Ордена Ленина
ИНСТИТУТ ПРИКЛАДНОЙ МАТЕМАТИКИ
имени М.В. Келдыша.
Академии Наук СССР

Н.А. Наумов, А.Г. Рубин, В.К. Смирнов

ОБ ОДНОМ СПОСОБЕ РЕАЛИЗАЦИИ ВХОДНЫХ ЯЗЫКОВ ДЛЯ СИМВОЛЬНОГО ПРОЦЕССОРА

Препринт № 146 за 1981 г.

Москва.

ОРДЕНА ЛЕНИНА
ИНСТИТУТ ПРИКЛАДНОЙ МАТЕМАТИКИ
имени М.В.Келдыша
АКАДЕМИИ НАУК СССР

Н.А.Наумов, А.Г.Рубин, В.К.Смирнов

ОБ ОДНОМ СПОСОБЕ РЕАЛИЗАЦИИ ВХОДНЫХ ЯЗЫКОВ ДЛЯ СИМВОЛЬНОГО ПРОЦЕССОРА

Москва 1981

УДК 681.14

Об одном способе реализации входных языков для символьного процессора. Наумов Н.А., Рубин А.Г., Смирнов В.К. ИПМ АН СССР. Препринт № 146, М., 1981 г., 27 стр., библиогр.: 85 назв.

В работе предлагается использовать конвертирование для реализации алгоритмических языков на специализированном символьном процессоре (СПП). Рассматривается сущность подобного метода повышения квалификации СПП и его преимущества на примере опытного варианта конвертера с некоторого подмножества языка лисп на язык рефал (ЛИРА), написанного на рефале. Приводятся результаты работы ряда объектных лисп-программ на макете СПП и оценивается ожидаемый выигрыш предлагаемого подхода по сравнению с программной реализацией лиспа.

Описанный конвертер может рассматриваться в качестве мобильной и открытой для пользователей реализации лиспа на ЭВМ, имеющих рефал-системы. Программа предъявляет минимальные требования к транслятору с языка рефал. В работе приводится краткий обзор литературы по использованию языка лисп в различных областях и библиография.

Ключевые слова и фразы: символьная обработка, конвертер, рефал, лисп, символьный процессор, компилятор, интерпретатор, макрогенератор, поиск по образцу, система программирования.

In this paper is proposed conversion technique as perspective realisation method for list structure or symbolic manipulation languages on special symbolic processor (SSP). Are discussed the advantages of the method in experimental LIRA-converter. Are discussed the results of execution of some object LISP-programms. Are appreciated the advantages of this method in compare to programm realisation of LISP language.

This portable, machine independent and opened for user LISP-converter is proposed for computers having refal-systems and making a minimum demands to refal-translator.

In this paper are produced short literature review and bibliography concerning LISP and its applications.

KEY WORDS: symbolic manipulation, converter, refal, LISP, symbolic processor, compiler, interpreter, macrogenerator, pattern matching.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Задачи неарифметического характера такие как трансляция, редактирование, аналитические преобразования занимают в настоящее время огромную часть машинного времени. Подключение к вычислительной системе специализированного процессора символьных преобразований может дать существенный выигрыш производительности системы.

В ИПМ АН СССР разрабатывается специализированный символьный процессор ССП [6.5-6.16], предназначенный для эффективного программирования и решения задач символьной обработки. В настоящее время экспериментальный вариант (макет) процессора реализован и эксплуатируется в институте. В качестве базового входного языка ССП используется язык рефал, получивший достаточно широкое распространение и реализованный на большинстве отечественных машин [6.1]. Технология и области его использования детально отражены в работах [6.1, 6.2]. В основу внутреннего языка ССП положен промежуточный язык компилятора с

рефала [6.I, 6.3], который расширен за счет добавления ряда операторов и встроенных функций, преимущественно являющихся микропрограммной реализацией машинных операций.

