УДК 681.3

И.Н. Парасюк, А.И. Провотар, В.А. Кондратенко

АКСИОМАТИЧЕСКИЕ МОДЕЛИ И МЕТОДЫ ПРОЕКТИРОВАНИЯ ЯЗЫКОВЫХ ТРАНСЛЯТОРОВ

Рассматриваются вопросы применения логических методов и методов теории формальных систем для построения языковых трансляторов. Предложен аналог метода резолюций для элементарных формальных систем.

Введение

Теория языковых трансляторов на сегодняшний день хорошо развита и нашла свое применение в многочисленных приложениях. Стержневой идеей ставших классическими методов построения трансляторов является описание языковых конструкций формальными грамматиками или автоматами различной сложности для решения проблем синтаксического анализа, перевода и компиляции. При этом соответствующие методы для каждой пары языков, вообще говоря, не будут универсальными, что делает их технологически трудоемкими (за исключением, быть может, случаев синтаксически управляемых трансляций).

Актуальность проблемы разработки технологически и функционально эффективных методов автоматизации программирования В.М. Глушков убедительно продемонстрировал еще в 1959 г. [1] при обсуждении новых принципов построения программирующих программ. Уместно отметить, что результаты этой работы были успешно применены в ряде проектов, в частности при построении семейства семантически управляемых автоматизированных систем обработки данных [2].

В настоящей статье предлагается подход к проектированию языковых трансляторов, который основан на логическом описании языков [3] и использовании универсальной составляющей различных этапов трансляции — методе резолюций [4]. В частности, представлен метод построения вывода из аксиом элементарных формальных систем (ЭФС) — аналог метода резолюций, формулами ЭФС описаны этапы синтаксического анализа и перевода. Применение подхода иллюстрируется на различных типах грамматик и синтаксически управляемых переводов, описанных формулами логики предикатов первого порядка [5].

Транслирующие грамматики

Транслирующей грамматикой или грамматикой перевода, как известно, называется контекстно-свободная грамматика, множество терминальных символов которой разбито на множество входных символов и множество символов действия. А цепочки языка, определяемого транслирующей грамматикой, называются последовательностями актов.

Процесс получения транслирующей грамматики продемонстрируем на простом модельном примере.

Допустим, нужно построить процессор, получающий в качестве входа инфиксное выражение и печатающий на выходе эквивалентное выражение в постфиксной польской записи. Кроме того, мы хотим, чтобы проектирование этого процессора основывалось на распознавателе, который каждый раз, когда распознан символ, вызывал процедуру печати и инициировал ее.

Грамматика G_0 для инфиксных выражений с начальным нетерминальным символом $<E>$ будет иметь следующий вид:

1. $<E> => <E> + <T>$.
2. $<E> => <T>$.
3. $<T> => <T>*<P>$.
4. $<T> => <P>$.
5. $<P> => (<E>)$.
6. $<P> => a$.
7. $<P> => b$.
8. $<P> => c$.

Для удобства изложенная грамматика содержит 3 конкретных имени констант a, b, c. Чтобы построить грамматику для последовательностей актов, опишем действия, соответствующие каждой правой части правил грамматики. Например, чтобы напечатать a после того, как оно прочитано, правило 6 заменяется на $P => a\{a\}$. Чтобы напечатать знак + после того, как напечатаны оба его операнда, правило 1 заменяется на $<E> => <E> + <T>\{+\}$. Новое правило 1 содержательно состоит в следующем: обработка нетерминала $<E>$ включает обработку нетерминала $<E>$, чтение операции +, обработку нетерминала $<T>$ и печать операции +. После аналогичных изменений в других правилах, новая грамматика примет следующий вид:

1. $<E> => <E> + <T>\{+\}$.
2. $<E> => <T>$.
3. $<T> => <T>*<P>\{*\}$.
4. $<T> => <P>$.
5. $<P> => (<E>)$.
6. $<P> => a\{a\}$.
7. $<P> => b\{b\}$.
8. $<P> => c\{c\}$.

Полученная новая грамматика и будет транслирующей грамматикой или грамматикой перевода.

Синтаксически управляемые переводы

Грамматику, транслирующую в цепочки, можно интерпретировать как способ описания перевода со входного языка на выходной. По сути, такие переводы управляются грамматикой и называются синтаксически управляемыми переводами (СУ-переводами) [6].

