модели, инструменты, приложения. – Элект-

ронная книга, 2006. – 220 с. – http://catscpp.googlecode.com/svn-history/r146/trunk/diploma/materials/ontologies_tesauruses.pdf.
17. *Gladun A., Rogushina J.* Use of Semantic Web Technologies and Multilinguistic Thesauri for Knowledge-Based Access to Biomedical Resources // International Journal of Intelligent Systems and Applications. – 2012, N 1. – P. 11–20. – http://www.mecs-press.org/ijisa/ijisa-v4-n1/IJISA-V4-N1-2.pdf
18. *Gladun A., Rogushina J.* Use of Semantic Web technologies in design of informational retrieval systems // in Book "Building and Environment", Nova Scientific Publishing, New-York, USA. – 2009. – P.89–103.
19. *Hartmann J., Palma R., Gomez-Perez A.* Ontology Repositories // in Book "Handbook on Ontologies", Edt. by S. Staab, R. Studer, Springer, 2009. – P. 551–572.
20. *Baclawski K., Schneider T.* The Open Ontology Repository Initiative: Requirements and Research Challenges // Proceedings of the 8th International Semantic Web Conference ISWC-2009, October 25, 2009, USA.
21. *Лесько О.Н., Рогушина Ю.В.* Использование онтологий для анализа семантики естественно-языковых текстов // Проблеми програмування. – 2009. – № 3. – С. 59–65.
22. *Leuf B., Cunningham W.* The Wiki way: collaboration and sharing on the Internet. – 2001. – http://www.citeulike.org/group/13847/article/7659081.
23. *Wagner C.* Wiki: A technology for conversational knowledge management and group collaboration // The Communications of the Association for Information Systems. – 2004. – V. 13(1). – P. 264–289.
24. *Рогушина Ю.В., Гладун А.Я.* Семантическая Википедия как источник онтологий для интеллектуальных поисковых систем // В кн.: Advanced Research in Artificial Intelligence. International Book Series "Information Science and Computing". ITHEA, Sofia, 2008. – P. 172–178.
25. *Гладун А.Я., Рогушина Ю.В.* Основи методології формування тезаурусів з використанням онтологічного та мереологічного аналізу // Искусственный интеллект. – 2008. – № 5. – С.112–124.
26. *Jansen B.J., Spink A., Saracevic T.* Real life, real users, and real needs: a study and analysis of user queries on the Web. – http://citeseer.nj.nec.com/jansen00real.html.
27. *Kobsa A.* User modeling: recent work, prospects and hazards. – http://zeus.gmd.de/~kobsa/papers/1993-aui-kobsa.pdf.
28. *Рогушина Ю.В.* Разработка средств персонификации интеллектуальных Web-приложений // Материалы V Международной научно-технической конференции «Открытые семантические технологии проектирвоания интеллектуальных систем» OSTIS-2015. – Минск: БГУИР, 2015. – С. 265–270.– http://www.conf.ostis.net/images/8/8b/OSTIS-2015.compressed.pdf.
29. *Rogushina J., Gladun A.* Ontology-based competency analyses in new research domains // Journal of Computing and Information Technology. – 2012. – V. 20, N. 4. – P. 277–293.
30. *Рогушина Ю.В., Гришанова І.Ю.* Літературний твір наукового характеру "Модель мультиагентної інформаційно-пошукової системи "МАІПС"("Модель МАІПС"). – Свідоцтво про реєстрацію авторського права на твір № 32068.
31. *Flesch* Reading Ease Readability Formula. – http://oleandersolutions.com/fleschreadingease.html.
32. *McLaughlin H.* SMOG grading a new readability formula // Journal of Reading. – 1969. – N 22. – P. 639–646.
33. *Davies J., Fensel D., van Harmelen F.* Towards the Semantic Web: Ontology-driven knowledge management // John Wiley & Sons Ltd,, England. – 2002. – 288 p.
34. *OWL* 2 Web Ontology Language Document Overview. W3C. 2009. – http://www.w3.org/TR/owl2-overview/.
35. *McCann R., Shen W., Doan A.* Matching schemas in online communities: A web 2.0 approach // Proc. of ICDE, 2008.
36. *Rao A.S., Georgeff M.P.* Modeling rational agents within a BDI-architecture // In R. Pikes and E. Sandewall, eds.. Proc. of Knowledge Representation and Reasoning (KR&R-91), Morgan Kaufmann Publishers: San Mateo, CA, April 1991. – P. 473–484.
37. *Cowles P.* Web Services and the Semantic Web. – http://ezolin.pisem.net/logic/ws_and_sw_rus.html.

References

1. *Amerland D.* Google Semantic Search: Search Engine Optimization (SEO) Techniques That Gets Your Company More Traffic, Increases Brand Impact and

Amplifies Your Online Presence. – Que Publishing, 2013. – 230 p.
2. *Lawrence S.* Context in the Web Search. – http://citeser.nj.nec.com/ lawrence00context.html.
3. *Berry M.W.* Survey of text mining // Computing Reviews 45.9, 2004. – 244 p.
4. *Andon P., Deretsky V.* Approach to Automatic Creation of Ontology from Documents for Improving Existent Information Retrieval // Proc.of 2-nd Balkan Conference in Informatics (BCI'2005) November 17–19, 2005. – P. 236–241.
5. *Cimiano P.* Ontology Learning and Population from Text: Algorithms, Evaluation and Applications. Ontology Learning and Population from Text: Algorithms, Evaluation and Applications. – Springer-Verlag New York, Inc. Secaucus, NJ, USA, 2006. – 347 p.
6. *Fensel D., Harmelen F., Horrocks I., McGuinness D., Patel-Schneider P.* OIL: An Ontology Infrastructure for the Semantic Web. – http://www.cs.man.ac.uk/ %7Ehorrocks/Publications/download/2001/IEEE-IS01.pdf.
7. *Gruber T.R.* What is an Ontology? – http://www-ksl.stanford.edu/kst/what-is-an-ontology.html.
8. *Uschold M., Grüninger M.* Ontologies: Principles, Methods and Applications // Knowledge Engineering Review. – 11(2). – 1996. – P. 93–155.
9. *Guarino N.* Formal Ontology in Information Systems // Formal Ontology in Information Systems. Proceedings of FOIS'98, Trento, Italy, 1998. – P. 3–15.
10. *Borgest N.M., Korovin M.D.* Ontologies: modern state, standards and support means, SRAU. – Samara, 2013. – 84 p. (in Russian)
11. *Keschev A.S., Artemieva I.L.* Mathematical models of domain ontologies. Part 1. Existing approaches for definition of «ontology» concept Science-technical information, series 2, 2001. – P. 20–27. (in Russian)
12. *ISO* 25964-1:2011, Thesauri and interoperability with other vocabularies. Part 1: Thesauri for information retrieval – Geneva: International Organization for Standards, 2011.
13. *Nariniani A.C.* Centaurus named TEON: thesaurus + ontology. – http://www.artint. ru/articles/narin/teon.htm. (in Russian)
14. *Braslavsky P.I., Goldshtein S.L., Tkachenko T.Ya.* Thesaurus as a mean of knowledge system deskribing // Information processes and systems, 1997. – N 11, series 2. – P. 16–22. (in Russian)
15. *Velichko V., Voloshin P., Svitla S.* Automated creation of thesaurus of domain terms for local retrieval systems. – www.foibg.com/ibs_isc/ibs-15/ibs-15.pdf. (in Russian)
16. *Dobrov B.V., Ivanov V.V., Lukashevich N.V., Soloviev V.D.* Ontologies and thesauri: models, instruments, applications. 2006. – 220 p. – http://window.edu.ru/resource/ 583/64583/files/Dobrov_978-5-9963-0007-5%2F1-2-3_cC0007-5.pdf. (in Russian)
17. *Gladun A., Rogushina J.* Use of Semantic Web Technologies and Multilinguistic Thesauri for Knowledge-Based Access to Biomedical Resources // International Journal of Intelligent Systems and Applications. – 2012. – N 1. – P. 11–20. – http://www.mecs-press.org/ijisa/ijisa-v4-n1/IJISA-V4-N1-2.pdf
18. *Gladun A., Rogushina J.* Use of Semantic Web technologies in design of informational retrieval systems // in Book "Building and Environment", 2009 Nova Scientific Publishing, New-York, USA. – P. 89–103.
19. *Hartmann J., Palma R., Gomez-Perez A.* Ontology Repositories // in Book "Handbook on Ontologies", Edt. by S.Staab, R.Studer, Springer, 2009. – P. 551–572.
20. *Baclawski K., Schneider T.* The Open Ontology Repository Initiative: Requirements and Research Challenges//Proceedings of the 8th International Semantic Web Conference ISWC-2009, October 25, 2009, USA.
21. *Lesko O.N., Rogushina Y.V.* Use of ontologies for analysis of natural language texts semantics // Problems in programming. – 2009. N 3. – P. 59–65. (in Russian)
22. *Leuf B., Cunningham W.* The Wiki way: collaboration and sharing on the Internet. – 2001. – http://www.citeulike.org/group/ 13847/article/7659081.
23. *Wagner C.* Wiki: A technology for conversational knowledge management and group collaboration // The Communications of the Association for Information Systems. – 2004. – V. 13(1). – P. 264–289.
24. *Rogushina Y.V., Gladun A.Ya.* Semantic Wikipedia as a source of ontologies for intelligent retrieval systems // Advanced Research in Artificial Intelligence. International Book Series "Information Science and Computing". ITHEA, Sofia, 2008. – P. 172–178. (in Russian)
25. *Gladun A.Ya., Rogushina Y.V.* Metodology bases of thesauri creation with use of ontological and mereological analysis // Artificial intelligence. – 2008. – N 5. – P. 112–124. (in Ukrainian)

26. *Jansen B.J., Spink A., Saracevic T.* Real life, real users, and real needs: a study and analysis of user queries on the Web. – http://citeseer.nj.nec.com/jansen00real.html.
27. *Kobsa A.* User modeling: recent work, prospects and hazards. – http://zeus.gmd.de/~kobsa/papers/1993-aui-kobsa.pdf.
28. *Rogushina Y.V.* Design of personification means of intelligent Web applications // Proc. of V scientific and technical conf. OSTIS-2015. – Minsk, 2015. – P. 265–270. (in Russian) – http://www.conf.ostis.net/images/8/8b/OSTIS-2015.compressed.pdf.
29. *Rogushina J., Gladun A.* Ontology-based competency analyses in new research domains // Journal of Computing and Information Technology. V.20, N. 4, 2012. – P. 277–293.
30. *Rogushina Y.V., Grishanova I.Y.* Literary work "Model of multiagent information retrieval system MAIPS ("MAIPS model"). – Copyright certificate of product registration N 32068. (in Ukrainian).
31. *Flesch* Reading Ease Readability Formula. – http://oleandersolutions.com/fleschreadingease.html.
32. *McLaughlin H.* SMOG grading a new readability formula // Journal of Reading. – 1969. – N 22. – P.639–646.
33. *Davies J., Fensel D., van Harmelen F.* Towards the Semantic Web: Ontology-driven knowledge management. – John Wiley & Sons Ltd,, England. – 2002. – 288 p.
34. *OWL* 2 Web Ontology Language Document Overview. W3C. 2009. – http://www.w3.org/TR/owl2-overview/.
35. *McCann R., Shen W., Doan A.* Matching schemas in online communities: A web 2.0 approach // Proc. of ICDE, 2008.
36. *Rao A.S., Georgeff M.P.* Modeling rational agents within a BDI-architecture // In R. Pikes and E. Sandewall, eds.. Proc. of Knowledge Representation and Reasoning (KR&R-91), Morgan Kaufmann Publishers: San Mateo, CA, April 1991. – P. 473–484.
37. *Cowles P.* Web Services and the Semantic Web. – http://ezolin.pisem.net/logic/ws_and_sw_rus.html.

Одержано 07.12.2015

Про автора:

Рогушина Юлія Віталіївна,
кандидат фізико-математичних наук,
старший науковий співробітник.
Кількість наукових публікації в українських виданнях – 100.
Кількість наукових публікацій в іноземних виданнях – 25.
Індекс Гірша – 10,
http://orcid.org/0000-0001-7958-2557.

Місце роботи автора:

Інститут програмних систем
НАН України,
03181, Київ-187,
проспект Академіка Глушкова, 40,
Тел.: 066 550 1999.
E-mail: ladamandraka2010@gmail.com

УДК 004.822, 681.3, 519.81

Е.П. Ильина

МЕТОДЫ И МОДЕЛИ ИСПОЛЬЗОВАНИЯ ЭКСПЕРТНО-АНАЛИТИЧЕСКОГО ЗНАНИЯ ДЛЯ ПОДДЕРЖКИ ПРИНЯТИЯ РЕШЕНИЙ В ОРГАНИЗАЦИИ. ЧАСТЬ 1. МОДЕЛИ ЗНАНИЯ О РЕШЕНИЯХ

В работе представлен аппарат моделей корпоративного знания организации для экспертно-аналитической поддержки принятия ее решений. Формализован концепт Организационное решение с учетом всех стадий жизненного цикла последнего: от постановки проблемы до анализа результатов. Описаны модели категорий концептов из состава онтологии поддержки принятия решений и модель поля решений организации. Система отношений между концептами онтологии составляет предмет рассмотрения следующей части статьи, наряду с формализмом гармонизации поля решений.

Ключевые слова: организационное решение, онтология сферы принятия решений, поле решений организации, категория концептов онтологии, частичное определение концепта, концептуально неполное знание.

Постановка задачи

Менеджмент организации при сегодняшних экономических реалиях должен искать нестандартные ответы на вызовы, которые не возникали или, по крайней мере, не доминировали ранее. В условиях долгосрочного глобального кризиса мировой экономики эти вызовы приобретают системный характер [1].

Одним из таких ответов стало создание и внедрение моделей управления, центрированных на системе решений организации [2, 3]. Следствием предложенного подхода является ряд специальных требований, адресованных самой системе организационных решений и средствам их поддержки. Требования к рациональной эффективности и качеству решений должны быть выявляемы и исследуемы на формальном уровне, а средства автоматизированной поддержки – ориентированы на их удовлетворение [4].

Одним из важнейших аспектов достижения качества решений организации в среде актуальных вызовов [1–3] является гармонизация разноаспектных противоречий. Их возникновение неизбежно сопутствует необходимой для менеджмента интеграции интеллектуальных ресурсов – концептуальных, целеопределяющих, задачных, информационных, методических. Кроме того, распределенность процессов принятия решений и динамичность потребности в их принятии выводят на первый план проблему гармонизации целостной системы решений, действующих в организации, при появлении нового решения.

Этот аспект становится одним из основных при осуществлении выбора вариантов воздействия на объекты управления, особенно, если речь идет о существенном характере влияний и существенных рисках, связанных с потерей эффективности. Необходимыми для решения данной проблемы являются формализация представления организационных решений, построение модели корпоративного поля решений, аксиоматизация критериев его гармоничности, а также предоставление их метризованной оценки. Создаваемый аппарат должен оперировать знаниями с разным уровнем определенности и знаниями, отражающими разные концептуальные взгляды, формируемые деятельностью их носителей.

Данная работа посвящена решению этих задач на основе использования специального аппарата онтологических моделей корпоративного решения, развивающего соответствующий формализм, предложенный ранее в [4, 5].

Первая часть включает рассмотрение структуры системы знаний организации и моделей концептов, составляющих онтологическую основу поля решений.

Поставленной задачей является предоставление формализма, который использован далее в данной работе для создания средств анализа решений организации. Он также может использоваться для распространения идей онтологизации решений проекта [6] на такой специфический объект как организационное решение. В следующей части будут рассмотрены система отношений между концептами, модели анализа гармоничности системы решений организации, а также аппарат интеграции концептуально различных точек зрения.

Концептуальный взгляд на организационное решение

Классические представления о решении могут быть сведены к его определению в виде

$$C:((\{A\},M),K)\to SA, \qquad (1)$$

где C – функция выбора, $\{A\}$ – множество альтернатив, M – модель предпочтений, K – массив используемых знаний об альтернативах, $SA\in\{A\}$ – выбранная альтернатива.

Собственно организационное решение D, которое является базовым концептом управления современной организации, соответствует композиции последовательного применения ряда операций, осуществляемых ЛПР и привлекаемыми экспертами

$$D=E_1*E_2*E_3*E_4*C*E_5\ , \qquad (2)$$

где E_1 – формализация проблемной ситуации; E_2 – целеполагание по поводу проблемной ситуации; E_3 – генерация множества альтернатив; E_4 – определение множества проектов выбора, произведенного разными носителями экспертных знаний; C – выбор (см. (1)), включающий интеграцию проектов и ее обоснование; E_5 – анализ результатов выполнения решения.

Помимо введения дополнительных отображений, по сравнению с (1), которые должны быть реализованы процессом принятия решения, в (2) производится фиксация вида альтернативы A как целевого воздействия GA на объект управления

$$GA:(ST_1(OB),PAR)\to ST_2(OB), \qquad (3)$$

где $ST_1(OB), ST_2(OB)$ – начальное и конечное состояния объекта управления OB;

PAR – параметры воздействия $PAR=<M,\mathrm{Res},S,T,F,RI>$, включающие реализующее мероприятие M, ресурсы Res, субъекты с их полномочиями S, срок выполнения T, факторы F, влияющие на успешность, а также нежелательные побочные влияния RI.

Для того, чтобы организационное решение рассматривалось в контексте современных требований к его качеству, на смену (2) должна прийти композиция, реализующая следующее преобразование

$$DA:(ST_1(OB),ST_1(ACA),PAR)\to \\ \to(ST_2(OB),ST_2(ACA)). \qquad (4)$$

Здесь ACA – актуальная среда управленческой модели организации состава

$$ACA=<OG,ZI,ODF>, \qquad (5)$$

где OG – система целей организации, ZI – система интересов стейкхолдеров, ODF – система решений.

При таком подходе появляется возможность преодоления следующих угроз качеству организационного решения: локальности критериев выбора (не учитывающих побочных влияний, оказываемых решением); предвзятости (игнорирования интересов, взглядов и мнений отдельных бизнес-ролей); отсутствия обратной связи (невозможности коррекции воздействия по результатам выполнения).

Далее предлагается формализм представления и анализа знаний организаций, поддерживающий процессы принятия решений, рассматриваемых в трактовке (4).

Модели знаний для экспертно-аналитического сопровождения системы организационных решений

Для обеспечения качества корпоративных организационных решений, системные и оценочные аспекты которого рассматривались в [4], необходимо создание экспертно-аналитической методологии, поддерживаемой средствами автоматизации. Такое эффективное экспертно-аналитическое сопровождение должно рекомендовать и предоставлять средства поддержки каждого из этапов [7, 8], соответствующих операциям из (2). Среди этих средств должны быть представлены компоненты:

– интеллектуального формирования информационных контекстов;

– выбор протокола процесса и участников для всех его ролей;

– поиск и построение эффективных моделей оценки вариантов;

– получение и обобщение экспертных мнений;

– выдача рекомендаций по аспектам и форматам аргументирования экспертных мнений;

– прогноз перспектив достижения компромисса и оценки его характеристик.

Функции и задачи интеллектуальной информационной технологии экспертно-аналитической поддержки организационных решений были рассмотрены в [9], вместе с описанием рамочного состава структур знаний этой технологии.

Рассмотрим формальный аппарат моделей знаний организации, обеспечивающий конструктивное оперирование ими для поддержки операций из (2) в заданной системе требований гармоничности относительно систем из (5). Определим пространство знаний организации о вырабатываемых и выполняемых решениях

$$KS = < OSP, OMT, OSDM, MPD, SF, MASF, AUSF >, \quad (6)$$

где OSP – онтология стандартных профилей организации; OMT – онтология имеющихся средств информирования и анализа; $OADM$ – онтология сферы принятия решений; MPD – множество моделей протоколов принятия решений, ассоциированное с правилами их применения; SF – поле решений; $MASF$ – модели анализа состояния SF; $AUSF$ – результаты выполнения аудитов состояния SF.

Онтология стандартных профилей организации опирается на состав и структуру подонтологий, выделявшихся ранее в онтологических проектах представления структуры и деятельности организации [10], а также отражает опыт описания организаций в системе SAP [11]. В ее состав с необходимостью входят подонтологии: Организационная структура, Кадры, Ресурсы, Функции, Продукция, Внешние контакты и влияния. В качестве связующего механизма между ними используется механизм Проекций, разработанный в проекте *OrgMaster* [12].

Онтология средств информирования и анализа включает сведения о методологиях, программных компонентах и информационных компонентах, доступных для использования в организации [8].

Онтология сферы принятия решений имеет состав:

$$OSDM = < CC_{st}, CAT, CC_{own}, CH, VP, MVP, REL >, \quad (7)$$

где CC_{st} – множество заимствованных концептов $C \in OSP$; CAT – множество категорий собственных концептов; CC_{own} – множество собственных концептов; CH – множество характеристик; VP – множество бизнес-обусловленных концептуальных точек зрения на предметную область решений организации; MVP – модели точек зрения; REL – множество отношений между концептами онтологии, представленных своими предикатами и метриками.

Множество характеристик CH подразделяется на следующие типы: ID – идентификационные, используемые для идентификации объекта; PAR – параметрические, задающие тип или неизменный

параметр объекта; *IND* – индикаторные, которые являются признаком состояния объекта, актуальным для функциональных нужд; *EV* – оценочные; *REP* – репрезентирующие, которые определяют актуальные для *OSDM* свойства объекта из CC_{st}. Элемент $H \in CH$ описывается своей моделью:

$$MCH(H) = < TCH, AZ, AR >, \qquad (8)$$

где *TCH* – один из вышеперечисленных типов; *AZ* – область значений; *AR* – множество категорий концептов, для которых характеристика актуальна.

Любая категория концептов $Cat \in CAT$ определяется обобщенной моделью своих концептов:

$$MGCat = \{MDP_i\}_{i=1,...,m}, \qquad (9)$$

где MDP_i – модель *i*-го частичного определения, *m* – количество частичных определений.

$$MDP_i = \{< r_j, A_i >\}_{i=1,...,md}, \qquad (10)$$

где r_j – имя j-й роли (j-го слота фрейма MDP_i);

A_j – модель интерпретации роли $INT(r_j)$ в модели концепта данной категории.

$$A_j = < SFORM, AINT, SM, MZ >, \qquad (11)$$

где *SFORM* – синтаксический формат; *AINT* – область интерпретации $(AIN \subseteq C_{st} \vee AINT \subseteq CC_{own} \vee AINT \subseteq CH)$; *SM* – параметризация модели формата; $MZ \subseteq MOD$ – модальности допустимого использования роли в определениях концептов.

$$MOD = \{AF, EN, UO, UN\}, \qquad (12)$$

где *AF* – полностью определенное знание; *EN* – пустое – из-за неактуальности роли; *UO* – неполное, открытое для пополнения; *UN* – неполное, но требующее достижения определенности для формирования экземпляра концепта.

Синтаксический формат

$$SFORM \in GSFORM \qquad (13)$$

$$GSFORM = \{S_i\}_{i-1,...,5},$$

где S_1 – одноэлементный, $SM = (r_j : C)$;
S_2 – множественный, $SM = (r_j \{C_1 ..., C_n\})$;
S_3 – кортеж, $SM = (r_j : < r_{jl} : C_l ..., r_{jn} : C_n >)$;
S_4 – множество кортежей

$$SM = (r_j : \{r_{j11} : C_1^1, ..., r_{jn1} : C_n^1 >, ..,$$

$$< r_{j1m} : C_1^m, ..., r_{jnm} : C_n^m > \ \});$$

S_5 – задание процедуры оценки значения указанного параметра

$$SM = (r_j : << X_i, ..., X_n > \rightarrow PROC \rightarrow P >),$$

где X_i – входной параметр процедуры $PROC$; P – оцениваемый параметр.

Модель концепта *C* категории *cat* $MC(C)$ задает интерпретацию модели категории

$$MC(C) = \{MDP_i(C)\}_{i=1..., m}, \qquad (14)$$

где $MDP_i(C) = \{INT(r_j)\}_{j=1,..., md}$,

$$INT(r_j) = < W_j, M_j, ME_j >.$$

При этом $W_j(Et_j)$ – выражение, формат которого соответствует параметризации (13), а $Et_j = \{Et_{jk}\}_{k=1,...,mk}$ множество термов, следующих в ( ) за знаком ":", причем $et_{kj} \in AINT_j$ (см.( )); M_j – значение модальности полноты знания, $M_j \in MZ$ (см. (12)); $ME \in$ (0,1) – модальность обязательности определенности значения.

В случае

$$SFORM_{ij} = S1,\ A_j = H,\ H \in CH$$

допустимо включение в определение PD_i выражения $W_j = < H, z(H) >$, в котором

$z(H)$ – значение характеристики H. Таким образом, значение отдельных параметров может быть зафиксировано непосредственно в модели концепта.