Для решения задач символьной обработки наряду с РЕФАЛ ом широко применяются и другие языки - ЛИСП, СНОБОЛ, КОМИТ и др. Сопоставление наиболее известных языков символьной обработки лиспа, снобола и рефала приводится в работах [4.2, 4.I3, 5.I9].

Наибольшей популярностью среди них пользуется, пожалуй, лисп, на котором создан целый ряд систем, имеющих практическое значение (см.п.2).

Учитывая общность областей применения указанных языков, их широкое распространение и большой объем накопленного на них программного обеспечения, а также специфику ССП естественным представляется стремление к реализации наиболее существенных из них в качестве дополнительных входных языков символьного процессора. Правомерно существование целой гаммы способов реализации новых языков на ССП. В частности, достаточно распространенным (но трудоемким) является путь, связанный с построением символьного процессора с настраиваемой архитектурой, системой команд. Появление методики динамического микропрограммирования позволяет достаточно легко изменить "на самом низком уровне" специализацию процессора, его языковую основу и тем самым получать требуемый лисп- , рефал- или снобол-процессор.

Среди возможных способов реализации нового дополнительного языка символьной обработки привлекательной представляется реализация основанная на конвертировании. Она позволяет максимально использовать уже существующее в настоящее время программное обеспечение для ССП и аналогичное обеспечение на других объектных машинах [6.I, 6.3]. С экономической точки зрения это даст возможность при минимальных временных затратах получить мобильную реализацию. С идеологической точки зрения - позволит отработать технологию и методику построения единой виртуальной машины для подмножества языков символьной обработки.

Настоящая работа посвящена рассмотрению такого способа реализации входных языков символьного процессора на примере Лиспа. Полученные результаты позволяют считать конвертирование вполне приемлемым способом реализации дополнительных языков для ССП, При этом авторы, естественно, не исключают других возможных способов реализации.

I. Конвертирование как способ реализации входных языков символьного процессора

В настоящее время при построении систем программирования для описания синтаксиса и семантики языков интенсивно применяются три основные схемы использования универсального метаязыка рекурсивных функций - рефала (L_R). В результате производится соответственно три вида программного продукта:

- по первой схеме на языке L_R создается компилятор с некоторого языка L_1 в другой язык L_2 ($L_2 \neq L_R$) и уровни L_1 и L_2 существенно различны;
- по второй схеме на языке L_R реализуется интерпретатор языка L_1 ;
- по третьей схеме на языке L_R пишется конвертер [6.17] с языка L_1 на язык L_2 и предполагается, что языки L_1 и L_2 одного уровня, например, конвертер АЛГОЛ-ФОРТРАН.

Особый интерес представляет случай $L_2 = L_R$, рассматриваемый в данной работе на примере мобильной реализации языка лисп (L_1). Указанный способ характерен тем, что метаязык используется одновременно и как инструментальный и как объектный язык. Поэтому условно принятый способ реализации (конвертирования) правомерно было бы назвать "компиляцией на себя".

Предлагаемый в работе конвертер можно отнести к конвертерам "с самообеспечением", поскольку, как указано выше, и трансляция и выполнение исходной программы (в нашем случае лисп-программы) осуществляется на одной и той же виртуальной L_R - машине (в отличие от других конвертеров, требующих двух виртуальных машин). В работе [5.18] отмечается, что предложенный здесь способ использования метаязыков в определенном смысле занимает промежуточное положение между первыми двумя схемами. Особенно интересным представляется применение комбинации первой и второй схемы и возможность получения при этом в предельном случае конвертера "с самообеспечением" и более эффективной реализации для входного языка L_1 .

Далее в работе предлагается конкретный пример реализации конвертера с подмножества языка ЛИсп на Рефал (ЛИРА), написанного на рефале и позволяющего получать по исходной лисп-программе функционально эквивалентную ей рефал-программу. Предварительно

детально обсуждаются приложения и характерные особенности лиспа. Достаточно универсальными с точки зрения реализации на ССП дополнительных языков символьной обработки являются приведенные ниже достоинства проекта, а также структура и функции конвертера. Несомненной при таком подходе является общность технологии подключения нового входного языка ССП, а также пути достижения действительно эффективной реализации.