Пусть Σ и Δ — два алфавита и $T \subseteq \Sigma^* \times \Delta^*$ — произвольное отношение на паре языков $L \subseteq \Sigma^*$, $l \subseteq \Delta^*$ (L — область определения, а l — множество значений отношения).

Отношение $T \subseteq \Sigma^* \times \Delta^*$ называется отношением СУ-перевода, если оно формально представимо [7].

Формально представимое отношение $T \subseteq \Sigma^* \times \Delta^*$ на паре языков $L \subseteq \Sigma^*$, $l \subseteq \Delta^*$ называется отношением трансляции, если область определения его разрешима.

Рассмотрим схему перевода, отображающую арифметические выражения языка $L(G_0)$ в постфиксные польские записи. Такая схема представляется правилами грамматики G_0 и соответствующими элементами перевода G_1 как показано ниже:

1. $<E> => <E> + <T>$;	$<E> => <E><T>+$.
2. $<E> => <T>$;	$<E> => <T>$.
3. $<T> => <T>*<P>$;	$<T> => <T><P>*$.
4. $<T> => <P>$;	$<T> => <P>$.
5. $<P> => (<E>)$;	$<P> => <E>$.
6. $<P> => a$;	$<P> => a$.
7. $<P> => b$;	$<P> => b$.
8. $<P> => c$;	$<P> => c$.

Схемы определяют отношения между цепочками, порождаемыми грамматиками G_0 и G_1 соответственно. Для этого необходимо найти вывод цепочки в грамматике G_0 и сопоставить его с выводом в грамматике перевода G_1. Например, цепочке $a+b*c$ будет соответствовать цепочка $abc*+$, так как существует вывод пары $(a+b*c, abc*+)$ вида

$$(<E>, <E>) \Rightarrow (<E>+<T>, <E><T>+) \Rightarrow$$
$$\Rightarrow (<T>+<T>, <T><T>+) \Rightarrow$$
$$\Rightarrow (<P>+<T>, <P><T>+) \Rightarrow$$
$$\Rightarrow (a+<T>, a<T>+) \Rightarrow$$
$$\Rightarrow (a+<T>*<P>, a<T><P>*+) \Rightarrow$$
$$\Rightarrow (a+<P>*<P>, a<P><P>*+) \Rightarrow$$
$$\Rightarrow (a+b*<P>, ab<P>*+) \Rightarrow (a+b*c, abc*+).$$

Схемой СУ-перевода есть пятерка $T = (N, \Sigma, \Delta, R, S)$,
где N — множество нетерминальных символов;

Σ, Δ — соответственно входной и выходной алфавиты;

R — конечное множество правил вида $A \to \alpha$, β, где $\alpha \in (N \cup \Sigma)^*$, $\beta \in (N \cup \cup \Delta)^*$ и вхождения нетерминалов в цепочку β образуют перестановку вхождений нетерминалов в цепочку α;

S — начальный нетерминал из N.

Переводом, определяемым схемой T, служит множество пар $\{(x, y) \mid (S,S) \Rightarrow^* (x, y), x \in \Sigma^* \text{ и } y \in \Delta^*\}$.

Схема СУ-перевода будет простой, если для каждого правила $A \to \alpha, \beta$ из R соответствующие друг другу вхождения нетерминалов встречаются в одном и том же порядке.

Переводы, определяемые простыми СУ-схемами, образуют важный класс переводов, потому что их технологически легко реализовать преобразователями с магазинной памятью. В частности, такие преобразователи получаются из автоматов с магазинной памятью, если их снабдить выходом и разрешить на каждом такте выдавать выходную цепочку конечной длины.

Легко показать, что справедлива

Теорема 1. СУ-перевод T простой тогда и только тогда, когда $T = \tau(P)$ для некоторого МП-преобразователя P.

Таким образом, построение транслятора, реализующего перевод, сводится к программному моделированию преобразователя с магазинной памятью.

