Рада Міністрів УРСР
АКАДЕМІЯ НАУК УРСР
ІНСТИТУТ КІБЕРНЕТИКИ
м. Київ, Лисогірська, 4
телеграфн. „Імпульс"

Совет Министров УССР
АКАДЕМИЯ НАУК УССР
ИНСТИТУТ КИБЕРНЕТИКИ
г. Киев, Лысогорская, 4
телеграфн. „Импульс"

тел. ком. 7-43-01, гор. 7-41-42

57/2125

„8" июня 1964 г.

ДИРЕКТОРУ ИНСТИТУТА МАТЕМАТИКИ
СО АН С С С Р
академику С.Л.СОБОЛЕВУ

Институт Кибернетики АН УССР направляет Вам доклад зав.отделом автоматизации программирования, кандидата физико-математических наук Е.Л.Ющенко (Рвачевой) на тему "Некоторые вопросы теории алгоритмических языков и автоматизации программирования" в связи с предполагаемой защитой диссертации на соискание звания доктора физико-математических наук по совокупности выполненных и опубликованных работ.

Просим Вас, как передовое предприятие, дать по докладу свое заключение. Предварительное согласие с зав.отделом проф. А.А.Ляпуновым и Ю.И.Журавлевым у нас имеется.

ДИРЕКТОР ИК АН УССР
академик АН УССР (В.М.ГЛУШКОВ)

Вх. № 1175
15 VI 1964 г.
Институт математики
СО АН СССР

Київ, 41, Голосієво, друкарня вид-ва УАСГН, 3-й уч. корпус, зам. 149—10000—62 р

17065

V37

Е.Л.Ющенко

Некоторые вопросы теории алгоритмических языков и автоматизации программирования

Доклад по выполненным и опубликованным работам, представляемым к защите вместо докторской диссертации

Киев, 1964

Работа выполнена в Институте
Кибернетики АН УССР

В В Е Д Е Н И Е

В последние годы все с большим успехом применяются электронные вычислительные машины (ЭВМ) в сложных научных исследованиях, в системах обработки данных, для анализа и прогнозирования различных областей знаний, автоматизации производственных процессов, моделирования мыслительных процессов и др. Основная трудность, с которой приходится сталкиваться при этом, состоит, как правило, в недостаточной формализации постановки задач, с одной стороны, и формализации процессов их решения, с другой.

Для описания алгоритмов в принципе могут быть использованы алгоритмические системы, развитые в основном в связи с вопросами оснований математики. Однако, языки классических алгоритмических систем для описания в них алгоритмов, встречающихся на практике, оказались фактически неприемлемыми.

В связи с этой с появлением ЭВМ возник новый прикладной раздел современной теории алгоритмов, называемый программированием. В задачу программирования входит разработка методов и приемов точного описания алгоритмов, реализуемых этими машинами, и прежде всего, алгоритмических языков и основанных на них систем автоматизации программирования.

Работы в этом направлении ведутся в СССР, начиная с 1952 года, главным образом, под руководством проф. А.А.Ляпунова, чл.-корр. АН СССР Л.В.Канторовича[1], проф. М.Р.Шура-Бура, канд. физ.-мат.наук А.П.Ершова и др.

Предложенный А.А.Ляпуновым операторный метод программирования получил широкое распространение в СССР и был положен в основу построения ряда систем автоматизации программирования[2]. Этот же метод, связанный с алгебраическим подходом к описанию программ по существу нашел развитие в многочисленных работах за рубежем[3].

Представленные к защите работы являются результатами исследований, выполненных в связи с разработкой вопросов теории алгоритмических языков и автоматизации программирования. В идейном плане эти работы также примыкают к исследованиям А.А.Ляпунова. Наибольшее внимание в них уделено разработке алгоритмического языка, названного адресным, методам составления трансляторов и их созданию для ряда отечественных машин (Урал, Киев, М-20, Днепр, Минск и др.).

---

1) Канторович Л.В., О проведении численных и аналитических вычислений на машинах с программным управлением, Изд.АН АРМ.ССР, № 2, 1957;

2) Камынин С.С., Любимский Э.З., Шура-Бура М.Р., Об автоматизации программирования при помощи программирующей программы, сб. "Проблемы кибернетики", вып. I, Физматгиз, 1958;
Ершов А.П., Программирующая программа для БЭСМ, Изд.АН СССР, 1958;
Жоголев Е.А., Система команд и интерпретирующая система для машины "Сетунь", Журнал вычисл.матем. и матем.физ., т.I, № 3, Изд.АН СССР, 1961, Система автоматизации программирования, под. ред. Н.П.Трифонова и М.Р.Шура-Бура, "Библиотека прикладного анализа и вычислительной математики" Физматгиз 1961 и др.

3) Автоматизация программирования, Сб.переводов под ред.А.П. Ершова, Физматгиз, 1961; Сообщение об алгоритмическом языке Алгол, Изд. АН СССР, 1960 и др.

Идея построения адресного языка изложена в работах В.С.Королюка[I)] и развита в [2], [7], [10] , а его разработка и внедрение в практику программирования осуществлены автором доклада и ее учениками ([8] , [26], [30], [31], [35], [39] и др.).

При создании адресного языка был использован опыт алгоритмизации многих задач вычислительного и логико-информационного характера, а также опыт конструирования ЭВМ. С другой стороны, работы по созданию адресного языка оказали свое влияние при выборе операций для новых конструируемых машин, в частности вычислительной машины "Киев", одним из руководителей разработки которой является автор доклада (см. [6], [9], [14], [15], [23]).

Адресный язык представляет собой практический универсальный алгоритмический язык, удобный для описания сложных арифметических и информационно-логических алгоритмов, что проверено опытом описания большого числа сложных алгоритмов в этом языке.

Благодаря использованию принципа адресности в его современном понимании, в адресном языке естественно описываются, с одной стороны, электронные вычислительные машины и, с другой, программирующие программы (ПП) для них (см. [8], [23], [26], [31]). Тем самым существенно упрощается задача построения ПП в машинных кодах.

I)В.С.Королюк,Про один спосіб програмування, ДАН УССР,№I2, I958;
Проблемы кибернетики, вып. 4, I960.

Разработке методов составления трансляторов и их созданию посвящены работы [3], [11], [12], [16] – [21], [23], [26] – [31], [38], [39].

Возможность описания алгоритмов ПП в адресном языке, являющемся для них же внешним языком, открывает естественный путь для решения актуальной проблемы автоматизации процесса составления самих ПП.

Рассмотрению этого вопроса уделено внимание в работах [26], [31], [38].

Основные результаты автора изложены в представляемых к защите работах (см. Приложение), и прежде всего в книге "Адресное программирование", Гостехиздат УССР, 1963, а также в книгах "Адресное программирование и особенности решения задач на машине "Урал", КВИРТУ, "Вычислительная машина "Киев" (соавтор В.М.Глушков); Техиздат УССР, 1962; „Управляющая машина широкого назначения и программирующая программа к ней", Издательство "Наукова думка", 1964, (группа авторов); "Элементы программирования", Физматгиз, 1961, II издание - 1963 год (соавторы - Б.В.Гнеденко, В.С.Королюк); "Программирующая программа для машины -Урал-1", Издательство "Наукова думка" 1964 (соавтор Т.А.Гринченко).

Ниже излагается краткое описание выполненных и опубликованных работ.

Представленные работы докладывались на научных конференциях [illegible] по Вычислительной математике и технике ВЦ АН УССР (Киев, 1958 и 1960 г.); на Всесоюзном совещании по вычислительной математике (Москва, 1959 г.);

на IУ Всесоюном математическом съезде (Ленинград, I96Iг.); на Международном коллоквиуме по основаниям математики, математической логике и программированию (Венгрия, Сегет, I962г); на П Всесоюзной конференции по автоматизации переработки информации (Москва, I963); на Конференции стран социалистического лагеря по автоматизации программирования и алгоритмическим языкам (Киев, I963); на совещании пользователей машин М-20 (Ленинград, I964) и др.

