УДК 004.424, 004.415

В.Г. Колесник

DS-ТЕОРИЯ. ИССЛЕДОВАНИЕ ФАКТОРОВ ФОРМАТИРОВАНИЯ Р-ДАННЫХ

В этой работе продолжается изложение теории схем декомпозиции как теории прикладных алгоритмов. Рассматривается механизм преобразования схемы декомпозиции в реальный прикладной алгоритм, обусловленный разнообразием способов хранения данных на носителях информации. Описана порция хранения данных как явление электронной обработки данных и предложено обобщенное понятие порции хранения. Описаны факторы, которые порождают необходимость форматирования и реформатирования данных. Описана обобщенная картина форматирования данных. Предложена группа алгоритмических конструкций, которые реализуют процедуры форматирования и реформатирования данных для общего случая. Уточняется тот факт, что в рамках теории схем декомпозиции существует значительно больше обстоятельств сочетания и взаимодействия порций хранения, чем описано в работе. Предложен механизм синтеза канонического алгоритма и алгоритмических конструкций, которые реализуют процедуры форматирования данных. Процесс внесения изменений в канонический алгоритм последовательный и методичный.

Ключевые слова: алгоритм, алгоритмическая конструкция, порция хранения, данное, узел, дерево, форматирование.

Введение

В этой работе наряду с [1–4] продолжается изложение теории схем декомпозиции как теории прикладных алгоритмов. Рассматривается еще один аспект механизма преобразования схемы декомпозиции (DS) в реальный прикладной алгоритм. В работе [2] было продемонстрирована возможность подобного преобразования. Канонический алгоритм (КА), построенный на основании дерева полной схемы декомпозиции (DPS), задает структуру программы в целом. DS имеет в качестве компонент (строительных клеток) алгоритмические конструкции узла (АКУ). АКУ преобразуется в текст программы. Одна АКУ реализуется одним параграфом. Вся DS превращается в логически завершенную программу. В работах [3, 4] рассмотрен аспект деления Р-данных (понятие введено в [1]) при хранении их на реальных носителях.

Группа алгоритмов, рассматриваемых в статье, имеют отношение к проектированию следующих алгоритмических явлений:

– отделение контента от форматирования [5];

– упаковка-распаковка данных в пакетах для передачи данных в сетях [6];

– конвертация данных [7];

– создание табличных выходных форм [8];

– стилевое форматирование вебстраниц [9] и т. п.

В настоящее время ни в практическом плане, ни на теоретическом уровне не существует обобщенной концептуальной алгоритмической модели, которая могла бы облегчить проектирование данной группы алгоритмов. Далее предпринимается попытка создать обобщенную картину этой группы алгоритмов, с последующей целью создать концептуальную алгоритмическую модель. Анализ проводится в контексте развития DS-теории.

Описание порций хранения Р-данных

Порция хранения. В DS-теории внешним носителем есть А-лента. Порция хранения Р-данных, которая записывается и считывается с А-ленты – это запись. Размер записи не оговаривается. Предполагается, что запись по размеру совпадает с размером, сохраняемого в ней Эл-данного или простого А-данного. Из записей на А-ленте составляются подобласти.

Но “запись” и “подобласть” – это не технические, а концептуальные понятия DS-теории. В практике электронной обработки данных обмен информации между компьютером и внешними устройствами выполняется блоками или порциями. В рамках статьи будет использоваться термин “порция хранения” (ПХ). Размеры ПХ, как правило, не совпадают ни с размерами записей (понятие DS-теории) и подобластей, ни с размерами их содержимого – Эл-данных или простых А-данных. Из-за этого необходимо включать в алгоритм конструкции форматирования Р-данных. Факторы, которые учитываются при размещении форматированных Р-данных в ПХ, называются алгоритмически релевантными факторами форматирования Р-данных (АРФФ).

При выводе Р-данных на реальные носители информации записи абстрактной длины и подобласти должны быть вложены в тот формат, в котором они будут храниться на этих носителях, – должны быть форматированы[1]. Если размер ПХ больше размера Р-данного, то форматирование заключается в том, чтобы собрать несколько Р-данных, чтобы укомплектовать ПХ и затем его вывести. Если размер ПХ меньше размера Р-данного, то форматирование заключается в том, чтобы разделить Р-данное на несколько ПХ, а потом их вывести. При вводе информации с реальных носителей должны быть восстановлены[2] абстрактные записи с Р-данными предназначенными для обработки. Если размер ПХ больше размера Р-данного, то реформатирование заключается в том, чтобы вычленить несколько Р-данных из укомплектованной ими ПХ для последующей их обработки. Если размер ПХ меньше размера Р-данного, то реформатирование заключается в том, чтобы собрать ПХ которыми укомплектовано Р-данное для последующей их обработки. Здесь очень коротко описаны процедуры форматирования-реформатирования (ФРФ) Р-данных. Основной аспект процедур ФРФ – это согласование размеров Р-данных и ПХ.

Цель работы. Показать обобщенную картину форматирования Р-данных, а также описать (очертить) группу алгоритмических конструкций, которые реализуют процедуры ФРФ Р-данных. Описать группу АРФФ и то, как эти факторы влияют на канонический алгоритм. Описать те изменения, которые необходимо внести в канонический алгоритм, чтобы организовать Р-данные для реализации расчета А-зависимостей в виде одного непрерывного потока – так, как будто они в естественной для DS-теории форме.

Изменения, которые должны быть внесены в канонический алгоритм, это операторы или алгоритмические конструкции (АК). Далее они будут либо полностью приведены, либо бегло описаны. Технически в условиях статьи невозможно показать весь набор изменений, обусловленных группой АРФФ. При анализе алгоритма объединения Р-данных будет обращаться внимание на то, появляется ли синергетический эффект [3].