2. Реализация конвертера ЛИСП-РЕФАЛ на символьном процессоре

2.1. Приложения и некоторые характерные особенности лиспа

Алгоритмический функциональный язык ЛИСП [1.1] является одним из наиболее популярных языков символьной обработки, наряду с такими языками, как Рефал и Снобол. Он широко используется:

- для аналитических вычислений (см.распространенные универсальные системы REDUCE и MACSYMA [3.1, 3.2], а также [3.4, 3.5, 3.3];
- для диалогового проектирования вычислительных процессов (пакетов прикладных программ) [5.6, 5.7];
- в задачах анализа текста естественного языка и машинного перевода [4.1 - 4.8];
- для доказательства теорем [4.9, 4.10, 1.1, 1.2] и использования нестандартных исчислений [4.11];
- для компиляции и интерпретации искусственных языков, включая языки программирования и макроязыки [4.1, 5.1 - 5.10];
- в интеллектуальных базах данных и ситуационном управлении [5.2, 4.2, 5.8, 5.9] с привлечением аппарата смешанных вычислений [5.14] в системах принятия решения в условиях неполностью заданной информации [5.15];
- во многих других областях приложения искусственного интеллекта [4.1, 4.2, 4.14].

"Чистый" лисп (без PROG - функции) является в полном смысле языком структурного программирования. Введением же первого расширения лиспа - PROG - функции, превращающей стандартную функциональную запись лисп-программы в нефункиональную в виде последовательности операторов, разработчиками была отдана дань

с одной стороны традиционному программированию, с другой стороны исторически сложившейся архитектуре существующих ЭВМ с более эффективной аппаратной поддержкой операторного программирования. В последнее время создаются новые архитектуры ЭВМ, поддерживающие функциональный стиль программирования и позволяющие при этом получать более эффективные программы.

Для "чистого" лиспа (как и для рефала), являющегося функциональным языком и связанного с обработкой списковых структур, рекурсивных по самой своей природе, естественна рекурсия. Обычно рекурсивное определение более компактно.

В предлагаемом в работе подходе к реализации лиспа существенным образом используется глубокая общность упомянутых языков.

Для языка лисп, как известно, не существует общепринятого стандарта (как, например, для ФОРТРАНа-IУ). Разработчики лисп-системы обычно реализуют ее более предпочтительным для них способом. Следует отметить тот факт, что в мире имеется огромное множество диалектов лиспа, а также способов его реализации на ЭВМ. В процессе данной работы авторы пользовались в основном руководством по широко известной лисп-системе [I.I, I.5] , а также описанием языка и реализации лиспа из [I.2]. На базе вышеупомянутых версий языка имеются довольно крупные библиотеки программного обеспечения на лиспе.

2.2. Цель проекта

Целью предлагаемой работы является:

- предоставление пользователям ССП еще одного распространенного языка символьных преобразований - ЛИСПа;

- получение собственного простого и "открытого" варианта реализации, доступного к расширению, изменению, оптимизации (получение отдельных временных характеристик системы и оценок по затратам памяти) и экспериментам;

- получение мобильного варианта реализации лисп-системы для различных объектных машин (БЭСМ-6, ЕС ЭВМ, Минск-32, БЭСМ-4, М-220, М-222, М-4030, ИРИС-80, IBM-360, SIEMENS-4004, ICL-4/70), в том числе и использование его на символьном процессоре;

- совершенствование внутреннего языка ССП в целях более эффективной реализации желаемого подмножества языков символьной

обработки;
- отработка технологии конвертирования с нового входного языка с использованием метаязыка рефал.