Для модели $MC(C)$ определены следующие структурные ограничения:

1) нецикличность

$$C \notin MC(C);$$

2) ограничение сложности, согласно которому вложенность в определениях концептов не превышает фиксированного порядка по собственным концептам онтологии сферы поддержки принятия решений *OSDM*.

Для последующего введения системы отношений в онтологии *OSDM* определим понятие экземпляров ее концептов.

Будем называть экземпляром $\uparrow C$ концепта C, который может снабжаться идентификатором X, обозначаясь как $X \uparrow C$, следующую конкретизацию $MC(C)$ (14).

$$X \uparrow C = \{con(MDP_i(C))\}_{i=1,\ldots,m},$$

$$con\,(MDP_i(C)) = \{con(INT(r_{ij}))\}_{j=1,\ldots,md},$$

$$con\,(INT(r_{ij})) = \{con\,(et_{ijk})\}_{k=1,\cdots\ldots,K},$$

где K – число термов в W_{ij}.

Конкретизация $con(et)$ принимает значения: $z \in Az(H)$, если $et = H$, $X_1 \uparrow C_1$ если $et = C_1$, неопределенное значение $(\bullet)$ (допустимо, если $ME_j = 0$).

Множество интенсионально заданных отношений *REL* будет рассмотрено в следующей части статьи.

Множество *MPD* в (6) включает элементы, каждый из которых описывает возможную организацию процесса принятия для регламентных и инновационных решений в ракурсах, рассмотренных ранее в [8, 9].

Поле решений *SF* представляет структуру знаний

$$\left(\{\uparrow DEC_i, PR_i, SG_i(T)\}_{i=1,NT}, T\right), \quad (15)$$

где T – момент времени; $\uparrow DEC_i$ – экземпляр решения, информация о котором имеется в момент T; PR_i – реализованный протокол его выработки; $SG_i\,(T)$ – статус решения (принятое, принимаемое на этапе E_k своего протокола, выполненное, выполняемое, отвергнутое, действующее).

Элементы (6) *MASF* и *AUSF* определяют способы и результаты оперирования знаниями, которые соответствуют задачам ИИТ ППОР [9] будут подробнее рассматриваться в следующих частях.

Категории концептов онтологии сферы поддержки принятия решений

К категориям концептов, составляющих онтологическую модель *OSDM* из (7), принадлежат Решение, Цель, Проблемная ситуация, Воздействие, Побочное влияние, Ценность, Модель ценности, Конгломерат целей, Конгломерат решений.

Такое выделение категорий связано с функциональными потребностями аналитики системы организационных решений и с поддерживаемыми протоколами их принятия. Состав их моделей учитывает результаты [8, 10, 13].

Описание моделей категорий, выполненное в аспекте их элементов, приведенных в (14), представлено в табл. 1–9.

Таблица 1. Модель категории G Цель $MG =< PG_1, PG_2, PG_3, PG_4 >$

Частичное определение	Роль				
	Обозначение	Семантика	Синтаксический формат	Область интерпретации	Допустимые модальности
PG_1 Направленность	R_1	Целевой объект	*S*1	Концепты-объекты управления	*AF*

Частичное определение	Роль				
	Обозначение	Семантика	Синтаксический формат	Область интерпретации	Допустимые модальности
	R_2	Изменяемые свойства объекта	*S*2	Характеристики	*AF*
	R_3	Направление изменения: снижение, повышение, стабилизация, разрушение, приближение, отклонение, минимизация, максимизация, достижение, недопущение	*S*1	Параметр	*AF*
	R_4	Критерий достижения: входные данные, метод оценки и индикатор	*S*5:R_{41} → R_{42} → R_{43}	R_{41} - Характеристики объектов R_{42} - Метод R_{43} - Параметр	*AF*, *UO*, *UN*
PG_2 Характеризация	R_1	Владелец цели	*S*1	Субъект	*AF*
	R_2	Тип условий актуализации	*S*1	Параметр	*AF*, *EN*
	R_3	Уровень конкретности (Видение, Миссия, Интерес, Цельпостановка, Конструктивная цель, Реализационная цель)	*S*1	Параметр	*AF*, *EN*, *UN*
	R_4	Тип постановки (точечная, интервальная)	*S*1	Параметр	*AF*, *UN*
	R_5	Класс периода актуальности	*S*1	Параметр	*AF*, *UN*
	R_6	Временной горизонт достижения цели или пересмотра	*S*1	Параметр	*AF*, *UN*
PG_3 Связи	R_1	Деревья целей, в состав которых входит	*S*2	Концепты категории Конгломерат целей	*AF*, *EN*, *UO*
PG_4 Позиция в деятельности	R_1	Сфера действия	*S*2	Элементы структуры организации, внешние объекты	*AF*, *UO*
	R_2	Тип стабильности (Постоянная, Ситуационная, Компромиссно пересматриваемая)	*S*1	Параметр	*AF*, *EN*
	R_3	Конструктивная поддержка	*S*2	Воздействия, Решения	*AF*, *EN*, *UO*, *UN*
	R_4	Способы поддержки (создание и выполнение функций, ресурсное обеспечение, организация взаимодействий)	*S*1		*AF*, *EN*, *UO*

Таблица 2. Модель категории *CG* Конгломерат целей

Частичное определение	Роль				
	Обозначение	Семантика	Синтаксический формат	Область интерпретации	Допустимые модальности
PCG_1 Состав	R_1	Источники формирования	*S*2	Решение, Документ, Коммуникация	*AF*
	R_2	Способ формирования (интеграция, раскрытие, обобщение версий)	*S*1	Параметр	*AF*
	R_3	Корневая цель	*S*1	Цель	*AF*
	R_4	Состав раскрывающих целей	*S*2	Цели	*AF*, *UO*, *UN*
PCG_2 Структура	R_1	Множество описаний положения элементов в формате (R_{11}: Цель, R_{12} : Предшественник, R_{13}: Последователь, R_{14} : Тип (основная или обеспечивающая))	*S*4	R_{i1}, R_{i2}, R_{i3} - Цель, R_{i4} - Параметр	*AF*, *UN*

Таблица 3. Модель категории *CA* Воздействие *CA MCA* = < PCA_1, PCA_2, PCA_3 >

Частичное определение	Роль				
	Обозначение	Семантика	Синтаксический формат	Область интерпретации	Допустимые модальности
PCA_1 Направленность и средства	R_1	Цель воздействия	*S*1	Цель	*AF*
	R_2	Реализуемые мероприятия	*S*2	Мероприятие	*AF, UN*
	R_3	Ресурсы	*S*2	Объекты ресурсов заданных видов	*AF, UO, UN*
	R_4	Время выполнения	*S*1	Характеристика	*AF, UN, EN*
PCA_2 Субъекты	R_1	Основные исполнители	*S*2	Субъект, элемент оргструктуры	*AF, UN, EN*
	R_2	Привлекаемые исполнители	*S*2	Субъект, элемент оргструктуры, внешний объект	*AF, UN, EN*
	R_3	Инициатор	*S*1	Субъект, элемент оргструктуры	*AF*
	R_4	Инстанция контроля и оценки	*S*2	Субъект, элемент оргструктуры	*AF, EN, UN*
PCA_3 Влияния	R_1	Факторы, определяющие выполнимость	*S*2	Факторы среды, Характеристики объектов управления	*AF, EN, UO*
	R_2	Факторы, влияющие на эффективность	*S*2	Факторы среды, Характеристики объектов управления	*AF, EN, UO*
	R_3	Возможные последствия	*S*2	Побочное влияние	*AF, UO, UN*

Таблица 4. Модель категории *PS* Проблемная ситуация $MPS =< PPS_1, PPS_2 >$

Частичное определение	Роль				
	Обозначение	Семантика	Синтаксический формат	Область интерпретации	Допустимые модальности
PPS_1 Проявление	R_1	Источник знаний	*S*2	Решение, Документ, Коммуникация	*AF*
	R_2	Тип неудовлетворительности положения дел (потеря эффективности в новых условиях, неиспользуемость новых возможностей, появление угроз, исчерпанность целей, потеря актуальности целей)	*S*1	Параметр	*AF*
	R_3	Сфера, в которой проявлена	*S*2	Сфера деятельности	*AF, UO, UN*
	R_4	Показатели и условия деятельности, на основе которых диагностирована ситуация	*S*2	Характеристики, Внешние факторы	*AF, UO*
PPS_2 Влияние	R_1	Конфликтующие элементы в формате {<элемент исходного или прогнозного положения дел, элемент фактического положения дел>}	*S*4	Цели, Функции, Характеристики, Факторы, Субъекты, Ресурсы	*AF, UO*

Частичное определение	Роль				
	Обозначение	Семантика	Синтаксический формат	Область интерпретации	Допустимые модальности
	R_2	Затрагиваемые интересы	*S*2	Цели, Субъекты	*AF, UO*

Таблица 5. Модель категории *UI* Побочное влияние $MUI =< PUI_1, PUI_2 >$

Частичное определение	Роль				
	Обозначение	Семантика	Синтаксический формат	Область интерпретации	Допустимые модальности
PUI_1 Эффект	R_1	Источник влияния	*S*1	Воздействие	*AF*
	R_2	Объект, испытывающий влияние	*S*1	Объект управления, Объект внешней среды	*AF*
	R_3	Изменяемая характеристика объекта	*S*2	Характеристики, Отношения, Параметр	*AF*
	R_4	Характер влияния (рост, снижение, разбалансирование, деструктурирование)	*S*1	Параметр	*AF*
	R_5	Существенность влияния	*S*1	Характеристика	*AF, EN*
PUI_2 Меры устранения	R_1	Пакет воздействий для нейтрализации	*S*2	Воздействие	*AF, EN, UO, UN*
	R_2	Оценка экономической приемлемости нейтрализации	*S*1	Характеристика	*AF, EN, UO*
	R_3	Приемлемость нейтрализации для стейкхолдеров	*S*1	Характеристика	*AF, EN, UO*

Таблица 6. Модель категории *D* Решение $MD =< PD_0, PD_1, PD_2, PD_3, PD_4, PD_5, PD_6, PD_7 >$

Частичное определение	Роль				
	Обозначение	Семантика	Синтаксический формат	Область интерпретации	Допустимые модальности
PD_0 Паспорт	R_1	Начало и конец процесса выработки	*S*3*:* R_{11} – дата начала, R_{12} – дата конца	Характеристика	*AF*
	R_2	Лицо, принимающее решение	*S*1	Субъект	*AF*
	R_3	Индекс критичности (да/нет)	*S*1	Параметр	*AF, UN, UO*
	R_4	Функциональная область	*S*1	Сфера деятельности	*AF, UN*
	R_5	Уровень управления	*S*1	Параметр	*AF, UN*
	R_6	Объект непосредственного воздействия	*S*1	Объект управления	*UN*
PD_1 Проблема	R_1	Разрешаемая проблемная ситуация	*S*1	Проблемная ситуация	*AF*

Частичное определение	Роль				
	Обозначение	Семантика	Синтаксический формат	Область интерпретации	Допустимые модальности
	R_2	Учитываемые проблемные ситуации	*S*2	Проблемная ситуация	*AF*, *EN*, *UO*
PD_2 Решаемая задача	R_1	Степень вмешательства (мониторинг, ослабление, разрешение)	*S*1	Параметр	*AF*
	R_2	Тип воздействия (прямое, косвенное)	*S*1	Параметр	*AF*
	R_3	Альтернативные цели	*S*3: R_{31} – цель, R_{32} – оценка перспективности	Цель, Характеристика	*AF*, *UN*
	R_4	Модель оценивания	*S*1	Модель ценности	*AF*
PD_3 Перспективные воздействия	R_1	Цель воздействия на проблемную ситуацию	*S*1	Цель	*AF*
	R_2	Пакет реализующих воздействий	*S*2	Воздействие	*AF*, *UN*
	R_3	Пакет приемлемых воздействий	*S2*	Воздействие	*AF*, *UN*
	R_4	Модель приемлемости	*S1*	Модель ценности	*AF*, *UN*
PD_4 Выбор	R_1	Модель выбора воздействия	*S*1	Модель ценности	*AF*, *UN*
	R_2	Учитываемые точки зрения	*S*2	Бизнес-перспектива онтологии	*AF*, *UN*
	R_3	Выбранное воздействие	*S*1	Воздействие	*AF*, *UN*
	R_4	Оценка качества выбора	*S*3	Характеристика	*AF*, *UN*
PD_5 Результаты выполнения	R_1	Удовлетворительность выполнения	*S*1	Характеристика	*AF*
	R_2	Удовлетворительность воздействия на проблемную ситуацию	*S*1	Характеристика	*AF*, *EN*
	R_3	Проявившиеся побочные влияния	*S2*	Побочные влияния	*AF*, *EN*
PD_6 Методы и средства	R_1	Рекомендованные методы	*S*2	Метод	*AF*
	R_2	Рекомендованные средства автоматизированной поддержки	*S*2	Программное средство, Информационный объект	*AF*
	R_3	Использованные методы	*S*2	Метод	*EN*, *UN*
	R_4	Использованные средства	*S*2	Программное средство, Информационный объект	*EN*, *UN*
	R_5	Оценка удовлетворительности поддержки	*S*1	Характеристика	*UN*
PD_7 Позиция по преемственности	R_1	Система решений, в которую целенаправленно включено данное	*S*1	Конгломерат решений	*AF*, *EN*
	R_2	Решение-предшественник	*S*1	Решение	*AF*, *EN*
	R_3	Развиваемые частичные определения	*S*2	Параметры	*AF*, *EN*

Частичное определение	Роль				
	Обозначение	Семантика	Синтаксический формат	Область интерпретации	Допустимые модальности
	R_4	Роли с дополняемым содержанием	*S*2	Параметры	*AF, EN*
	R_5	Роли с конкретизируемым содержанием	*S*2	Параметры	*AF, EN*
	R_6	Роли с пересматриваемым содержанием	*S*2	Параметры	*AF, EN*

Таблица 7. Модель категории *DC* Конгломерат решений $MDC =< PDC_1, PDC_2 >$

Частичное определение	Роль				
	Обозначение	Семантика	Синтаксический формат	Область интерпретации	Допустимые модальности
PDC_1 Характеризация	R_1	Формат возникновения (плановый, инновационный уникальный, инновационный тиражируемый)	*S1*	Параметр	*AF*
	R_2	Инициирующий фактор (регламентирующий документ, решение, коммуникация)	*S*1	Документ, Решение, Коммуникация	*AF*
	R_3	Уровень планирования	*S*1	Параметр	*AF*
	R_4	Интервал выполнения	*S*3: R_{41} – начало, R_{42} – конец	Характеристика	*AF, UN*
PDC_2 Структура	R_1	Начальное решение	*S*1	Решение	*AF*
	R_2	Состав конгломерата	*S*2	Решение	*AF, UO, UN*

Таблица 8. Модель категории *VAL* Ценность $MVAL =< PVAL_1, PVAL_2 >$

Частичное определение	Роль				
	Обозначение	Семантика	Синтаксический формат	Область интерпретации	Допустимые модальности
$PVAL_1$ Содержание	R_1	Цель, которой соответствует ценность	*S*1	Цель	*AF*
	R_2	Объекты, для которых актуальна ценность как характеристика	*S*2	Объект управления	*AF, UO, UN*
	R_3	Модели ценности	*S*2	Модель ценности	*AF, UO, UN*
$PVAL_2$ Сфера	R_1	Субъекты интересов, основанных на ценности	*S*2	Стейкхолдер	*AF, UO, UN*
	R_2	Субъекты, руководствующиеся ценностью в своей деятельности	*S2*	Элемент оргструктуры	*AF, UO, UN*
	R_3	Информационные источники	*S*2	Документ, Информационный объект, Коммуникация	*AF, UO, UN*

Таблица 9. Модель категории MV Модель ценности $MV =< PMV_1, PMV_2, PMV_3 >$

Частичное определение	Роль				
	Обозначение	Семантика	Синтаксический формат	Область интерпретации	Допустимые модальности
PMV_1 Генезис	R_1	Источник знаний о модели	$S1$	Документ, Решение	*AF*
	R_2	Дерево целей, соотнесенное с моделью	$S1$	Конгломерат целей	*AF, UN*
	R_3	Базовая ценность, раскрытие которой осуществляется	$S1$	Ценность	*AF*
	R_4	Объем дерева N	$S1$	Параметр	*AF*
PMV_2 Структура	R_1	Характеристика каждой ценности из состава модели ее позицией (оконечная, функциональный узел, вспомогательный узел), шкалой и информационным контекстом	$S4$: R_{1i1} – ценность, R_{1i2} – позиция, R_{1i3} – шкала, R_{1i4} – источник, $i \in_{1,n}$	Ценность, Параметр, Информационный объект	*AF, UO, UN*
PMV_3 Аргументация	R_1	Характеризация каждой ценности весовым коэффициентом, ценностью-предшественником, методом оценивания, конкретизируемым элементом цели-предшественника, аргументом актуальности	$S4$: R_{1i1} – ценность, R_{1i2} – предшественник, R_{1i3} – метод, R_{1i4} – роль в цели, R_{1i5} – концепт, подтверждающий актуальность	Ценность, Метод, Параметр, Цель, Документ, Воздействие, Побочное влияние	*AF, UN*

Модели категорий, представленные в табл. 1–9, служат основой описания онтологии *OSDM* (7) конкретной организации, формирования поля ее решений и оперирования ими при создании интеллектуальной информационной технологии [9] для этой организации.

Язык оперирования знаниями из состава *OSDM* и *SF* будет рассмотрен в следующей части статьи, как и формализм анализируемых отношений и свойств.

Заключение

В работе рассмотрено представление организационного решения как формального объекта, ориентированного на сохранение опыта организации и на выполнение аналитических операций для повышения качества ее деятельности.

Акцент сделан, прежде всего, на отображение аспектов и характеристик решений, выделяемых и активно используемых в современной теории менеджмента и в передовых технологиях управления проектами.

Это отражено в предложенных структурах знаний, которые осуществляют:

– позиционирование решения в системе целей, интересов и ценностей организации;

– поддержку накопления и использования информации, позволяющей осуществлять мониторинг его качества на всех этапах жизненного цикла;

– сосуществование структур неполного и противоречивого экспертного знания о предметной области для его ситуативной компромиссной интеграции;

– оперирование полем решений организации как целостной системой, анализ состояния которой обеспечивает эффективный выбор вариантов управляющих воздействий и аудит качества управления в организации.

Во второй части статьи будут представлены аналитические механизмы, основанные на использовании предложенных моделей.

1. *Ананьин В.* Бережливая информатизация. Ч.1 [Электронный ресурс] // Intelligent Enterprise. – 2009. – № 12. – Режим доступа: http://www.management.com.ua/ims/ims155.html.
2. *Mc Carty I., Menicou M.* A Classification Schema of Manufacturing Decisions for the GRAI Enterprise Modelling technique // Computers in Industry. – 2002. – 47. – P. 339–355.
3. *Frankel E.G.* Quality Decision Management – The Heart of Effective Futures-Oriented Management: A Primer for Effective Decision-Based Management. New York: – Springer, 2008. – 110 p.
4. *Ильина Е.П.* Управление качеством организационных решений на основе формализованного корпоративного знания. Ч 2. Качество организационных решений и его поддержка // Математические машины и системы. – 2014. – № 2. – С. 84–96.
5. *Ильина Е.П.* Методы автоматизированного управления экспертизами при концептуальной неоднородности экспертных взглядов // Проблеми програмування. – 2007. – № 4. – С. 35–46.
6. *Towards* a Semantic Decision Representation Format // W3C Incubator Group Report 17 April 2012 http://www.w3.org/2005/Incubator/decision/XGR-decision/.
7. *Мазур И.И., Шапиро В.Д., Ольдерогге Н.Г. и др.* Корпоративный менеджмент. Справочник для профессионалов / Иван Иванович Мазур (ред.) – М.: Высшая школа, 2003. – 1076 с.
8. *Ильина Е.П.* Управление качеством организационных решений на основе формализованного корпоративного знания. Ч 1. Онтология организационных решений // Математические машины и системы. – 2014. – № 1. – С. 129 – 142.
9. *Ильина Е.П., Синицын И.П., Яблокова Т.Л.* Принципы построения интеллектуальной информационной технологии поддержки решений в организации // Проблемы программирования. – 2015. – № 2. – С. 63–75.
10. *Uschold M. et al.*The Enterprise Ontology. – AIAI_TR-1998. – 61 p. – *Available at* http://www.aiai.ed.ac.uk/project/enterprise/enterprise/ontology.html.
11. *SAP* NetWeaver Master Data Management. Режим доступа: http://www.sap.com/pc/tech/enterprise-information-management/software/master-data/index.html
12. *Гаврилова Т.А., Кудрявцев Д.В.* Информационные технологии управления знаниями // В: *Мильнер Б.З.* Инновационное развитие: экономика, интеллектуальное развитие, управление знаниями. – М.: – Инфра-М, 2009. Гл. 25.
13. *Popova V., Sharpanskykch A.* Formal modelling of organizational goals based on performance indicators // In: Data & Knowledge Engineering. – 2011.– Vol. 70. – P. 335–364.

References

1. *Ananin V.* "Economical informatisation". [In Russian] In: [Electronic Resource] // Intelligent Enterprise. – 2009. – N 12. – Mode of access to text: http://www.management.com.ua/ims/ims155.html.
2. *Mc Carty I., Menicou M.* A Classification Schema of Manufacturing Decisions for the GRAI Enterprise Modelling technique // Computers in Industry. – 2002. – 47. – P. 339–355.
3. *Frankel E.G.* Quality Decision Management – The Heart of Effective Futures-Oriented Management: A Primer for Effective Decision-Based Management, New York: – Springer, 2008 – 110 p.
4. *Ilina E.P.* Management of quality of organization decisions grounded on formalized corporation knowledge. P2. Quality of organization decisions and its support [In Russian] In: Mathematic machines and Systems.–2014. – N 2. – P. 84–96.

5. *Ilina E.P.* Methods for automated management of expert decision making under conceptual heterogeneity of expert views [In Russian] In: Problems in Programming. – 2007. – N 4. – P. 35–46.
6. *Towards* a Semantic Decision Representation Format // W3C Incubator Group Report 17 April 2012 In: [Electronic Resource] http:// www.w3.org/2005/Incubator/decision/XGR-decision/.
7. *Mazur I.I., Shapiro V.D., Olderogge N.G.* Corporative management. Handbook for Professionals [In Russian]. – Moscow: High School, 2003. – 1076 p.
8. *Ilina E.P.* Management of quality of organization decisions grounded on formalized corporation knowledge. P1. Ontology of organization decisions [In Russian] In: Mathematic machines and Systems. – 2014. – N 1. – P. 129–142.
9. *Ilina E.P., Sinitsyn I.P., Yablokova T.L.* Designing principles of the Intelligent information technology for organization decisions [In Russian] In: Problems in Programming. – 2015. – N 2. – P. 63–75.
10. *Uschold M. et al.* The Enterprise Ontology. – AIAI_TR-1998. – In: [Electronic Resource] 61 p. – Mode of access to text: http://www.aiai.ed.ac.uk/project/enterprise/enterprise/ontology.html.
11. *SAP* NetWeaver Master Data Management. In: [Electronic Resource] Mode of access to text: http://www.sap.com/pc/tech/enterprise-information-management/software/master-data/index.html.
12. *Gavrilova T.A., Kudriavtsev D.V.* Information technologies for knowledge management. [In Russian] In: Innovative development and knowledge management". – Moscow: Infra – M, 2009. Chap.25.
13. *Popova V., Sharpanskykch A.* Formal modelling of organizational goals based on performance indicators // In: Data & Knowledge Engineering. – 2011.– Vol. 70. – P. 335–364.

Получено 01.12.2015

Об авторе:

Ильина Елена Павловна,
кандидат физико-математических наук,
старший научный сотрудник,
ведущий научный сотрудник.
Количество научных публикаций в украинских изданиях – более 50.
http://orcid.org/0000-0002-1528-366X

Место работы автора:

Институт программных систем
НАН Украины,
03187, Киев-187,
Проспект Академика Глушкова, 40.
Тел.: 526 4188.
E-mail: sec_kiev@ukr.net

УДК 681.3

А.Ю. Дорошенко, П.А. Іваненко, О.М. Овдій, О.А. Яценко

АВТОМАТИЗОВАНЕ ПРОЕКТУВАННЯ ПРОГРАМ ДЛЯ РОЗВ'ЯЗАННЯ ЗАДАЧІ МЕТЕОРОЛОГІЧНОГО ПРОГНОЗУВАННЯ

Розроблено засіб автоматизованого конструювання паралельного коду для середовища OpenMP на основі високорівневих алгебро-алгоритмічних специфікацій. Застосування засобу демонструється на прикладі задачі моделювання циркуляції атмосфери, що представлений як сервіс у складі Інтернет-порталу з надання послуг метеопрогнозу. Здійснена генерація програмного коду та наведено результати експерименту з виконання розробленої паралельної програми прогнозування на мультипроцесорній платформі.