Структура порции хранения. В рамках DS-теории предполагается, что все Р-данные состоят из знаков (цифры, буквы, знаки препинания, специальные символы). Понятие “знак” есть прототип понятия “байт”. С точки зрения внутреннего закодированного представления, информацию аудио-, видео- и любого другого вида файлов можно рассматривать как состоящую из знаков. Все Р-данные могут быть измерены количеством представляемых ими знаков. Аналогично могут быть измерены и ПХ количеством знаков, кото-

[1] Форматирование текста, выполняемое редакторами текстов, – это частный случай форматирования Р-данных. Генерация кода этих программ, выполняемая транслятором, содержит процедуры форматирования. Вывод сообщения на экран монитора в процессе диалога в социальной сети тоже включает процедуру форматирования в понимании DS-теории.

[2] Не существует термина общего плана для всех алгоритмических явлений, связанных с разделением контента и форматирования, которым можно было бы обозначить процесс восстановления Р-данного – снятие или упразднение формы, в которой оно сохранялось. Наиболее близкое по смыслу понятие – “деконструкция”, но это из другой сферы деятельности. Понятия “распаковка”, “разархивирование”, “реформатирование” или “деформатирования” (unformat) – ассоциируются с этим процессом, но не совпадают с его сутью полностью. В DS-теории с некоторой степенью условности используется понятие “реформатирование”.

рые они содержат на носителе. ПХ – это обобщения таких понятий как:

- магнитная лента – бобина, файл, блок;
- магнитный диск (физическая структура) – диск, дорожка, сектор;
- магнитный диск (логическая структура) – диск, логический диск, сектор;
- электронный документ – страница, строка, знак;
- бумажный документ – [книга], [раздел], страница, строка, знак и т. п.

Многообразие порций хранения порождается многообразием видов носителей (рис. 1). Различной может быть также вложенность ПХ. Так, например, страницу текста можно рассматривать состоящей из строк, но также состоящей из абзацев и наоборот. Компоненты, из которых состоят более сложные ПХ, могут быть различных видов и, возможно, различной длины. В файлах – это записи разных типов, а в документе строки различной длины. Наряду с вложенностью Р-данных существует вложенность реальных ПХ и вложенность абстрактных ПХ. Вложенность ПХ (структура ПХ) на А-ленте изображается деревом[3]. В практике обработки данных могут использоваться ПХ, для измерения которых могут потребоваться и более двух координат. Если для определения размера реального носителя используются две координаты, то сложность структуры ПХ возрастает[4].

С точки зрения длины простейшими Р-данными (ПРД) – Эл-данные и простые А-данные. Внутреннее представление числовой, текстовой информации или информации иных видов в DS-теории не рассматривается. Р-данные – это только цепочки знаков или символов. Длина их известна перед началом обработки.

Простейшие Р-данные в практике электронной обработки могут быть также переменной и неопределенной длины. Размеры их должны быть известны перед началом работы с ними. В данной работе предполагается, что перед началом работы по ФРФ, длина Р-данного известна. Ситуация, в которой для определения Р-данных переменной или неопределенной длины, необходимо выполнять предварительный проход вдоль цепочки Р-данных на А-ленте или в А-памяти в работе не рассматривается.

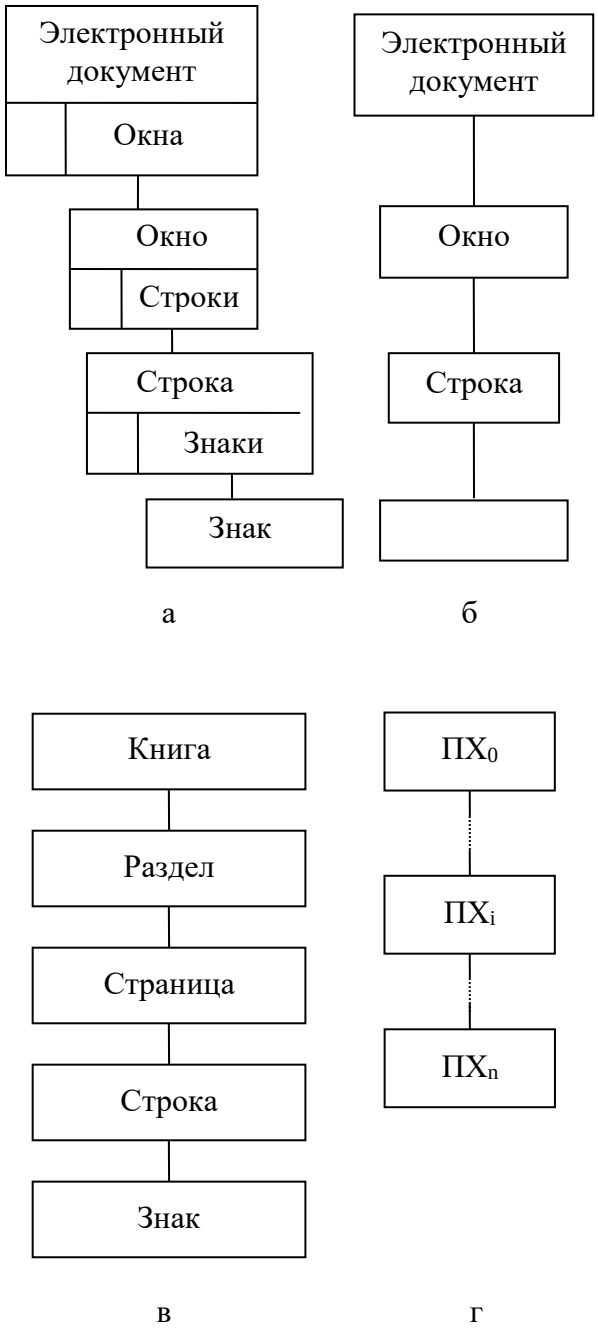

Рис. 1. Виды порций хранения (ПХ)

[3] На рисунке 1, а показана структура ПХ в формате FS [1]. На рис. 1, б и 1, в показаны структуры ПХ в упрощенном виде. На рис. 1, г показана структура ПХ в общем виде для случая, когда в каждой ПХ содержится компонента только одного вида – в дереве только одна ветвь.

[4] В работе не рассматриваются ПХ, размеры которых определяются двумя координатами.