I. АДРЕСНЫЙ ЯЗЫК

Адресный язык весьма близок к обычному языку математических формул. В нем приняты символы, которыми в математике обозначают величины, векторы, функции и др. Кроме того в нем вводятся специальные понятия и соответствующие символы, благодаря которым язык становится более приспособленным для описания алгоритмов.

Основные понятия и средства адресного языка

Одним из основных понятий адресного языка является штрих-операция, задающая отображение некоторого множества A, называемого множеством адресов, в множество B, называемое множеством содержимых этих адресов. Указанное отображение называют адресным. Символ операции $'$ (штрих) ставится сверху слева от аргумента:

$$'a = b,$$

где

a - аргумент ($a \in A$);

b - результат операции ($b \in B$).

В абстрактных рассмотрениях множества A и B могут иметь любую (не обязательно числовую) природу, т.е. функция $'$ (штрих) может быть определена на произвольном множестве аргументов A и иметь значения в произвольном конечном множестве B.

Повторное применение операции $'$ (штрих) приводит к понятию адреса второго ранга. Так, если $'a = b$ и $b \in \mathcal{A}$, т.е. оно, в свою очередь, является адресом, то имеется такое c, что

$$'b = c.$$

Это символически записывается так:

$$''a = {}^2a = 'b = c.$$

В этом случае a называется адресом второго ранга (или адресом адреса или фиксатором) c .

Аналогично вводится понятие адресов более высоких рангов.

Для упорядочения элементов в множествах вводятся операции следования. В каждом конкретном случае операция следования должна, вообще говоря, описываться алгоритмически.

Для определения и изменения адресного отображения применяется операция засылки, символ которой " $\Rightarrow$ ". Запись операции

$$a \Rightarrow b$$

(читается так: " a заслать по адресу b ") означает, что:

I) элемент b включается в множество адресов $\mathcal{A}$;

2) элемент a включается в множество содержимых $\mathcal{B}$;

3) устанавливается соответствие $'b = a$;

4) все ранее установленные соответствия вида $'x = y$, где $x \neq y$, остаются неизменными.

Справа и слева от знака $\Rightarrow$ в формуле может стоять адрес любого ранга ${}^n a$.

В этом случае выполнение операции засылки состоит в том, что извлекается содержимое (слева и справа от знака $\Rightarrow$) до тех пор, пока над элементами уже не будет стоять знак штрих и, наконец, производится соответствующая засылка.

Для указания порядка выполнения формул некоторые из них могут отмечаться метками с последующим троеточием.

Отдельное действие может описываться меткой без последующего троеточия. В этом случае она называется меткой безусловного перехода. При реализации алгоритма выполнение такого действия состоит в переходе к формуле, помеченной той же меткой.

Отдельную формулу алгоритма может составлять и адрес любого ранга (в этом случае содержимое его по данному рангу является меткой). Выполнение такой формулы состоит в извлечении содержимого по данному рангу и в переходе к формуле алгоритма, помеченной меткой, определяющейся этим содержимым.

Формула останова обозначается символом „!“ и может составлять отдельную строку алгоритма.

Выполнение формулы останова означает окончание алгоритмического процесса.

Перечисленные средства адресного языка, при наличии бесконечной памяти, достаточны для записи любого алгоритма, который может быть записан какими-либо другими алгоритмическими средствами. Оказывается даже, что достаточно употреблять операцию извлечения содержимого из адреса не выше второго ранга.

Допустимые формулы языка и определение адресного алгоритма

Адресный язык предназначен для алгоритмизации задач и программирования на вычислительных машинах. Поэтому, несмотря на полноту его основных средств, для удобства записи в нем алгоритмов и их обозримости в язык введены понятия адресов любого ранга и дополнительные выразительные средства.

Общепринятое понятие функции и соответствующего ей выражения, построенного из символов одноместных и двуместных опе-

раций, скобок и величин в адресном языке расширено путем включения в качестве одной из допустимых элементарных операций штрих-функции. Такие функции называются адресными. С помощью адресных функций записываются другие допустимые формулы языка: вычисляемого и относительного перехода, засылки, замены, вхождения подпрограмм, предикатные и циклирования.

Изменение адресного отображения осуществляется с помощью формул засылки и обмена. С помощью формул циклирования можно описывать циклические процессы переработки информации, а с помощью формул вхождения выделять отдельные вычисления в подпрограммы; предикатные формулы, формулы относительного и вычисляемого перехода дают возможность осуществлять в алгоритмах произвольные разветвления.

Запись адресного алгоритма состоит из исходного адресного отображения, конечного числа формул и указания множества адресов результативного отображения.

Начальное адресное отображение задается в виде последовательностей равенств

$$'a = b,$$

где a - адрес; b - содержимое.

Результативное адресное отображение задается перечислением адресов, содержимые которых после окончания работы алгоритма представляют решение задачи. Порядок выполнения алгоритмов определяется следующими правилами:

I. Первой выполняется строка, помеченная специально указанной начальной меткой. Если такого указания нет, то пер-

вым выполняется действие первое в записи.

2. Формулы вычисляемого перехода, списки формул с переходом, формулы вхождения и циклирования, а также предикатные формулы, значения которых содержат формулы перехода, указывают себе строку-преемницу.

3. После выполнения строки, не указывающей своей приемницы, выполняется следующая в записи строка.

4. Формула останова означает конец работы алгоритма.

Содержательно адресный алгоритм представляет собой некоторую переработку информации о задаче, в результате которой получается ее решение. Эта переработка имеет форму установления и изменения некоторого адресного отображения.

Информация о задаче задается некоторым адресным отображением. Поэтому во многих случаях алгоритм начинается установлением некоторого исходного адресного отображения посредством засылок или же таблицей отображения.

Результатом работы алгоритма является содержимое адресов, указанных в качестве результативного множества.

Алгоритмическая полнота адресного языка

Вопрос об алгоритмической полноте адресного языка впервые в 1/ рассмотрен В.С.Королюком 2/. В дальнейшем в работе автора доклада [22] (см.также [26]) этот результат был обобщен для расширенного адресного языка.

Выясняется, что для описания произвольной машины Тьюринга, т.е. для построения любого мыслимого алгоритма, достаточны лишь немногие из средств адресного языка:

1/ В.С.Королюк, "Про один спосіб програмування", ДАН УССР, № 12, 1958

2) В.С.Королюк, " О понятии адресного алгоритма", Проблемы кибернетики, вып. 4, 1960.

I) засылка по адресу $a \Rightarrow b$;

2) засылка по адресу, являющемуся содержимым другого адреса $a \Rightarrow {}'b$;

3) безусловный переход по формуле, содержащей адрес второго ранга, ${}^2a \circ {}'b$;

4) сдвиг фиксатора ${}'a + {}'\delta \Rightarrow a$;

5) остановка.

Следует особо подчеркнуть одно из основных свойств адресного языка, заключающееся в удобстве написания в нем произвольных обобщенных алгоритмов, т.е. алгоритмов, решающих неограничено широкий класс задач. При этом вся информация не только о конкретных условиях задачи; но и о классе задач естественным образом может быть отнесена к исходному адресному отображению, т.е. включена в исходную информацию о задаче. Алгоритм же может быть записан общий, не зависимый от решаемой задачи.

На практике главная трудность решения конкретной задачи переносится при этом на заполнение начального массива памяти, но это не умаляет значения сделанного замечания. Адресный язык допускает перенесение любого объема исходной информации в адресное отображение с целью упрощения и обобщения записи алгоритма.