Важно отметить, что для произвольного нового входного языка автоматически становятся доступными все возможности, которые доступны в рефале (более подробно см.ниже).

2.3. Достоинства проекта

Наряду с выбранным методом реализации лиспа, как отмечалось, существует целый ряд других возможных способов его реализации, имеющих как определенные достоинства, так и недостатки. Предложенный в данной работе подход близок к задаче разработки компилирующей системы. В литературе [I.8], например, отмечается, что программы, изготовленные лисп-компилятором, выполняются в I0-I00 раз быстрее, чем при интерпретации, и используют меньше памяти.

Принятый способ реализации имеет следующие преимущества:

I. Простота реализации, основанная на глубокой общности рассматриваемых языков и благодаря этому компактности и обозримости конвертера, а также на мощном сервисном обеспечении рефал-систем для современных ЭВМ.

2. Мобильность и доступность реализации, связанная:
- с разработкой конвертера на метаязыке, входящем в состав систем программирования на различных объектных машинах;
- с использованием в программе конвертера минимальных требований к транслятору с языка рефал (с точки зрения входного языка и машинных операций).

3. Использование единого базового входного языка (рефал). Это обстоятельство представляется удобным с различных точек зрения, среди которых немаловажными являются, удобство эксплуатации и возможность получения единой эффективной [6.I3] параллельной реализации аппликативного языка [6.I4, 6.I5] с учетом методов и технологии [6.I1].

4. Предоставление пользователю объединенной системы програм-

мирования ЛИсп-РефАл (ОСП ЛИРА) на различных объектных машинах. Это обстоятельство представляется одним из наиболее существенных с точки зрения развития и применения различного рода МО, ориентированного на обработку символьной информации.

Объединенная система программирования приводит к взаимному обогащению отдельно взятых систем программирования для ЛИСПа и рефала. Она позволяет:

4.I. Совместно использовать имеющееся МО, разработанное на этих языках.

4.2. Создавать новое программное обеспечение с возможностью выборочного написания отдельных фрагментов программы на более предпочтительном языке.

4.3. Использовать уже имеющиеся на ССП и других объектных машинах СП, в которые включен рефал (редакторы связей, загрузчики, библиотеки объектных модулей и пр.).

Подготовка программ в ОСП ЛИРА на языках лисп и рефал предусматривается в двух режимах:

- раздельная компиляция (лисп-программа обрабатывается конвертером, рефал-программа обрабатывается рефал-компилятором);

- совместная компиляция (программа состоит из фрагментов, написанных на различных языках, обрабатываемых конвертером).

Для объединения программных модулей используется общий редактор связей.

5. Принятый вариант реализации автоматически гарантирует введение в лисп-систему таких мощных средствах как:

5.I. Развитый в рефале эффективный аппарат поиска по образцу, который, как утверждается в литературе, так или иначе приходится вводить в лисп-системы различными способами [4.I, 5.9-5.I2].

5.2. Разнообразные стандартные функции (машинные операции) на объектах машинах для работы:

- с арифметикой с многократной точностью;
- с текстами (размножение, подсчет термов, символов и др.);
- с различными структурами данных (статические, динамические ящики, стеки, таблицы и др.);
- с терминалом.

5.3. Трассировка и отладка лисп-программ, используя аппарат прокрутки рефала.

5.4. Измерение временных характеристик и профиля работы лисп-программ, используя аппарат, имеющийся в рефале. Практика показывает, что такой аппарат позволяет удобно исследовать временные характеристики работы функций и провести необходимые оптимизации. Здесь следует указать на возможность использования (пока еще упрощенных) моделей [6.I5] для исследования ожидаемого ускорения от распараллеливания рефал-программ (т.е. в нашем случае эквивалентных им лисп-программ).

5.5. Сборка мусора с использованием соответствующих средств, заложенных в рефале [6.I].