Более совершенны атрибутные транслирующие грамматики [8], иначе называемые контекстно-зависимыми. Они сохраняют в неизменном виде все преимущества КС-грамматик и в то же время снабжены специальными средствами описания нетерминальных символов и символов действия в этих грамматиках, с помощью которых эти компоненты обеспечиваются необходимыми сведениями для корректной трансляции языков программирования. В качестве специальных средств описания выступают атрибуты — синтезируемые и наследуемые [6]. Т.е. в АТГ существует возможность сопровождать каждый из нетерминальных символов и символов действия неограниченным количеством атрибутов, выражающих те или иные свойства терминальных символов, которые подставляются вместо указанных нетерминалов. Именно эти атрибуты обеспечивают возможность порождения корректных фраз (в синтаксическом смысле) в естественных языках, а также возможность обеспечения операндов необходимыми адресами в языках программирования.

Резолютивный вывод для реализации синтаксически управляемых переводов

Суть этого подхода состоит в следующем. После того как правила транслирующей грамматики преобразованы в правила логики предикатов, необходимо ввести дополнительные, обусловливающие порядок (стратегию) применения резолютивного вывода для доказательства истинности полученной формулы. Под порядком применения резолютивного вывода подразумевается очередность расположения дизъюнктов, принадлежащих логической модели, в итоговом списке дизъюнктов S этой модели. От этой очередности зависит эффективность доказательства теоремы. Исследования этой проблемы показали, что наиболее эффективным доказательство становится в случае, когда порядок резолютивного вывода совпадает с порядком вывода исследуемой цепочки языка при ее распознавании в определенной грамматике. Иными словами, когда осуществляется распознавание только с помощью левого посимвольного вывода заданной цепочки в этой грамматике путем подстановок начиная с символа S и подстановок, соответствующих правилам рассматриваемой грамматики, вплоть до получения целевой цепочки.

Реализация универсального транслятора в любой Prolog-системе [6] осуществляется путем построения систем логических аксиом, соответствующих правилам грамматик входного и выходного языков, и последующим преобразованием аксиом — приведением их к виду, удобному для применения правила резолюций.

Покажем, что справедлива следующая теорема.

Теорема 2. Для каждой грамматики G существует формальная система (E), содержащая предикат, представляющий язык $L(G)$.

Доказательство. Проведем доказательство для случая регулярных и КС-грамматик. Пусть $G = (K, V, \sigma, P)$ — регулярная грамматика. Каждому нетерминалу $\theta \in V$ грамматики G ставим в соответствие предикатный символ θ' так, что различным нетерминальным символам алфавита V соответствуют различные предикатные символы. Пусть $V' = \{\theta' / \theta \in V\}$ — полученное множество предикатных символов. Построим формальную систему (E), аксиомы которой строятся по продукциям грамматики G следующим образом: продукцию вида $\theta \to a$ заменяем на аксиому $\theta' a$, продукцию $\theta \to \varphi a$ заменяем аксиомой $\varphi' x \to \theta' ax$, продукцию $\theta \to \theta a$ — аксиомой $\theta' x \to \theta' ax$, где φ, θ — произвольные нетерминальные символы; a — терминальный символ грамматики G; x — переменная.

Пусть z — произвольная цепочка, выводимая в G. Полный вывод цепочки z можно представить в виде

$$\sigma \Rightarrow \psi_1 t_1 \Rightarrow \psi_2 t_2 \Rightarrow \ldots \Rightarrow \psi_n t_n \Rightarrow z,$$

где ψ_i — нетерминальные символы; t_i — цепочки терминальных символов. Этому выводу соответствует последовательность продукций грамматики G

$$\sigma \to \psi_1 v_1,\ \psi_1 \to \psi_2 v_2,\ \ldots,\ \psi_{n-1} \to \\ \to \psi_n v_n,\ \psi_n \to v_{n+1},$$

причем $t_1 = v_1$, $t_2 = v_1 v_2$ ... $t_n = v_1 v_2 \ldots v_n$, $z = v_1 v_2 \ldots v_{n+1}$. Заменив каждую продукцию этой последовательности согласно указанным выше преобразованиям, получим последовательность аксиом системы (E)

$$\psi_1' x \to \sigma' v_1 x,\ \psi_2' x \to \psi_1' v_2 x,\ \ldots,\ \psi_n' x \to \\ \to \psi_{n-1}' v_n x,\ \psi_n' v_{n+1}.$$

Тогда полный вывод цепочки z в (E) можно представить в виде

$$\psi_n' v_{n+1} \equiv> \psi_{n-1} v_n v_{n+1} \equiv> \ldots \equiv> \psi_2' v_3 \ldots v_{n+1} \equiv> \\ \equiv> \psi_1' v_2 \ldots v_{n+1} \equiv> \sigma' v_1 \ldots v_{n+1}.$$

Таким образом, предикат σ' представляет язык $L(G)$.