Это свойство адресного языка позволяет, при пользовании им, абстрагироваться от конкретных особенностей алгоритмов и исследовать общие проблемы теории алгоритмов.

Классификация алгоритмических языков по принципу адресности

К алгоритмическим языкам предъявляется ряд требований, среди которых наряду с алгоритмической полнотой существенное значение имеет простота достижения общности записи алгоритмов в этих языках.

При построении алгоритмического языка в качестве исходного часто берется алгебраический язык формул. Для указания хода преобразования информации вводятся специальные операторы. Однако в алгоритмических языках в отличие от исходного алгебраического языка уточняется понятие памяти посредством введения принципа адресности.

В настоящее время, в связи с развитием методов автоматизации программирования, появился ряд алгоритмических языков. Эти языки в различной степени не зависят от конкретных языков ЭВМ, отличаются способами записи, набором операций и операторов и имеют различную направленность: для описания вычислительных процессов, преобразования буквенной информации, описания производственно-экономических задач и т.п.

Несмотря на все разнообразие создающихся языков их классификацию можно осуществить по степени использования в них принципа адресности [26], [35].

Для всякого алгоритма существует некоторая совокупность возможных исходных данных $\mathcal{J}$ - множество конструктивных объектов, к которым имеет смысл применять данный алгоритм. Для

описания заданного алгоритма используется некоторая совокупность конструктивных объектов K - множество структурных элементов алгоритма.

Множества J и K могут пересекаться. При этом, чем больше используются элементы исходной информации в качестве структурных элементов алгоритма, тем, вообще говоря, менее общей будет запись алгоритма.

Из множества K выделим следующие подмножества:

I. Непустое множество $A \subset K$, называемое множеством адресов.

2. Непустое множество O, называемое множеством операций. Это множество состоит из подмножеств O_J и O_K. Множество O_K представляет набор алгоритмических операций, не зависящих от природы исходной информации и удовлетворяющих свойству алгоритмической полноты. Элементы множества O_J вводятся для удобства записи исходной информации.

3. Некоторое непустое множество B - совокупность вспомогательных элементов, как-то: метки, разделители и т.п.

Обычно в начале работы алгоритма устанавливается соответствие между множеством J и некоторым подмножеством множества А. При применении к элементам множества А как к аргументам специальной операции из O_K, называемой штрих-операцией, получим множество значений G_J, называемое множеством содержимых.

Отображение множества А в G_J называется адресным отображением.

В начале работы алгоритма, если $i \in J$, существует такое

$g \in G$, что $i = g$. В процессе работы алгоритма адресное отображение изменяется и по окончании работы, если $r \in R$ (R – результативное множество), существует такое $g_r \in G$, что $g_r = r$. Задачей всякого алгоритма является получение для заданного множества J результативного множества R .

Для достижения большей общности записи алгоритмов разделяем запись алгоритмов на две части – статическую и динамическую.

Статическая часть S служит для установления исходного адресного отображения; она описывается на некотором языке соотношений, в котором явно представлены элементы исходной информации.

Динамическая часть D описывается на языке процессов. Исходную информацию здесь представляют элементы множества J. Динамическая часть алгоритма при переходе на язык машины представляется в виде программы, а статическая – в виде исходного адресного отображения.

Как структурные элементы алгоритмов элементы множества G могут играть различную роль в записи алгоритмов, определяемую степенью использования в языке принципа адресности. Различаем три ступени языка.

I. Множество G представляет только исходную информацию. Поэтому имеет смысл штрих-операция только первого ранга, и в множестве O_K необходима операция переадресации (замены). С другой стороны, чем разнообразнее природа исходной информации, тем естественнее вводить более широкие наборы операций O_J. В статической части устанавливается соответствие лишь между исходной информацией и множеством A ; удельный вес ее не-

значителен и поэтому деление на S и D часто вообще не делается. Общность, достигаемая на этой ступени, соответствует общности, достигаемой в алгебре при введении обозначения чисел буквами: выражения, отличающиеся содержимым адресов и совпадающие по записи, на этой ступени считаются подобными.

2. Содержимые могут представлять собой адреса. При этом в статической части устанавливается соответствие не только между J и A, но и между некоторым подмножеством $A_g \subset A$ и множеством A, т.е. G_j и A пересекаются и их общей частью является A_g. Во множестве O_K может быть введена штрих-операция второго ранга. Набор операций O_j определяется, как и на I ступени, природой исходной информации. Вообще язык I ступени является подъязыком II ступени. Язык II ступени отличается от языка I ступени высокой общностью записи в нем алгоритмов:подобными считаются выражения не только отличающиеся лишь содержимым адресов, но и адресами (например, $^2a \times ('b - {}^2d)$ и $^2b \times ('k - {}^2e)$); при этом важную роль играет статическая часть, за счет которой можно уменьшать динамическую.

3. Содержимые представляют и элементы множества O. На этой ступени допускается пересечение множеств O и G_j. При этом из записи алгоритмов можно вообще исключить элементы множества O_j, относя их к исходному адресному отображению. Тогда динамическая часть становится не зависящей от исходной информации. Запись алгоритмов на этой ступени максимально общая: на этой ступени подобными выражениями считаются выра-

ния совпадающие лишь по форме (например) $^{3}a+('b-{}^{2}d)$ и $^{2}c:({}^{2}e+'d)$). На этой ступени запись алгоритмов можно сделать единой для всех мыслимых алгоритмов, записываемых на языке П ступени, и тем более - I ступени.

Если же в записи алгоритмов допустить множество O_j, то язык П ступени будет подъязыком Ш ступени, т.е. получается достаточно гибкий универсальный язык.

2. АДРЕСНОЕ ПРОГРАММИРОВАНИЕ

Благодаря близости адресного языка к общепринятому символическому языку математики при записи алгоритмов в нем не возникает никаких специфических, связанных именно с языком трудностей.

В то же время специальные понятия и символы адресного языка отражают общие принципы, принятые во всех современных вычислительных машинах. Поэтому алгоритм, записанный на этом языке, удобен для машинного осуществления и близок к программе любой машины.

УРОВНИ И СТИЛИ АДРЕСНОГО ЯЗЫКА

Несмотря на то, что адресный язык основан на принципе адресности, тем не менее, к упорядоченности множества адресов в нем не предъявляется никаких требований и адресное отображение может задаваться на произвольном множестве адресов. Для задач, по своему существу связанных с упорядоченностью элементов, могут вводиться операции следования, не описанные алгоритмически.

Уровень упорядоченности адресов, уровень алгоритмизации введенных в них операций следования можно считать уровнем языка.

Можно определить следующие три уровня адресного языка [21],[26].

I. Общеалгоритмический уровень. Принимается наиболее естественное для данной задачи множество адресов и в тех случаях, когда этого требует задача, вводятся операции следования, описываемые общематематическими средствами (например, с помощью индексов).

На этом уровне адресный язык наиболее близок к АЛГОЛУ-60.

2. Уровен условных адресов. Предполагается упорядочение отдельных массивов адресов, обрабатываемых алгоритмом. Обычно массивы представляют собой арифметические последовательности адресов, первый (или нулевой) из которых задается. Последовательности, определяемые разными начальными адресами, предполагаются непересекающимися. Операции следования описываются таким образом уже алгоритмически.

На этом уровне адреса упорядочиваются, только исходя из требований задачи. Взаимная же упорядоченность массивов и другие вопросы, связанные с фактическим распределением памяти, не решаются. Тем не менее уже на этом уровне решаются практически все основные вопросы, относящиеся к области программирования для конкретных машин, например вопрос о выборе параметров, о рациональном распределении памяти, о рациональном построении циклов, экономии рабочих ячеек и др.