5.6. Расширение лисп-системы не только средствами рефала, но и за счет использования развитого аппарата машинных операций в рефал-системе, который обеспечивает необходимый стандартный интерфейс языка программирования (лисп) с различными инструментальными языками ССП и других объектных машин (например, ассемблер. Астра, Фортран или микроассемблер, расширяющийся язык микропрограммирования более высокого уровня [6.4, 6.I0] и др.)

5.7. Контекстный макрогенератор для лисп-программ.

Достоинства системы, имеющей в своем составе макрогенератор широко известны. В виде макроопределений могут быть оформлены многие стандартные процедуры, выполняемые при решении определенного класса задач. С использованием макросредств могут быть разработаны дополнительные программы, расширяющие возможности системы или выполняющие самостоятельные функции, могут быть реализованы новые языки программирования.

В настоящей работе благодаря преимуществам принятой схемы реализации лиспа целесообразно использовать рефал с его мощными средствами поиска по образцу, обеспечивающими рекурсивные контекстные макровызовы, в качестве макрогенератора лисп-системы.

I. Макроопределения оформляются в виде цепочки рефал-функций, первой из которых присвоено имя MACRO. Головная функция MACRO и все необходимые связанные с ней макрофункции оформляются отдельным модулем (модулями) лисп-системы и должен быть предварительно скомпилирован рефал-транслятором.

2. После получения очередного исходного S - выражения конвертер предварительно вызывает макрогенерацию, обращаясь

к рефал-функции, имеющей стандартное имя MACRO . Результатом работы вызванной таким образом цепочки рефал-функций будет сгенерированный текст (макрорасширение) на лиспе, которое в свою очередь подвергается обычной обработке конвертером.

3. Конвертер инициирует обращения к функции MACRO до тех пор, пока не встретит в анализируемом тексте особого признака конца генерации (/ENDM/) либо сигнала об ошибке, обнаруженной на этапе макрогенерации (/ERRM/).

Перечислим несколько простейших случаев приложения макросредств в предлагаемой системе:

- использование в исходной лисп-программе арифметических и прочих функций в естественной (математической) записи, например: M! вместо (FACT M);

- временное запоминание отдельных фрагментов исходной лисп-программы с последующим их восстановлением (реализуется, например, через копилку рефала);

- использование нового исходного языка (в частности, настройка системы на желаемый диалект лиспа), который сводится к выбранному стандарту, например, иллюстрируется макроопределением:

```
MACRO
EO('DEF 'EF(EX)EY) = ('SEXPR 'EF('LAMBDA'(EX)EY))
EI = /ENDM/ EI
```

- расширение возможностей программиста, связанное, например, с введением стандартных списков в лисп-программах, записью комментариев и пояснений в произвольной точке исходной программы и пр.

Чрезвычайно важным является тот факт, что предлагаемый аппарат макрогенерации позволяет достаточно легко приспособить и применить лисп-систему в решении задач, актуальных для пользователей.

5.8. Использование аппарата верификации программ [6.18].

3. Структура и функции конвертера

Конвертер ЛИРА (K_R), написанный на рефале, как подчеркивалось, осуществляет перевод лисп-программы P_L , обрабатывающей некоторые исходные данные D' (заданные "статически" с помощью

QUOTE - функции), в функционально эквивалентную рефал-программу P_R, т.е. формально выполняется следующее преобразование:

$$P_R(D')=K_R(P_L(D'))$$

Собственно конвертер состоит из группы операторов, осуществляющих ввод исходного текста $P_L(D')$ на лиспе, элементарный синтаксический контроль и сжатие его. Вслед за этим производится анализ исходного выражения и подробная проверка скобочной структуры S - выражений лиспа.

Далее выполняется генерация предложений рефала для $P_R(D')$ осуществляющих преобразования, эквивалентные исходным, и вывод их в файл для последующей обработки рефал-транслятором. На последнем этапе производится подключение базовых функций лиспа, подготовленных на рефале.