Форматирование Р-данных

Отношения между структурами ПХ и Р-данными. Процедурам ФРФ могут подвергаться Р-данные всех видов, это: простые А-данные (рис. 2, а), расширенные А-данные (рис. 2, б), сложные А-данные (рис. 2, в), простые С-свойства, расширенные С-свойства (рис. 2, г), сложные С-свойства. ПХ, для которых применяются процедуры ФРФ, могут быть простыми (рис. 2, д), составными (рис. 2, е) а также могут быть компонентами более сложных ПХ (рис. 2, ж). Таким образом, общий вид взаимодействия Р-данных и ПХ посредством процедур ФРФ показан на рис. 2, з. Двунаправленная двойная стрелка на этом рисунке и в дальнейшем будет указывать на то, какое именно Р-данное взаимодействует с конкретной ПХ. Это значит, что это Р-данное будет при вводе изменено в соответствии с форматом указанной ПХ. При выводе это Р-данное будет реформатировано (извлечено или освобождено от формата ПХ для последующей обработки). ПХ, на которое указывает стрелка, называется взаимодействующей порцией хранения (ПХВ). Простейшие, несоставные ПХ обозначаются аббревиатурой ПХП, – это листья в дереве, которое изображает структуру ПХ.

Все ПХ имеют свои атрибуты. Существование их связано со способом хранения информации на носителях. Иногда объем ПХ задается Р-данным и учитывая это соответствующий атрибут ПХ формируется перед началом и после форматирования Р-данного. Связь Р-данного и соответствующего ПХ обозначается аббревиатурой ПХК (порция хранения, коррелирующая с Р-данным). На чертеже такая связь обозначается двунаправленной одинарной стрелкой. ПХК всегда больше ПХВ. В терминах деревьев это значит, что

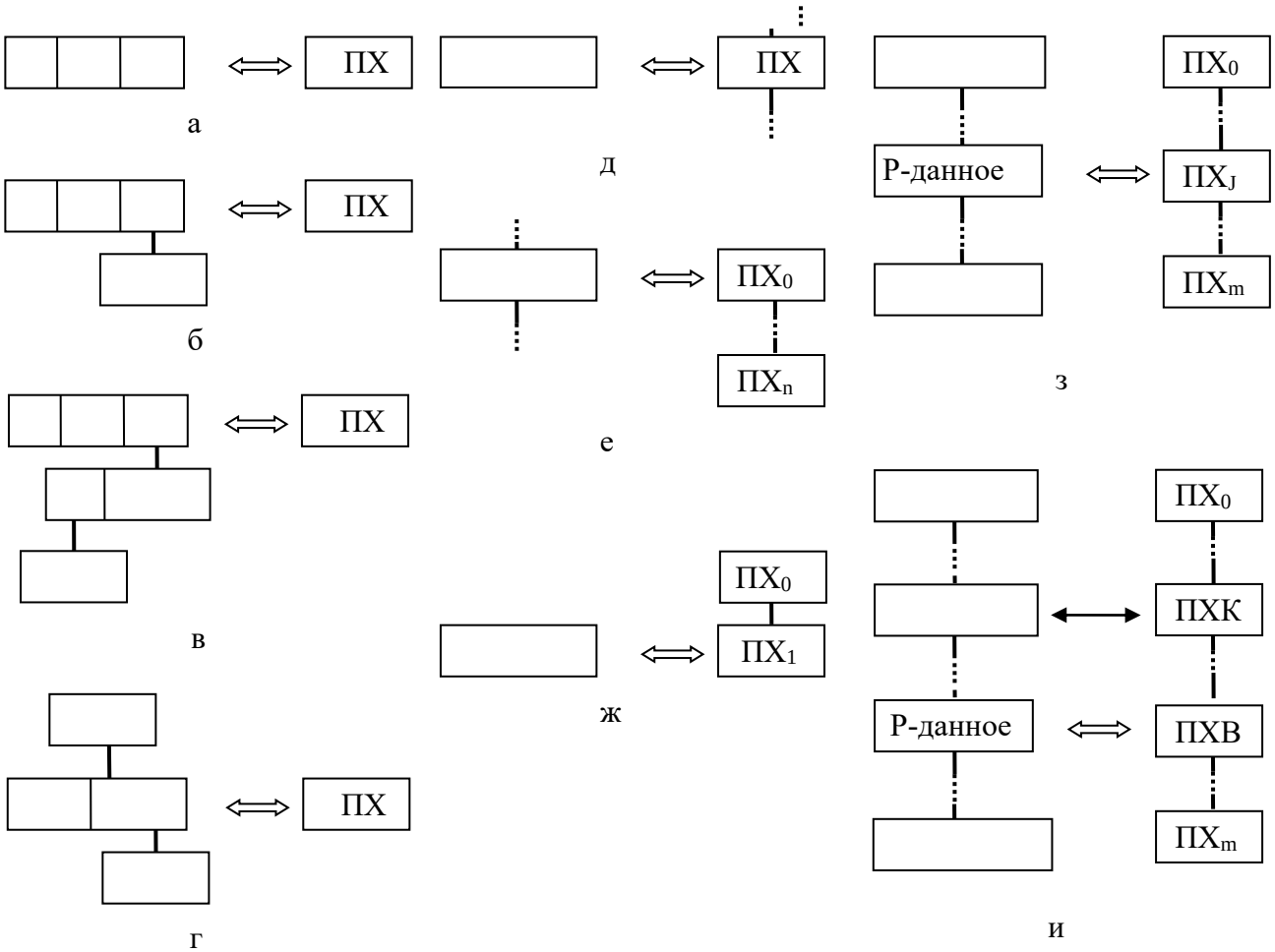

Рис. 2. Варианты взаимодействия Р-данных и ПХ

узел ПХК всегда расположен в дереве выше, чем узел ПХВ (рис. 2, и). ПХК может быть более одной.

Операторы форматирования. В работе [2] используются операторы READ и WRITE, которые являются компонентами гипотетического языка, используемого только для иллюстрации алгоритмических конструкций в рамках изложения DS-теории. Эти операторы обеспечивают ввод и вывод записей с А-ленты [2]. Но с точки зрения АРФФ, операторы READ используются для ввода ПХ (WRITE – для вывода ПХ), а для АК, которые реализуют процедуры ФРФ, вводятся операторы UNPACK – для реформатирования Р-данных и PACK – для форматирования[5] Р-данных.