Этот уровень языка можно считать основным.

3. Уровень конкретных адресов. Предполагается, что множество адресов, исключая программные регистры, линейно упорядочено. Обычно имеется в виду выполнение данного алгоритма на конкретной машине, в связи с чем и решаются вопросы определения операций следования.

Абстрактный адрес может составлять и часть ячейки машинной памяти или нескольких таких ячеек.

В адресном языке заложена возможность перехода от уровня к уровню, начиная от самого абстрактного алгоритмического языка и кончая полным распределением адресов для данной машины. Это делает адресный язык полностью пригодным в качестве входного языка для программирующих программ. Три описанных выше уровня, конечно, не исчерпывают всех возможных промежуточных. В зависимости от уровня принятого входного языка создаются более или менее сложные ПП. Некоторые из таких ПП работают в ряде организаций (например, в ИК АН УССР ПП для "Урал-I" с I959 года, ПП для "Киева" с I959 года, ПП для М-20 с I96I года, ПП для УМШН с I964 года и др.).

Свободно используя средства адресного языка (речь идет о применении его операторов), можно записать один и тот же алгоритм самыми различными способами. Перевод со свободного адресного языка на язык конкретной машины довольно сложен. Поэтому вводятся так называемые стили языка. Стиль предполагает такие ограничения в применении выразительных средств свободного языка, которые делают перевод с данного стиля на язык конкретной машины более простым.

Перевод с общего языка в определенный стиль его возможен на любом уровне, но правила такого перевода должны быть строго сформулированы. Ограничения, налагаемые на язык, могут быть настолько сильными, что станет возможным поформульный перевод записи алгоритма в команды машины, т.е. запись алгоритма в данном стиле будет отличаться от машинной программы

только кодировкой. В этом смысле язык любой машины является адресным.

Таким образом благодаря широте и гибкости адресного языка можно произвольно выбрать границу, до которой необходимо вручную доводить программы задач с тем, чтобы дальнейшую их обработку проводила программирующая программа.

Вместе с тем при составлении адресной программы решаются все вопросы, связанные с рассматриваемой задачей и общими для всех машин особенностями ее решения. Поэтому на уровне адресного программирования возможен обмен информацией для различных машин.

ЗАДАЧА ОБОЗРЕВАНИЯ ИНФОРМАЦИИ

При решении любой задачи требуется обозревание относящейся к ней информации. Это легко достигается в алгоритмах, если информация невелика и связь между элементами множества, определяющая операции следования в ней, проста. При большом объеме информации и сложных операциях следования для составления схем обозревания необходимо использовать адреса второго и высших рангов.

В [10] рассмотрены схемы обозревания информации, используемые в программах широкого круга математических и информационно-логических задач.

В [26] показано, что применяя стандартные способы кодирования и схемы обозревания информации по адресам высших рангов, можно передавать все необходимые данные для подпрограмм с помощью всего лишь одного-двух адресов, хранящих информацию о распределении этих данных по адресам.

Удобство применения адресного языка для описания алгоритмов, содержащих схемы обозревания массивов сложной структуры и, в частности, с помощью стандартных способов их кодирования, проверено на большом круге задач, значительная часть которых возникла из практики. Таковы задачи линейной алгебры, строительной механики и прикладной теории упругости, математического программирования, календарного планирования, теории расписаний, планирования производства, обработки информации, задачи, связанные с проблемой раскладки, задачи типа перевода и многие другие. Рассмотрению ряда таких задач посвящена глава V в [26] ; см.также I)

I) Сборники стандартных подпрограмм для ЭВМ "Урал" под ред. Е.Л.Ющенко, Изд. КвИРТУ, Киев, ч.I, 1962; 4.П,1963;
В.А.Михайлов, В.М.Москаленко, Зб.праць з обчислювальної матем. і техн. т, П, 1961, 61-67;
А.А.Стогний, там же, 68-76; Н.И.Шлаховая, там же 78-82;
Л.Г.Дмитриев, П.М.Сосис, Программирование расчета производственных конструкций, Изд. литерату(ры) по строительству и архитектуре УССР, Киев, 1963 и др.

О ПРЕОБРАЗОВАНИИ АДРЕСНЫХ АЛГОРИТМОВ

Всякий алгоритм, в частности, адресный, допускает бесконечно много различных представлений в классе эквивалентных ему алгоритмов. Однако, программы, построенные по любому из эквивалентных алгоритмов и решающие одну и ту же задачу, вообще говоря, существенно отличаться друг от друга и будут более или менее благоприятными, например, для реализации на ЭВМ.

Отсюда естественно возникает важная проблема эквивалентных преобразований алгоритмов, рассмотренная, например, в работах А.А.Ляпунова[I)], Ю.И.Янова [2)], А.Н.Криницкого[3)]. Исследование эквивалентных преобразований алгоритмов заслуживает большого внимания, в первую очередь, в связи с возможностью их использования для поиска как рационального процесса построения программ, так и результатов этого процесса.

Так, например, преобразуя алгоритмы и оценивая с различных точек зрения процесс составления программы, а также сами программы, к которым приводят эти алгоритмы, можно, не прибегая к их явному построению, выбрать те программы, которые в некотором смысле являются оптимальными.

Автором доклада исследовались эквивалентные преобразования алгоритмов с двух точек зрения: формально-логической на уровне тождественных преобразований логических граф-схем ал-

---

I)А.А.Ляпунов, О логических схемах программ, Проблемы кибернетики, вып. I,1958.

2) Ю.Н.Янов, О логических схемах алгоритмов, там же.

3) Н.А.Криницкий и др. Программирование, СМБ, Физматгиз, 1963.

горитмов в смысле Л.А.Калужнина[4] и содержательной, учитывающей индивидуальные особенности решаемой задачи.

В основу метода формально-логических преобразований логических граф-схем алгоритмов (в частности минимизации их по операторам действия [36] было положено сведение этой задачи к аналогичной задаче для некоторых автоматов Мура и к применению к ним аппарата теории абстрактных автоматов, разработанного, например, в работах В.М.Глушкова.

Содержательные эквивалентные преобразования алгоритмов еще полностью не систематизированы. Тем не менее широкие возможности, открывающиеся при их описании в адресном языке показаны на ряде важных задач [26] таких, как эквивалентные преобразования с одного уровня (стиля) адресного языка в другой, внутри одного уровня (стиля) и др. Последнее существенным образом используется, например, при построении систем автоматизации программирования, а также при исследовании вопросов, связанных с автоматизацией построения трансляторов.

АДРЕСНЫЕ АЛГОРИТМЫ И МАТЕМАТИЧЕСКИЕ МАШИНЫ

Любой алгоритм описывается конечным числом операций. Следовательно, в принципе может быть построена идеальная адресная машина (ИАМ), для которой данный адресный алгоритм, как и все алгоритмы, составленные из тех же действий и использующий те же адреса, является программой [13]. С этой целью

---

4) Л.А.Калужнин, Об алгоритмизации математических задач, Проблемы кибернетики, вып. П, 1959 г.

в набор элементарных операций ИАМ надо включить все встречающиеся в алгоритме действия и обеспечить соответствующую их кодировку.

Реальные ЭВМ отличаются от ИАМ, помимо ограниченности памяти, тем, что набор их элементарных операций, хотя и является полным, ограничен небольшим числом. Ради простоты записи в качестве адресов машин используется обычно начальный отрезок чисел натурального ряда, длина которого характеризует объем оперативного запоминающего устройства (ОЗУ). Однако в множестве ячеек, используемых для хранения информации в машине, существенное место принадлежит ее программным регистрам (наименования - адреса этих регистров должны выбираться существенно отличными от адресов ЗУ).