Ключові слова: паралельні обчислення, метеорологічне прогнозування, синтез програм, алгебра алгоритмів, Інтернет-портал.

Вступ

Системи прогнозування з року в рік стають все більш складними, використовувані ними математичні моделі вдосконалюються, обсяги обчислювальних даних стрімко зростають. Не винятком є і задача метеорологічного прогнозування. Враховуючи великий попит на метеорологічні прогнози в різних галузях людської діяльності, проектування та розробка програм для розв'язання цієї задачі є надзвичайно актуальними. Крім того, достовірність та оперативність отримання даних метеорологічних прогнозів мають велике, а іноді і критичне значення, що висуває високі вимоги до швидкісних характеристик таких програм.

Задача метеорологічного прогнозування є складною прикладною обчислювальною задачею з високими вимогами до точності отриманих результатів та жорсткими часовими обмеженнями. Великий обсяг обчислень над великими масивами даних потребує впровадження паралельних реалізацій та забезпечення їх оптимізації.

У попередній роботі авторів [1] було запропоновано реалізацію Інтернет-порталу для надання послуг метеорологічного прогнозування, яка поєднує комплексність використання адекватних фізичних моделей атмосферних процесів з ефективними обчислювальними схемами та методами програмування високопродуктивних обчислень на мультипроцесорних системах, що дає змогу досягати належного ступеня точності, повноти та своєчасності інформації, необхідної для задоволення потреб широкого кола користувачів. Дана робота поширює можливості попередньої на середовище програмування OpenMP з використанням більш потужної мультипроцесорної техніки та розширеної постановки прикладної задачі.

Одним з перспективних напрямів у розробці та дослідженні систем паралельних і розподілених обчислень нині є побудова програмних абстракцій у вигляді алгебро-алгоритмічних мов і моделей, що ставить своєю метою розвиток архітектурно- і мовно-незалежних засобів програмування для мультипроцесорних обчислювальних систем і мереж. У роботах [2–9] були запропоновані теорія, методологія та інструментарій для автоматизованого проектування паралельних програм, що ґрунтуються на засобах високорівневої алгеброалгоритмічної формалізації та автоматизації перетворень програм. Зокрема, розроблено інструментальну систему автоматизації програмування, названу Інтегрованим інструментарієм Проектування та Синтезу програм (ІПС) [2–6]. В системі спільно використовуються три форми подання алгоритмічних знань про предметні області: аналітична (формула в алгебрі алгоритмів), природно-лінгвістична (текстова) та графова (граф-схеми). ІПС ґрунтується на методі діалогового конструювання синтаксично правильних програм,

що орієнтований на виключення виникнення синтаксичних помилок в процесі проектування. Інструментарій забезпечує покрокову розробку програм, починаючи від формальної специфікації і закінчуючи кодом цільовою мовою програмування (Java, C++). У роботах [7–9] розглядається нова версія системи синтезу програм – Онлайновий Діалоговий конструктор Синтаксично Правильних програм (ОДСП). Особливість інструментарію ОДСП полягає у використанні Web-технологій та у розподіленій архітектурі системи.

У попередніх роботах [2–9] системи ІПС та ОДСП були застосовані для розробки паралельних застосувань для багатоядерних центральних процесорів та графічних прискорювачів, а також розподілених та сервісно-орієнтованих програм для Грід. Розроблена інтеграція ІПС та системи переписувальних правил TermWare [5, 10] для автоматизації трансформації паралельних програм. Предметною областю застосування інструментальних засобів, зокрема, є задача метеорологічного прогнозування. Наприклад, у роботі [8] виконане автоматизоване проектування паралельної програми для розв'язання двовимірної задачі конвективної дифузії, яка виникає при математичному моделюванні циркуляції атмосфери в метеорології. У даній статті наведено результати застосування інструментарію для генерації паралельної програми для тривимірного випадку згаданої задачі [11–13]. Розроблена багатопотокова програма реалізована мовою C++ і використовує засоби OpenMP [14]. У роботі також виконане проектування сервісу, який використовує розроблену паралельну програму, і входить до складу вищезгаданого Інтернет-порталу з надання послуг метеорологічного прогнозування.

Запропонований у даній статті підхід є близьким до робіт, що відносяться до алгебраїчного програмування [15], синтезу програм на основі високорівневих специфікацій [16, 17], а також автоматизованої генерації OpenMP програм [18–21] та Java застосувань [22–24]. Відмінність нашого підходу полягає у використанні специфікацій алгебри Глушкова, поданих у природно-лінгвістичній формі, що полегшує розуміння алгоритмів і досягнення необхідної якості програм. Іншою перевагою розроблених засобів є застосування методу діалогового конструювання синтаксично правильних програм, що виключає можливість виникнення синтаксичних помилок у процесі проектування схем.

1. Алгебра алгоритмів та інструментальні засоби автоматизованої розробки програм

В основу пропонованого підходу до проектування паралельних програм покладений апарат систем алгоритмічних алгебр (САА) та їх модифікацій [2]. Модифіковані САА (САА-М) призначені для формалізації процесів мультиобробки, що виникають при конструюванні програмного забезпечення в мультипроцесорних системах. На САА-М ґрунтуються розроблені інструментальні засоби автоматизованого проектування та генерації програм [2–9].

1.1. Основні операції алгебри алгоритмів. САА-М є двоосновною алгеброю $< Op, \mathrm{Pr}; \Omega >$, де Op – множина операторів; Pr – множина логічних умов (предикатів); Ω – сигнатура, що складається з логічних операцій (диз'юнкції, кон'юнкції, заперечення, лівого множення оператора на умову) та операторних операцій (композиції, альтернативи, циклу та ін.). Оператори та предикати можуть бути базисними або складеними.

На САА-М ґрунтується алгоритмічна мова САА/1 [2], призначена для багаторівневого структурного проектування і документування послідовних та паралельних алгоритмів і програм. Перевагою її використання є можливість опису алгоритмів у природно-лінгвістичній формі, зручній для людини, що полегшує досягнення необхідної якості програм. Подання операторів мовою САА/1 називаються САА-схемами.

Далі наведено перелік назв та специфікації основних операторних операцій сигнатури САА-М, поданих у природньо-лінгвістичній формі.

1. Композиція (послідовне виконання операторів):

оператор 1 ПОТІМ *оператор* 2

або

оператор 1; *оператор* 2

2. Альтернатива (умовний оператор):

ЯКЩО *умова* ТО
  оператор 1
ІНАКШЕ
  оператор 2
КІНЕЦЬ ЯКЩО

3. Цикл:

ПОКИ *умова виконання циклу*
ЦИКЛ *оператор*
КІНЕЦЬ ЦИКЛУ

4. Операція виконання одного з *n* операторів за істинності відповідної умови:

ВИБІР ([*умова* 1] → *оператор* 1,
  [*умова* 2] → *оператор* 2,
  ...
  [*умова n*] → *оператор n*)

5. Асинхронна диз'юнкція – паралельне виконання *n* операторів (потоків), де *i* – номер потоку:

ПАРАЛЕЛЬНО (*i* = 0, ..., *n*–1)
(
  оператор
)

6. Синхронізатор, що виконує затримку обчислень доти, поки значення умови не стане істинним:

ЧЕКАТИ *умова*

Приклад використання наведених конструкцій наведено у розділі 2.

Відмітимо, що операції САА-М також можуть бути подані англійською мовою, як, наприклад, у роботі [9].

1.2. Додаткові операції алгебри для об'єктно-орієнтованої парадигми. У сигнатуру САА-М також входять операції, призначені для формалізації основних понять об'єктно-орієнтованого програмування (класів, об'єктів та ін.). Відмітимо, що для сумісності з мовою програмування Java в сигнатуру включені операції для визначення анотацій. Анотації є спеціальною формою синтаксичних метаданих, яка може бути додана у програмний код, і використовується для його аналізу, компіляції або виконання [25]. Анотованими можуть бути пакети, класи, методи, змінні та параметри. Далі наведено перелік основних операцій САА-М для визначення класів, що використовуються у даній роботі.

1. Визначення анотованого класу (в узагальненому вигляді):

Annotated class X implements Y (
  Class name, *Interface name*)
<Annotations>
  Annotation (*Annotation text*);
  Annotation (*Annotation text*);
  ...
<Fields>
  operators
<Methods>
  operators

де *Class name* – ім'я класу, який реалізує інтерфейс *Interface name*; <Annotations>, <Fields>, <Methods> – рядки, після яких необхідно вказати оператори визначення анотацій, полів даних та методів класу, відповідно; *Annotation text* – текст анотації.

2. Анотації специфікуються за допомогою конструкції

Annotation (*Annotation text*),

де *Annotation text* – параметр, у якому зазначається текст анотації.

Прикладом визначення анотації для класу може бути конструкція:

Annotation (*Service*).

3. Ініціалізоване поле даних:

Field initialized (*Modifiers*, *Field type*,
  Field name)
<Initialization>
  operator

Ця конструкція використовується для визначення поля класу, якому присвоюється певне початкове значення. Тут *Modifiers* – список модифікаторів доступу (наприклад, public, static, final і т. п.); *Field*

type – тип даних поля; *Field name* – назва поля; <Initialization> – рядок, після якого потрібно вказати оператор або вираз, значення якого буде використане для ініціалізації поля.

4. Визначення анотованого поля даних:

Field annotated (*Modifiers*, *Field type*, *Field name*)
<Annotations>
Annotation (*Annotation text*);
Annotation (*Annotation text*);
...

Тут словом annotated позначене поле, відмічене анотаціями (наприклад, *Resource*), які необхідно вказати після рядка <Annotations>.

5. Визначення анотованого методу:

Method annotated (*Return type*, *Method name*, *Parameters list*),
<Annotations>
Annotation (*Annotation text*);
Annotation (*Annotation text*);
...
<Body>
operators

Тут *Return type* – тип значення, яке повертає метод; *Method name* – назва методу; *Parameters list* – список формальних параметрів; <Annotations> – рядок, після якого зазначаються анотації до методу (наприклад, *Override*, *Transactional*); <Body> – рядок, після якого вказуються оператори тіла метода.

6. Конструкція для виклику методу екземпляра (об'єкта) класу має вигляд:

Call instance method (*Instance name*, *Method name*, *Arguments*),

де *Instance name* – ідентифікатор екземпляра класу; *Method name* – назва методу класу; *Arguments* – список фактичних параметрів методу.

Приклад використання операцій, розглянутих у даному підрозділі, наведено у розділі 3.

1.3. Інструментальні системи автоматизованої розробки програм. Розроблені інструментальні засоби підтримки проектування та генерації програм ґрунтуються на використанні розглянутих засобів САА-М та методу діалогового конструювання синтаксично правильних програм (ДСП-методу) [2]. На відміну від традиційних синтаксичних аналізаторів, ДСП-метод орієнтований не на пошук і виправлення синтаксичних помилок, а на виключення можливості їх появи в процесі побудови алгоритму. Основна ідея методу полягає в порівневому конструюванні схем зверху вниз шляхом суперпозиції мовних конструкцій САА-М. На кожному кроці конструювання система в діалозі з користувачем надає на вибір лише ті конструкції, підстановка яких у текст алгоритму, що проектується, не порушує синтаксичну правильність схеми. На основі побудованої схеми алгоритму виконується автоматична генерація тексту програми цільовою мовою програмування. Відображення операцій САА-М у текст мовою програмування подане у вигляді шаблонів і зберігається в базі даних інструментарію.

Згаданий підхід був реалізований в системі ІПС, розглянутій у роботах [2–6]. Основними компонентами системи ІПС є такі:

- діалоговий конструктор синтаксично правильних програм (ДСП-конструктор), призначений для діалогового проектування схем алгоритмів та синтезу програм мовами Java та C++;
- редактор граф-схем;
- база даних алгеброалгоритмічних специфікацій, у якій зберігається текст конструкцій САА і базисних елементів схем, а також їх програмні реалізації;
- генератор САА-схем за гіперсхемами.

Для автоматизації виконання трансформацій алгоритмів система ІПС застосовується спільно з системою TermWare [10], що ґрунтується на парадигмі переписувальних правил.

У роботах [7–9] розглядається нова версія системи ІПС, названа онлайновим діалоговим конструктором синтаксично правильних програм. Система ОДСП при-

значена для діалогового проектування, генерації й запуску програм. Основна відмінність інструментарію ОДСП у порівнянні з системою ІПС полягає у сервісно-орієнтованій архітектурі інструментарію та спрямованості на багатокористувальницьке використання інструментарію через Інтернет. Іншою відмінністю є орієнтація системи ОДСП на парадигму об'єктно-орієнтованого програмування, тоді як інструментарій ІПС в основному спрямований на імперативну парадигму.

У попередніх роботах [2–9] засоби САА-М та розроблені інструментальні системи використовувались для автоматизації проектування та генерації паралельних програм для багатоядерних центральних процесорів та графічних прискорювачів, зокрема, у предметній області метеорологічного прогнозування.

У наступних розділах виконаний подальший розвиток створених засобів у напрямку їх застосування для тривимірного випадку задачі прогнозування, а також автоматизації розробки програмного забезпечення для Інтернет-порталу з надання послуг прогнозування погоди.

2. Проектування паралельної схеми алгоритму для розв'язання тривимірної задачі конвективної дифузії

У даному розділі розглядається застосування апарату САА-М та системи ІПС для розробки паралельної програми розв'язання тривимірної задачі конвективної дифузії, яка виникає при математичному моделюванні циркуляції атмосфери в метеорології.

2.1. Математична модель. Постановка задачі конвективної дифузії детально викладена в [11]. У згаданій роботі розглянута модель циркуляції атмосфери, яка описується такими основними фізичними законами:

1) збереження кількості руху

$$\frac{dV}{dt} = -\frac{1}{\rho} gradp - 2\Omega \times V + F_g + \frac{1}{\rho} F_f, \quad (1)$$

2) збереження маси (нестисливість середовища)

$$divV = 0, \quad (2)$$

3) збереження теплової енергії

$$\frac{dT}{dt} = \frac{\delta}{c_V \rho}, \quad (3)$$

4) рівняння стану повітря

$$p = \rho R_C T, \quad (4)$$

де V – вектор швидкості у неінерційній системі координат, ρ – густина повітря, p – атмосферний тиск, Ω – вектор кутової швидкості обертання Землі, F_g – сила тяжіння, F_f – щільність сили тертя, T – абсолютна температура, δ – притік тепла до одиниці об'єму повітря, c_V – питома теплоємність сухого повітря за сталого об'єму, R_C – питома газова стала сухого повітря.

У моделях, що описують макромасштабну циркуляцію, використовують сферичну систему координат [26]. Рівняння (1) – (3) після деяких спрощень матимуть вигляд

$$\frac{\partial u}{\partial t} + \frac{u}{a\cos\varphi}\frac{\partial u}{\partial \lambda} + \frac{v}{a}\frac{\partial u}{\partial \varphi} + w\frac{\partial u}{\partial z} = -\frac{1}{\rho a\cos\varphi}\frac{\partial p}{\partial \lambda} + \left(v\sin\varphi - w\cos\varphi\right)\left(2\omega + \frac{u}{a\cos\varphi}\right) + \frac{1}{\rho}F_\lambda, \quad (5)$$

$$\frac{\partial v}{\partial t} + \frac{u}{a\cos\varphi}\frac{\partial v}{\partial \lambda} + \frac{v}{a}\frac{\partial v}{\partial \varphi} + w\frac{\partial v}{\partial z} = -\frac{1}{\rho a}\frac{\partial p}{\partial \varphi} - u\left(2\omega + \frac{u}{a\cos\varphi}\right)\sin\varphi - \frac{wv}{a} + \frac{1}{\rho}F_\varphi, \quad (6)$$

$$\frac{1}{a\cos\varphi}\frac{\partial u}{\partial \lambda} + \frac{1}{a}\frac{\partial v}{\partial \varphi} + \frac{\partial w}{\partial z} - \frac{v}{a}tg\varphi = 0, \quad (7)$$

$$\frac{\partial T}{\partial t} + \frac{u}{a\cos\varphi}\frac{\partial T}{\partial \lambda} + \frac{v}{a}\frac{\partial T}{\partial \varphi} + w\frac{\partial T}{\partial z} = \frac{\delta}{c_V \rho}, \quad (8)$$

де λ – довгота, φ – широта, z – висота над рівнем моря, $V=(u,v,w)$, a – радіус Землі, ω – швидкість добового обертання Землі, $F_f=(F_\lambda,F_\varphi,F_z)$.

Паралельна чисельна реалізація цієї моделі здійснюється відповідно до трирівневого алгоритму [12], який передбачає розпаралелювання за рівняннями, просторовими напрямками та підобластями. В алгоритмі використовується модифікований адитивно-усереднений метод розщеплення (МАУМ) [13]. До розрахункової області застосовуються покоординатні декомпозиції [27]: розбиття уздовж λ для підзадач за напрямками φ та z; розбиття уздовж φ для підзадач за напрямком λ. Тому в програмі використані два формати подання даних в масивах: основний $[\lambda][\varphi][z]$ та допоміжний $[\varphi][z][\lambda]$. Із рівнянь (5) та (6) визначалися горизонтальні складові руху u та v, із (7) – вертикальна складова w, із (8) – температура T, а із (4) – густина ρ.

2.2. Схема паралельного алгоритму. У даному підрозділі наведена розроблена САА-схема паралельного алгоритму для розв'язання вищерозглянутої задачі конвективної дифузії. Схема сконструйована із використанням інструментальної системи ІПС [2–6]. На основі схеми виконана генерація програмного коду мовою C++ з використанням засобів OpenMP [14]. САА-схема складається з двох складених операторів: "Основна схема" та "Паралельні обчислення", що подані на рис. 1 та рис. 2, відповідно. Згадані складені оператори відповідають функціям *main* та *CalcParallelPart* у програмі мовою C++.

У складеному операторі "Основна схема" виконується виклик операторів ініціалізації даних (параметрів, масивів і т. п.), відліку часу виконання програми, складеного оператора виконання паралельних обчислень, порівняння обчислених значень та збереження результуючих даних у файли. У цій схемі вказані такі змінні: *Proc_Num* – кількість паралельних потоків; *Subdomains* – кількість підобластей; *M_prm* – *m*-параметр МАУМ; *TmKnotsPer12h* – кількість часових точок для 12-годинного періоду; *TmLimCalc* – кінцеве значення часу в умові *m*-циклу; *tau* – часовий крок; *U*, *V* – горизонтальні складові руху, *W* – вертикальна складова; *P* – атмосферний тиск; *T* – абсолютна температура.

СХЕМА ПАРАЛЕЛЬНИЙ АЛГОРИТМ ДЛЯ РОЗВЯЗАННЯ 3-ВИМІРНОЇ ЗАДАЧІ КОНВЕКТИВНОЇ ДИФУЗІЇ

"Основна схема"

==== "Визначити змінну (*Proc_Num*) типу (int) = (1)"
ПОТІМ
"Встановити початкові параметри (*Subdomains*, *M_prm*, *TmKnotsPer12h*, *TmLimCalc*, *tau*)"
ПОТІМ
"Почати відлік часу"
ПОТІМ
"Завантажити дані в масиви для (*P*, *W*, *U*, *V*, *T*)"
ПОТІМ
(*Proc_Num* := "Паралельні обчислення")
ПОТІМ
"Закінчити відлік часу та вивести результат"
ПОТІМ
"Порівняти інтерпольовані та обчислені значення в нормі L2"
ПОТІМ
"Зберегти дані у файли для 3-вимірної задачі конвективної дифузії"

Рис. 1. Початок САА-схеми для задачі конвективної дифузії: складений оператор "Основна схема"

```
"Паралельні обчислення"
==== "Визначити константу (sdm_sz) типу (int) = ((SZ_X1-1)/Subdomains+1)"
     ПОТІМ
     "Визначити змінну (NmbThreads) типу (int)"
     ПОТІМ
     "Визначити масив (pOut[DIR][EVL_MT_VAL][MAX_SUBDMN]) типу (double)"
     ПОТІМ
     "(NmbThreads) := (Subdomains * DIR * EVL_MT_VAL)"
     ПОТІМ
     ПАРАЛЕЛЬНО (proc = 0, ..., NmbThreads-1)
     (
       "Ініціалізація змінних та масивів"
       ПОТІМ
       "Коментарій (***** m-цикл *****)"
       ПОТІМ
       "(n) := (M_prm)"
       ПОТІМ
       ПОКИ '(n * tau) <= (TmLimCalc)'
       ЦИКЛ
            "Встановити значення для U, V, T, P, D, W, F, C та граничні умови"
            ПОТІМ
            ВИБІР ['(dir) = (0)'] →
                     "Обчислити задачі для напрямку (Lam)",
                      ['(dir) = (1)'] →
                      "Обчислити задачі для напрямку (Fi)",
                      ['(dir) = (2)'] →
                      "Обчислити задачі для напрямку (Z)")
            ПОТІМ
            "Поміняти місцями значення в масивах pR1 та pR2"
            ПОТІМ
            ЧЕКАТИ 'Обробка в усіх потоках закінчена'
            ПОТІМ
            "Обчислити середні значення на основі отриманих результатів для
             кожного напрямку"
            ПОТІМ
            ЧЕКАТИ 'Обробка в усіх потоках закінчена'
            ПОТІМ
            "Занести результати в глобальні масиви"
            ПОТІМ
            ЧЕКАТИ 'Обробка в усіх потоках закінчена'
            ПОТІМ
            "Збільшити (n) на (M_prm)"
       КІНЕЦЬ ЦИКЛУ
       ПОТІМ
       "Звільнити пам'ять для масивів для 3-вимірної задачі конвективної дифузії"
     )
     ПОТІМ
     "Повернути значення (NmbThreads)";
```

Рис. 2. Продовження САА-схеми для задачі конвективної дифузії: деталізація складеного оператора "Паралельні обчислення"

Схема "Паралельні обчислення" (див. рис. 2) починається з ініціалізації змінних і масивів, після чого виконується *m*-цикл МАУМ. У тілі *m*-циклу спочатку здійснюється підготовка до основних обчислень: постановка граничних умов, обчислення коефіцієнтів рівнянь, вертикальної складової швидкості і т. п. Далі обчислюються підзадачі для кожного з напрямків λ, φ та z. Потім виконується усереднювання результатів, які відповідають різним напрямкам, зі збереженням граничних умов. У кінці циклу отримані результати записуються в загальні масиви.

Окрім вищезазначених, у схемі паралельних обчислень вказані такі додаткові змінні та константи: *proc* – номер потоку; *NmbThreads* – кількість паралельних потоків; *sdm_sz* – розмір підобластей у напрямку λ; *SZ_X*1 – кількість точок скінченно-різницевої сітки у напрямку λ; *pOut, pR*1, *pR*2 – масиви для зберігання проміжних результатів; *DIR* = 3 кількість напрямків (λ, φ, z); *EVL_MT_VAL* = 3 – кількість метеорологічних величин в еволюційному рівнянні; *MAX_SUBDMN* – максимальна кількість підобластей; *n* – змінна *m*-циклу; *D* – густина повітря; *F* – коефіцієнт рівнянь; *C* – адвекція; *dir* – цілочисельна змінна для зберігання значення поточного напрямку (дорівнює 0 для λ; 1 для φ; 2 для z). Кількість паралельних потоків в алгоритмі встановлюється згідно формули

$$(NmbThreads) := (Subdomains * DIR * * EVL\_MT\_VAL). \quad (9)$$

Як згадувалося вище, на основі побудованої САА-схеми в системі ІПС виконана генерація програмного коду мовою C++ із використанням OpenMP [14]. OpenMP є сукупністю директив компілятора (прагм), бібліотечних процедур і змінних оточення, що призначена для програмування багатопотокових застосувань на багатопроцесорних системах із загальною пам'яттю. У процесі генерації коду ДСП-конструктор інструментарію ІПС використовує шаблони програмних реалізацій операцій САА-М та базисних понять із бази даних. Зокрема, операцію асинхронного виконання потоків (ПАРАЛЕЛЬНО) для задачі, що розглядається, було реалізовано із використанням OpenMP директиви *#pragma omp parallel*. Для реалізації операції синхронізації (ЧЕКАТИ) було застосовано директиву *#pragma omp barrier*, яка виконує затримку обчислень доти, поки всі потоки не завершать свою роботу.

Результати проведеного експерименту з виконання згенерованої програми на мультипроцесорній платформі розглядаються у розділі 4.