ПХ вводится оператором READ в поименованный участок А-памяти [2]. С этим участком соотносятся три служебных Эл-данных: длина участка (префикс имени – LNC_), адрес внутри участка (префикс имени – APH_) и количество переносимых символов (префикс имени LFR_). После ввода выполняется реформатирование информации в Р-данное. Перед выводом в участок ПХ вводится форматированная информация и затем оператором WRITE выводится на А-ленту. С учетом обработки ПХ операторы READ и WRITE дополняются атрибутами INTO и FROM соответственно.

READ *имя_ленты* INTO *имя_участка_ПХ* – ввод ПХ с А-ленты *имя_ленты* в участок ПХ *имя_участка_ПХ*;

WRITE *имя_ленты* FROM *имя_участка_ПХ* – вывод ПХ на А-ленту *имя_ленты* с участка ПХ *имя_участка_ПХ*.

Р-данное, подвергающееся процедуре ФРФ, рассматривается как строка символов. С ним соотносится два служебных Эл-данных: длина Р-данного (префикс имени – LNR_) и адрес внутри участка занимаемого Р-данным (префикс имени – ARD_). Формат оператора PACK следующий: PACK *имя_Р-данного* INTO *имя_участка_ПХ* (*количество_переносимых_символов*) – перенос Р-данного *имя_Р-данного* или его фрагмента в участок ПХ *имя_участка_ПХ*.

Перед выполнением оператора PACK должны быть определены: адрес в участке Р-данного, количество переносимых символов и адрес в участке ПХ.

Формат оператора UNPACK следующий: UNPACK *имя_Р-данного* FROM *имя_участка_ПХ* (*количество_переносимых_символов*) – перенос содержимого участка ПХ *имя_участка_ПХ* или его части в Р-данное *имя_Р-данного*. Перед выполнением оператора UNPACK должны быть определены: адрес в ПХ, количество переносимых символов и адрес в участке Р-данного.

Принцип работы операторов PACK и UNPACK рассматривается далее.

Описание алгоритмов форматирования Р-данных

Отношения между ПРД и ПХ. Есть три варианта реформатирования Р-данных при вводе. Во всех случаях оператор UNPACK переносит информацию из ПХВ в Р-данное. Во всех вариантах A1 – имя А-ленты, NP – имя участка ПХВ, NRD – имя ПРД.

1. Длина ПРД равна длине ПХВ. АК имеет вид:

```
LFR_NP = LNC_NP
READ A1 INTO NP
UNPACK NRD FROM NP (LFR_NP)
```

Алгоритм 1

2. Длина ПРД больше длины ПХВ. АК имеет вид:

```
LFR_NP = LNC_NP
READ A1 INTO NP
ARD_ NRD = 1
PERFORM AA UNTIL (ARD_NRD ≥
LNR_NRD)
    {Адрес очередной части Р-данного
    больше размера Р-данного}
```

[5] Операторы PACK и UNPACK гипотетические и могут иметь большее или меньшее функциональное наполнение, но вложено именно такое, что позволит лучше иллюстрировать АК реализующие процедуры ФРФ.

```
    …
AA: BEGIN
  UNPACK NRD FROM NP (LFR_NP)
  ARD_ NRD = ARD_ NRD + LNC_NP
        END AA
```

Алгоритм 2

После выполнения оператора UNPACK увеличен адрес для загрузки очередной порции информации в Р-данном на размер ПХВ.

3. Длина ПРД меньше длины ПХВ. АК имеет вид:

```
{Перед первым выполнением UNPACK}
LFR_NP = LNC_NRD
READ A1 INTO NP
APH_NP = 1
…
UNPACK NRD FROM NP (LFR_NP)
APH_NP = APH_NP + LNC_NRD
IF (APH_NP ≥ LNC_NP)
{Адрес очередной части ПХВ больше размера ПХВ}
  READ A1 INTO NP
  APH_NP = 1
ENDIF
```

Алгоритм 3

После выполнения оператора UNPACK увеличен адрес для чтения очередной порции информации в ПХВ на размер Р-данного.

Есть три варианта форматирования Р-данных при выводе. Во всех случаях оператор PACK переносит информацию из Р-данного в ПХВ. Во всех вариантах A1 – имя А-ленты, NP – имя участка ПХВ, NRD – имя ПРД.

1. Длина ПРД равна длине ПХВ. АК имеет вид:

```
LFR_NP = LNC_NRD
PACK NRD INTO NP (LFR_NP)
WRITE A1 FROM NP
```

Алгоритм 4

2. Длина ПРД больше длины ПХВ. АК имеет вид:

```
LFR_NP = LNC_NP
ARD_NRD = 1
PERFORM AA (ARD_NRD ≥ LNR_NRD)
    {Адрес очередной части Р-данного
    больше размера Р-данного}
…
AA: BEGIN
  PACK NRD INTO NP (LFR_NP)
  WRITE A1 FROM NP
  ARD_NRD = ARD_NRD + LNC_NP
END AA
```

Алгоритм 5

После выполнения оператора PACK увеличен адрес в Р-данном на размер ПХВ для чтения очередной порции информации.

3. Длина ПРД меньше длины ПХВ. АК имеет вид:

```
{Перед первым выполнением PACK}
LFR_NP = LNC_NRD
APH_NP = 1
        …
PACK NRD INTO NP (LFR_NP)
APH_NP = APH_NP + LNR_ARD
IF (APH_NP ≥ LNC_NP) THEN
{Адрес очередной части ПХВ больше размера ПХВ – ПХВ заполнена}
  WRITE A1 FROM NP
  APH_NP = 1
ENDIF
```

Алгоритм 6

После выполнения оператора PACK увеличен адрес на размер ПРД в участке ПХВ для загрузки очередной порции информации.

Форматирование С-данного при выводе[6]. При форматировании с последующим выводом простого или расширенного С-данного нет смысла сопоставлять его размер с размером ПХВ, так как размер С-данного не постоянный. Кроме того, обработка С-данного происходит последовательным перебором его компонент – простейших Р-данных (Эл-данных и простых А-данных) и, соответственно, последовательно форматируются его компоненты. Именно их размеры и следует сопостав-

[6] Далее в работе рассматриваются только процедуры форматирования при выводе.