Пусть $G = (K, V, \sigma, P)$ — произвольная КС-грамматика. Преобразуем ее к виду в нормальной форме Хомского. Тогда продукции грамматики будут иметь один из следующих видов: $\theta \to \varphi\psi$, $\theta \to a$.

Как и в случае регулярных грамматик, каждому нетерминалу θ сопоставим различные предикатные символы. Пусть $V' = \{\theta' / \theta \in V\}$ — полученное множество предикатных символов. Построим формальную систему (E), схема P' которой строится по схеме P грамматики G следующим образом: продукцию вида $\theta \to a$ заменяем аксиомой $\theta' a$, продукцию $\theta \to \varphi\psi$ — аксиомой $\varphi' x \to \psi' y \to \theta' xy$, где φ, θ, ψ — произвольные нетерминальные символы; a — терминальный символ грамматики G; x, y — переменные.

Пусть z — произвольная цепочка, выводимая в G. Полный вывод цепочки z можно представить в виде

$$\sigma \Rightarrow \varphi_1\varphi_2 \Rightarrow \varphi_1\psi_2\psi_3 \Rightarrow \ldots \Rightarrow \varphi_1\psi_2 \ldots \theta_n \Rightarrow \\ \Rightarrow \varphi_1\psi_2 \ldots t_n \Rightarrow \ldots \Rightarrow t_1 t_2 \ldots t_n = z,$$

где φ_i, ψ_i, ..., θ_i — нетерминальные символы, t_i — терминальные символы. Этому выводу соответствует последовательность продукций грамматики G

$$\sigma \to \varphi_1\varphi_2,\ \varphi_2 \to \psi_2\psi_3,\ \ldots,\ \gamma_{n-1} \to \theta_{n-1}\theta_n,\ \theta_n \to \\ \to t_n,\ \theta_{n-1} \to t_{n-1},\ \ldots, \varphi_1 \to t_1.$$

Заменив каждую продукцию этой последовательности согласно указанным выше преобразованиям, получим последовательность аксиом системы (E)

$$\varphi_1' \to \varphi_2' \to \sigma' \mathrm{xy},\ \psi_2' \mathrm{x} \to \psi_3' \mathrm{y} \to \varphi_2' \mathrm{xy},\ \ldots,$$

$$\theta_{n-1}' \mathrm{x} \to \theta_n' \mathrm{y} \to \gamma_{n-1}' xy,\ \theta_n' t_n,\ \ldots,\ \varphi_1' t_1.$$

Тогда полный вывод цепочки z в системе (E) можно представить в виде

$$\theta_n' t_n \equiv> \gamma_{n-1}' \equiv> \ldots \equiv> \varphi_2' t_2 \ldots t_n \equiv> \sigma' t_1 t_2 \ldots t_n.$$

Таким образом, предикат σ' представляет язык $L(G)$. Теорема доказана.

На основании доказанной теоремы можно произвольную грамматику преобразовать в систему логических аксиом. Это, в частности, относится и к СУ-переводам — формальным моделям трансляторов, определяемых парой грамматик — входного языка и перевода.

Например, КС-грамматика с системой продукций $\sigma \to ab$, $\sigma \to a\sigma b$ может быть преобразована в формальную систему с аксиомами $\sigma' ab$, $\sigma' x \to \sigma' axb$. СУ-перевод, реализующий отношение между арифметическими выражениями в алфавите $\{a, b, c\}$ и их постфиксными записями может быть преобразован в формальную систему (E), содержащую логические аксиомы вида

1. Pa,a.
2. Pb,b.
3. Pc,c.
4. $Ex,y \to Px,(y)$.
5. $Px,y \to Tx,y$.
6. $Tx,y \to Pt,z \to Tx{*}t, xz{*}$.
7. $Tx,y \to Ex,y$.
8. $Ex,y \to Tt,z \to Ex{+}t, yz{+}$.