3. Проектування онлайнового сервісу для надання послуг метеорологічного прогнозування

У даному розділі проілюстроване застосування засобів алгебри алгоритмів та системи ОДСП на прикладі розробки онлайнового сервісу, який входить до складу сервісно-орієнтованої системи «Оракул-Портал», що призначена для надання послуг метеорологічного прогнозування [1]. Згаданий портал складається з декількох модулів-сервісів і використовує паралельні обчислення на мультипроцесорній платформі. Структура системи детально розглянута у роботі [1]. До складу порталу входять сервіси управління метеорологічними прогнозами та розрахунку прогнозів. База даних (БД) системи містить вихідні метеодані, постачальником яких є метеорологічний центр у м. Офенбах [28], а також дані уточнюючих прогнозів, розрахованих на основі вихідних даних. Значна частина системи написана мовою Java, крім власне паралельних програм для виконання обчислення прогнозу, які розроблені мовою C++ із використанням OpenMP. На даний момент використовуються три варіанти програм для обрахунку прогнозів: двовимірного, тривимірного та періодичного. Вище у розділі 2 даної роботи наведено побудовану САА-схему паралельного алгоритму для автоматизованої генерації програми для тривимірного випадку задачі прогнозування.

Далі на рис. 3 показано побудовану в системі ОДСП схему для Java класу *Prediction3DServiceImpl*, який реалізує один із сервісів системи «Оракул-Портал».

```
 Annotated class X implements Y (Prediction3DServiceImpl, Prediction3DService)
<Annotations>
    Annotation (Service)

<Fields>

    Field initialized (private static final, Logger, LOGGER)
    <Initialization>
        Call instance method (Logger, getLogger, Prediction3DServiceImpl.class);

    Field annotated (public, Prediction3DRepository, prediction3dRepository)
    <Annotations>
        Annotation (Resource)

<Methods>

    Method annotated (Prediction3D, findById, Long id)
    <Annotations>
        Annotation (Override);
        Annotation (Transactional)
    <Body>
        Call instance method (LOGGER, debug, "Trying get prediction3d #" + id +
                                                    " from DB");
        Return (Call instance method (prediction3dRepository, findById, id));

    Method annotated (void, savePrediction, Prediction3D prediction)
    <Annotations>
        Annotation (Override);
        Annotation (Transactional)
    <Body>
        Assign (prediction, Call instance method (prediction3dRepository, save,
                                                    prediction));
        Call instance method (LOGGER, debug,"Prediction3D #"
                                        + prediction.getId() + " has been saved to DB");

    Method annotated (Long, createPrediction, Prediction3D prediction)
    <Annotations>
        Annotation(Override)
    <Body>
        Assign (prediction, Call instance method (prediction3dRepository, save, prediction));
        Return (Call instance method (prediction, getId));

    Method annotated (void, deletePrediction, Long id)
    <Annotations>
        Annotation(Override)
    <Body>
        Call instance method (prediction3dRepository, delete, id);
```

Рис. 3. Схема класу Prediction3DServiceImpl

При конструюванні наведеної схеми використані об'єктно-орієнтовані операції CAA-M, розглянуті у підрозділі 1.2. Клас *Prediction3DServiceImpl* визначено за допомогою конструкції "Annotated class X implements Y", у параметрах якої вказано назву класу та назву реалізованого ним інтерфейсу *Prediction3DService*. Після заголовка класу наведено опис анотації, визначення полів та методів класу.

Полями класу є ініціалізоване поле *LOGGER* та анотоване поле *prediction3DRepository*. Поле *LOGGER* є екземпляром Java класу *Logger*, методи якого призначені для запису повідомлень у журнал. Поле *prediction3DRepository* представляє базу даних порталу.

Клас *Prediction3DServiceImpl* також містить чотири методи. Метод *findById* призначений для пошуку (за номером *id*) обрахованого прогнозу у базі даних. Методи *savePrediction*, *createPrediction* та *deletePrediction* призначені для збереження, створення та видалення прогнозів з БД відповідно.

На основі побудованої схеми в системі ОДСП виконана автоматична генерація програмного коду класу *Prediction3DServiceImpl* мовою Java.

4. Результати чисельних експериментів

Проведено два незалежних експерименти з виконання OpenMP програми для розв'язання тривимірної задачі метеорологічного прогнозування, згенерованої за побудованою у розділі 2 схемою, на двох мультипроцесорних системах. Перша система є одним з вузлів кластера Інституту програмних систем НАНУ і складається із двох чотирьох-ядерних процесорів Quad Core Intel Xeon E5405 із частотою 2 ГГц (всього 8 ядер на вузлі). Як другу мультипроцесорну систему використано один з вузлів суперкомп'ютерного обчислювального комплексу СКІТ-4 Інституту кібернетики імені В.М. Глушкова НАН України [29]. Згаданий вузол містить два шестиядерних процесори Intel Xeon X5675, із частотою 3.07 ГГц (всього 12 фізичних ядер на вузлі; 24 логічних ядра – у режимі *hyper-threading*).

У процесі експерименту були встановлені такі значення параметрів програми:

термін часу, на який виконується розрахунок прогнозу погоди – 12 годин;

m-параметр модифікованого адитивно-усередненого методу розщеплення [11–13] *M_prm* = 10;

кількість точок часової сітки *TmKnotsPer12h* = 4320;

кінцеве значення часу *TmLimCalc* = 43200 сек;

часовий крок *tau* = 10 сек;

кількість підобластей *Subdomains* = 2, кількість потоків *NmbThreads* = 18 (див. формулу (9) у підрозділі 2.2).

Просторова область визначення задачі відповідала наявним архівним даним і становила:

$$\lambda \in [0^\circ, 90^\circ],\ \varphi \in [0^\circ, 90^\circ],$$

$$z \in [8000; 18000].$$

Розмір вхідної скінченно-різницевої сітки обрано $182 \times 182 \times 11$ точок.

У таблиці подано отримані результати виконання програми на 1 та 8 ядрах процесорів кластера ІПС НАНУ та 1 та 12 фізичних ядрах кластера СКІТ-4. Як видно з таблиці, проведені експерименти демонструють гарний ступінь розпаралелюваності обчислень. Краще значення прискорення на кластері СКІТ-4 пояснюється більшою кількістю ядер та наявністю вищезгаданого режиму *hyper-threading*.

Таблиця. Результати виконання паралельної програми для розв'язання задачі конвективної дифузії на кластерах ІПС та СКІТ-4

Мультипроцесорна система	Час $T(1)$ виконання на 1 ядрі, сек	Час $T(N)$ виконання на N ядрах, сек	Мультипроцесорне прискорення, $Sp = \frac{T(1)}{T(N)}$
Кластер ІПС, кількість фізичних ядер $N = 8$	1504,80	259,43	5,80
Кластер СКІТ-4, кількість фізичних ядер $N = 12$	988,20	83,88	11,78

Висновки

Виконане автоматизоване конструювання високорівневих алгебро-алгоритмічних специфікацій програмного забезпечення для розв'язання задачі метеорологічного прогнозування. Прогнозування здійснюється на основі паралельної реалізації задач моделювання циркуляції атмосфери на мультипроцесорній платформі. Виконане проектування сервісу, що входить до складу розроблюваного Інтернет-порталу [1] з надання послуг метеопрогнозу. Виконана генерація програмного коду за побудованими специфікаціями на основі використання розроблених інструментальних засобів автоматизованого проектування та синтезу програм. Перевагою використовуваного в інструментарії підходу до проектування програм є використання мовних конструкцій, близьких до природної мови, а також застосування методу, який забезпечує синтаксичну правильність алгоритмів та програм, що проектуються.

Проведено нові експерименти з виконання розробленої паралельної програми для розв'язання задачі метеорологічного прогнозування на мультипроцесорній платформі, що показали гарний ступінь розпаралелюваності обчислень.

1. *Дорошенко А.Ю., Іваненко П.А., Овдій О.М. та інші.* До створення Інтернет-порталу надання послуг метеорологічного прогнозування на мультипроцесорній платформі // Проблеми програмування. – 2015. – № 3. – С. 24–32.
2. *Андон Ф.И., Дорошенко А.Е., Яценко Е.А. и др.* Алгеброалгоритмические модели и методы параллельного программирования. – Киев: Академпериодика, 2007. – 631 с.
3. *Дорошенко Е.А., Яценко Е.А.* О синтезе программ на языке Java по алгеброалгоритмическим спецификациям // Проблеми програмування. – 2006. – № 4. – С. 58–70.
4. *Дорошенко А.Е., Жереб К.А., Яценко Е.А.* Формализованное проектирование эффективных многопоточных программ // Там само. – 2007. – № 1. – С. 17–30.
5. *Яценко Е.А.* Интеграция инструментальных средств алгебры алгоритмов и переписывания термов для разработки эффективных параллельных программ // Там само. – 2013. – № 2. – С. 62–70.
6. *Дорошенко А.Ю., Бекетов О.Г., Яценко О.А. та інші.* Розробка сервісно-орієнтованих засобів для запуску паралельних програм на мультипроцесорному кластері // Там само. – 2014. – № 4. – С. 3–14.
7. *Иовчев В.А., Мохница А.С.* Инструментальные средства алгебры алгоритмики на платформе Web 2.0 // Там само. – 2010. – № 2–3. – С. 547–555.
8. *Дорошенко А.Ю., Бекетов О.Г., Яценко О.А. та інші.* Автоматизована генерація паралельних програм для графічних прискорювачів на основі схем алгоритмів // Там само. – 2015. – № 1. – С. 19–28.
9. *Андон Ф.И., Дорошенко А.Е., Яценко Е.А. и др.* Инструментальные средства автоматизации параллельного программирования

на основе алгебры алгоритмов // Кибернетика и системный анализ. – 2015. – № 1. – С. 162–170.
10. *Doroshenko A., Shevchenko R.* A rewriting framework for rule-based programming dynamic applications // Fundamenta Informaticae. – Amsterdam: IOS Press, 2006. – Vol. 72, N 1–3. – P. 95–108.
11. *Черниш Р.І.* Паралельна реалізація моделі макромасштабної циркуляції атмосфери // Вісник Київського національного університету імені Тараса Шевченка: серія фізико-математичні науки. – 2009. – № 2. – С. 155–158.
12. *Черниш Р.І., Тирчак Ю.М., Іваненко П.А.* Побудова паралельного алгоритму чисельного розв'язання багатовимірної задачі моделювання навколишнього середовища // Проблеми програмування. – 2009. – № 1. – С. 85–91.
13. *Прусов В.А., Дорошенко А.Е., Черныш Р.И.* Метод численного решения многомерной задачи конвективной диффузии // Кибернетика и системный анализ. – 2009. – № 1. – С. 100–107.
14. *OpenMP* Application Program Interface [Електронний ресурс]. – Режим доступу: http://www.openmp.org/mp-documents/ OpenMP4.0.0.pdf. – 04.11.2015 р.
15. *Sannella D., Tarlecki A.* Foundations of algebraic specification and formal software development. – Berlin: Springer-Verlag, 2012. – 594 p.
16. *Flener P.* Achievements and prospects of program synthesis // Lecture Notes in Artificial Intelligence. – Berlin: Springer-Verlag, 2002. – Vol. 2407. – P. 310–346.
17. *Gulwani S.* Dimensions in program synthesis // Proc. 12th Int. ACM SIGPLAN symposium on Principles and Practice of Declarative Programming, Hagenberg, Austria (26–28 July, 2010). – New York: ACM, 2010. – P. 13–24.
18. *Raghesh A.* A framework for automatic OpenMP code generation. A project report [Електронний ресурс]. – Режим доступу: http://www. cse.iitm.ac.in/~raghesh/raghesh-a-masters-thesis.pdf. – 04.11.2015 р.
19. *Hu K., Zhang T., Yang Z.* Multi-threaded code generation from Signal program to OpenMP // Frontiers of Computer Science. – Berlin: Springer-Verlag, 2013. – Vol. 7, N 5. – P. 617–626.
20. *PLUTO* – an automatic parallelizer and locality optimizer for multicores [Електронний ресурс]. – Режим доступу: http://pluto-compiler.sourceforge.net/ – 04.11.2015 р.
21. *Par4all* [Електронний ресурс]. – Режим доступу: – http://www.par4all.org/ – 04.11.2015 р.
22. *Kabir M.H.* Automatic construction of Java programs from functional program specifications // International Journal Of Advanced Chemical Science and Applications. – Bhubaneswar: IRD India, 2015. – Vol. 6, N 4. – P. 65–72.
23. *Schoeberl M., Brooks C., Lee E.A.* Code generation for embedded Java with Ptolemy // Proc. 8th IFIP Workshop on Software Technologies for Future Embedded and Ubiquitous Systems, Waidhofen, Austria (13–15 October, 2010). Lecture Notes in Computer Science. – Berlin: Springer-Verlag, 2010. – Vol. 6399. – P. 155–166.
24. *Herrington J.* Code-generation techniques for Java [Електронний ресурс]. – Режим доступу: http://www.onjava.com/pub/a/onjava/ 2003/09/03/generation.html. – 04.11.2015 р.
25. *Oracle* Java documentation. The Java Tutorials. Lesson: Annotations [Електронний ресурс]. – Режим доступу: https://docs. oracle.com/javase/tutorial/java/annotations/ – 04.11.2015 р.
26. *Белов П.Н.* Практические методы численного прогноза погоды. – Ленинград: Гидрометеорологическое издательство, 1967. – 336 с.
27. *Черниш Р.І.* Покоординатна декомпозиція області для еволюційних задач математичної фізики // Вісник Київського національного університету імені Тараса Шевченка: серія фізико-математичні науки. – 2008. – № 4. – С. 191–194.
28. *Wetter* und Klima – Deutscher Wetterdienst – Startseite (веб-сайт метеорологічного центру у м. Офенбах) [Електронний ресурс]. – Режим доступу: http://www.dwd. de. – 04.11.2015 р.
29. *Суперкомп'ютер* ІК НАН України. – http://icybcluster.org.ua.

References

1. *Doroshenko A.Yu., Ivanenko P.A., Ovdii O.M.* at all. Creation of an Internet portal providing meteorological forecasting services on multi-processor platform // Problems in programming. – 2015. N 3. – P. 24–32 (in Ukrainian).
2. *Andon P.I. et al.* Algebra-algorithmic models and methods of parallel programming. Kiev: Academperiodika, 2007 (in Russian).

3. *Doroshenko, A.Yu. & Yatsenko O.A.* About the synthesis of Java programs by algebra-algorithmic specifications // Problems in programming. – 2006. – N 4. – P. 58–70 (in Russian).
4. *Doroshenko, A.Yu., Zhereb K.A & Yatsenko O.A.* Formalized design of efficient multi-threaded programs // Problems in programming. – 2007. – N 1. – P. 17–30 (in Russian).
5. *Yatsenko O.A.* (2013) Integration of algebra-algorithmic tools and term rewriting for efficient parallel programs development // Problems in programming. – 2013. – N 2. – P. 62–70 (in Russian).
6. Doroshenko A.Yu., Beketov O.G., Yatsenko O.A. at all. Development of the service-oriented soft-ware for launching parallel programs on a multiprocessor cluster // Problems in programming. – 2014. – N 4. – P. 3–14 (in Ukrainian).
7. Iovchev V.O. & Mokhnitsa O.S. Algebra-algorithmic tools on Web 2.0 platform // Problems in programming. – 2010. – N 2. – P. 547–555 (in Russian).
8. *Doroshenko A.Yu., Beketov O.G., Ivaniv R.B. at all.* Automated generation of parallel programs for graphics processing units based on algorithm schemes. Problems in programming. – 2015. – N 1. – P. 19–28 (in Ukrainian).
9. *Andon, P.I., Doroshenko, A.Yu., Beketov, O.G., Iovchev, V.O. & Yatsenko O.A.* (2015) Software tools for automation of parallel programming on the basis of algebra of algorithms // Cybernetics and systems analysis. (1). P. 162–170 (in Russian).
10. *Doroshenko A. & Shevchenko R.* (2006) A rewriting framework for rule-based programming dynamic applications. Fundamenta Informaticae. Amsterdam: IOS Press. 72 (1-3). – P. 95–108.
11. *Chernysh R.I.* (2009) Parallel implementation of atmospheric macroscale circulation model. Bulletin of the University of Kiev, Series: Physics & Mathematics. (2). – P. 155–158 (in Ukrainian).
12. *Chernysh R.I., Tyrchak Yu.M. & Ivanenko, P.A.* (2009) Construction of parallel algorithm for numeric solving multidimensional problem of environmental modeling // Problems in programming. (1). – P. 85–91 (in Ukrainian).
13. *Prusov, V.A., Doroshenko, A.Yu. & Chernysh, R.I.* (2009) A method for numerical solution of a multidimensional convection-diffusion problem // Cybernetics and systems analysis. (1). – P. 100–107 (in Russian).
14. *OPENMP*. (2015) OpenMP Application Program Interface. [Online] Available from: http://www.openmp.org/mp-documents/ OpenMP4.0.0.pdf [Accessed: 4 November 2015].
15. *Sannella D. & Tarlecki A.* (2012) Foundations of algebraic specification and formal software development. Berlin: Springer-Verlag.
16. *Flener P.* (2002) Achievements and prospects of program synthesis. Lecture Notes in Artificial Intelligence. Berlin: Springer-Verlag. 2407. – P. 310–346.
17. *Gulwani S.* (2010) Dimensions in program synthesis. In Proc. 12th Int. ACM SIGPLAN symposium on Principles and Practice of Declarative Programming. Hagenberg, Austria, 26-28 July 2010. New York: ACM. – P. 13–24.
18. *Raghesh A.* (2011) A framework for automatic OpenMP code generation. A project report. [Online]. Available from: http://www.cse.iitm.ac.in/~raghesh/raghesh-a-masters-thesis.pdf [Accessed: 4 November 2015].
19. *Hu K., Zhang T., Yang Z.* Multi-threaded code generation from Signal program to OpenMP // Frontiers of Computer Science. – Berlin: Springer-Verlag, 2013. – Vol. 7, N 5. – P. 617–626.
20. *PLUTO-COMPILER.* (2015) *PLUTO – an automatic parallelizer and locality optimizer for multicores.* [Online] Available from: http://pluto-compiler.sourceforge.net/ [Accessed: 4 November 2015].
21. *PAR4ALL.* [Online] Available from: http://www.par4all.org/ [Accessed: 4 November 2015].
22. *Kabir M.H.* Automatic construction of Java programs from functional program specifications // International Journal Of Advanced Chemical Science and Applications. – Bhubaneswar: IRD India, 2015. – Vol. 6, N 4. – P. 65–72.
23. *Schoeberl M., Brooks C., Lee E.A.* Code generation for embedded Java with Ptolemy // Proc. 8th IFIP Workshop on Software Technologies for Future Embedded and Ubiquitous Systems, Waidhofen, Austria (13–15 October, 2010). Lecture Notes in Computer Science. – Berlin: Springer-Verlag, 2010. – Vol. 6399. – P. 155–166.
24. *Herrington J.* (2003) Code-generation techniques for Java. [Online] Available from: http://www.onjava.com/pub/a/onjava/ 2003/09/03/generation.html [Accessed: 4 November 2015].
25. *ORACLE.* (2015) Java documentation. The Java Tutorials. Lesson: Annotations. [Online]

Available from: https://docs.oracle.com/javase/tutorial/java/annotations/ [Accessed: 4 November 2015].
26. *Belov P.N.* (1967) Practical methods for numerical weather prediction. Saint Petersburg: Hydrometeorological publishing. (in Russian).
27. *Chernysh R.I.* (2008) Coordinate-wise decomposition of area for evolutionary tasks of mathematical physics. Bulletin of the University of Kiev, Series: Physics & Mathematics. (4). – P. 191–194 (in Ukrainian).
28. *DWD* (2015) Wetter und Klima – Deutscher Wetterdienst. [Online] Available from: http://www.dwd.de [Accessed: 4 November 2015].
29. *ICYBCLUSTER.* (2015) Supercomputer of IC [Online] Available from: http://icybcluster.org.ua [Accessed: 4 November 2015].

Одержано 23.11.2015

***Про авторів*:**

Дорошенко Анатолій Юхимович,
доктор фізико-математичних наук,
професор, завідувач відділу теорії
комп'ютерних обчислень Інституту
програмних систем НАН України,
професор кафедри автоматики і управління в технічних системах НТУУ "КПІ".
Кількість наукових публікацій в
українських виданнях – понад 150.
Кількість наукових публікацій в
іноземних виданнях – понад 30.
Індекс Гірша – 3.
http://orcid.org /0000-0002-8435-1451,

Іваненко Павло Андрійович,
молодший науковий співробітник
Інституту програмних систем
НАН України.
Кількість наукових публікацій в
українських виданнях – понад 30.
Кількість наукових публікацій в
іноземних виданнях – 2.
http://orcid.org/0000-0001-5437-9763.

Овдій Ольга Михайлівна,
молодший науковий співробітник
Інституту програмних систем
НАН України.
Кількість наукових публікацій в
українських виданнях – 16.
Кількість наукових публікацій в
іноземних виданнях – 2.
http://orcid.org/0000-0002-8891-7002.

Яценко Олена Анатоліївна,
кандидат фізико-математичних наук,
старший науковий співробітник Інституту
програмних систем НАН України.
Кількість наукових публікацій в
українських виданнях – 28.
Кількість наукових публікацій в
іноземних виданнях – 12.
http://orcid.org/0000-0002-4700-6704.

Місце роботи авторів:

Інститут програмних систем
НАН України,
03187, м. Київ-187,
проспект Академіка Глушкова, 40.
Тел.: (044) 526 3559.

E-mail: doroshenkoanatoliy2@gmail.com,
paiv@ukr.net,
olga.ovdiy@gmail.com,
oayat@ukr.net

УДК 004.7

O.P. Ignatenko

GAME THEORETIC MODELING OF AIMD NETWORK EQUILIBRIUM

This paper deals with modeling of network's dynamic using game theory approach. The process of interaction among players (network users), trying to maximize their payoffs (e.g. throughput) could be analyzed using game-based concepts (Nash equilibrium, Pareto efficiency, evolution stability etc.). In this work we presented the model of TCP network's dynamic and proved existence and uniqueness of solution, formulated payoff matrix for a network game and found conditions of equilibrium existence depending of loss sensitivity parameter. We consider influence if denial of service attacks on the equilibrium characteristics and illustrate results by simulations.
Key words: game theory, network model, Nash equilibrium, network simulation.

Introduction

It is almost impossible now to imagine our life without computer networks. Only for several decades, the Internet has rapidly transformed the ways in which individuals, societies and even governments communicate, exchange information and conduct their economic and social activities. And this process is far from the ending. As was envisioned recently by Google's executive chairman Eric Schmidt: "the internet will disappear as everything in our life gets connected. There will be so many devices, sensors, things that you are wearing, things that you are interacting with that you won't even sense it. It will be part of your presence all the time."

It seems that future Internet has to be self-organizing, self-protecting, and self-optimizing.

This next-generation communication environment will include interaction of intelligent devices that are capable of autonomously take decisions within highly dynamic and rapidly changing digital world. However, the continuous (and successful till now) development of networks, which is accompanied by exponential growing of their complexity, heterogeneity and distributiveness and which appears natural at the present time, creates new challenging problems. As mentioned in [1], to address these problems with appropriate approach we need to develop a new set of models based on control theory, game theory, and network optimization.

First, let us introduce a problem on a high level. Consider a interaction between selfish users in a network (network here is some common pool with limited resources). Each user can adopt a method of action (strategy) which have influence on whole network and other users. The examples of actions are: choose protocol, change rate or route of data flows. For every possible combination of adopted strategies (outcome) there is a reward or utility for each user, which indicates his/her preferences over outcomes. Selfishness (or rationality in game theory terminology) means that a user wants to maximize utility. For instance, when user wants to download big file he prefers to receive as much network resources as possible. However, if user wants to read news then small but stable connection is sufficient. This situation leads to obvious conflict when summary users' demand is bigger then network supply. If this happens network drops or delays users' data, so generally it is not good for users. From the other side, network underloading (when demand is smaller then network capacity) is also undesirable because leads to inefficiency of resource using.

Delivering information about network state to end user is a challenging problem and crucial part of any feedback based protocol. As a rule user has knowledge about successful delivery of his data (in other words he knows that network is probably underloaded) and about overload event (if he doesn't receive successful ACK – acknowledgement packet) with some delay. This type of information is called binary feedback. The natural rate

control based on this information called AIMD (additive increase, multiplicative decrease) scheme. There are another possibilities, but it was proved that AIMD algorithm will oscillate near the point of effective (all bottle-necks will be loaded) and fair (in some sense) allocation of network resources. AIMD was the core of first version of first successful protocol – TCP, which still carries 70 percent of the Internet traffic. Nowadays, TCP isn't one protocol but big family (number keeps increasing) of algorithms with different implementations of the origin idea.

Protocol development went through the competitive evolution between different protocols, abandonment of some of them and appearance of new ones. The possibility to deploy new versions of protocols gives user control to improve performance of his connection by choosing suitable algorithm. When many users are trying to achieve better performance it is difficult to predict consequences of such a competition. There is a problem how to ensure stable, fair and effective network behavior in the situation of dynamic and antagonistic interaction of selfish users. First natural approach to address this problem with optimization framework was developed in work by Kelly et al. [2]. Later it was shown that congestion control, routing and scheduling in wired and wireless networks can be thought of as fair resource allocation. The protocols in this framework are nothing else as algorithms that allow a decentralized solution of the problem. This idea to consider network as an algorithm for solving maximization problem of total network utility (sum of users' utilities) proved to be very fruitful [3]. The limitation in this approach is that protocol (in centralized or decentralized manner) dictates what strategy user should use.