лять с размерами ПХВ. Далее рассматриваются пять вариантов (алгоритмов) форматирования. Во всех вариантах A1 – имя А-ленты, NP – имя участка ПХВ, NRD – имя ПРД – компоненты С-данного, NRC – имя С-данного.

1. Длина ПРД (компоненты С-данного) равна длине ПХВ. АК имеет вид:

```
AA: BEGIN
  LFR_NP = LNC_NRD
  PERFORM BB ({до конца С-данного})
  …
END AA
BB: BEGIN
  PACK NRD INTO NP (LFR_NP)
  WRITE A1 FROM NP
END BB
```

Алгоритм 7

2. Длина ПРД больше длины ПХВ. АК имеет вид:

```
AA: BEGIN

LFR_NP = LNC_NP
PERFORM BB ({до конца С-данного})
…
END AA
BB: BEGIN
  ARD_NRD = 1
  PERFORM CC (ARD_NRD ≥ LNR_NRD)
  {до конца Р-данного адрес очередной части Р-данного больше размера Р-данного}
  WRITE A1 FROM NP
END BB

CC: BEGIN
  PACK NRD INTO NP (LFR_NP)
  ARD_NRD = ARD_NRD + LFR_NP
END CC
```

Алгоритм 8

3. Длина ПРД меньше длины ПХВ. АК имеет вид:

```
AA: BEGIN

  {Перед первым выполнением PACK}
  LFR_NP = LNC_NRD
  APH_NP = 1
  PERFORM BB ({до конца С-данного})
  …
END AA
BB: BEGIN
  PACK NRD INTO NP (LFR_NP)
  APH_NP = APH_NP + LNR_ARD
  IF (APH_NP ≥ LNC_NP) THEN
  {Адрес очередной части ПХВ больше размера ПХВ – ПХВ заполнена}
    WRITE A1 FROM NP
    APH_NP = 1
  ENDIF
END BB
```

Алгоритм 9

В следующем варианте учитывается то, что размер ПРД может быть не кратный размеру ПХВ, но последний фрагмент ПРД заполнит ПХВ частично.

4. Длина ПРД больше длины ПХВ. PRIZN – рабочее Р-данное. АК имеет вид:

```
AA: BEGIN
  APH_NP = 1
  ARD_NRD = 1
  PERFORM BB ({до конца С-данного})
  WRITE A1 FROM NP
  …
END AA
BB: BEGIN
  PRIZN = 0
 {Р-данное не упаковано}
  PERFORM CC ({до конца С-данного})
                OR (PRIZN = 1)
  {Р-данное упаковано}
END BB

CC: BEGIN
  IF (LNR_NRD - ARD_NRD + 1 > LNC_NP – APH_NP + 1) THEN
  {Остающееся количество символов Р-данного больше свободного участка ПХВ}
  LFR_NP = LNC_NP - APH_NP + 1
  ELSE
    {Остающееся количество символов Р-данного меньше или равно свободному участку ПХВ}
    LFR_NP = LNR_NRD - ARD_NRD + 1
  ENDIF
  PACK NRD INTO NP (LFR_NP)
  APH_NP = APH_NP + LFR_NP
  IF (APH_NP ≥ LNC_NP) THEN
```

```
   {ПХВ заполнена}
   WRITE A1 FROM NP
   APH_NP = 1
  ENDIF
  ARD_NRD = ARD_NRD + LFR_NP
  IF (ARD_NRD > LNR_NRD) THEN
   PRIZN = 1
   ARD_NRD = 1
  ENDIF
END CC
```

Алгоритм 10

Если форматируется С-данное (NRC), но с ПХ взаимодействует компонента С-данного – Р-данное (NRD), то в параграфе АА нужно убрать операторы:

```
APH_NP = 1
ARD_NRD = 1
…
WRITE A1 FROM NP
```

А параграф ВВ будет иметь вид:

```
BB: BEGIN
 APH_NP = 1
 ARD_NRD = 1
 PERFORM CC (ARD_NRD ≥ LNR_NRD)
 {Адрес очередной части Р-данного больше размера Р-данного – Р-данное упаковано}
 WRITE A1 FROM NP
END BB
```

Алгоритм 11

Длина Р-данного меньше длины ПХВ и, что размер Р-данного может быть не кратный размеру ПХВ. Данный случай – это частный случай ситуации в п. 4.

С-данное в последних пяти ситуациях – это FS двух уровней. АКУ-обусловленность для этих алгоритмов сохранялась бы в том случае, если бы DA тоже была бы двух уровней. Но в п. 4 появляется параграф СС – это вследствие того, что размер Р-данного превосходит размер ПХВ. АЗ-параграфом здесь есть параграф ВВ.

Алгоритм форматирования будет сохраняться и в том случае, если в ПХВ будут форматироваться и перемещаться более одного ПРД и хотя бы одно из них будет отличаться размером от остальных. При этом хотя бы одно из ПРД имеет размер не кратный размеру ПХВ.

Форматирование FS в простых ПХ. Если форматированию подлежит сложное С-свойство (FS трех уровней – рис. 3, а), то на каком бы уровне не находились ПРД, в АЗ-параграфах будут АК реализующие процедуры ФРФ. При этом не имеет значения соотношения размеров между ПРД и ПХ, так как применяется одна и та же АК – параграф СС алгоритма 10. Так как каждый из таких параграфов сочленяется с DA иерархическим сочленением, то АКУ-обусловленность программы отсутствует. Соответствующий DA показан на рис. 3, б.