Покажем, что в этой формальной системе существует вывод пары ($a{+}b{*}c$, $abc{*}{+}$). Действительно, имеем

9. Tb, b (правило заключения 2, 5).
10. $Tb,b \to Pc,c \to Tb{*}c,bc{*}$ (аксиома 6).
11. $Pc,c \to Tb{*}c, bc{*}$ (правило заключения 9, 10).
12. $Tb{*}c,bc{*}$ (правило заключения 3, 11).
13. Ta,a (правило заключения 1, 5).
14. Ea,a (правило заключения 7, 13).
15. $Ea,a \to Tb{*}c, bc{*} \to Ea{+}b{*}c, abc{*}{+}$ (аксиома 8).
16. $Tb{*}c,bc{*} \to Ea{+}b{*}c, abc{*}{+}$ (правило заключения 14, 15).
17. $Ea{+}b{*}c, abc{*}{+}$ (правило заключения 12, 16).

Далее осуществляется преобразование аксиом формальных систем в множество дизъюнктов по правилу $\alpha \to \beta = \neg\alpha \vee \beta$. Преобразованная система (E) будет иметь вид

1. Pa,a.
2. Pb,b.
3. Pc,c.
4. $\neg Ex,y \vee Px,(y)$.
5. $\neg Px,y \vee Tx,y$.
6. $\neg Tx,y \vee \neg Pt,z \vee Tx{*}t, xz{*}$.
7. $\neg Tx,y \vee Ex,y$.
8. $\neg Ex,y \vee \neg Tt,z \vee Ex{+}t, yz{+}$.

Если аксиомы системы представлены в таком виде, то для доказательства выводимости пар цепочек входного и выходного языков можно применять метод резолюций. Покажем, например, что в полученной формальной системе существует вывод пары ($a{+}b{*}c$, $abc{*}{+}$). Для этого добавим к имеющемуся множеству дизъюнктов

9. $\neg\, E\, a{+}b{*}c, abc{*}{+}$

и построим следующий вывод

10. $\neg Pb,b \vee Tb,b$ (аксиома 5).
11. Tb,b (правило резолюций 2, 10).
12. $\neg Tb,b \vee \neg Pc,c \vee Tb{*}c, bc{*}$ (аксиома 6).
13. $\neg Pc,c \vee Tb{*}c, bc{*}$ (правило резолюций 11, 12).
14. $Tb{*}c, bc{*}$ (правило резолюций 3, 13).
15. $\neg Pa,a \vee Ta,a$ (аксиома 5).
16. Ta,a (правило резолюций 1, 15).
17. $\neg Ta,a \vee Ea,a$ (аксиома 7).
18. Ea,a (правило резолюций 16, 17).
19. $\neg Ea,a \vee \neg Tb{*}c,bc{*} \vee Ea{+}b{*}c, abc{*}{+}$ (аксиома 8).
20. $\neg Tb{*}c,bc{*} \vee Ea{+}b{*}c, abc{*}{+}$ (правило резолюций 18, 19).
21. $Ea{+}b{*}c, abc{*}{+}$ (правило резолюций 14, 20).
22. $\square$ (правило резолюций 9, 21).

Вывод пустого дизъюнкта означает, что пара ($a{+}b{*}c$, $abc{*}{+}$) принадлежит СУ-переводу.

Пусть Σ и Δ — два алфавита. Гомоморфизмом называется любое отображение $h: \Sigma \to \Delta^*$.

Язык L характеризует отношение T, если существуют такие два гомоморфизма h_1 и h_2, что $T = \{(h_1(x), h_2(x)) / x \in L\}$.

Теорема 3. Если отношение T характеризуется языком L, то T — СУ-перевод.

Доказательство. Отношение гомоморфизма формально представимо. Это следует из аксиом

Ee, e;

$Ea_1, h(a_1)$;

$Ea_2, h(a_2)$;

. . . .

$Ea_n, h(a_n)$.

Кроме того, $T = \{(h_1(x), h_2(x)) / x \in L\}$ и отношения $y_1 = h_1(x)$, $y_2 = h_2(x)$ представимы предикатами H_1 и H_2 соответственно в формальных системах (E_1) и (E_2). Добавив к аксиомам этих систем

$$Lx \to H_1x,y_1 \to H_2x,y_2 \to Ty_1,y_2,$$

получим формальную систему, в которой предикатом T представляется отношение T. Теорема доказана.

Пусть K — алфавит из m символов $a_1, a_2, ..., a_m$; T — СУ-перевод на паре языков L, l.