It is natural to assume that users try to improve performance of their connection by choosing suitable protocol. The problem here lies in interaction between different implementations of TCP which could be "unfriendly". This means that one implementation is more "aggressive" and another is more "peaceful" in competition for resources. The question of protocols interaction is quite complex. Building analytic model for predicting network behavior for different protocols is a challenging problem. There are many approaches of investigation of complex networks from different directions (static, dynamic, deterministic etc). There is, however, novel systematic approach towards network modeling – the game theory. Game theory addresses problems in which multiple players with conflicting goals compete with each other. The evolutionary games concept is a part of game theory that focuses on studying interactions between populations rather than individual players. One of the earliest publications about the use of evolutionary games in networking is [4] that study through simulations some aspects of competition between TCP users. For this model it was shown that dynamic of this process described by difference equation has stable solution and users payoffs are forming a structure of evolutionary game known as Hawk-Dove game. Also there were identified conditions under which equilibrium is evolutionary stable.

There is another possible reason of inefficient, unstable or unpredictable network behavior – security violation. Unfortunately, networks have many security issues: illegal data access, viruses, network attacks, etc. One of the most dangerous attacker's activities are Denial of Service (DoS) attacks. DoS attack aims to stop the service provided by a target. When the traffic of a DoS attack comes from multiple sources, it called a Distributed Denial of Service (DDoS) attack. By using multiple attack sources, the power of a DDoS attack is amplified and the problem of defense is made more complicated. Currently we have numerous DoS attack types. Each attack uses some special exploit of Internet protocols or software weaknesses. Recently novel type of attack was developed. This low-rate attacks, using carefully calculate timing, imply significant inefficiencies that tremendously reduce system capacity or service quality. In the literature, this kind of network intrusion is called shrew attack or Reduction of Quality (RoQ) attack. This constant development of new attacks demands new solutions especially in attack detection area.

Intrusion Detection Systems (IDSs) is a software which is used to monitor events occurring in a network. An IDS is also used to analyze these events in order to determine whether an attack has occurred. Once an attack is detected, a report is sent to the network administrator. Current IDSs are not very sophisticated and rely on ad hoc schemes and experimental algorithms. Due to these, IDSs need theoretical tools to handle sophisticated, organized attacks. Game theoretic approaches have been proposed by many researchers to improve network security, for example to analyze high level "security investment game", but these models usually don't include network dynamics.

Game theory provides mathematical base for analyzing and modeling security problems with many agents which could interact in complex, dynamic environment. The advantage of game theory approach is s possibility of analyzing many different scenarios before adopting a certain strategy. Using mathematical modeling we can simulate network topology, controlling algorithms and users' actions. This model could greatly improve network administration by predicting future security problems and likely behavior of users *before* we actually start to build our network.

On the other hand network security measurements involve risk assessment. For example, one of the metrics is the probability of it being attacked. If we adopt game theory view on network dynamic then we can formulate conditions when interaction between rational users leads to an equilibrium state of the network. Network attack is a result of malicious actions of attacker. Attack changes equilibrium characteristics and could be therefore detected.

In this work we describing integrated approach based on game theory models. First, we introduce formal model of TCP network dynamic. We mainly focus on AIMD behavior because it is the most important mechanism of TCP congestion control. Using dynamic systems theoretic results we show existence and stability of network resources allocation point. Then we consider game between users in the network and formulate conditions for Nash equilibrium existence and uniqueness. We introduce network attacker into system and estimate attack influence on equilibrium characteristics. Finally, we show simulation results and make conclusions of future trends.

1. Game theory. Definitions

We will limit our scope with non-cooperative games in strategic or normal form. A non-cooperativeness here does not imply that the players do not cooperate, but it means that any cooperation must be self-enforcing without any coordination among the players. Strict definition is as follows.

A non-cooperative game in strategic (or normal) form is a triplet

$$G = \{\mathrm{N}, \{S_i\}_{i\in\mathrm{N}}, \{u_i\}_{i\in\mathrm{N}}\},$$

where:

– N is a finite set of players, i. e., $\mathrm{N} = \{1,...,N\}$;

– S_i is the set of admissible strategies for player i;

– $u_i : S \to R$ is the utility (payoff) function for player i, with $S = S_1 \times ... \times S_N$ (Cartesian product of the strategy sets).

A game is said to be static if the players take their actions only once, independently of each other. In some sense, a static game is a game without any notion of time, where no player has any knowledge of the decisions taken by the other players. Even though, in practice, the players may have made their strategic choices at different points in time, a game would still be considered static if no player has any information on the decisions of others. In contrast, a dynamic game is one where the players have some information about each others' choices and can act more than once, and where time has a central role in the decision-making. When dealing with dynamic games, the choices of each player are generally dependent on some available information. There is a difference between the notion of an action and a strategy. A strategy can be seen as a mapping from the information available to a player to the action set of this player.

Based on the assumption that all players are rational, the players try to maximize their payoffs when responding to

other players' strategies. Generally speaking, final result is determined by non-cooperative maximization of integrated utility. In this regard, the most accepted solution concept for a non-cooperative game is that of a Nash equilibrium, introduced by John F. Nash. Loosely speaking, a Nash equilibrium is a state of a non-cooperative game where no player can improve its utility by changing its strategy, if the other players maintain their current strategies. Formally, when dealing with pure strategies, i.e., deterministic choices by the players, the Nash equilibrium is defined as follows:

A pure-strategy Nash equilibrium (NE) of a non-cooperative game

$$G = \{\mathrm{N}, \{S_i\}_{i\in\mathrm{N}}, \{u_i\}_{i\in\mathrm{N}}\}$$

is a strategy profile $s^* \in S$ such that for all $i \in \mathrm{N}$ we have the following:

$$u_i(s_i^*, s_{-i}^*) \geq u_i(s_i, s_{-i}^*) \text{ for all } s_i \in S_i .$$

Here $s_{-i} = [s_j]_{j\in\mathrm{N}, i\neq j}$ denotes the vector of strategies of all players except i. In other words, a strategy profile is a pure-strategy Nash equilibrium if no player has an incentive to unilaterally deviate to another strategy, given that other players' strategies remain fixed.

Another important concept is Pareto-dominance, which allow two strategies to be compared. The strategy profile $s^* \in S$ Pareto-dominates

$$s \in S \text{ if for all } i \in \mathrm{N} \;\; u_i(s^*) \geq u_i(s).$$

The strategy profile $s^* \in S$ is a Pareto-optimal profile if it is dominated by no other profile. In Pareto-optimal profile no player could make his payoff better without worsen payoff of some other player. Now consider the notion of best response. The best response (BR) of player *i* to the strategy profile s_{-i} is a correspondence

$$BR_i(s_{-i}) = \arg\max_{s_i\in S_i} u_i(s_i, s_{-i}).$$

The *BR* is a correspondence that is a set-valued function. In practice this means that for some situations player has (possible) many strategies with the same payoff. Using best response notion we can characterize Nash equilibrium as follows. A pure-strategy Nash equilibrium of a non-cooperative game

$$G = \{\mathrm{N}, \{S_i\}_{i\in\mathrm{N}}, \{u_i\}_{i\in\mathrm{N}}\}$$

is a strategy profile

$$s^* \in S \text{ such that } s^* \in BR(s^*).$$

The strong side of Nash concept is that every game has at least one NE (under mild assumptions). From the other hand it is common situation to meet many NEs or to have Pareto-dominated NE.

Let us define the last metric. From network performance point of view it is important to measure the aggregated payoff. The social optimum of a game is a maximum of the sum of the utilities of all players. Any social optimum is Pareto optimal.

2. Game theory for security problems

Let us fix following notations to explain attack-defense interaction in networks. **Network** is a collection of nodes and links. Node can be a server, user or router. Legitimate users are rational and interested in minimizing their own costs. Any user that launches an attack on a network is called **attacker**. IDS is a hardware or software system used to monitor the events occurring in a network or computer system. The main purpose of IDS is of course detection of attack. There are two possible issues: false alarms and missing detection.

Several different approaches have been proposed for detecting intrusions. A currently widely used method is to check monitored events (packets in the network, log files, etc.) against a known list of security attack signatures. This approach has the advantage of enjoying a relatively small false alarm rate and ease of implementation. The disadvantages are the need to maintain and update the attack signature database, and the restriction to detection of only the known attacks documented in the database. These information structures are also useful in detecting more organized multistep attacks. An alternative approach is the anomaly detection, where changes in the patterns of

nominal usage or behavior of the system are detected. Although this approach increases the probability of detecting undocumented new attacks it is difficult to implement, and has often a higher false alarm rate. We introduce an idea to develop game theory based detection of anomalies. A significant shortcoming of the current IDSs is the lack of a unifying mathematical framework to put the pieces into a perspective. Game theory can provide a basis for development of formal decision and control mechanisms for intrusion detection. Specifically, game theoretic models can be used to address issues like the following:

– Develop game model of network using huge amounts of data from detection mechanisms.

– Finding weaknesses and possible targets of an attacker in a large complex system.

– Reconfiguring the security system given the severity of attacks and making decisions on trade-offs like increasing security versus increasing system overhead or decreasing efficiency.

– Deciding on where to allocate or reallocate limited resources in real time to detect significant threats to vital subsystems in a large networked system.

– Analyzing of and modeling the interaction between different types of protocols, allocation algorithms and detection schemes.

Game theory provides a framework to model interaction between selfish, competetive users, malicious attackers and system administrator. Three key elements of such a system are: network dynamic model, game model and scenarios of malicious actions (attacks). Network dynamic modeling is a challenging problem, which was developed last decades. The work of F. Kelly et al. [2] was the first example of considering of Internet network resource allocation as an optimization problem. Later many authors [see for example 5–10] have developed generalizations of this framework. There are many approaches to investigation of complex networks from different angles (static, dynamic, deterministic etc.) using control theory, Petri nets, Markov chains etc.

The evolutionary games concept is a part of game theory that focuses on studying interactions between populations rather than individual players. One of the earliest publications about the use of evolutionary games in networking is [11] that study through simulations some aspects of competition between TCP users. The evolutionary games based on the concept of the ESS (Evolutionary Stable Strategy), defined in 1972 by the biologist Maynard Smith [12]. Fundamental survey of applications of game theory to networks is [9]. In this paper we develop the line of research presented in [13] by Altman et al. We consider a model of users which are using different TCP connections. For this model it was shown that dynamic of this process described by difference equation has a stable solution and users payoffs are forming a structure of evolutionary game known as Hawk-Dove game. Also there were identified conditions under which equilibrium is evolutionary stable.

Considering distributed network of selfish players (e.g. the Internet) we meet problem of efficiency measuring. It is obvious that centralized planning could optimize overall performance of the system. Unfortunately, there are many reasons why it is not possible to construct centralized control over the Internet. However, game theory paradigm generated unexpected idea for dealing with such complex decentralized systems [15]. If game has unique NE and users are rational players then network will operate near this point even without coordination. Game structure, defined by utilities and strategies determine network evolution toward equilibrium point. So it is important to characterize the equilibrium efficiency and to find conditions of existence and uniqueness. A well-known way of characterizing the efficiency of the NE is to calculate whether or not it is Pareto-optimal. However, it is not uncommon for non-cooperative game to have not Pareto-optimal NE. In the work [14] of Papadimitriou was introduced a concept of Price of Anarchy measure. Price of Anarchy (PoA) is equal to the ratio of the highest value of the social optimum to the lest optimal NE of the game. Another important metric is the Price of

Stability which is defined similarly by replacing the denominator of the PoA with the best NE of the game.

We propose a concept for improving security through developing a game theoretic model for better understanding of processes in networks. On the first stage we build comprehensive network model with definite static game structure, which could be dependent on different parameters. Nash equilibriums determine possible dynamic of associated repeated game, so we could calculate metrics and characteristics. Based on this calculation IDS make decisions about anomalies and intrusions.

3. Network dynamic model

We start with notation of network modeling. All propositions in this and following chapters could be found in [16, 17] with detailed explanation.

Consider a network with M nodes. Every node has at least one service link with limited overall capacity p_i, $i = 1,...,M$ (e.g. processing rate, CPU time or network bandwidth). Let $I = \{1,...,M\}$, $K = \{1,...,L\}$ be sets of indexes of nodes and service links respectfully. There are N users, connected to this network. Let $x_j(t)$ be the transmission rate of j user, where $j \in J = \{1,...,N\}$. There is natural assumption about vector of rates $\bar{x} = (x_1,...,x_N)$: $x \in R_+^N$. Users choose their rates $x_j(t)$ at moment t. This means that packets streaming through link $k \in K$ with summary rate $y_k = \sum_{j \in s(k)} x_j(t)$, where $s(k)$ is a set of indexes of users, which use this link. In our simplified model there are no queues and information delays. If sum of transfer rates y_k less then node capacity p_k, then all packets are served. If summary rate of flows using the node's links is equal or bigger than the node capacity then overload event occurs (overload here is a synonym of packet loss). We will assume that routing is deterministic and uncontrolled and information about overload delivers to users momentarily. Let us fix the following notation throughout this paper. Denote $u_k(t)$, $k \in K$ as the service rate of k's link. The constituency matrix is the $M \times L$ matrix C whose c_{ij} element is equal to 1 if i's link belongs to j's node and otherwise is 0. Now we define a control set $U \subset R_+^K$, which contains all possible service rates for the system. Let U be a convex compact set from R_+^K and for any $u \in U$ the inclusion $\alpha u \in U$ holds for any $\alpha \in [0,1]$.Let P be $diag\{p_1,..., p_M\}$ – diagonal matrix. The routing matrix R is the $M \times M$ matrix defined for $i, j \in P$. Element r_{ij} is equal to 1 if the output of i's link is the input of j's link and otherwise is 0. The input matrix A is the $L \times N$ matrix defined for $i \in K, j \in J$. Element a_{ij} is equal to 1 if j's user uses i's link and otherwise is 0.

Overload conditions. When the system produces overload and how one can analytically predict it? This is important problem of network modeling.

Proposition 3.1. (Stability condition) If for $x(t)$, $t \in [t_0, t_1]$ there exist $u \in U$, $\alpha \in [0,1)$, such that

$$P^{-1} \sum_{k=0}^{M-1} \left(R^T\right)^k A\bar{x}(t) = \alpha u,$$

then the system doesn't produce any overload events.

If rates x satisfy stability condition then network will be lossless. But from practical point of view, there are many problems with applicability of this condition. First, in real network each user doesn't have information about system's current state and about rates of other users so he cannot calculate proper rate. Second, user cannot choose any rate he wants (at least in TCP scheme). Instead he chooses protocol, controlling his rate.

Geometric approach. Let $\bar{x}_0$ be an initial vector of rates and $\bar{\alpha}$, $\bar{\beta}$ vectors of parameters. According to original AIMD scheme user rates are increasing between overloads with rate $\bar{\alpha}$. When overload occurs rate drops to $\bar{\beta}x$. Now we will put into formal definitions.

Denote W as a set

$$\left\{ w \in R_+^N \mid P^{-1} \sum_{k=0}^{M-1} \left(R^T\right)^k A\, w \in U \right\}.$$

Let us define function $\alpha(x, X) = \max\{\alpha \geq 0 : \alpha x \in X\}$ and set $V = \{v \in U : \alpha(v, U)v = v\}$. Set V is a subset of boundary of U, which belongs to $\operatorname{int} R_+^N$ (V is "active" in the sense that in these and only in these points overload are happened).

Define t_i, $i \geq 1$ as a first moment of time $t_i > t_{i-1}$, such that $x(t_i) \in V$. We will assume that the RTT (round trip times) are the same for all connections and losses are synchronized: when the combined rates attain capacity, all connections suffer from a loss. Consider following equation

$$\dot{\bar{x}}(t) = \bar{\alpha} - \sum_{i=1}^{N_t} (I - B)x(t_i)\delta(t - t_i), \qquad (1)$$

where δ is delta-function, $B = diag\{\beta_1, ..., \beta_N\}$, $N_t = \max\{n : t_n \leq t\}$. Equation (1) is well-defined Caratheodory equation with discontinuous right-hand side, differential equations with impulses have been examined in many papers, which cannot all be referenced here. It is known that there is an almost continuous solution (continuous in all points except a set of measure zero)

$$x(t) = \alpha t - \sum_{i=1}^{N_t} (I - B)x(t_i)\eta(t - t_i), \qquad (2)$$

where η is the Heaviside step function. Explicit formula (2) is not very practical but gives us important information about solution existence and its continuity in almost all points.

Condition 3.1. For any $x \in W$, such that $P^{-1} \sum_{k=0}^{M-1} (R^T)^k Ax \in V$ it is true that $Bx \in W$.

Let us explain Condition 2.1 informally. W is the vector set of possible users rates. W is convex compact set and $x(t) \in W$ for $t \geq t_0$ $t \geq t_0$. As mentioned $x(t)$ is an almost continuous function, and drops only happened when $x(t) \in V$. After drop event users rates equal to $Bx(t)$. The condition 2.1 means that after applying decreasing operator B user rate still will be in the admissible set W.

Main result. Now we can formulate the main result of this section – existence and uniqueness of the limit solution.

Proposition 3.2. Let us consider admissible pair $\bar{\alpha}$, $\bar{\beta}$. If Condition 3.1 holds then for any $\bar{x}_0 \in W$ solution of (1) exists and is converging to unique periodical solution $\hat{x}(t)$.

Using this property, we can calculate $\bar{x}^*$ directly $c\bar{x}^* = (I - B)^{-1}\bar{\alpha}T = T\bar{\gamma}$.

Now we consider a competition between users which use AIMD version of TCP with different parameters. Their connections are sharing a common network. We will assume that users send their packets exactly the same way, so we can reduce network topology to single link type with capacity c, calculated from the solution (2).

4. NETWORK GAME MODEL

In order to formulate game for our dynamic system in strategic form we must specify the players, their strategies, and their potential payoffs. We assume that there are N AIMD strategies s_i with control parameters (α_i, β_i), $i = 1, ..., N$. Denote S as a set of all possible strategies. We consider payoff of the form

$$J_i(s) = Thp_i(s) - \lambda R(s),$$

where $\bar{s} = (s_1, ..., s_N)$ – vector of strategies; $Thp_i(s) = 0.5(1 + \beta_i)x_i$ – average throughput of i's player; $\lambda \geq 0$ – tradeoff parameter (sensitivity to losses); $R(s) = \dfrac{1}{T(s)}$ – loss rate.

Example. Let us calculate payoffs for two strategies:

$$J_1(s_i, s_i) = J_2(s_i, s_i) = \frac{(1 + \beta_i)}{4} c - \lambda \frac{2\gamma_i}{c},$$

$$J_1(s_1,s_2)=\frac{(1+\beta_1)\gamma_1 c}{2(\gamma_1+\gamma_2)}-\frac{\lambda}{c}(\gamma_1+\gamma_2),$$

$$J_1(s_2,s_1)=\frac{(1+\beta_2)\gamma_2 c}{2(\gamma_1+\gamma_2)}-\frac{\lambda}{c}(\gamma_1+\gamma_2),$$

$$J_2(s_1,s_2)=J_1(s_2,s_1),\ J_2(s_2,s_1)=J_1(s_1,s_2).$$

Equilibrium in N protocols game. Consider game with N AIMD strategies. We assume that all s_i are ordered lexicographically, $s_1 \geq s_2 \geq ... \geq s_N$, where $s_i \geq s_j$ means that $\alpha_i \geq \alpha_j$ and $\beta_i \geq \beta_j$. In other words protocols are sorted by aggressiveness ordering.

Proposition 4.1. If λ is sufficiently small than the most aggressive protocol is dominant strategy.

Proof. Suppose $\alpha_1 \geq \alpha_i$, $\beta_1 \geq \beta_i$ for all $i=2,...,N$. Consider payoffs for the first player $J_1(s_1,s_{-1})$ and $J_1(s_j,s_{-1})$. Let us find period for both strategy profiles:

$$T(s_1,s_{-1})=\frac{c}{\gamma_1+A},$$

where $A=\sum_k \gamma_k$ is sum, defined by strategy set s_{-1}, $T(s_j,s_{-1})=\frac{c}{\gamma_j+A}$. Note, that $T(s_1,s_{-1})<T(s_j,s_{-1})$.

Calculate throughputs:

$$Thp_1(s_1,s_{-1})=\frac{(1+\beta_1)x_1^*}{2}=\frac{\gamma_1(1+\beta_1)c}{2(\gamma_1+A)}=$$

$$=\frac{(1+\beta_1)c}{2(1+A/\gamma_1)},$$

$$Thp_1(s_j,s_{-1})=\frac{(1+\beta_j)x_j^*}{2}=\frac{(1+\beta_j)c}{2(1+A/\gamma_j)}.$$

Calculate payoffs:

$$J_1(s_1,s_{-1})=Thp_1(s_1,s_{-1})-\frac{\lambda}{c}(\gamma_1+A),$$

$$J_1(s_j,s_{-1})=Thp_1(s_j,s_{-1})-\frac{\lambda}{c}(\gamma_j+A).$$

Condition of dominating of first strategy is

$$Thp_1(s_1,s_{-1})-\frac{\lambda}{c}(\gamma_1+A)>$$

$$>Thp_1(s_j,s_{-1})-\frac{\lambda}{c}(\gamma_j+A),$$

$$\frac{(1+\beta_1)c}{2(1+A/\gamma_1)}-\frac{(1+\beta_j)c}{2(1+A/\gamma_1)}>\frac{\lambda}{c}\gamma_1,$$

$$\lambda<\frac{c^2}{\gamma_1}\left[\frac{(1+\beta_1)c}{2(1+A/\gamma_1)}-\frac{(1+\beta_j)c}{2(1+A/\gamma_j)}\right].$$

And since expression in right side is positive we obtain the result.

For more subtle results about equilibrium's characteristics see [16].

Nash Mixed and Pure in Two Protocols Game. Here we investigate the game for two protocols and find conditions for Nash equilibrium. From definition it is clear that $J_i(s_k,s_k)=J_j(s_k,s_k)$ – we will write just

$$J(s_k,s_k),\ J_i(s_k,s_p)=J_j(s_p,s_k),$$

$$j\in\{1,\ 2\}\setminus i.$$

Using standard techniques for calculating Nash we obtain:

$$J_1(s_1)=pJ(s_1,s_1)+(1-p)J(s_1,s_2),$$

$$J_1(s_2)=pJ(s_2,s_1)+(1-p)J(s_2,s_2)$$

assuming the probability of player 2 using the first strategy is p. In Nash equilibrium the payoff can't be further increased, so these two values should be indistinguishable, which leads to the following equation

$$pJ(s_1,s_1)+(1-p)J(s_1,s_2)=$$

$$=pJ(s_2,s_1)+(1-p)J(s_2,s_2).$$

Or, after solving it for p:

$$p=$$

$$=\frac{J(s_1,s_2)-J(s_2,s_2)}{(J(s_1,s_2)-J(s_2,s_2))+(J(s_2,s_1)-J(s_1,s_1))}.$$

Taking into account that p is a probability, we impose a natural restrictions on it: $0\leq p\leq 1$, where cases with $p=1$ or $p=0$ result in game having a pure-strategy equilibrium (with dominant strategy s_1 and

s_2, respectively), and $0 < p < 1$ corresponds to the case of mixed-strategy Nash equilibrium.

Should we investigate the conditions for the former, we get

$$J(s_1, s_2) - J(s_2, s_2) =$$

$$= \frac{-\lambda(\alpha_2(1-\beta_1) + \alpha_1(1-\beta_2))}{c(1-\beta_2)(1-\beta_1)} +$$

$$+ \frac{c\alpha_1(1+\beta_1)(1-\beta_2)}{2(\alpha_2(1-\beta_1) + \alpha_1(1-\beta_2))} + \frac{2\lambda\alpha_2}{c(1-\beta_2)} +$$

$$+ \frac{1}{4}C(1+\beta_2).$$

$$J(s_2, s_1) - J(s_1, s_1) =$$

$$= \frac{-\lambda(\alpha_2(1-\beta_1) + \alpha_1(1-\beta_2))}{c(1-\beta_2)(1-\beta_1)} +$$

$$+ \frac{c\alpha_2(1+\beta_2)(1-\beta_1)}{2(\alpha_2(1-\beta_1) + \alpha_1(1-\beta_2))} +$$

$$+ \frac{2\lambda\alpha_1}{c(1-\beta_1)} + \frac{1}{4}C(1+\beta_1).$$

Consequently,

$$p = 1 - \frac{2\alpha_2(1-\beta_1)}{\alpha_2(1-\beta_1) - \alpha_1(1-\beta_2)} -$$

$$- \frac{4\lambda(\alpha_1+\alpha_2) + c^2(\beta_1^2 - 1)}{c^2(1-\beta_1)(\beta_1 - \beta_2)} +$$

$$+ \frac{4\lambda\alpha_2}{c^2(1-\beta_1)(1-\beta_2)}.$$

Considering the case where game has pure-strategy equilibrium, we get two possible conditions: $p = 1$ or $p = 0$.