Произвольная FS может содержать в каждом узле на каждом уровне произвольное количество ПРД. Кроме того, во всех узлах кроме листьев могут отсутствовать ПРД. Лист должен содержать хотя бы одно ПРД. Пример, на рис. 3, в

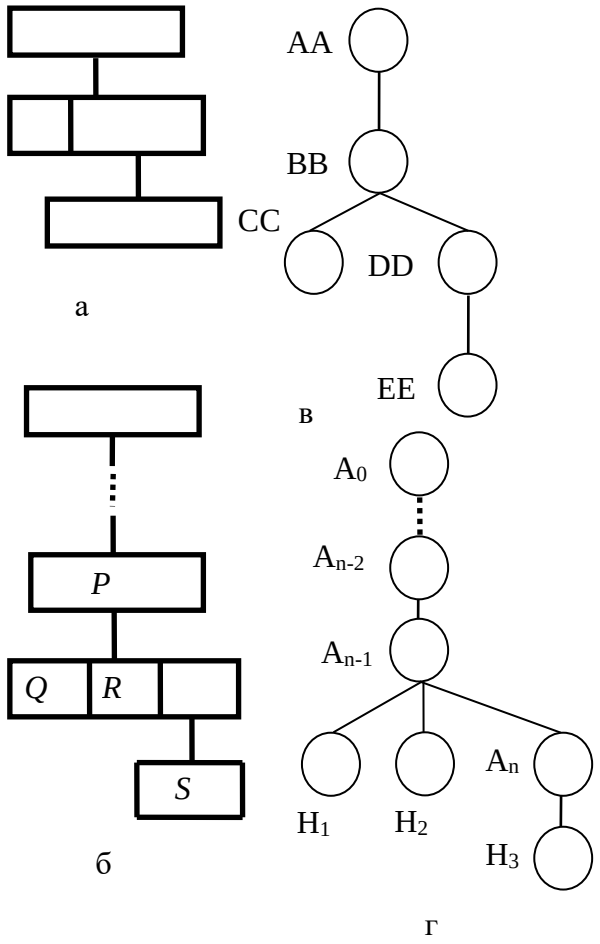

Рис. 3. Форматирование FS

узел уровня n-1 содержит два ПРД – Q и R; узлы нулевого и n-2 уровней не содержат ПРД; узел уровня n содержит одно ПРД – S. DA программы реализующей процедуры ФРФ при выводе для соответствующей FS будет содержать в каждом АЗ-параграфе цикл для форматирования ПРД. Тело цикла – это вышеупомянутый параграф СС алгоритма 10. DA для форматирования FS на рис. 3, в показана на рис. 3, г.

ПХВ может соотносится с С-данными не только предпоследних, но любых уровней FS. Узел, который соотносится с ПХВ, где бы он не находился в FS, повлечет в DA изменения только в АЗ-параграфе. Это начало и завершение работы с ПХВ – те действия, что выполняют операторы APH_NP = 1, ARD_NRD = 1 и WRITE A1 FROM NP в параграфе АА алгоритма 10.

Описание алгоритмов взаимодействия с составными ПХ

Форматирование Р-данных с составными ПХ. Как упоминалось выше, ПХ могут быть составными. Этот фактор тоже рассматривается при построении АК, реализующих процедуры ФРФ. Анализ АК проводится в следующих ситуациях.

ПРД переносится в ПХВ. Размеры Р-данного и ПХВ совпадают. ПХВ состоит из кратного количества ПХП меньших размеров (рис. 4, а). Во всех вариантах A1 – имя А-ленты, NP – имя участка ПХВ, NPP – имя участка ПХП, NRD – имя ПРД. ПХВ содержит кратное количество ПХП. АК имеет вид:

```
AA: BEGIN
LFR_NPP = LNC_NPP
ARD_NRD = 1
APH_NP = 1
PERFORM BB (ARD_NRD ≥ LNR_NRD)
    {До конца Р-данного}
WRITE A1 FROM NP
…
END AA
BB: BEGIN
  PACK NRD INTO NP (LFR_NPP)
  ARD_NRD = ARD_NRD + LNC_NPP
  ARD_NP = ARD_NP + LNC_NPP
END BB
```

Алгоритм 12

Параграф BB необходим для того, чтобы заполнять ПХВ фрагментами ПРД равными по длине размерам ПХП. Эта ситуация отличается параграфом алгоритма 12 от ситуации реализуемой алгоритмом 4. Соответственно, в алгоритмах 5 и 6 тоже появится параграф, который выполняет ту же функцию форматирование ПХП. То есть, то обстоятельство, что ПХВ составная, влечет в DA дополнительный параграф. Этот дополнительный параграф наряду с фактором несовпадения размеров ПРД и ПХ тоже влечет нарушение АКУ-обусловленности.

Анализ АК для форматирования С-данного показывает, что необходимо учитывать размеры компонент Р-данного и размеры ПХП. Прежде всего эти размеры определяют содержимое АК. Далее рассматриваются три ситуации форматирования С-данных в сложных ПХ.

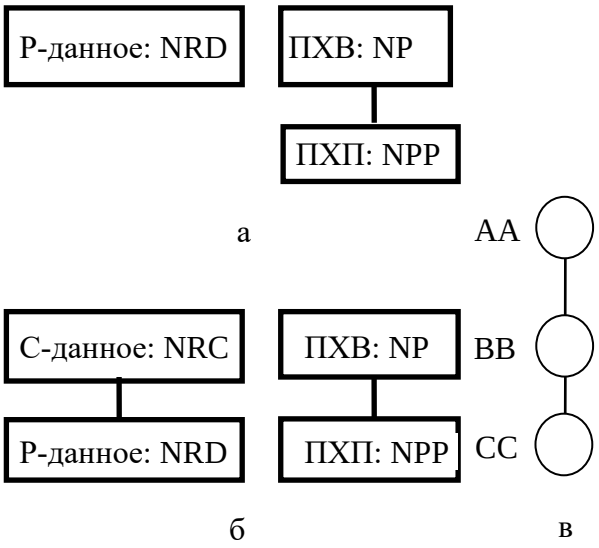

Рис. 4. Форматирование составных ПХ

1. С-данное форматируется и переносится в составную ПХВ (рис. 4, б). Компонента С-данного – ПРД имеет длину больше длины ПХП. Прототипом алгоритма необходимого для этой ситуации есть алгоритм 8. После внесения соответствующих изменений АК имеет вид:

```
AA: BEGIN
  LFR_NP = LNC_NP
  PERFORM BB ({до конца С-данного – NRC})
  …
END AA
BB: BEGIN
  ARD_NRD = 1
  PERFORM CC ({до конца С-данного – NRC})
            OR (ARD_NRD ≥ LNR_NRD)
            {до конца Р-данного – NRD}
  WRITE A1 FROM NP
END BB

CC: BEGIN
  PACK NRD INTO NP (LFR_NP)
  ARD_NRD = ARD_NRD + LFR_NP
END CC
```

Алгоритм 13

Параграф BB здесь появляется для того, чтобы разделить на части и форматировать ПРД (NRD). Этот параграф нарушает АКУ-обусловленность алгоритма. Дерево алгоритма не совпадает с деревом FS.