В предлагаемой системе специально реализованы на рефале следующие встроенные базовые функции лиспа: **QUOTE, CAR, CDR, CONS, LIST, PRINT, ADDI, SUBI** встроенные предикаты: **ATOM, EQ, NULL, NOT, OR, AND, ZEROP, ONEP,** функция **COND**, а также композиции функций **CAR, CDR** практически произвольной глубины (например, **CADDAR**). Функции, определяемые программистом специально для решения поставленной перед ним задачи, могут строиться на базе вышеперечисленных встроенных функций.

Процесс генерации конвертера K_C (на языке сборки С) для ССП или другой объектной машины (ОМ) можно представить схематично (рис.I). На первой стадии производится получение программы конвертера K_C' на языке сборки после обработки текста этой программы, написанной на рефале (K_R). Затем выполняется сборка-редактирование (S) готовой программы конвертера K_C на объектной машине с подключением готовой библиотеки машинных операций на ОМ.

На рис.2 схематично проиллюстрирована работа программного комплекса, включающего конвертер, на объектной машине.

Процесс работы конвертера K_C на интерпретаторе языка сборки С на объектной машине осуществляется в несколько этапов.

На I этапе, как отмечалось, конвертером K_C выполняется перевод лисп-программы P_L, содержащей "статически" определенные исходные данные D' (введенные в P_L через **QUOTE**), в функционально эквивалентную рефал-программу $P_R(D')$ с использо-

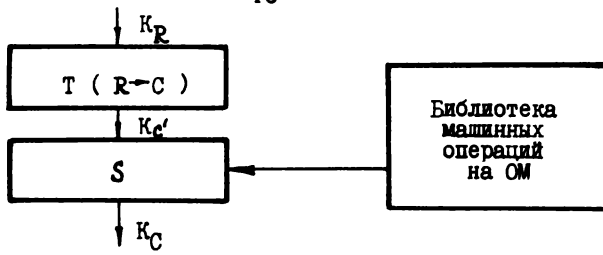

Рис.I Генерация конвертера ЛИРА на объектной машине.

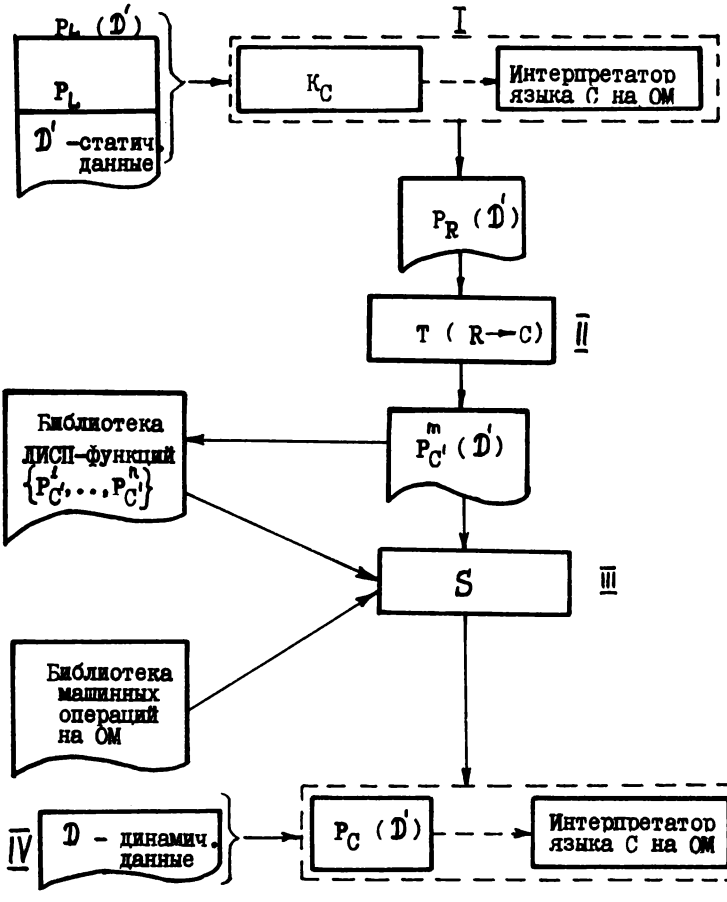

Рис.2 Схема работы системы на объектной машине.