Solving the equations, we find the values of λ that correspond to the case of dominant strategy:

$$\lambda =$$

$$= \frac{c\beta_1\beta_2(\alpha_2\beta_1(1-\beta_1+2\beta_2) - \alpha_1(1-\beta_2)(1+\beta_1))}{4(\alpha_2(1-\beta_1) - \alpha_1(1-\beta_2))(\alpha_2(1-\beta_1) + \alpha_1(1-\beta_2))},$$

$$\lambda =$$

$$= \frac{c\beta_1\beta_2(\alpha_2\beta_1(1+\beta_2) + \alpha_1(1-\beta_2)(1+2\beta_1-\beta_2))}{4(\alpha_2(1-\beta_1) - \alpha_1(1-\beta_2))(\alpha_2(1-\beta_1) + \alpha_1(1-\beta_2))}. \quad (3)$$

Now, for the game to have mixed-strategy equilibrium the following system of inequalities must hold:

$$p < 1 \text{ and } p > 0.$$

After solving this system for λ we get

$$\frac{C^2\bar{\beta}_1\bar{\beta}_2(\alpha_1\bar{\beta}_2(1+\beta_1) + \alpha_2\bar{\beta}_1(\beta_1 - 2\beta_2 - 1))}{4(\alpha_1^2\bar{\beta}_2^2 - \alpha_2^2\bar{\beta}_1^2)} <$$

$$< \lambda <$$

$$< \frac{C^2\bar{\beta}_1\bar{\beta}_2(\alpha_1\bar{\beta}_2(1+2\beta_1-\beta_2) - \alpha_2\bar{\beta}_1(1+\beta_2))}{4(\alpha_1^2\bar{\beta}_2^2 - \alpha_2^2\bar{\beta}_1^2)}. \quad (4)$$

Proposition 4.2. If λ satisfies (4) then there is Nash equilibrium in mixed strategies. If λ satisfies (3) then there is Nash equilibrium in pure strategies.

Extension for Protocols Parameters. The game settings in previous sections were limited by aggressive ordering of protocols. In this section we weaken this condition to cover protocol parameters relation that falls beyond the "aggressive-peaceful" scheme, namely situation when $\alpha_1 \le \alpha_2$ and $\beta_1 \ge \beta_2$.

Applying the same considerations as above, we get the same results for pure-strategy Nash equilibria, but for mixed-strategy equilibrium an additional constraint emerges.

Since we're looking for cases with $0 < p < 1$, we get the following conditions for $p > 0$:

$$\begin{cases} J(s_1, s_2) - J(s_2, s_2) > 0, \\ (J(s_1, s_2) - J(s_2, s_2)) + (J(s_2, s_1) - J(s_1, s_1)) > 0. \end{cases}$$

Similarly, $p < 1$ holds when

$$(J(s_1, s_2) - J(s_2, s_2)) + (J(s_2, s_1) - J(s_1, s_1)) >$$

$$> J(s_1, s_2) - J(s_2, s_2).$$

(It can be shown that other case with $J(s_1, s_2) - J(s_2, s_2) < 0$ results $\lambda < 0$, which has no physical sense, recalling that λ is an error weight).

So, in the end we have the following system of inequalities:

$$\begin{cases} J(s_1, s_2) - J(s_2, s_2) > 0, \\ J(s_2, s_1) - J(s_1, s_1) > 0. \end{cases}$$

Or, after replacement of J and transformations

$$\begin{cases} \lambda < \\ < \dfrac{C^2\bar{\beta}_1\bar{\beta}_2(\alpha_1\bar{\beta}_2(1+2\beta_1-\beta_2)-\alpha_2\bar{\beta}_1(1+\beta_2))}{4(\alpha_1^2\bar{\beta}_2^2-\alpha_2^2\bar{\beta}_1^2)}, \\ \lambda > \\ > \dfrac{C^2\bar{\beta}_1\bar{\beta}_2(\alpha_1\bar{\beta}_2(1+\beta_1)+\alpha_2\bar{\beta}_1(\beta_1-2\beta_2-1))}{4(\alpha_1^2\bar{\beta}_2^2-\alpha_2^2\bar{\beta}_1^2)}. \end{cases}$$

Replacing

$$\mu_1 = \\ = \frac{C^2\bar{\beta}_1\bar{\beta}_2(\alpha_1\bar{\beta}_2(1+2\beta_1-\beta_2)-\alpha_2\bar{\beta}_1(1+\beta_2))}{4(\alpha_1^2\bar{\beta}_2^2-\alpha_2^2\bar{\beta}_1^2)},$$

$$\mu_2 = \\ = \frac{C^2\bar{\beta}_1\bar{\beta}_2(\alpha_1\bar{\beta}_2(1+\beta_1)+\alpha_2\bar{\beta}_1(\beta_1-2\beta_2-1))}{4(\alpha_1^2\bar{\beta}_2^2-\alpha_2^2\bar{\beta}_1^2)}.$$

We get two possible solutions to the system above:

$$\begin{cases} \alpha_2(1-\beta_1)-\alpha_1(1-\beta_2)>0, \\ \mu_2<\lambda<\mu_1, \end{cases}$$

or

$$\begin{cases} \alpha_2(1-\beta_1)-\alpha_1(1-\beta_2)<0, \\ \mu_1<\lambda<\mu_2. \end{cases}$$

Since $\alpha_1 \le \alpha_2$ and $\beta_1 \ge \beta_2$ then $\mu_2 > \mu_1$, the actual solution is

$$\begin{cases} \alpha_2(1-\beta_1)-\alpha_1(1-\beta_2)<0, \\ \mu_1<\lambda<\mu_2. \end{cases}$$

Proposition 4.3. If $\alpha_1 \le \alpha_2$ and $\beta_1 \ge \beta_2$ and

$$\alpha_1 < \alpha_2 < \frac{\alpha_1(1-\beta_2)}{1-\beta_1},\ \mu_1<\lambda<\mu_2,$$

then there is evolutionary stable equilibrium in mixed strategies.

Formulated conditions are consistent with the previous result with regards to protocol parameters specifics.

5. NETWORK ATTACKS SIMULATIONS

We study in this section numerically dynamic system (1) and equilibriums of defined game with replicator dynamics. The practical value of these results could be divided on two parts. Firstly, this is analytical tool for predicting shares of network resources for given set of AIMD protocols. Secondly, we can model users' behavior (taking into account usual game theory assumptions about rationality, common knowledge etc.) using replicator dynamic equation. This equation is rather quality solution tool that show a dynamic and shares of network resources for each users group.

Solution for dynamic system. Numerical simulations were made using Wolfram Mathematica environment. On the picture below we show convergence of AIMD scheme for 2 and 3 dimensions.

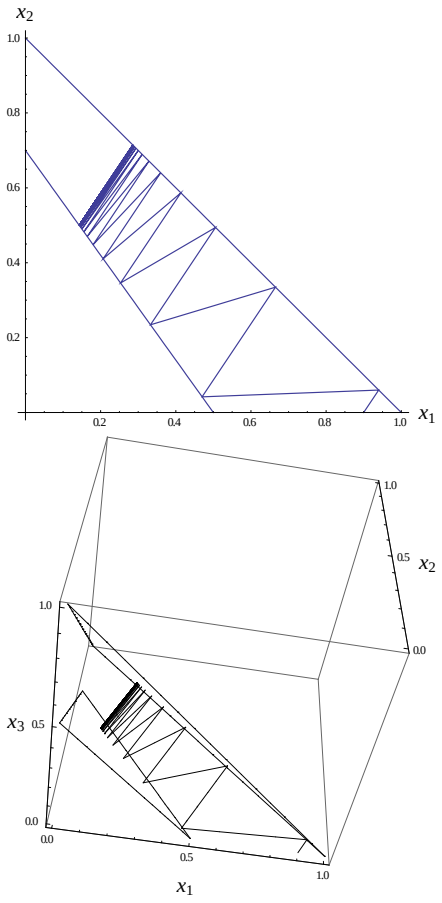

Fig. 1. Simulations results for 2-d and 3-d systems

Attack modeling. To test theoretic model described above, a simulation model was developed under NS-3 Network Simulator package. The specifics of the model, as well as simulation results, are described below.

For the purposes of testing, a simple topology consisting of four nodes was built (fig. 2) – the nodes represent sender, router, receiver and attacker.

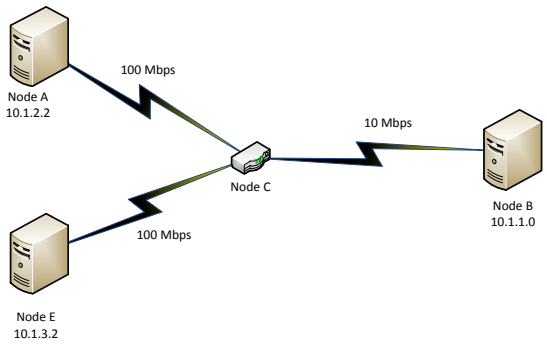

Fig. 2. Testbed topology

The basic workflow is as follows: the sender (Node A) uploads a large file to some storage, controlled by file server (Node B), that resides in a different subnet. Thus, a large volume of traffic is generated and routed between the adjacent subnets by router (Node C). Then an attacker from the sender's subnet steps in (Node E). His goal is to disrupt a client-server operations by performing a denial-of-service attack, but, as router comes equipped with basic flood-detection capabilities, attacker won't be able to perform a full-scale UDP or ICMP DoS attack, and has to resort to different means.

He chooses a particularly stealthy approach known as low-rate TCP denial-of-service attack, which exploits the weakness of TCP retransmission mechanism to cause a significant service degradation or even a full outage. The idea is the following: as TCP employs an exponential backoff technique for retransmission of packets presumed to be lost, it is enough for an attacker to cause a short-term service outage with traffic spike and then maintain this state by sending the same spikes on the exact moments the client attempts to retransmit a packet. In more detail, if the client detects a packet loss at time t, it is enough for an attacker to perform short-term DoS in moments t+RTO, (t+RTO)+2*RTO, ((t+RTO)+2*RTO)+4*RTO and so on, where RTO is a value of TCP retransmission timeout. Moreover, lots of TCP implementations have the default RTO value of 1 second, which makes the described attack feasible even for networks with large amount of clients, as they are likely to have close RTO values.

The setup of NS-3 model is as follows: at time 0 the client at Node A establishes the connection with the server at Node B and starts sending data using TCP. Then, at 20 seconds from the beginning of a simulation, an attacker at Node E kicks in, periodically sending 30 traffic spikes of predefined length, separated by periods of silence. We denote the period between traffic spikes as T, and the length of the spike itself as τ. Obviously, different values and ratio between T and τ would yield different results in terms of attack success, with most prominent being achieved as T nears RTO. Depicted on fig. 3 are sample graphs of client congestion windows dynamics under different values of T with τ being the same value of 0.1.

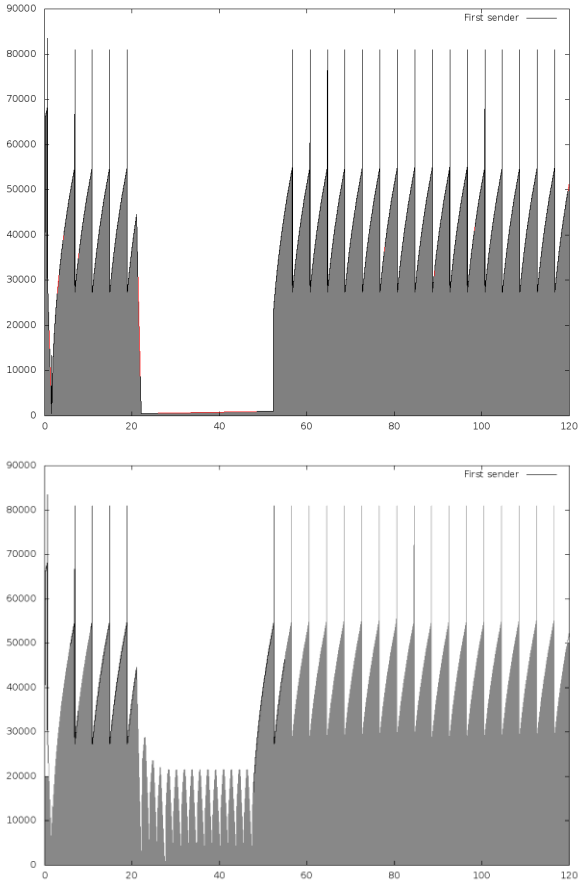

Fig. 3. Congestion window dynamics at T = 0.9 and T = 1

As shown on graphs, the least goodput (and the most successful denial-of-service) is achieved with T being equal to default RTO value, which is consistent with theoretically predicted results.

Further, we investigate a client throughput change during attacks with different T values. The resulting graph, presented at fig. 4, shows the performance degradation of client on 10 Mbps link under different attacks with the same total amount of traffic sent.

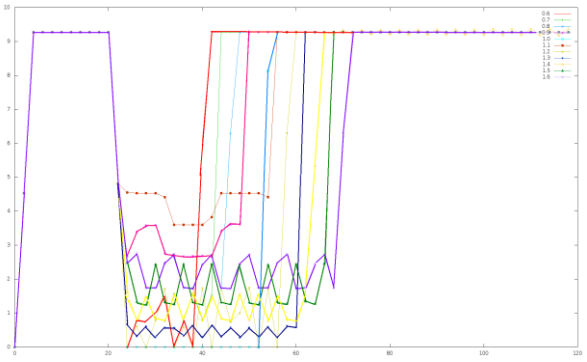

Fig. 4. Client throughput depending on attack parameters

CONCLUSIONS

In this paper, we have presented an overview of approaches to deal with security problems using a game-theoretic framework. The general objective was to identify and address the security and efficiency problems, where game theory can be applied to model and evaluate security problems and consequently used to design efficient network control solution. The application of game theory is an emerging field in network security, with only a few papers published so far.

Game theory provides a (parametric) model, which is refined and calculated using statistical data from real network. This model is actually a set of three layers of models, discovering system's dynamic and users' behavior from different angles. The first layer is a game between user and network. Solution is the protocol – strategy of user data flow. The second later is a game among users. Each user tries to receive maximal resource. This is non-cooperative game. Rational behavior leads to the Nash equilibrium (through its computing can be very complex). Network tries to balance users and achieve effective NE. System allocation algorithms efficiency is measured by PoA or PoS metrics. The last layer is a game with attacker. Attacker wants to disrupt network and prevent users from receiving any resources. This is game with pure conflict (zero-sum game). Analysis of resulting NE gives us metric to measure strength of attack and detect weaknesses of system.

1. *Sandberg, Henrik, Saurabh Amin, and K. Johansson.* Cyberphysical Security in Networked Control Systems: An Introduction to the Issue // Control Systems, IEEE 35.1. – 2015. – P. 20–23.
2. *Kelly, Frank P., Aman K. Maulloo, and David KH Tan.* Rate control for communication networks: shadow prices, proportional fairness and stability // Journal of the Operational Research society. – 1998. – P. 237–252.
3. *Shakkottai, Srinivas, Srinivas Govindaraju Shakkottai, and Rayadurgam Srikant.* Network optimization and control. Now Publishers Inc, 2008.
4. *Manshaei, Mohammad Hossein, et al.* Game theory meets network security and privacy // ACM Computing Surveys (CSUR). – 2013. – 45.3, 25.
5. *Liang Xiannuan, and Yang Xiao.* Game theory for network security // Communications Surveys & Tutorials, IEEE. – 2013. –15.1. P. 472–486.
6. *La Richard J.* Role of network topology in cybersecurity // Decision and Control (CDC). – 2014 IEEE 53rd Annual Conference on. IEEE, 2014.
7. *Kelly Frank P., Aman K. Maulloo, and David KH Tan.* Rate control for communication networks: shadow prices, proportional fairness and stability // Journal of the Operational Research society. – 1998. – P. 237–252.
8. *Mo J., Walrand J.* Fair end-to-end window-based congestion control // IEEE/ACM Transactions on Networking. – 2000. – 8. – P. 556–567.
9. *Paganini F., Doyle J.C., Low S.H.* Scalable laws for stable network congestion control // Proc. of IEEE Conference on Decision and Control. – 2001. – 1. – P. 185–190.

10. *Low S.H., Srikant R.* A Mathematical Framework for Designing a Low-Loss, Low-Delay Internet // Network and Spatial Economics. – 2004. – 4 (1). – P. 75–102.
11. *Altman E. et al.* The evolution of transport protocols: An evolutionary game perspective // Computer Networks. – 2009. – T. 53, N 10. – C. 1751–1759.
12. *Smith John Maynard.* Evolution and the Theory of Games. Cambridge university press, 1982.
13. *Altman E., Bonneau N., Debbah M., Caire G.* An evolutionary game perspective to ALOHA with power control, in: Proceedings of the 19th International Teletraffic Congress, Beijing, 29 August–2 September, 2005.
14. *Papadimitriou Christos.* Algorithms, games, and the internet // Proceedings of the thirty-third annual ACM symposium on Theory of computing. ACM, 2001.
15. *Han Zhu, et al.* Game theory in wireless and communication networks. Cambridge University Press, 2012.
16. *Ignatenko Oleksii, and Oleksandr Synetskyi.* Evolutionary Game of N Competing AIMD Connections // Information and Communication Technologies in Education, Research, and Industrial Applications. Springer International Publishing. – 2014. – P. 325–342.
17. *Andon F.I., and Ignatenko O.P.* Modeling conflict processes on the internet // Cybernetics and Systems Analysis. – 2013, 49.4. – P. 616–623.

Date received 11.12.2015

Information about author:

Ignatenko Oleksii,
Cand. Sc. (Phys. & Math.),
senior research fellow,
Publications: 34,
7 – in indexed scientific libraries.
http://orcid.org/0000-0001-8692-2062

Affiliation:

Institute of Software Systems
NAS Ukraine,
03187, Kyiv - 187,
Academician Glushkov ave, 40.
Tel.: (044) 526 6025.
E-mail: o.ignatenko@gmail.com

УДК 004.942+519.837.2

А.Л. Яловець

МУЛЬТИАГЕНТНЕ МОДЕЛЮВАННЯ ПОСЛІДОВНИХ БАГАТОЕЛЕМЕНТНИХ ЯПОНСЬКИХ АУКЦІОНІВ

Досліджуються особливості моделювання аукціонів з точки зору імітаційного (мультиагентного) моделювання. Наводиться характеристика аукціонів як об’єкта моделювання. Виконується постановка задачі та пропонується метод побудови механізму проведення послідовних багатоелементних японських аукціонів, який забезпечує використання агентами домінуючих стратегій та дозволяє побудувати оптимальний аукціон. Експериментально підтверджується ефективність запропонованого метода.
Ключові слова: моделювання, японський аукціон, дизайн механізму, агент, домінуючі стратегії.

Вступ

Однією з актуальних проблем, що активно досліджується у світі, є моделювання аукціонів. Актуальність цього напряму досліджень є наслідком сучасних загальносвітових тенденцій. По-перше, з розвитком Інтернет-технологій з’явилися та поширилися різнорідні on-line аукціони (е-аукціони – наприклад, eBay.com, tcwc.com, iauction.com, klik-klok.com). По-друге, теорія аукціонів, як розділ теорії ігор, на сьогоднішній день дала поштовх багатьом напрямам фундаментальних досліджень: від досліджень з розробки методів цінового формування в різних галузях господарства (з урахуванням конкуренції та монополізації ринків) до досліджень, орієнтованих на нецінові проблеми розподілу, які в тому числі включають і так звані «війни на виснаження», дослідження механізмів моделювання яких на сьогодні є вкрай важливим і для України.

Зріст інтересу до цих проблем також підтверджується множиною виданих за останні роки книг (наприклад, [1–7]), та присудженням Нобелевської премії з економіки у 1996 та 2007 рр. за результати саме з галузі теорії аукціонів. Зазначимо, що в економіці теорія ігор вивчає функціонування економічних систем за умов «недосконалого ринку» й ігрові моделі аукціонів, зокрема, є прикладами успішного використання ігрового підходу в економіці.

На сьогоднішній день у світі існує два домінуючих підходи щодо використання мультиагентного підходу для моделювання аукціонів: моделювання е-аукціонів та моделювання переговорів у мультиагентних системах на основі аукціонів. Якщо при моделюванні е-аукціонів [8], як правило, моделюється лише механізм аукціону зі сторони аукціоніста (тобто дії покупців не моделюються, а реєструються як зовнішні дії), то при моделюванні переговорів на основі аукціонів [9, 10], в рамках яких агенти конкурують за ресурси або послуги, відбувається моделювання всіх учасників аукціону. Водночас, незважаючи на існуючі досягнення, за межами досліджень, на наш погляд, залишається розгляд мультиагентного моделювання аукціонів з точки зору імітаційного моделювання динамічних систем. Пропонуванню одного з можливих підходів до такого моделювання аукціонів і присвячено дану статтю.

1. Загальна характеристика аукціонів як об’єкта моделювання

1.1. Загальні відомості про аукціони та їх сутність. *Аукціон* (від латинського *auctio* – продаж з публічного торгу) – це спосіб продажу деяких товарів за цінами, що встановлюються покупцями за результатами торгів. Аукціон – це процедура, за результатами якої приймається рішення про те, кому передати лот і скільки переможець має заплатити. Учасники повідомляють про свою готовність платити і указане рішення приймається виключно на основі отриманих сигналів.

Всі товари попередньо ділять на лоти (стандартні за кількістю та іншим озна-

кам партії товарів). Лотом може бути і одиничний товар. Кожний лот має свій номер (ідентифікатор), під яким він представлений на торгах.

Продавець в аукціонах може встановити *резервну* і *початкову* ціни. Резервна ціна характеризує цінність товару для продавця. Початкова ціна – це ціна, з якої починається торг. Якщо резервна ціна оголошується перед початком аукціону, вона стає початковою ціною. Якщо торги проводяться з початкової ціни і заключна ціна (ціна продажу) становить значення, менше за резервну ціну, лот знімається з продажу. Організатор аукціону встановлює також крок торгів. Величину кроку торгів доцільно ув'язувати з початковою ціною, оскільки при початковій ціні, близької до вартості об'єкта, та великим кроком торгів аукціон може виграти учасник з не самою високою оцінкою об'єкта.

Розрізняють два основних критерії якості аукціонної процедури – ефективність та оптимальність. Аукціон називають *ефективним,* якщо об'єкт достається учаснику з найвищою оцінкою. Аукціон є *оптимальним*, якщо він максимізує очікуваний прибуток продавця.

1.2. Типи аукціонів. Обґрунтування вибору досліджуваного типу аукціону. На сьогоднішній день відомі чотири основних типи аукціонів:

- *англійський аукціон* (відкритий аукціон з підвищенням ціни). Цей тип аукціону здійснюється в реальному масштабі часу за безпосередньою участю гравців (усний аукціон) або інтерактивно у віддаленому доступі (за електронними торгами). При цьому аукціоніст виходить з низької початкової ціни за лот (деякий товар) та поступово підвищує ціну, в процесі чого претенденти, не спроможні оплатити товар за поточною ціною торгів, вибивають, поки єдиний претендент не залишиться, який і виграє (придбає аукціонний товар) за остаточною ціною;

- *голландський аукціон* (відкритий аукціон зі зниженням ціни). Цей тип аукціону також здійснюється в реальному масштабі часу, але, на відміну від англійського аукціону, аукціоніст починає з високої початкової ціни за лот та поступово знижує її до тих пір, доки деякий претендент не буде згоден оплатити ціну лота за поточною ціною торгів. Ця ціна і є остаточною ціною торгів. Назва даного типу аукціону походить від аукціонів, що проводяться в Голландії при оптовій торгівлі квітами;

- *закритий аукціон першої ціни*. При цьому типі аукціону кожний з претендентів одночасно подає аукціоністу «запечатану пропозицію» з вказівкою пропонованої ціни за лот. В свою чергу, аукціоніст, одночасно відкриваючи наявні пропозиції, визначає покупця, яким виявляється той, хто запропонував найвищу ціну. Ця ціна і є остаточною ціною торгів;

- *закритий аукціон другої ціни* (аукціон Вікрі). Цей тип аукціону подібний до аукціону першої ціни і переможцем торгів також вважається той покупець, який запропонував найвищу ціну за лот, але переможець платить ціну, яка є другою найвищою ціною зі всіх пропозицій, що надійшли. Даний тип аукціону названий у честь його автора, Нобелевського лауреата Вільяма Вікрі.