2. С-данное форматируется и переносится в составную ПХВ (рис. 4, б). Компонента С-данного – ПРД имеет длину меньше длины ПХП. АК имеет вид:

```
AA: BEGIN

LFR_NP = LNC_NRD
PERFORM BB
          ({до конца С-данного – NRC})
…
END AA
BB: BEGIN
  ARD_NRD = 1
  PERFORM CC
            ({до конца С-данного – NRC})
       OR (APH_NPP ≥ LNC_NPP)
                {до конца ПХП – NPP}
  WRITE A1 FROM NP
END BB

CC: BEGIN
  PACK NRD INTO NP (LFR_NP)
  APH_NPP = APH_NPP + LFR_NP
END CC
```

Алгоритм 14

Параграф BB здесь появляется для того, чтобы форматировать и объединить ПРД (NRD). Этот параграф нарушает АКУ-обусловленность алгоритма.

3. В этом варианте учитывается то, что размер ПРД может быть не кратный размеру ПХП, но последний фрагмент ПРД заполнит ПХВ частично. АК имеет вид:

```
AA: BEGIN
  APH_NP = 1
  ARD_NRD = 1
  PERFORM BB ({до конца С-данного})
  WRITE A1 FROM NP
  …
END AA
BB: BEGIN
  PRIZN = 0
  {ПРД не отформатировано}
  PERFORM CC ({до конца С-данного})
           OR (PRIZN = 1)
  {ПРД отформатировано}
END BB

CC: BEGIN
  IF (LNR_NRD - ARD_NRD + 1 > LNC_NP
- APH_NP + 1) THEN
  {Остающееся количество символов Р-
данного больше свободного участка ПХВ}
    LFR_NP = LNC_NP - APH_NP + 1
  ELSE
    LFR_NP = LNR_NRD - ARD_NRD + 1
  ENDIF
  PACK NRD INTO NP (LFR_NP)
  APH_NP = APH_NP + LFR_NP
  IF (APH_NP > LNC_NP) THEN
  {Адрес очередной части ПХВ больше
размера ПХВ – ПХВ заполнена}
    WRITE A1 FROM NP
    APH_NP = 1
  ENDIF
  ARD_NRD = ARD_NRD + LFR_NP
  IF (ARD_NRD > LNR_NRD) THEN
    PRIZN = 1
    ARD_NRD = 1
  ENDIF
END CC
```

Алгоритм 15

В процессе построения алгоритмов 13 и 14 сопоставлялись размеры двух совокупностей: размер ПРД и размер ПХП. Большая из них порождала параграф (и, соответственно, дополнительный цикл). Этот цикл должен был заполнить большую из совокупностей компонентами меньшей совокупности. В алгоритме 15 дополнительный параграф (и цикл) порожден тем, что в процессе форматирования компонентами, заполняющими ПХП, становятся фрагменты ПРД независимо от размеров ПРД и ПХП. 2, з и 2, и).

Форматирование FS в сложных ПХ. Как упоминалось выше как FS, так и структуры ПХ, в которые форматируются Р-данные, могут иметь произвольное количество уровней. При этом необходимо учитывать размеры между узлами дерева последовательно всех уровней (рис. 2, з и 2, и).

Общий подход рассматривается на примере (рис. 5). FS имеет три уровня и структура порций хранения имеет три уровня. С-данные А и В имеют состоят из произвольного количества компонент. ПХ М состоит из 30 компонент. Длина ПРД С – 200 знаков, а ПХП N может содержать 400 знаков. Параграф нулевого уровня содержит цикл который будет работать до завершения обработки С-данного нулевого уровня FS – А. Вместе с завершением этого цикла будет завершено форматирование исходной FS. Параграф первого уровня тоже содержит цикл. Управлять этим циклом будет большая из совокупностей – С-данное В (на первом уровне FS) или ПХ М на первом уровне структуры хранения ПХ. Здесь возможны следующие варианты:

1. B > M;
2. B < M;
3. B = M.

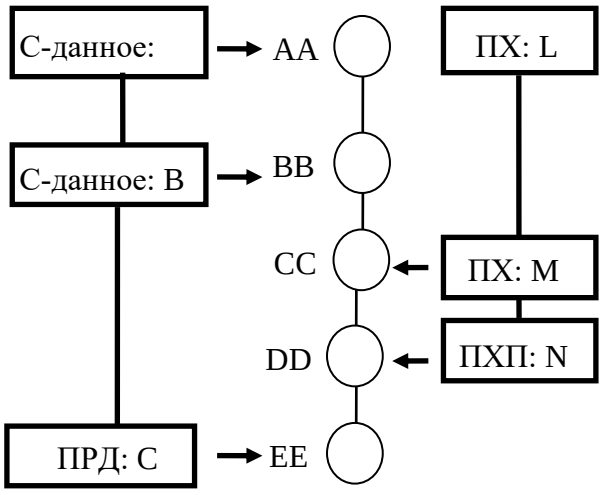

Рис. 5. Форматирование FS в многоуровневых структурах ПХ

В первом случае цикл будет выполняться до завершения обработки С-данного В. В примере имеет место именно эта ситуация. Построение DA продолжается построением очередного параграфа на уровне 2. Для этого сопоставляются очередные совокупности или ПРД. В данном случае сопоставляются ПРД С и ПХ М.