Моделирование работы любой реальной ЭВМ на ИАМ сводится к выбору из ее элементарных операций только тех, которые входят в набор данной реальной машины. Порождаемый при этом язык представляет собой некоторый стиль адресного языка и адресная программа, записанная при соблюдении требований данного стиля, для превращения в программу реальной машины нуждается только в перекодировке.

С другой стороны, любую элементарную операцию ИАМ можно воспроизвести некоторой последовательностью операций - программой реальной ЭВМ (с учетом объема памяти, ее упорядоченности и наличия тех или иных программных регистров).

Использование адресов высшего ранга и схем обозревания информации по ним позволяет записывать адресные алгоритмы в форме, не изменяемой в процессе их выполнения. При этом произ-

вольный адресный алгоритм можно представить в эквивалентной ему форме, в которой все входящие в запись алгоритма адресные функции имеют ранг не выше второго [22] . Однако, алгоритмы, описываемые с помощью адресных функций только первого ранга и не включающие формул замены (переадресации), представляют собой весьма узкий класс алгоритмов. Таким образом, включение в набор операций машины для его алгоритмической полноты операций обращения по адресу второго ранга или замены (или операций, эквивалентных им) обязательно.

Первая из этих операций обладает известным удобством, поскольку пользоваться программами, не изменяющимися в процессе их выполнения, практически удобнее. Кроме того, такие программы проще для схемной реализации.

Приведенные соображения были учтены при выборе операций для машины "Киев" [23] . Прежде всего следует назвать операцию по адресу второго ранга (код операции Ф), в том числе условного перехода по адресу высшего ранга, а также операции с адресным регистром, разработка которых была осуществлена автором совместно с В.С.Королюком [14].

По команде Ф машиной выполняются операции:

$$'(('(' \mathcal{K}_{\underline{I}} \oplus '\mathcal{E}_1 \cdot 'M)_{\underline{II}}) \Rightarrow '\mathcal{K}_{\underline{III}} \oplus '\mathcal{E}_3 \cdot 'M .$$

$$'('\mathcal{K}_{\underline{I}} \oplus '\mathcal{E}_1 \cdot M)_{\underline{II}} \Rightarrow M$$

$$'C \oplus 1 \Rightarrow C$$

$$^2C \Rightarrow \mathcal{K}$$

Здесь: M - регистр модификации адреса: C - счетчик команд; K - регистр команд; римские индексы I, II, III обозначают соответствующие адреса (ячейки ЗУ или регистров); $\mathcal{E}_i$ - разряды признаков модификации соответсвующих адресов.

С помощью операции Ф и операций над регистром адреса была технически реализована выдвинутая автором идея [I5] создания в машине сменно-спаянной опреативной памяти, в которой (в виде сменных блоков) размещаются те или иные программы или подпрограммы.

3. АДРЕСНЫЙ ЯЗЫК И МЕТОД УНИВЕРСАЛЬНЫХ ПРОГРАММИРУЮЩИХ ПРОГРАММ

Одним из наиболее перспективных направлений в автоматизации программирования является метод универсальных программирующих программ, основанный на использовании алгоритмических языков.

Задачей программирующей программы является перевод исходной информации - записи алгоритма в некотором абстрактном алгоритмическом языке, выбранном в качестве внешнего языка системы автоматизации программирования, - в рабочую программу этого алгоритма на языке конкретной машины.

Среди бесконечного множества возможных алгоритмических языков, одинаково удобных для человека, адресный язык является наиболее "обращенным" к современным электронным вычислительным машинам, поскольку в нем в наибольшей мере учитываются особенности их естественных внутренних языков - принцип адресности и принцип программного управления.

Указанные особенности адресного языка, как внешнего языка систем автоматического программирования, представляют известные преимущества.

Применению адресного языка в связи с проблемой автоматизации программирования и, в частности, в связи с автоматизацией процесса составления трансляторов, посвящены работы автора [12], [21], [26], [29], [38] и совместные работы [16-19], [23], [30], [31], [39].

УРОВНИ И СТИЛИ АДРЕСНОГО ЯЗЫКА И ПРОБЛЕМА АВТОМАТИЗАЦИИ ПРОГРАММИРОВАНИЯ

Задачей программирующей программы является перевод алгоритмов с общего алгоритмического (например, адресного) языка на язык конкретной машины. Однако, этот перевод может быть разделен на этапы введением ряда стилей, все более и более приближающихся к языку машины.

Широта и гибкость адресного языка позволяют произвольно выбирать точку, до которой необходимо вручную доводить программы задач с тем, чтобы дальнейшую обработку проводила программирующая программа.

Создание сложных программирующих программ, переводящих запись алгоритма с самого абстрактного алгоритмического языка на язык конкретной машины, следует считать не единственным и, возможно, не лучшим путем автоматизации программирования. Этот путь обладает рядом недостатков.

I. Для разработки таких программ, их проверки и отладки необходимы длительные сроки, поэтому вопрос о текущей автоматизации сдвигается на чрезмерно далекие сроки.

2. Такие программы неизбежно будут чрезмерно большими по объему. Следовательно, они будут работать с многократными обращениями к внешней памяти (большую часть времени машина при этом тратить на пересылку информации из одной памяти в другую).

3. Ввиду большого расстояния между входными и выходными языками ПП трудно добиться оптимального программирования с ее помощью для всех задач.

4. Как и во всех больших программах, имеется большая вероятность ошибок, которые трудно локализовать.

В работах автора [12], [23], [26] предложен другой путь автоматизации программирования.

Предлагается установить несколько стандартов входного языка, каждый из которых характеризуется распределением памяти (уровнем) и ограничением выразительных средств (стилем). Одним из стандартов является свободный язык. Переработка записи алгоритма в программу машин сводится к обработке его несколькими программами перевода.

Преимущества такого способа состоят в следующем:

I. Автоматизация программирования производится в короткие сроки и дает значительные результаты уже на первых порах, что проверено на ряде ПП в ИК АН УССР.

2. Программы перевода, полностью находящиеся в оперативной памяти, работают быстро.

3. Ввиду меньшего размера ПП ошибки в них менее вероятны.

4. Такая система автоматизации способна к развитию и усовершенствованию. Любую программу перевода можно изменить, не нарушая работы всей системы; каждое такое улучшение немедленно дает свои результаты.

5. Предложенная система не исключает первого способа автоматизации. Достаточно склеить несколько программ перевода,

чтобы получить программирующую программу первого типа. Правильность такого подхода к автоматизации программирования подтверждена опытом работы ПП с двумя внешними языками, созданной под руководством автора доклада для машины М-20 [29],[39]. Первый уровень входного языка представляет собой некоторый стиль адресного языка (используется с начала 1962 года), и служит промежуточным языком для второго уровня. Последний представляет собой сокращенный язык Алгол-60 (используется с 1964 года).

Граница, до которой необходимо вручную доводить программы задач с тем, чтобы дальнейшую их обработку проводила программирующая программа, должна выбираться в зависимости от того, насколько сложную программирующую программу целесообразно осуществить в данный момент.

О принципах построения программирующих программ

Алгоритмы перевода ПП ввиду особой структуры алгоритмических языков являются более простыми, более определенными и исчерпывающими, чем алгоритмы перевода, с одного живого языка на другой.

Работа ПП основывается на заранее установленных основных соответствиях между определенными синтаксическими выражениями алгоритмического языка, которые назовем представимыми, и кодами машины. Количество и характер представимых выражений должны быть достаточными для расчленения любого алгоритма на цепочку таких выражений.

ПП должны в ходе своей работы представлять исходную информацию в виде такой цепочки и, используя основные соответствия, заменять эти выражения одно за другим кодами машины. Выбор системы представимых выражений (т.е. установление основных соответствий) определяет структуру и принципы построения ПП.