Традиційно [1] при теоретичному дослідженні аукціонів розглядають різновид англійського аукціону, який називається *японським аукціоном*, та полягає у тому, що ціна за лот підвищується безперервно з постійним ціновим кроком торгів, і претенденти, які не спроможні оплатити лот за поточною ціною, вибувають остаточно. Кожний з претендентів приймає участь у торгах з початку торгів, жодний інший потенційний учасник не може увійти до торгів у процесі їх проведення; жодний претендент не може різко підвищити ціну шляхом пропонування «підвищеної ставки». Далі в наших дослідженнях ми будемо розглядати саме *японський аукціон*.

Для кожного з названих типів аукціону можуть існувати різні аспекти розгляду, пов'язані з особливостями проведення аукціону. За цією ознакою аукціони можна розподілити на одноелементні (single-unit) та багатоелементні (multi-unit), де під одноелементними розуміються аукціони, на яких на протязі досліджуваного

проміжку часу торгується один неподільний лот (single indivisible unit); відповідно під багатоелементними – аукціони, на яких на протязі досліджуваного проміжку часу торгується багато неподільних лотів. Зазначимо [1], що на сьогоднішній день переважна більшість досліджень присвячена одноелементним та багатоелементним аукціонам для випадку, коли кожний покупець претендує лише на один лот з багатьох пропонованих.

Для багатоелементного аукціону має значення організація процесу торгівлі. Відповідно до цього існують одночасні (simultaneous) та послідовні (sequential) багатоелементні аукціони, розбіжність між якими випливає з їх назв: якщо одночасні аукціони передбачають одночасний незалежний продаж *m* лотів на різних аукціонах, то послідовні аукціони передбачають продаж *m* лотів на аукціонах, які проводяться послідовно в часі (тобто послідовно проводиться *m* аукціонів).

1.3. Визначення основних властивостей досліджуваних аукціонів як різновиду ігор. Визначимо властивості ігор, які ми будемо розглядати, виходячи з класифікації ігор. Як відомо [11], ігри класифікуються наступним чином:

- за кількістю гравців: ігри 1; 2; *n* гравців;
- за кількістю стратегій: скінченні та нескінченні ігри. Якщо у гравців скінченна кількість стратегій, то така гра є скінченною, в іншому випадку – нескінченною;
- за характером взаємовідносин між гравцями: кооперативні та некооперативні ігри. В кооперативній грі гравці можуть заключати угоди з метою збільшення своїх виграшів, в некооперативній – ні;
- за кількістю ходів: одноходові та багатоходові. Багатоходові поділяються на:
  - позиційні, коли декілька гравців послідовно виконують ходи і виграш гравців залежить від стратегії вибору ходів (наприклад, шашки, шахи);
  - стохастичні, коли в грі є ходи, які виконуються випадковим чином та існує ймовірність повернення на попередні позиції;
  - диференційні, коли ходи виконуються безперервно і поведінка гравців описується диференційними рівняннями.
- за інформованістю гравців: ігри з досконалою та недосконалою інформацією. У грі з досконалою інформацією на кожному кроці гравцям відомо, які ходи були зроблені раніше (наприклад, шашки, шахи). У грі з недосконалою інформацією гравці можуть не знати, в якій позиції вони знаходяться (що відповідає, наприклад, стохастичним іграм). До ігор з недосконалою інформацією зводяться ігри з неповною інформацією (також відомі як байєсовські ігри). На відміну від ігор з недосконалою інформацією, де неповна інформованість гравців виникає у процесі гри, в іграх з неповною інформацією неповна інформованість деяких гравців виникає ще до початку гри, як наслідок асиметричної інформованості гравців (наприклад, на аукціоні покупець знає про якість товару менше, ніж продавець).

Отже, ігри, на основі яких ми будемо досліджувати проблеми моделювання аукціонів, мають наступні властивості: за кількістю гравців – *n* гравців; за кількістю стратегій – скінченні стратегії; за характером взаємовідносин між гравцями – некооперативні ігри; за кількістю ходів – багатоходові позиційні; за інформованістю гравців – ігри з досконалою та неповною інформацією (гравцям відомі попередні ходи, але невідомі майбутні ходи ані свої, ані чужі; жодний з гравців не має інформації про гроші, наявні в інших гравців, та про розподіл їх уподобань щодо товарів, які продаються на аукціоні).

При моделюванні аукціонів також вирішується задача *дизайну механізму* («mechanism design») аукціону [7, 11, 12]. Дизайн механізму – це розділ теорії ігор, в якому вивчаються конфліктні ситуації з вираженою стратегічною поведінкою всіх гравців, щодо яких необхідно визначити правила прийняття рішень та здійснення платежів таким чином, щоби рівноважним став такий наслідок, в якому досягає свого максимуму деяка функція соціального вибору. Традиційно в основі дизайну механізму лежить *принцип правдивості*

(truthfulness), який дозволяє обмежитись пошуком тільки такого механізму, який спонукає всіх гравців чесно повідомляти свої типи.

Сутність задачі дизайну механізму полягає у наступному: необхідно побудувати механізм, в якому той чи інший рівноважний стан системи буде оптимальним відносно тієї чи іншої мети. В загальному випадку у якості мети виступає *функція соціального вибору*, під якою розуміється функція $f: T_1 \times ... \times T_n \to O$, яка обирає той чи інший бажаний результат $f(t)$ при даних типах $t = (t_1,...,t_n)$ гравців. Тобто функція соціального вибору – це те, що необхідно отримати від механізму, який розробляється. Але при цьому кожний гравець буде намагатися використовувати *домінуючі* стратегії (тобто максимізувати власний прибуток), і ці зворотні задачі необхідно узгодити. Саме на рішення цієї задачі і націлене поняття дизайну механізму.

1.4. Деякі узагальнення. На основі викладеного можна зробити наступні висновки щодо об'єкта моделювання:

1. Як об'єкт моделювання доцільно вибрати *послідовні багатоелементні японські аукціони*, в яких кожний покупець може претендувати на k лотів, де $1 \le k \le m$; m – загальна кількість лотів (дорівнює кількості аукціонів, що відбудуться).

2. Обраний тип аукціону з точки зору теорії ігор належить до класу *некооперативних динамічних ігор з досконалою та неповною інформацією*.

3. Будемо вважати *принцип правдивості* недіючим, оскільки в реальній дійсності гравці не надають жодної інформації щодо свого типу. З цього випливає задача розробки *дизайну механізму* аукціону та *домінуючих стратегій* для гравців (агентів), узгоджуваних з механізмом аукціону.

4. При формалізації типу гравця будемо виходити з припущення, що тип відповідає відношенню гравця до ризику. Відомо [12], що гравці (агенти) можуть мати три стани, що описують їх відношення до ризику:

- *обережний* (risk-averse), коли агент готовий заплатити за лот суму, строго меншу за ту, яку він запланував заплатити;
- *нейтральний до ризику* (risk-neutral), коли агент готовий заплатити за лот саме заплановану суму;
- *ризикований* (risk-loving) або *не нейтральний до ризику*, коли агент готовий заплатити за лот суму, що перевищує заплановану суму.

Для цілей моделювання будемо вважати, що тип агента описується двома станами щодо його відношення до ризику: {*нейтральний до ризику*, *ризикований*}.

2. Постановка задачі моделювання японського аукціону

В пропонованій постановці будемо виходити з припущення, що розглядаються гравці (далі – агенти) одного типу (*нейтральні до ризику*). Будемо також вважати, що резервна та початкова ціни на будь-який лот збігаються.

Задамо множини, які описують основні вхідні параметри проведення послідовного багатоелементного японського аукціону (далі – аукціону):

$N = \{1,\ldots,n\}$ – кінцева множина агентів, що приймають участь в аукціоні;

$L = \{1,\ldots,l\}$ – кінцева множина лотів, які торгуються на аукціоні;

$K_i \subseteq L$ – кінцева множина лотів, у торгах щодо придбання яких приймає участь i-ий агент. Можливо, що $K_i \cap K_n \neq \varnothing$, де $i, n \in N$, $i \neq n$;

R_i – ресурс (в загальному випадку в грошовому еквіваленті), наявний у i-го агента для придбання лотів множини K_i;

$V_i = \{(k_i^1, v_i^1),\ldots,(k_i^j, v_i^j)\}$ – кінцева множина пар, у кожній з яких подано пріоритет (суб'єктивна зважена оцінка цінності лота) v_i^j i-го агента щодо придбання відповідного j-го лота $k_i^j \in K_i$, $0 < v_i^j \le 1$, $\sum_{j=1}^{k} v_i^j = 1$, $k = card(K_i)$, $i \in N$;

$U = \{(p_1, c_1),\ldots,(p_m, c_m)\}$ – кінцева

множина пар, в кожній з яких подано початкову ціну p_j та крок торгів c_j щодо продажу відповідного j-го лота множини L (де кожному елементу множини U може бути зіставлений чітко визначений елемент множини L), $card(U) = card(L)$.

Під максимальним значенням ресурсу, виділеного i-им агентом на придбання j-го лота, будемо розуміти параметр $R_i^j = R_i * v_i^j$.

Вважатимемо, що жодному агенту невідомі пріоритети та ресурси інших агентів та подальші дії як свої, так і інших агентів, але йому відомі власні пріоритети і ресурси та попередні дії як свої, так і інших агентів. Кожний агент також володіє інформацією про початкові (поточні) ціни на лоти та про учасників торгів і перелік лотів, що мають торгуватися на аукціоні.

Виходячи з названих умов необхідно визначити домінуючі стратегії агентів та винайти дизайн механізму аукціонних торгів. Як показано в п. 1.3, побудова механізму має узгоджуватись з використовуваними агентами домінуючими стратегіями, в рамках яких кожний з агентів буде намагатися максимізувати власний прибуток. До того ж механізм має дозволяти формувати оптимальний аукціон.

Для побудови механізму будемо також виходити з наступних передумов:

1. Кожний i-ий агент, зареєстрований як учасник торгів за j-ий лот, має прийняти в торгах фактичну участь та намагатися зробити свою останню пропозицію в момент часу t проведення аукціону, коли виконується одна з двох умов: $z_j(t) - c_j \le R_i^j < z_j(t_1)$ або $z_j(t_e) \le R_i^j$ (якщо $z_j(t_e) \ge R_{-i}^j$), де $z_j(t)$ – значення ціни за j-ий лот в момент часу t проведення аукціону; $z_j(t_1)$ – значення ціни за j-ий лот у момент часу t_1, в який була подана наступна пропозиція після пропозиції, що надійшла в момент часу t; $z_j(t_e)$ – значення ціни за j-ий лот у момент часу t_e завершення торгів (ціна, яку платить переможець); c_j – крок торгів за j-ий лот; R_{-i}^j – максимальне значення ресурсу, виділене на придбання j-го лоту будь-яким не i-им агентом, що приймає участь в аукціоні.

2. Агенти, які мають більші ресурси на лот, вступають у аукціон пізніше, ніж інші агенти.

Перша передумова зумовлює необхідність автоматичного формування домінуючих стратегій агентів у рамках використовуваного механізму проведення аукціону. Друга передумова узагальнює практичний досвід проведення аукціонів [6] та впливає на визначення порядку участі агентів в аукціонних торгах.

Зауважимо, що прибуток переможця (i-го агента) в придбанні j-го лота визначатиметься як $R_i^j - z_j(t_e)$, а прибуток продавця – як $z_j(t_e) - p_j$, де p_j – резервна (у нас, як показано вище, вона збігається з початковою) ціна за j-ий лот.

Як наслідок, при побудові механізму необхідно вирішити наступну задачу оптимізації (при цьому мають виконуватись вищезазначені передумови 1, 2):

$$\begin{aligned} R_i^j - z_j(t_e) &\to \max, \\ z_j(t_e) - p_j &\to \max. \end{aligned} \qquad (1)$$

Слід зауважити, що переможець зацікавлений у мінімізації $z_j(t_e)$, а продавець, навпаки – в максимізації $z_j(t_e)$. Тобто побудований механізм має забезпечувати врівноваження між цими суперечними намірами учасників аукціону.

3. Метод побудови механізму проведення аукціону

Виходячи з наведеної постановки задачі розроблено метод побудови механізму проведення аукціону, який ґрунтується на наступних поняттях: *мотивація* агента, *коефіцієнт впевненості* агента і *коефіцієнт пасивності участі* агента в аукціоні.

Будемо називати *мотивацією* i-го агента щодо придбання j-го лота пара-

метр M_i^j, що є часткою пріоритету агента щодо j-го лота за відношенням до інших пріоритетів агента:

$$M_i^j = \frac{v_i^j}{1 - v_i^j}.$$

Цей параметр може тлумачитися як суб'єктивний рівень зацікавленості *i*-го агента в придбанні j-го лота і є незмінним на всьому протязі торгів за j-ий лот. Якщо $v_i^j = 1$ (тобто коли $card(K_i) = 1$), M_i^j присвоюється значення 1.0.

Під *коефіцієнтом впевненості* *i*-го агента щодо придбання j-го лота в момент часу t будемо розуміти параметр $kv_i^j(t)$, який визначається наступним чином:

$$kv_i^j(t) = \frac{R_i^j - z_j(t)}{R_i^j},$$

де $z_j(t)$ – значення ціни за j-ий лот у момент часу t проведення аукціону. Коефіцієнт впевненості може тлумачитися як міра спокою *i*-го агента у момент часу t щодо можливості придбання j-го лота (зі зростом t параметр $kv_i^j(t)$ зменшується; $0 < kv_i^j(t) < 1$). У випадку, коли $kv_i^j(t) \le 0$, участь *i*-го агента в подальшому аукціоні щодо придбання j-го лота неможлива (у зв'язку з закінченням ресурсу *i*-го агента).

Під *коефіцієнтом пасивності участі* *i*-го агента у торгах за j-ий лот у момент часу t будемо розуміти параметр $kp_i^j(t)$, який визначається як $kp_i^j(t) = M_i^j * kv_i^j(t)$. Коефіцієнт пасивності участі *i*-го агента може тлумачитись як ступінь його активності щодо придбання j-го лота в момент часу t проведення аукціону; $0 < kp_i^j(t) < 1$. Чим менший цей коефіцієнт, тим вища активність *i*-го агента в аукціоні і тим сильніше його намагання зробити ставку в момент часу t проведення аукціону.

Метод побудови механізму проведення аукціону ґрунтується на визначенні коефіцієнтів пасивності участі для кожного агента, який не вибув з аукціону в кожний момент виконання ставок за лот та порівнянні значень двох найменших коефіцієнтів. Якщо агент, що має найменший коефіцієнт з порівнянних, останнім робив ставку, то обирається інший агент з наступним найменшим коефіцієнтом, який і робить ставку за лот. В іншому випадку для виконання ставки обирається агент з найменшим коефіцієнтом.

4. Імітаційне моделювання послідовного багатоелементного японського аукціону

Запропонований метод побудови механізму моделювання аукціону реалізовано в прототипі мультиагентної системи (МАС) «Аукціонъ». Для реалізації використано мову логічного програмування PDC Visual Prolog з міркувань необхідності вести і обробляти БД аукціону (Prolog є реляційною мовою) та можливості засобами мови описувати логіку поведінки агентів.

Засобами МАС «Аукціонъ» забезпечується як процес формування і перегляду даних про агентів та про пропоновані лоти, так і власно виконання моделювання аукціону (див. рисунок) на створених та збережених даних.

Розглянемо результати мультиагентного моделювання послідовного багатоелементного японського аукціону (див. таблицю), виконаного засобами МАС «Аукціонъ» на прикладі вхідних даних, наведених далі, та покажемо, що в рамках запропонованого механізму забезпечується підтримка домінуючих стратегій агентами та вирішується задача оптимізації (1).

Вхідні дані аукціону:

Дані про лоти:

Лот 1: початкова ціна – 100; крок торгів – 10; учасники: Агент 1, Агент 2, Агент 3, Агент 4, Агент 6.

Лот 2: початкова ціна – 200; крок торгів – 20; учасники: Агент 2, Агент 3, Агент 5.

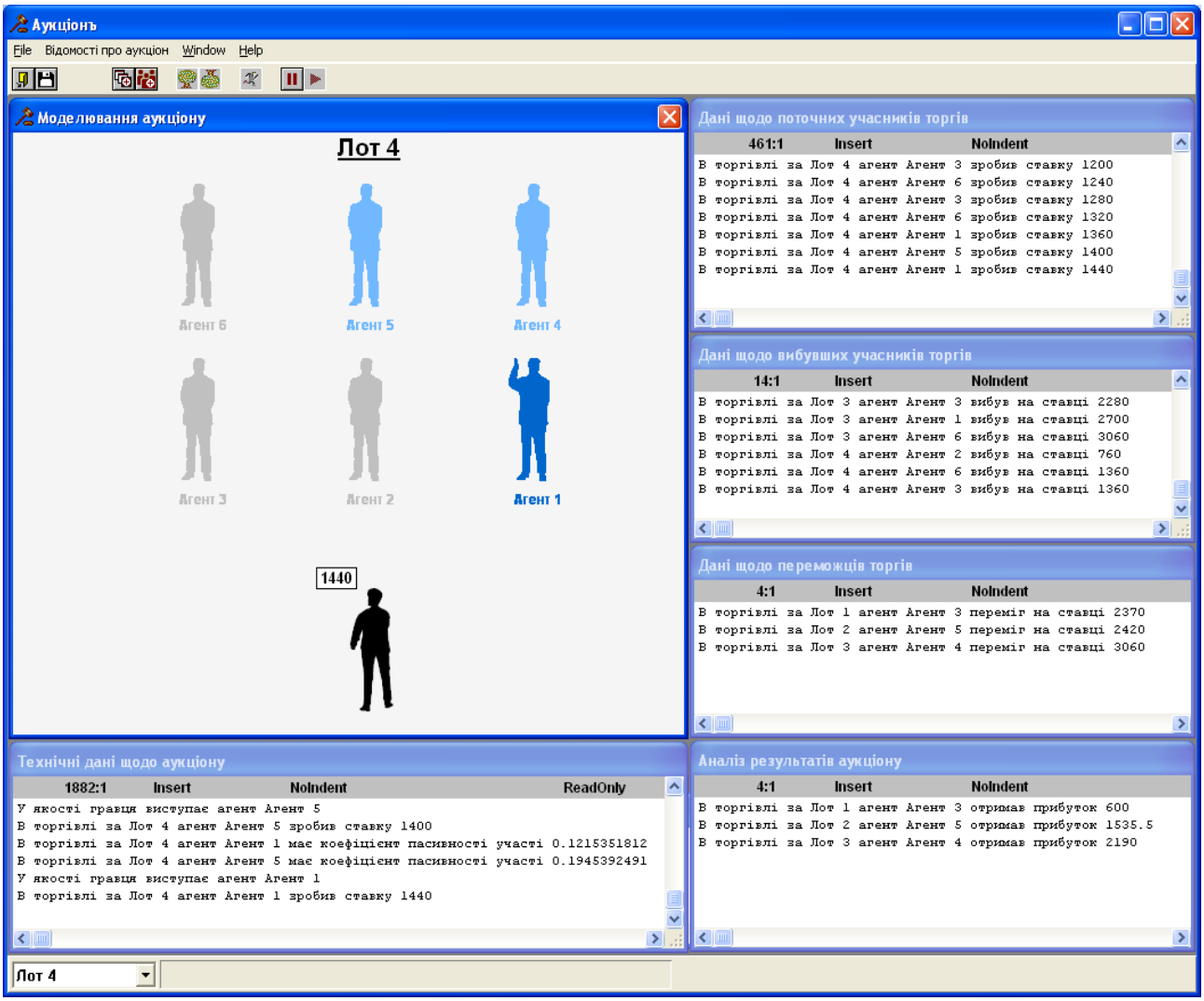

Рисунок. Інтерфейс процесу мультиагентного моделювання японського аукціону

Лот 3: початкова ціна – 300; крок торгів – 30; учасники: Агент 1, Агент 3, Агент 4, Агент 5, Агент 6.

Лот 4: початкова ціна – 400; крок торгів – 40; учасники: Агент 1, Агент 2, Агент 3, Агент 4, Агент 5, Агент 6.

Дані про агентів:

Агент 1: Ресурс – 6700; розподіл пріоритетів: Лот 1 – 0.3, Лот 3 – 0.4, Лот 4 – 0.3.

Агент 2: Ресурс – 3000; розподіл пріоритетів: Лот 1 – 0.4, Лот 2 – 0.35, Лот 4 – 0.25.

Агент 3: Ресурс – 9000; розподіл пріоритетів: Лот 1 – 0.33, Лот 2 – 0.27, Лот 3 – 0.25, Лот 4 – 0.15.

Агент 4: Ресурс – 10500; розподіл пріоритетів: Лот 1 – 0.15, Лот 3 – 0.5, Лот 4 – 0.35.

Агент 5: Ресурс – 8790; розподіл пріоритетів: Лот 2 – 0.45, Лот 3 – 0.25, Лот 4 – 0.3.

Агент 6: Ресурс – 6770; розподіл пріоритетів: Лот 1 – 0.35, Лот 3 – 0.45, Лот 4 – 0.2.

З таблиці випливає, що стратегії агентів, забезпечувані запропонованим механізмом проведення аукціону, є домінуючими, оскільки вони дозволяють максимізувати прибуток (для переможця) або надавати максимально можливу (або близьку до неї) пропозицію (для агентів, що вибули з аукціону). Крім того, запропонований механізм дозволяє реалізувати оптимальний аукціон, оскільки він максимізує прибуток продавця. Так, у всіх випадках переможець (i-ий агент) придбав лот за найнижчою можливою ціною (більшою за попередню ціну за j-ий лот лише на значення одного кроку торгів), а продавець продав лот за максимально можливу ціну, яка відповідає вимозі передумови 1 (див. п. 2): $z_j(t_e) \le R_i^j$, де $z_j(t_e) \ge R_{-i}^j$. Очевидно також, що запропонований механізм задовольняє умовам (1).

Таблиця. Результати моделювання послідовного багатоелементного японського аукціону

Дані по лоту \ Агенти		**Агент 1**	**Агент 2**	**Агент 3**	**Агент 4**	**Агент 5**	**Агент 6**
Лот 1	Максимальна кількість ресурсів на лот	2010	1200	2970	1575	–	2369.5
	Максимальна ставка, зроблена агентом	2010	1190	2370	1560	–	2360
	Порядок вибуття агентів з торгів	3	1	5	2	–	4
	Отриманий прибуток	–	–	600	–	–	–
Лот 2	Максимальна кількість ресурсів на лот	–	1050	2430	–	3955.5	–
	Максимальна ставка, зроблена агентом	–	1020	2400	–	2420	–
	Порядок вибуття агентів з торгів	–	1	2	–	3	–
	Отриманий прибуток	–	–	–	–	1535.5	–
Лот 3	Максимальна кількість ресурсів на лот	2680	–	2250	5250	2197.5	3046.5
	Максимальна ставка, зроблена агентом	2640	–	2250	3060	2160	3030
	Порядок вибуття агентів з торгів	3	–	2	5	1	4
	Отриманий прибуток	–	–	–	2190	–	–
Лот 4	Максимальна кількість ресурсів на лот	2010	750	1350	3675	2637	1354
	Максимальна ставка, зроблена агентом	2000	680	1280	2640	2600	1320
	Порядок вибуття з торгів	4	1	2	6	5	3
	Отриманий прибуток	–	–	–	1035	–	–

Висновки

В статті як приклад некооперативних динамічних ігор з досконалою та неповною інформацією розглянуто послідовні багатоелементні японські аукціони, для яких поставлено та вирішено задачу побудови механізму проведення аукціонів, який узгоджує у собі використання агентами їх домінуючих стратегій та дозволяє сформувати оптимальний аукціон. Розроблені методи реалізовано в прототипі МАС «Аукціонъ». Виходячи з аналізу сучасного стану досліджень теорії аукціонів можна стверджувати, що запропонований підхід до мультиагентного моделювання аукціонів є новим, а розроблений метод побудови механізму проведення аукціонів створює передумови для подальших досліджень у напрямку розробки методів незалежного формування агентами домінуючих стратегій у рамках такого механізму, що в перспективі дозволить формулювати та вирішувати задачі моделювання одночасних багатоелементних аукціонів як основи для моделювання війн на виснаження для загального випадку.

1. *Klemperer P.* Auctions: Theory and Practice. – Princeton University Press, 2004. – 239 p.