Теорема 4. Если L и l — разрешимые языки над алфавитом K, то СУ-перевод T разрешим.

Доказательство. Пусть T — предикат, представляющий отношение T; L' и l' — предикаты, представляющие дополнения языков L и l соответственно; L и l — предикаты, представляющие собственно языки.

Отношение "x непосредственно следует за y" в лексикографическом упорядочении представляется посредством предиката S в формальной системе, аксиомами которой являются

1. $S\alpha a_1, \alpha a_2$.
2. $S\alpha a_2, \alpha a_3$.

. . . .

3. $S\alpha a_{m-1}, \alpha a_m$.
4. Sa_m, a_1a_1.
5. $Sx,y \to Sxa_m,ya_1$.

Добавим к этим аксиомам следующие:

6. $Sx,y \to Qy,x$.
7. $Qx,y \to Qy,z \to Qx,z$.

Тогда Q представляет отношение "$x < y$". Добавим

8. $Qx,y \to Dx,y$.
9. $Qy,x \to Dx,y$.

Тогда предикат D представляет отношение "$x \neq y$". Добавим к этим аксиомам следующие

10. $l'y \to T'x,y$.
11. $L'x \to T'x,y$.
12. $Lx \to ly \to Tx,z \to Dy,z \to T'x,y$.

Получим формальную систему, в которой предикат T' представляет отношение T'. Теорема доказана.

Доказанные теоремы — обоснование примененимости метода резолюций для построения лингвистических трансляторов.

Метод резолюций в случае логики предикатов, вообще говоря, не заканчивает свою работу. Поэтому на входной и выходной языки должны быть наложены некоторые ограничения, а именно: разрешимость входного языка, которая обеспечивает завершенность фазы синтаксического анализа, и формальная представимость отношения на языках, которая обеспечивает завершенность фазы перевода.

Эти условия выполняются в случае разрешимости отношения СУ-перевода. Но алгоритм трансляции, с учетом дополнительных условий, может быть описан другими предикатами в системах логического программирования.

Выводы

Показана возможность применения логических формализмов различных типов для реализации синтаксических анализаторов и перевода. Описанные результаты, по мнению авторов, — это лишь первые шаги в теоретических исследованиях на пути применения аксиоматического похода к построению языковых трансляторов. Основную роботу — формирование соответствующих методологических и технологических основ новой информационной технологии — еще предстоит сделать. Это направление исследований актуальное, перспективное и многообещающее.

1. *Глушков В.М.* Об одном методе автоматизации программирования // Пробл. кибернетики. — №2. — С.181—184.
2. *Парасюк И.Н., Сергиенко И.В.* Пакеты программ анализа данных: технология разработки. — М: Финансы и статистика, 1988. — 160 с.
3. *Bratko I.* Prolog Programming for Artificial Intelligence: Addison Wesley, M.A., 1986. — 560 с.
4. *Чень Ч., Ли Р.* Математическая логика и автоматическое доказательство теорем. — М: Наука, 1983. — 360 с.
5. *Тей А., Грибомон П., Яцы Ж.* Логический подход к искусственному интеллекту — М: Мир, 1990. — 430 с.
6. *Ахо А., Ульман Дж.* Теория синтаксического анализа, перевода и компиляции. — М: Мир, 1978. — Т.1. — 612с.
7. *Р. Смальян.* Теорія формальних систем. — М: Наука, 1981. — 208с.
8. *Семантика* языков программирования / Под ред. В.М. Курочкина. — М: Мир, 1980. — 396 с.
9. *Lewis P.M., Rozenkrantz D.J., Stearns R.E.* Attributed Translation. General Electric Company, Research and Development Center, New York 12345, Received August 30, 1973 // J. of Computer and System Sciences. 1974. — **9**, N.3, Dec. — 142 p.

Получено 04.09.03

Об авторах

Парасюк Иван Николаевич,
член-корреспондент НАНУ, доктор техн. наук, профессор, заведующий отделом

Провотар Александр Иванович,
доктор физ.-мат. наук, профессор, заведующий кафедрой НУ им. Т. Шевченко.

Кондратенко Виктория Александровна,
канд. физ.-мат. наук, ведущий инженер-программист

Место работы авторов:
Институт кибернетики им. В.М. Глушкова НАН Украины,
просп. Академика Глушкова, 40,
Киев-187, 03680, Украина
Тел. 266 0458