Большинство созданных до сих пор ПП основано на одном и том же принципе. В них множество представимых выражений составляется из синтаксических выражений, переводимых на машинный язык одной или несколькими командами. Работа ПП, основанной на этом принципе, сводится к следующему.

I. ПП, перебирая исходную информацию, отыскивает в ней представимое (в известном смысле, первое) выражение (требуется, чтобы в любой исходной информации всегда имелись такие выражения).

2. В формируемую программу записывается приказ (группа приказов), соответствующий найденному выражению.

3. Соответственно преобразуется исходная информация, и цикл работы ПП начинается снова с п.I.

4. Требуется, чтобы в результате преобразования информации в ней до ее полного исчерпания появились представимые выражения. Предполагается, что выполнение представимых выражений в преобразуемом алгоритме в порядке их нахождения ПП является одним из допустимых в алгоритме порядков действий.

На этом принципе построены, например ПП, разработанные в ряде организаций (см.сноску на стр. 2), а также разработанные под руководством автора доклада ПП с входным адресным языком для машин Урал-I [16] , Киев [30] и упомянутая ранее

ПП для машины М-20 с двумя входными языками - адресным и сокращенным Алголом [29] , [39] .

Выполнение перечисленных требований достигается за счет достаточной сложности ПП (как правило, за счет предварительного перевода алгоритмов в запись на некотором упрощенном промежуточном языке - внутреннем языке транслятора) или за счет дополнительных требований к записи алгоритмов во внешнем языке (например, бесскобочная инверсная запись формул по Лукасевичу, которая принята для ПП-АК [16], [29]).

Исходя из специфики алгоритмических языков, в которых каждый из синтаксических знаков (или их некоторая последовательность) в отличие от естественных языков несет свою собственную информацию о смысле, не зависимую от сочетаний с другими знаками, в [23] предложен следующий принцип поэлементного перевода адресных формул в машинный код.

Можно считать представимыми выражениями минимальные выражения, составленные из основных символов принятого в языке алфавита и несущие синтаксическую информацию. Если, например, рассматриваются простейшие арифметические формулы в скобочной записи, то представимым будет каждое из следующих выражений: величина, адрес величины, знаки (, +, - и др.).

Принятое множество представимых выражений определяет основные соответствия. Прежде всего, представимые выражения так малы, что им соответствуют не команды машины, а обычно, часть команды - отдельный код. Кроме того, в число представимых элементов входят, например, скобки, определяющие не действия, а порядок их выполнения. Им должно быть поставлено в

соответствие, кроме, может быть, кодов, еще и установление некоторого порядка записи команд в формируемой программе.

Работу ПП, основанной на этом принципе, можно характеризовать следующим:

I. Для формирования программы отдельной формулы исходного алгоритма выделяется рабочее поле. В это поле вносятся те или иные записи в зависимости от просматриваемого в данный момент элемента. Отдельным элементам может соответствовать не только внесение кодов в рабочее поле, но и определение мест в поле для внесения следующих кодов.

2. Исходная информация просматривается однократно, в естественном порядке, поэлементно.

3. Ни одна часть формируемой программы не может считаться, вообще говоря, законченной, пока не запрограммирована вся формула. (Поформульная обработка информации вызвана тем, что формула, во всяком случае, имеет законченный смысл).

Этот принцип поэлементного перевода алгоритмов в машинные программы положен в основу ряда ПП, разработанных под руководством автора доклада и М.М.Бушко-Жук (ПП для машины "Днепр", Урал-I, Минск-I, Урал-2 и др.).

Сформулированный принцип построения ПП позволяет по-новому подойти к решению вопроса об автоматизации программирования. Несмотря на сложность ПП, в процессе их работы далеко не полностью используются возможности машины (по набору операций, в гибкости связей отдельных устройств). Поэтому, учитывая частое применение ПП, естественно ставить вопрос о ее создании в виде запаянной схемы.

ОБ АВТОМАТИЗАЦИИ ПРОЦЕССА СОСТАВЛЕНИЯ ТРАНСЛЯТОРОВ

В общем случае ПП представляют собой сложные информационно-логические алгоритмы типа алгоритмов перевода с одного языка на другой.

Вместе с тем, хотя каждая ПП и является средством автоматизации программирования, для ее практического применения необходимо составлять ее рабочую программу в машинном языке в то время, когда необходимый транслятор еще отсутствует. Как средство автоматизации при отсутствии средств автоматизации, рабочая программа алгоритма ПП, вообще говоря, составляется вручную. Громоздкость и длительность таких работ при бурном развитии современной вычислительной техники обычно существенно снижает ожидаемый эффект от создаваемых систем автоматизации программирования.

Отсюда возникает важная проблема автоматизации самого процесса составления ПП - автоматизации программирования.

В работах автора и ее ученика [redacted] В.П.Семика (см.[26],[31],[38]) намечены следующие пути решения данной проблемы.

I. Метод применения "общемашинного" стиля в качестве промежуточного языка для ПП.

2. Метод применения одной ПП для создания других ПП.

3. Метод применения универсального моделирующего адресного алгоритма (УМ) для построения программирующих программ.

Первый из указанных методов основан на выделении некоторого общемашинного стиля адресного языка, включающего в себя характерные особенности машинных языков вцелом, и на разделении процесса работы ПП на следующие два этапа, обычно применяемые при машинном переводе с помощью языка-посредника:

а) эквивалентные преобразования записи алгоритмов с избранного в качестве внешнего стиля языка в общемашинный стиль;

б) перевод с общемашинного стиля языка в язык конкретной машины. Работа ПП для разных ЭВМ отличается по существу лишь вторым этапом, т.к. преобразования, указанные в а) являются общими для всех трансляторов с данного входного языка. Таким образом, алгоритмы, реализующие эти преобразования, и разработанные при составлении ПП для одной какой-либо машины (алгоритмы анализа), могут быть непосредственно использованы при составлении ПП для других машин.

Свободному адресному языку соответствует машинный язык с нефиксированной структурой команд ИАМ. Каждой записи алгоритмов с нефиксированной структурной команд соответствует определенная комбинация команд машинных стилей. Из соответствия записи алгоритмов в обобщенном и фиксированных стилях языка (основные из них: одно - , двух - и трехадресные стили) следует, что работа ПП для разных ЭВМ отличается по существу лишь вторым из указанных этапов, который условно назовем этапом кодирования.

Составив для заданных машин библиотеку алгоритмов кодирования, можно построить комплексную ПП, составляющую рабочие программы для одной из избранной группы машин или одновременно для нескольких из них.

При построении конкретной ПП разделение во времени указанных этапов работы может оказаться нерациональными, в связи с тем алгоритмы эквивалентных преобразований и кодирования могут объединяться.

Используя соответствие между стилями, можно из объединенного алгоритма для одной ЭВМ строить алгоритмы для других машин. Этот прием был использован в ИК АН УССР при составлении адресного алгоритма ПП для машин "Днепр" и "Минск" при наличии соответствующей ПП для машины Урал-I.

В основу второго и третьего из указанных методов положена идея предварительного описания ПП в языке, являющемся их внешним языком.

Осуществление этой идеи в принципе возможно для любого универсального языка. Однако далеко не каждый такой язык оказывается практически приемлемым для этой цели.

Благодаря использованию принципа адресности программирующие программы в адресном языке описываются весьма естественно (см.[21],[31]), в связи с чем этот язык оказывается вполне пригодным для указанной цели.

Обозначим через ПП(А-В-С) программирующую программу с внешним языком А для машины В, описанную в языке С.

Пусть α - алгоритмический язык; А, В, С - языки некоторых машин.