2. *Krishna V.* Auction Theory. – Elsevier Inc., 2010. – 323 p.
3. *Lusk J.L., Shogren J.F.* Experimental Auctions. Methods and Applications in Economic and Marketing Research. – Cambridge University Press, 2007. – 304 p.
4. *Menezes F.M., Monteiro P.K.* An Introduction to Auction Theory. – Oxford University Press, 2008. – 184 p.
5. *Milgrom P.* Putting Auction Theory to Work. – Cambridge University Press, 2004. – 368 p.
6. *Mochón A., Sáez Y.* Understanding Auctions. – Springer, 2015. – 148 p.
7. *Multiagent* Systems: Algorithmic, Game-Theoretic, and Logical Foundations/ Shoham Y., Leyton-Brown K. – Cambridge University Press, 2008. – 532 p.
8. *Jureta I., Kolp M., Faulkner S.* Multi-Agent Patterns for Deploying Online Auctions // Selected readings on electronic commerce technologies: contemporary applications / Edited by Hu W.-Ch. – IGI Global, 2009. – P. 130–146.
9. *Filzmoser M.* Simulation of Automated Negotiation. – Springer-Verlag, 2010. – 248 p.
10. *Sierra C., Noriega P.* Agent-Mediated Interaction. From Auctions to Negotiation and Argumentation // Foundations and Applications of Multi-Agent Systems (UKMAS Workshops 1996–2000 Selected Papers). – Springer-Verlag, 2002. – P. 27–48.
11. *Писарук Н.Н.* Введение в теорию игр. – Минск: БГУ, 2015. – 256 с.
12. *Николенко С.И.* Теория экономических механизмов. – М.: БИНОМ, 2009. – 207 с.

References

1. *Klemperer P.* Auctions: Theory and Practice. – Princeton University Press, 2004. – 239 p.
2. *Krishna V.* Auction Theory. – Elsevier Inc., 2010. – 323 p.
3. *Lusk J.L., Shogren J.F.* Experimental Auctions. Methods and Applications in Economic and Marketing Research. – Cambridge University Press, 2007. – 304 p.
4. *Menezes F.M., Monteiro P.K.* An Introduction to Auction Theory. – Oxford University Press, 2008. – 184 p.
5. *Milgrom P.* Putting Auction Theory to Work. – Cambridge University Press, 2004. – 368 p.
6. *Mochón A., Sáez Y.* Understanding Auctions. – Springer, 2015. – 148 p.
7. *Multiagent* Systems: Algorithmic, Game-Theoretic, and Logical Foundations/ Shoham Y., Leyton-Brown K. – Cambridge University Press, 2008. – 532 p.
8. *Jureta I., Kolp M., Faulkner S.* Multi-Agent Patterns for Deploying Online Auctions // Selected readings on electronic commerce technologies: contemporary applications / Edited by Hu W.-Ch. – IGI Global, 2009. – P. 130–146.
9. *Filzmoser M.* Simulation of Automated Negotiation. – Springer-Verlag, 2010. – 248 p.
10. *Sierra C., Noriega P.* Agent-Mediated Interaction. From Auctions to Negotiation and Argumentation // Foundations and Applications of Multi-Agent Systems (UKMAS Workshops 1996-2000 Selected Papers). – Springer-Verlag, 2002. – P. 27–48.
11. *Pisaruk N.N.* Introduction to Game Theory. – Minsk: BGU, 2015. – 256 p. (in Russian).
12. *Nikolenko S.I.* Theory of Economical Mechanisms. – Moscow: BINOM, 2009. – 207 p. (in Russian).

Одержано 11.01.2016

Про автора:

Яловець Андрій Леонідович,
доктор технічних наук,
заступник директора інституту.
Кількість наукових публікацій в українських виданнях – 100.
Кількість наукових публікацій в іноземних виданнях – 5.
http://orcid.org/0000-0001-6542-3483

Місце роботи автора:

Інститут програмних систем
НАН України.
03187, Київ-187,
проспект Академіка Глушкова, 40.

Тел.: (044) 526 15 38.
E-mail: yal@isofts.kiev.ua

УДК 004.4

Об’єктно-компонентна розробка змінюваних програмних систем / К.М. Лавріщева, О.О. Слабоспицька, А.Ю. Стеняшин, А.Л. Колесник. – С. 3–16.

Об'єктно-компонентний метод моделювання програмних систем Лавріщевої – Грищенка розвинуто узгодженими моделями варіабельності систем та їх варіантної конфігураційної збірки з компонентів, а також алгеброю операцій ізоморфного перетворення нерелевантних типів даних для цих компонентів. Запровадження до моделі системи точок варіантності з їх варіантами забезпечує змінюваність систем та стійку взаємодію їх компонентів. Описано реалізацію формального апарата в автоматизованому конфігураторі систем та його апробацію в інструментально-технологічному комплексі ІПС НАНУ і експериментальній фабриці програм КНУ ім. Т. Шевченка.

Ключові слова: об'єкт, компонент, об'єктно-компонентний метод, модель варіабельності, артефакт, точка варіантності, готовий ресурс, управління моделями, конфігураційна збірка.

UDC 004.4

Object-component development of changeable software systems / E. Lavrischeva, O. Slabospitskaya, A. Stenyashin, A. Kolesnyk. – P. 3–16.

Complementary limitations of both Software Product Lines industrial technologies and Lavrischeva – Grishenrko object-component method concerning changeable software development are elicited such as the lack of formalisms for program assets building and ill predictability of this build features.

To cope with the limitations universal Model of Software Family Variant Features is proposed expanding its traditional feature model for basic development artifacts. For assets being considered as reusable Components final Changeable Software Object-Component Model is elaborated including the universal model above being adjusted as Software Variability Object-Component Model. The Algebra is depicted for the operations of both the Components configuring and data types transforming over their interaction within changeable software system.

These operations are proposed to incorporate into the target process for Changeable Software Family proactive and informed Variability management being represented with its technological chart. The process proposed composes the functions for variability Planning, Implementing and Control as well as Family model/consist Evolving up to the Control results. The functions listed are performed within common information environment structured accordingly to Variant Features Model or its object-component adjustment.

Trial software tool for configuring Components in the above process is probed. The usage is depicted of both the framework proposed and this tool over technological lines being implemented in Software Systems Institute of NAS of Ukraine Instrumental-technological complex for changeable software configuring from the components.

Key words: object, component, object-component method, variability model, artifact, variability point, asset, variability model, configuration build.

УДК 004.42:510.69

Алгебры квазиарных и би-квазиарных реляций / Н.С. Никитченко, С.С. Шкильняк. – С. 17–28.

Предложено понятие квазиарной реляции (отношения), введены операции над такими реляциями, описаны алгебры квазиарных реляций. Доказан изоморфизм алгебры квазиарных реляций и первопорядковой алгебры тотальных однозначных квазиарных предикатов. Построены алгебры бикквазиарных реляций, заданные на множествах пар квазиарных реляций. Определены различные подклассы таких алгебр, иссследованы их связи с алгебрами частичных однозначных, тотальных неоднозначных, частичных неоднозначных, монотонных, антитонных квазиарных предикатов.
Ключевые слова: алгебра, логика, изоморфизм, реляция, квазиарный предикат.

UDC 004.42:510.69

Algebras of quasiary and of bi-quasiary relations / M.S. Nikitchenko, S.S. Shkilniak. – P. 17–28.

The notion of quasiary relation which can be considered generalization of the notion of traditional n-ary relation is proposed. A number of algebras of quasiary relations is built and investigated. Alongside with conventional operations of union, intersection, and complement, special nominative operations of renomination and quantification are defined for quasiary relations. The isomorphism between the algebra of quasiary relations and the first-order algebra of total single-valued quasiary predicates is proved. Algebras of bi-quasiary relations defined over sets of pairs of quasiary relations are built. The isomorphism between algebras of bi-quasiary relations and algebras of quasiary predicates is proved. The following subclasses of algebras of bi-quasiary relations are specified: algebras of partial single-valued (functional), total, total many-valued bi-quasiary relations. For all defined subclasses their counterparts of the classes of algebras of quasiary predicates are described. Also subalgebras of the algebra of bi-quasiary relations induced by upward closedness and downward closedness are investigated.
Key words: algebra, logic, relation, isomorphism, quasiary predicate.

УДК 004.42:510.69

Отношения логического следствия в логиках квазиарных предикатов / О.С. Шкильняк. – С. 29–43.

Изучаются отношения логического следствия в логиках тотальных однозначных, частичных однозначных, тотальных неоднозначных и частичных неоднозначны предикатов. Наряду из ранее рассмотренными отношениями типов *T*, *F*, *TF*, *IR*, *DI*, для логик квазиарных предикатов предлжены и исследованы отношения типов $T \vee F$ и *C*. Описаны свойства отношений логического следствия. Приведены примеры, свидетельствующие о различии рассмотренных отношений. Показана не-

UDC 004.42:510.69

Logical consequence relations in logics of quasiary predicates / O.S. Shkilniak. – P. 29–43.

Logical consequence is one of the most fundamental concepts in logic. A wide use of partial (sometimes many-valued as well) mappings in programming makes important the investigation of logics of partial and many-valued predicates and logical consequence relations for them. Such relations are a semantic base for a corresponding sequent calculi construction. In this paper we consider logical consequence relations for composition nominative logics of total single-valued, partial single-valued, total

транзитивность отношений типов $T\vee F$ и C, возможность моделирования отношений типа C с помощью отношений типа TF. Установлены соотношения между различными отношениями логического следствия.
Ключевые слова: логика, предикат, семантика, логическое следствие.

many-valued and partial many-valued quasiary predicates. Properties of the relations can be different for different classes of predicates; they coincide in the case of classical logic. Relations of the types *T*, *F*, *TF*, *IR* and *DI* were investigated in the earlier works. Here we propose relations of the types $T\vee F$ and C for logics of quasiary predicates. The difference between these two relations manifests already on the propositional level. Properties of logical consequence relations are specified for formulas and sets of formulas. We consider partial cases when one of the sets of formulas is empty. It is shown that relations ${}^{P}|=_{T\vee F}$ and ${}^{R}|=_{C}$ are non-transitive, some properties of decomposition of formulas are not true for ${}^{R}|=_{C}$, but at the same time the latter can be modelled through ${}^{R}|=_{TF}$. A number of examples demonstrates particularities and distinctions of the defined relations. We also establish a relationship among various logical consequence relations.
Key words: logic, predicate, semantics, logical consequence.

УДК 00.007.3

Алгоритм поиска связей и зависимостей в данных Web-страниц /
А.Н. Глибовец. – С. 44–50.

Методы добычи и анализа данных – относительно новая и перспективная отрасль компьютерных наук, нашла свое применение в системах информационного поиска. В работе предложен алгоритм поиска связей и зависимостей в коллекциях Web-страниц. Алгоритм не предусматривает поиска релевантных ресурсов. Эту функцию выполняет поисковая система. Она также производит очистку, интеграцию и выбор данных.

Особенностью алгоритма является использование уже существующего хранилища данных (поисковая система или хранилище данных), языковая независимость и простота реализации.

Ключевые слова: поисковая система, PatternRecognition, алгоритм поиска, связи и зависимости.

UDC 00.007.3

Algorithms of relationships and dependencies search in Web-pages /
A.M. Glybovets. – P. 44–50.

Methods of extraction and analysis of data – a relatively new and promising branch of computer science, has found its application in information retrieval systems. An algorithm of relationships and dependencies searching in the collections of Web pages. The algorithm does not provide relevant search resources. This function is performed by the search engine. It also produces cleaning, integration, and data selection.

A special feature of the algorithm is to use the existing data store (search engine or data storage), language independence and ease of implementation.

Key words: search engine, PatternRecognition, search algorithms, relationships and dependencies.

УДК 681.3

Аналитический обзор методов и средств информационного поиска в Semantic Web / И.Ю. Гришанова. – С. 51–72.

В статье изложены и проанализированы методы и средства информационного поиска в среде Semantic Web. Представлены базовые понятия информационного поиска, задачи, модели и классификация систем информационного поиска по различным признакам. Приведены примеры существующих современных поисковых систем, а также выделены этапы развития и перечислен перечень функциональных и архитектурных признаков 3-х поколений поисковых систем. Предложенная модель информационного поиска для новой среды Semantic Web и Web вещей расширяет классификацию поисковых систем и модель поиска с учетом возможности поиска новых обьектов, которые стали доступными в Web, и использования знаний, представленных в Semantic Web.
Ключевые слова: информационный поиск, семантический поиск, поисковые системы, Semantic Web.

UDC 681.3

Analytical review on information retrieval methods and applications in the Semantic Web / I.Y. Grishanova. – P. 51–72.

The article describes and analyzes the Information Retrieval (IR) methods and applications in the environment of Semantic Web. The author provided the basic Information Retrieval concepts, problems, models and classification of IR systems on various grounds. Examples of existing modern search engines, as well as highlighted the stages of development and listed a list of functional and architectural features of 3-rd search engines generation. The proposed model of IR extends the classification of search engines and search model with the possibility of finding new objects that have become available in the web, and use knowledge represented in the Semantic Web.
Key words: information retrieval, semantic search, search engines, Semantic Web.

УДК 681.3

Использование онтологий для персонифицированного поиска знаний в естественноязыковых текстов / Ю.В. Рогушина. – С. 73–88.

Предложенный в работе подход к персонификации поиска информационных ресурсов и информационных объектов, который базируется на построении и использовании тезауруса задачи пользователя, позволяет использовать знания относительно предметной области поиска и структуры информационных объектов, представленные с помощью соответствующих онтологий. Приведенные определения семантического поиска, его субъектов и компонентов позволяют более четко формулировать проблемы, связанные с поиском информации в открытой среде Web. Программная реализация

UDC 681.3

Use of ontologies for personalized search of knowledge for natural language texts / Y. Rogushina. – P. 73–88.

The paper analyzes the problems of search personalization of information resources and information objects which is based on the construction and use of user task thesaurus. This thesaurus allows the use of knowledge about search domain and structure of information objects represented by some appropriate ontologies. The definitions of semantic search, its subjects and components allow more articulate issues related to the information retrieval in the Web open environment. Software implementation of the proposed approach confirms the effectiveness of its practical use.

предложенного подхода подтверждает эффективность его практического использования.
Ключевые слова: семантический поиск, информационный объект, онтология, тезаурус задачи.

Keywords: semantic search, information object, ontology, task thesaurus

УДК 004.822, 681.3, 519.81

Методи та моделі використання експертно-аналітичного знання для підтримки прийняття рішень в організації. Частина 1. Моделі знань про рішення / О.П. Ільїна. – С. 89–101.

В роботі надано апарат моделей корпоративного знання організації для експертно-аналітичної підтримки прийняття її рішень. Формалізовано концепт Організаційне рішення з урахуванням усіх стадій життєвого циклу останнього від постановки проблеми до аналізу результатів. Описано моделі категорій концептів зі складу онтології підтримки прийняття рішень та модель поля рішень організації. Систем відношень між концептами онтології становить предмет розгляду наступної частини статті, як і формалізм гармонізації рішень.
Ключові слова: організаційне рішення, онтологія сфери прийняття рішень, поле рішень організації, категорія концептів онтології, часткове означення концепту, концептуально неповне знання.

UDC 004.822,681.3,519.81

Methods and Models for Employment of the Expert Analytical Knowledge in Organization Decision Making. Part I. Decisions Knowledge Models / E.P. Ilina. – P. 89–101.

The paper is devoted to the formal representation of the organization decision by providing of the special kind of the organization knowledge models. Being employed in the organization information system these models give an ability for reusing of organization experience. Moreover they help to support analytical operations for quality management of organization decisions and decision making processes. The main components of the proposed knowledge models system are the Ontological Model for Decisions Support and the Decision Field of Organization. First of them includes such the categories of concepts as Gall, Gall Tree, Operation, Problem Situation, Side Influence, Decision, Package of Decisions, Value, Value tree. Their models are represented as the systems of the partial definitions for the concept that belong to the certain category and the set of rules for interpretation of each role position in such partial definition. Incomplete conceptual knowledge may be also represented in these models. Models proposed may be used for representation of the Organization Decisions Field and for constructing the mechanism of its analysis that is a subject of the next part of this article.
Key words: organization decision, ontology of decision making, decision field of organization, ontological concepts category, partial definition of concept, conceptually incomplete knowledge.

УДК 681.3

Автоматизированное проектирование программ для решения задачи метеорологического прогнозирования / А.Е. Дорошенко, П.А. Иваненко, О.М. Овдей, Е.А. Яценко. – С. 102–115.

Разработано средство автоматизированного конструирования параллельного кода для среды OpenMP на основе высокоуровневых алгебро-алгоритмических спецификаций. Применение средства демонстрируется на примере задачи моделирования циркуляции атмосферы, представленном как сервис в составе Интернет-портала для предоставления услуг метеопрогноза. Осуществлена генерация программного кода и приведены результаты эксперимента по выполнению разработанной параллельной программы прогнозирования на мультипроцессорной платформе.
Ключевые слова: параллельные вычисления, метеорологическое прогнозирование, синтез программ, алгебра алгоритмов, Интернет-портал.

UDC 681.3

Automated program design for solution of weather forecasting problem / A.Yu. Doroshenko, P.A. Ivanenko, O.M. Ovdiy, O.A. Yatsenko. – P. 102–115.

The facilities for automated design of parallel code for OpenMP environment on the basis of high-level algebra-algorithmic specifications are developed. The application of the facilities is illustrated on an example of a problem of atmosphere circulation modeling, which is represented as a service, belonging to the Internet-portal for providing meteorological forecasting services. The generation of program code was implemented and the results of the conducted experiment, which consisted in execution of the developed parallel weather forecasting program on a multiprocessor platform, are given.
Key words: parallel computing, meteorological forecasting, software synthesis, algorithms algebra, web portal.

УДК 004.7

Теоретико-ігрове моделювання рівноваги у AIMD мережах / О.П. Ігнатенко. – С. 116–128.

В даній роботі досліджується моделювання динаміки мережі на основі теоретико-ігрового підходу. Процес взаємодії між користувачами, що намагаються максимізувати свої виграші (наприклад, частку мережі) допускає представлення у формі гри та застосування методів аналізу рівноваги. В роботі пропонується модель TCP мережі та доведено існування і єдність точки стійкого розподілу ресурсів, побудована матриця мережевої гри та знайдені умови існування рівноваги в залежності від чутливості користувачів до наявності помилок. Розглянуто також вплив атак на характеристики рівноваги та проведене імітаційне моделювання.

УДК 004.7

Теоретико-игровое моделирование равновесия в AIMD сетях / А.П. Игнатенко. – С. 116–128.

В данной работе исследуется моделирования динамики сети на основе теоретико-игрового подхода. Процесс взаимодействия между пользователями, которые пытаются максимизировать свои выигрыши (например, долю сети) допускает представление в форме игры и применение методов анализа равновесия. В работе предлагается модель TCP сети и доказано существование и единственность точки устойчивого распределения ресурсов, построена матрица сетевой игры и найдены условия существования равновесия в зависимости от чувствительности пользователей к наличию ошибок. Рассмотрены также влияние атак на характеристики рав-

Ключові слова: теорія ігор, модель мережі, рівновага Неша, імітаційне моделювання.

новесия и проведено имитационное моделирование.

Ключевые слова: теория игр, модель сети, равновесие Нэша, имитационное моделирование.

УДК 004.942+519.837.2

Мультиагентное моделирование последовательных многоэлементных японских аукционов / А.Л. Яловец. – С. 129–137.

Исследуются особенности моделирования аукционов с точки зрения имитационного (мультиагентного) моделирования. Приводится характеристика аукционов как объекта моделирования. Выполняется постановка задачи и предлагается метод построения механизма проведения последовательных многоэлементных японских аукционов, который обеспечивает использование агентами доминирующих стратегий и позволяет построить оптимальный аукцион. Экспериментально подтверждается эффективность предложенного метода.

Ключевые слова: моделирование, японский аукцион, дизайн механизма, агент, доминирующие стратегии.

UDC 004.942+519.837.2

Multiagent modeling of sequential multi-unit japanese auctions / A.L. Yalovets. – P. 129–137.

Features of modeling of auctions are investigated from the point of view of simulation (multiagent) modeling. The characteristic of auctions as object of modeling is resulted. Statement of a task carries out and the method of construction of the mechanism of carrying out of sequential multi-unit japanese auctions, which provides use by agents of dominant strategies is offered and allows to construct optimal auction. Efficiency of the suggested method experimentally proves to be true.

Key words: modeling, japanese auction, mechanism design, agent, dominant strategies

ДО УВАГИ АВТОРІВ!

У журналі "Проблеми програмування" публікуються наукові матеріали, які раніше не публікувалися в інших виданнях.

Мова статті: українська, російська, англійська. Обсяг 6–16 с.

Разом с надрукованим текстом статті необхідно подати до редакції файли, які мають бути у форматі *. doc і підготовлені за допомогою текстового редактора Microsoft Word версії 6.0 і вище.

Автори можуть користуватися електронною поштою і також телефаксом для ділової переписки та передачі до редакції тексту статті та правки при коректурі. E-mail редакції: tsok@isofts.kiev.ua. FAX: +380 (44) 526 6263, Телефон: 526 5065.

1. Оформлення файлу з текстом статті.

При підготовці файлу використовуються: стиль нормальний (звичайний) або normal; шрифт Times New Roman, розмір шрифта 12 пт.; рядкова відстань одинарна; абзацний відступ 1,25 см. Формат паперу А4, поля документа – 2 см. Текст статті після анотації має бути оформлений у 2 колонки, ширина яких – 7,86 см, а пробіл між ними – 1,27 см.

Послідовність розміщення та оформлення матеріалу статті.

УДК: індекс за універсальною десятковою класифікацією.

Автори: ініціали та прізвища авторів, курсив (світлий).

Заголовок 1 **(назва доповіді):** не містить абревіатур та строго відповідає змісту статті. Шрифт 15 пт, напівжирний, регістр верхній.

Анотація **(мовою статті):** 50–100 слів, не містить абревіатур, зрозумілих з контексту статті. Шрифт 10 пт, звичайний.

Ключові слова **(мовою статті):** не більше10 слів, не містить абревіатур, зрозумілих з контексту статті, подаються в називному відмінку, розділені комами. Шрифт 10 пт, звичайний.

Заголовок 2 **(назва розділу):** шрифт 14 пт, напівжирний; абзац із центральним вирівнюванням, без переносів. Заголовки нижчого рівня (пункти і т.п.) у самостійний абзац не виділяються і проходять першим реченням текстового абзацу, шрифт 12 пт, напівжирний.

Основний текст статті, має такі необхідні елементи:

постановка проблеми в загальному вигляді і її зв'язок з важливими науковими або практичними завданнями;

аналіз останніх досліджень і публікацій, у яких розпочато рішення даної проблеми і на які спирається автор, виділення невирішених раніше частин загальної проблеми, яким присвячується дана стаття;

формулювання цілей статті (постановка задачі);

виклад основного матеріалу дослідження з повним обґрунтуванням отриманих наукових результатів;

висновки з даного дослідження і перспективи подальших розробок у даному напрямку;

подяка (за наявності такої).

Формули мають бути набрані в редакторі формул Microsoft Equation Editor 2.0 і вище.

Рисунки мають бути створені вбудованим редактором Word Picture або експортовані з прикладних програм Windows у графічних форматах (bmp, pcx, gif, tif). Розташування рисунків на сторінці центральне. Підрисуночний текст центрований.

Таблиці мають бути підготовлені стандартним вбудованим в Word інструментарієм “Таблиця”.

Література**:** нумерований список джерел згідно ДСТУ ГОСТ 7.1:2006, шрифт 11 пт.

Література англійською мовою (References): список використовуваних джерел згідно ***Harvard Style***. Джерела з заголовками на латиниці наводяться без перекладу. Для літератури джерел на мовах, що не використовують латинський алфавіт, необхідно

забезпечити переведення назв джерел і вказати після них у дужках мову оригіналу. Прізвища та ініціали авторів, слід транслітерувати за правилами як для закордонного паспорта. Приклади оформлення бібліографічних посилань згідно з вимогами ***Harvard Style*** наведені в багатьох публікаціях, наприклад, за електронною адресою http://www.staffs.ac.uk/assets/harvard_referencing_examples_tcm44-39847.pdf

Дані про авторів**:** мають починатися рядком "Про авторів:", напівжирний курсив. Далі вказуються для кожного з авторів ПІБ повністю, наукове звання, посада, адреса, кількість публікацій в українських виданнях (приблизна), кількість публікацій в іноземних індексованих виданнях (приблизна), індекс Гірша (за наявності), обов'язково номер ORCID (сайт ORCID http://orcid.org/).

Дані про місце роботи авторів**:** починаються рядком "Місце роботи авторів:", далі вказуються місце роботи, адреса, телефон, факс, електронна пошта, контактний телефон.

3. Оформлення файлу з анотаціями.

Файл з анотаціями містить інформацію двома мовами та має бути оформлений у дві колонки: УДК; назва статті; прізвища та ініціали авторів; текст анотації, ключові слова

Вимоги до анотації англійською мовою: обсяг від 100 до 250 слів, інформативність, оригінальність (не є калькою української або російськомовної анотації), змістовність (відображає основний зміст статті і результати досліджень), структурованість (дотримується логіки опису результатів у статті).

Примітка: Підписний індекс журналу "Проблеми програмування" – **90853.**