Во втором случае цикл будет выполняться до завершения обработки ПХ М. Построение DA продолжается построением очередного параграфа на уровне 2. Для этого сопоставляются очередные совокупности или ПРД. В данном случае сопоставляются С-данное В и ПХ N.

В третьем случае (Р-данное полностью форматируется в ПХ) цикл будет выполняться до завершения обработки С-данного В. Для построения очередного параграфа сопоставляются очередные совокупности или ПРД, которые до этого шага не сравнивались. В данном случае сопоставляются ПРД С и ПХП N.

При сравнении ПРД и ПХ (не ПХП), учитывается, что ПРД представляется как одна компонента. Соответственно, при сравнении С-данного и ПХП, учитывается, что ПХП представляется как одна компонента. Третий случай предполагает, что количество компонент в С-данном равно количеству компонент в ПХ или ПХ определяется количеством компонент С-данного.

На рис. 5 показано как построена DA для конкретного примера. Стрелки указывают на то, какой объект порождает соответствующий параграф.

Более сложная ситуация – это когда количество уровней как в дереве FS, так и в дереве структуры ПХ – произвольное. В этом случае DA формируется в результате последовательного перебора и сравнения объектов соотносящимися с узлами дерева FS с одной стороны и объектов соотносящимися с узлами дерева структуры ПХ, с другой стороны. Завершается перебор и сравнение в тот момент, когда сопоставляются ПРД с одной стороны и ПХП, с другой стороны. В результате сравнения порождается один или два параграфа в зависимости от соотношения значений ПРД и ПХП. Алгоритмы для возможных вариантов описаны выше.

Общий случай предполагает следующее. Количество уровней в деревьях FS и структуры ПХ может не совпадать. ПХ могут быть различных типов и размеров. Как дерево FS, так и дерево структуры ПХ могут иметь произвольное количество ветвей. ПХ могут быть сгруппированы на А-ленте в подобласти.

Может быть более одной структуры ПХ в которые форматируются FS.

Совмещение дерева FS и дерева структуры ПХ, порождает DA. Другими словами производится синтез дерева FS и дерева структуры ПХ или просто – производится синтез деревьев.

Выводы

Описаны порции хранения Р-данных как явление электронной обработки данных и предложено обобщенное понятие порции хранения. Описаны факторы, которые порождают необходимость форматирования и реформатирования Р-данных. Описана обобщенная картина форматирования Р-данных. Предложена группа алгоритмических конструкций, которые реализуют процедуры ФРФ Р-данных для некоторого уровня обобщения. Несоответствие размеров Р-данных и ПХ порождает дополнительные параграфы в программе (узлы в DA), что нарушает АКУ-обусловленность канонического алгоритма. Процесс внесения изменений в канонический алгоритм обусловленный результатами целенаправленных исследований.

1. *Колесник В.Г.* DS-теория как прототип теории прикладных алгоритмов // Проблеми програмування. – 2012. – № 1. – С. 17–33.
2. *Колесник В.Г.* DS-теория. Представление канонического алгоритма с помощью алгоритмического языка // Проблеми програмування. – 2015. – № 1. – С. 3–18.
3. *Колесник В.Г.* DS-теория. Исследование факторов деления Р-данных с целью генерации прикладных алгоритмов. Первая часть // Проблеми програмування. – 2015. – № 3. – С. 3–12 .
4. *Колесник В.Г.* DS-теория. Исследование факторов деления Р-данных с целью генерации прикладных алгоритмов. Вторая часть // Проблеми програмування. – 2015. – № 4. – С. 3–13.
5. *Mark Baker*. Algorithms in Structured Writing: Processing Structured Text [Electronic resource] — Mode access: http://techwhirl.com/16098-2/
6. *Основы* локальных сетей: Пакеты, протоколы и методы управления обменом. [Электронный ресурс] // Режим доступа: http://www.intuit.ru/studies/courses/57/57/lecture/1678?page=2
7. *Data* Translation. 1 EDI Source. [Электронный ресурс] // – Режим доступа: http://www.1edisource.com/what-is-edi-translation
8. *System* Analysis and Design. Input/Output and Form Design [Электронный ресурс] // Режим доступа: http://www.systemanalysisanddesigns.com/inputoutput-and-form-design/
9. Справочник CSS. WebReference.ru. [Электронный ресурс] // URL: https://webref.ru/css

References

1. *Kolesnyk V.G.* DS-theory as a prototype of the theory of applied algorithms // Problems in programming. – 2012. – № 1. – P. 17–33. (In Russian).
2. *Kolesnyk V.G*. DS-theory. Presentation of canonical algorithm by means of algorithmic language // Problems in programming. – 2015. – № 1. – P. 3–18. (In Russian).

3. *Kolesnyk V.G.* DS-theory. Research of r-data division factors in order to generate applied algorithms. Part 1 // Problems in programming. – 2015. – № 3. – P. 3–12. (In Russian).
4. *Kolesnyk V.G.* DS-theory. Research of r-data division factors in order to generate applied algorithms. Part 2 // Problems in programming. – 2015. – № 4. – P. 3–13. (In Russian).
5. Mark Baker. Algorithms in Structured Writing: Processing Structured Text [Electronic resource] — Mode access: http://techwhirl.com/16098-2/
6. Fundamentals of LANs: Packages, protocols and exchange management. [Electronic resource] — Mode access: http://www.intuit.ru/studies/courses/57/57/lecture/1678?page=2 (In Russian).
7. Data Translation. 1 EDI Source. [Electronic resource] // – Mode access: http://www.1edisource.com/what-is-edi-translation
8. System Analysis and Design. Input/Output and Form Design [Electronic resource] // Mode access: http://www.systemanalysisanddesigns.com/inputoutput-and-form-design/
9. CSS Reference. WebReference.ru. [Electronic resource] // URL: https://webref.ru/css (In Russian).

Получено 29.12.2015

Об авторе:

Колесник Валерий Георгиевич,
старший научный сотрудник
кафедры АПП.
Количество научных публикаций в
украинских изданиях – 24.
http://orcid.org/0000-0002-2313-9852.

Место работы автора:

Донбасская государственная
машиностроительная академия.
г. Краматорск, ул. Академическая, 72,
п/я 13.