Чтобы применить ПП(α - А- В) (или ПП (α - А -А)) для составления ПП (α - С - А) или ПП(α - С - С) составляем вначале алгоритм ПП (α - С -α). Перевод этого алгоритма с помощью ПП(α - А - В) на машине В дает ПП(α - С - А). Повторный перевод того же алгоритма ПП(α - С-α) с помощью полученной ПП (α - С - А) (на машине А).

~~ПП С-А~~ дает программирующую программу в языке машины С для машины С (ПП(α-С-С)).

Пример использования одной ПП для построения других рассмотрен в работе [31] . А именно, ПП для машины "Киев", реализованная вручную в языке машины "Киев", вначале используется для создания ПП для машины "Днепр" в языке машины "Киев"; последняя затем используется для построения ПП для машины "Днепр" в языке той же машины "Днепр".

В связи с последним заметим, что не всякая ЭВМ оказывается практически удобной для реализации ПП. Так, например, управляющая машина широкого назначения "Днепр", предназначенная для управления различными технологическими процессами, хотя и является универсальной вычислительной машиной, имеет набор операций, существенно обращенный в сторону задач управления и весьма неудобный для описания задач типа задач перевода. Для автоматического составления программы для такой машины естественно использовать какую-либо другую машину, с более гибким набором операций, в связи с чем в качестве такой машины и была выбрана машина "Киев".

Построение ПП для машины "Днепр" с реализацией на какой-либо другой машине тем более представляется целесообразным, поскольку эта машина на практике может находиться в замкнутой цепи управления каким-либо процессом и ее применение для составления новых программ (например, в целях совершенствования тех же алгоритмов управления) в этом случае весьма нежелательно.

Другая идея использования записи ПП в ее внешнем языке состоит в применении универсального [illegible] алгоритма, ра-

ботающего по принципу моделирования,т.е. решающего произвольную задачу по исходному (моделируемому) алгоритму ее решения,описанному во внешнем языке, и по информации к нему.

Задавая,с одной стороны, в качестве моделируемого алгоритма алгоритм какой-либо ПП для машины А ,описанный в алгоритмическом языке ПП(α-А-α), и с другой - его же в качестве исходной информации для самого себя, с помощью этого универсального алгоритма получаем транслятор в машинном коде машины ПП(α-А-А). В принципе, результатом работы универсального моделирующего алгоритма может быть как решение некоторой (например,вычислительной) задачи,так и ее программа или транслятор, с помощью которого может быть составлена эта программа.

В действительности же, преследуя цель - автоматическое составление ПП, в набор операций моделирующего алгоритма достаточно включить операции, необходимые лишь для моделирования трансляторов.Заметим также, что особых требований относительно качества и скорости работы к универсальному алгоритму не пред"является. Качество создаваемых им программ (в том числе и ПП) полностью определяется качеством исходного алгоритма ПП(α-А-α).

ЗАКЛЮЧЕНИЕ

Представленные работы являются результатами исследований, выполненных в связи с разработкой вопросов теории алгоритмических языков и автоматизации программирования.

Наибольшее внимание в работах уделено разработке алгоритмического адресного языка, методам составления трансляторов и их созданию для ряда отечественных ЭВМ.

Отличительной особенностью исследований является их практическая направленность. Основные теоретические результаты получили непосредственное применение при разработке ряда программирующих программ и при алгоритмизации многих производственных процессов, экономических и других информационно-логических задач, а также в практике конструирования ЭВМ.

Основные результаты, полученные в работах, следующие:

I. Разработан алгоритмический адресный язык и основанный на нем метод программирования для ЭВМ. Доказана его алгоритмическая полнота.

2. Рассмотрены вопросы эквивалентных преобразований и оптимизации алгоритмов.

3. На основе использования принципа адресности предложена классификация алгоритмических языков.

4. На основе использования адресов высших рангов разработаны стандартные методы задания информации о массивах сложной структуры, применяя которые можно передавать информацию для подпрограмм с помощью одного-двух адресов.

5. На обширном круге сложных математических и информационно-логических задач показана эффективность использования адресного языка.

6. На основе выбора уровней и стилей в алгоритмическом языке предложен поэтапный метод автоматизации программирования.

7. Разработаны новые принципы организации работы трансляторов.

8. Предложены методы автоматизации построения трансляторов.

9. Показаны удобства использования адресного языка для описания ЭВМ с программным управлением.

IO. Исследован и теоретически обоснован выбор некоторых математических параметров ЭВМ "Киев".

Автором написано 6 монографий и учебников (некоторые в соавторстве с В.М.Глушковым, Б.В.Гнеденко, В.С.Королюком и др.), получивших широкое распространение среди математиков-программистов, аспирантов, научных работников. Из них учебник "Элементы программирования" рекомендован МВССО РСФСР в качестве учебного пособия для вузов (I изд. вышло в I96I г. [redacted]; II изд. - I963 г. [redacted]) и переводится на венгерский, ческий и др. языки.

Под руководством автора доклада:

I. Осуществлено внедрение адресного метода программирования.

2. На основе использования адресного языка осуществлена алгоритмизация ряда сложных математических и информационно-логических задач, в том числе сложных технологических процессов, экономических, проектно-конструкторских и др. работ.

3. Разработана система автоматического программирования с входным адресным языком - трансляторы для ЭВМ "Урал-I","Киев", М-20, УМШН (Днепр), Минск-I и др.

4. Разработан транслятор с сокращенного языка Алгол-60 на М-20, использующий в качестве языка-посредника адресный язык.

5. Осуществлено внедрение указанных трансляторов в практику решения задач в ряде институтов АН УССР (Институт кибернетики,Математики,Высокомолекулярных соединений, Полупроводников, Электродинамики и др.),НИИ Экспериментального проектирования (Киев),Научно-исследовательском геолого-разведовательном институте (Киев),Тресте Геофизразведка,Укрпроектстальконструкция, КГУ, БПИ (Бердянск), Академия с.х.наук и др.

6. Разработана и создана ЭВМ "Киев" (руководство осуществлялось коллективом авторов).

7. Созданы библиотеки стандартных и тестовых программ для машин "Киев", "Днепр" и др.

8. Разработаны и внедрены методы автоматизации процесса составления программирующих программ.

Приложение I

КРАТКАЯ ХАРАКТЕРИСТИКА

некоторых трансляторов, разработанных в

Институте Кибернетики АН УССР

I. ПП-УМШН - транслятор с сокращенного адресного языка для управляющей машины широкого назначения "Днепр", реализованный на машине "Киев" [3I] .

Работа ПП основана на принципе поэлементного перевода исходного алгоритма; для арифметических и логических выражений принята строго скобочная запись формул.

Начало РП может указываться произвольно.

Скорость работы - 60 команд в минуту (включая время ввода-вывода) ПП использует ОЗУ и 2 магнитных барабана. Объем ПП-II30 ячеек.

2. Трансляторы для машины М-20.

а) Транслятор с адресного языка ПП-М [39].

Для арифметических и логических выражений принята обычная скобочная запись формул.

ПП осуществляет экономию приказов РП. При ее работе используется 2 зоны магнитной ленты (МЛ) по 770 кодов, и одна - 200 кодов , один магнитный барабан и 900 ячеек МОЗУ.

ПП состоит из двух блоков "Автокодировщик" и программирующий блок, которые хранятся на МЛ и вызываются поочередно в МОЗУ.

Скорость работы - 130 команд в мин (включая время ввода-вывода).

Помимо РП, ПП выдает таблицу распределения операторов, соответствия истинных и буквенных адресов и начал массивов. ПП допускает автоматическое использование библиотечных программ системы ИС-2м, имеющих стандартное обращение.

б) Транслятор с сокращенного Алгола ПП-М2; Используется в качестве промежуточного языка язык ПП-М.

Помимо указанных блоков ПП-М, имеет блок перевода алгольных алгоритмов в адресные, занимающий еще одну зону МЛ в 1500 кодов.

3. Трансляторы для машины Урал-1.

а) ПП- транслятор с сокращенного адресного языка [30]. Составляет РП в режиме фиксированной и плавающей запятой. Для арифметических и логических выражений принята инверсиая бесскобочная запись формул. Исходная информация вводится по зонам до 440 кодов в каждой.

Для реализации плавающего режима мантисса числа помещается в полную ячейку, порядок - в короткую.

Начало РП, исходных данных и массива рабочих ячеек задаются произвольно. Объем ПП-1240 ячеек ОЗУ; при работе ПП используется только оперативная память; скорость работы - 15 команд в мин. (без учета времени ввода-вывода).

б) ПП-ЭРА - транслятор с сокращенного адресного языка (более свободного, чем входной язык ПП-АУ)[26].

Для логических и арифметических выражений принята строго скобочная запись формул (максимальная глубине скобок - 10).

ПП основана на принципе поэлементного перевода исходного алгоритма; выдает РП в режиме фиксированной запятой; для плавающего режима ПП использует моделирующую программу (объем которой 254 команды), в силу чего РП в плавающем режиме, составленная ПП-ЭРА, работает медленее, чем соответствующая РП, выданная ПП-АУ. В отличие от ПП-АУ мантисса и порядок числа при работе ПП-Эра помещаются в одну ячейку (28 разрядов отводится под мантиссу, 6- под порядок).

Начало РП и массива рабочих ячеек задаются произвольно.

Объем - II52 ячейки ОЗУ; при работе используется только ОЗУ. Скорость работы - 40 команд в мин.

Приложение 2

Список научных трудов по теме диссертации

№ п/п	Наименование работы	Печатная рукопись	Название издательства, журнала	Год издания, № журнала	Количество печатных листов	Примечание /указать соавторов/
I.	Основы программирования	печатная	Сб.по вычислит. техн.и матем. Изд.КВИРТУ	1958	4,75	
2.	Об адресном программировании*/	-"-	Труды ВЧ 06669	1959,вып.36	0,4	В.С.Королюк
3.	Программирующая программа для машины "Киев"*/	-"-	-"-	-"-	0,6	Л.Н.Иваненко
4.	Определение принадлежности точки границе области*/	-"-	-"-	-"-	0,2	
5.	Программирование таблиц для оптимальных методов статистического приемочного контроля*/	-"-	Вопросы вычисл. техн.и матем., Сб.трудов ВЦ АН УССР	1958 Вып.3	0,6	В.С.Королюк, Л.П.Нижник
6.	Цифровая автоматическая машина "Киев"*/	-"-	Сб.трудов конф. по вычисл.матем. и техн.ВЦ АН УССР	1960	2,0	В.М.Глушков и др.
7.	Адресное программирование*/	-"-	-"-	1960	0,4	В.С.Королюк
8.	Адресное программирование и особенности решения задач на машине "Урал"*/	-"-	Изд.КВИРТУ	1960	14,5	
9.	Математичні параметри універсальної цифрової автоматичної машини "Київ"*/	-"-	Зб.праць ОЦ АН УРСР	1961 Т.2	0,2	В.М.Глушков, Б.В.Гнеденко
10.	Питання теорії і практики програмування*/	-"-	-"-	1961 т.I	2,0	В.С.Королюк
11.	Основні питання побудови програмуючої програми для машини "Київ"*/	-"-	-"-	1961,т.II	0,2	Л.И.Иваненко
12.	Адресный язык и проблема автоматизации программирования*/		Доклад на IУ Всесоюзном матем. с"езде	1961		

1	2	3	4	5	6	7
13.	Адресні алгоритми та цифрові математичні машини*/	печатная	Зб.праць ОЦ АН УРСР	1961, т.П	0,3	
14.	Групові операції машини "Київ"*/	-"-	-"-	-"-	0,3	В.С.Королюк,К.А. Шкабара
15.	Бібліотека підпрограм машини "Київ"*/	-"-	-"-	-"-	0,2	
16.	Програмуюча програма,інформацією для якої служить адресний алгоритм*/	-"-	-"-	1961 т.Ш	1,0	Л.П.Бистрова
17.	Алгоритм формальної перевірки привильності дужкового та бездужкового запису формул*/	-"-	-"-	-"-	0,4	
18.	Алгоритм перекладу дужкового запису формул в бездужковий*/	-"-	-"-	-"-	0,4	О.І.Костюченко
19.	Элементы программирования/учебное пособие/*/	-"-	Физматгиз	1961	21,75	Б.В.Гнеденко, В.С.Королюк
20.	Адресні алгоритми та математичні машини*/	-"-	ДАН УССР	1962, № 6	0,2	
21.	Рівні та стилі адресної мови та проблема автоматизації програмування*/	-"-	-"-	1962, №7	0,2	
22.	Про повноту засобів адресної мови*/	-"-	-"-	1962,№ 10	0,2	
23.	Вычислительная машина "Киев"*/	-"-	Укртехиздат	1962	12,0	В.М.Глушков
24.	Адресное программирование*/	-"-	Кибернетика на транспорте,Материалы заочн.семинара, тема 6,КДНТП	1962	2,0	
25.	Про один новий метод розв"язання задачі лінійного програмування*/	-"-	Обчисл.матем.і техн. Сб.ИК АН УССР	1963	0,2	
26.	Адресное программирование*/	-"-	Укртехиздат	1963	18,0	
27.	Моделирование режима плавающей запятой на машине "Киев"*/	-"-	Обчисл.матем.і техн. Сб.ИК АН УССР	1963	0,2	Л.И.Иваненко, И.В.Сергиенко
28.	Элементы программирования /П изд./*/	-"-	Физматгиз	1963	21,75	Б.В.Гнеденко, В.С.Королюк

1	2	3	4	5	6	7
29.	Малые трансляторы Института кибернетики АН УССР*)	руконись	Доклад на Конференции стран социалист. лагеря "Методы автоматического программирования и алгоритмические языки"	Киев 1963	0,6	
30.	Программирующая программа для машины "Урал-1"*)	печатная	Изд."Наукова думка"	1964	5,5	Т.А.Гринченко
31.	Управляющая машина широкого назначения "Дніпро" и программирующая программа к ней*)	-"-	-"-	1964	17,0	Б.Н.Малиновский и др.
32.	Про побудову координатних сімей, природно пов"язаних з задачою	принято к печати	ДАН УССР		0,2	В.Л.Рвачов
33.	Об аналитическом описании геометрических об"ектов	-"-	Кибернетика и техн. вычислений, Сб.работ ИК АН УССР		0,8	В.Л.Рвачев
34.	Применение адресного языка для описания информационно-логических алгоритмов*)	-"-	Сб.работ П Всесоюзн. совещания по автом. переработки информации		0,8	В.П.Семик
35.	Адресный язык и принцип адресности в алгоритмических языках*)	-"-	-"-		0,5	В.П.Семик
36.	К вопросу о минимизации логических-граф-схем обучения*)	-"-	Научный совет по киберн. АН УССР		1,0	В.Н.Редько
37.	О геометрических приложениях алгебры логики	-"-	-"-		1,0	В.Л.Рвачев
38.	Про автоматизацію процесу складання трансляторів для ЕОМ*)	-"-	ДАН УССР			
39.	Программирующая программа с входным адресным языком для одной Ш-адресной машины*)	-"-	Изд."Наукова думка"			Л.П.Бабенко и др.

*) Звездочкой отмечены работы, представляемые к защите