ванием интерпретатора языка сборки на объектной машине.

На II этапе последняя обрабатывается транслятором (компилятором) Т с рефала на язык сборки, в результате чего создается модуль исходной программы на языке сборки $P^{m}_{C}(D')$. Указанный модуль может быть записан в библиотеку лисп-функций на языке сборки или сразу же отправлен на очередную стадию обработки.

На III этапе осуществляется редактирование и сборка готовой скомпилированной программы $P_C(D')$ с использованием имеющейся библиотеки лисп-функций и библиотеки машинных операций на ОМ.

На заключительном IУ этапе программа $P_C(D')$ выполняется через интерпретатор языка сборки на ОМ, в случае необходимости (по логике работы программы) на этой стадии осуществляется ввод "динамических" данных в программу пользователя (через **READ** - функцию).

В дальнейшем целесообразно расширение системы и совершенствование лисп-машины (см.п.4). Особый интерес представляет направление работ, связанное с непосредственной микропрограммой (аппаратной) интерпретацией основных базисных функций лиспа, удельный вес которых по полученным данным составляет более 40% от общего времени счета.

В дополнении к рассмотренному описанию в системе можно выделить следующие две компоненты:

I. Лисп - компилятор правомерно представить состоящим из лисп-конвертера K_R и трнаслятора (компилятора) Т из рефала в язык сборки ($R \longrightarrow C$) на объектной машине. Тогда преобразования исходного текста (лисп-программы) в результирующий (программу на языке сборки) может быть описано следующим образом:

$$P_R(D') = K_R(P_L(D'))$$
$$P_C(D') = T(P_R(D'))$$

или, что эквивалентно:

$$P_C(D') = T(K_R(P_L(D')))$$

где: $P_L(D')$ - исходная лисп-программа;

$P_R(D')$ - эквивалентная конвертированная программа;

$$P_R(D') \sim P_L(D');$$

$P_C(D')$ - эквивалентная скомпилированная программа:

$$P_C(D') \sim P_L(D').$$

2. Лисп-интерпретатор можно представить как совокупность специально скомпилированных рефал-функций $\{F^i_C\}$ на языке сбор-

ки и интерпретатора языка сборки на объектной машине.

4. Оценка эффективности реализации лиспа на макете ССП

Теперь оценим выигрыш, который можно получить при исполнении лисп-программы на символьном процессоре по сравнению с ее исполнением на программной лисп-системе, работающей на ЭВМ, аппаратура которой сопоставима с аппаратурой символьного процессора. Для проведения такой оценки будем использовать гипотетическую программную лисп-систему для ЭВМ ЕС-1030. Ее производительность рассчитаем ниже исходя из измеренной производительности программной лисп-системы ВЦ АН СССР [I.2, 2.I-2.4] для ЭВМ БЭСМ-6.

Для проведения оценки были выбраны 3 программы на языке лисп (предлагалось, что удельные веса этих программ одинаковы для смеси задач):

I) Перестановка элементов списка PERM(N)[I.3]. Список содержит N элементов и задается явно через свои элементы функцией QUOTE .

Испытывалось два примера: PERM(4) и PERM(5) .

2) Доказательства теорем общезначимости логических формул исчисления предикатов I-го порядка без кванторов. Для этих целей была использована программа из [I.I], работающая по алгоритму Вана. Программа содержит более I0 функций.

Для доказательства использовались две теоремы:

TI: $P \longrightarrow P \vee Q$

T2: $A \vee \neg B \longrightarrow (P \wedge Q) \longrightarrow (P \equiv Q)$

3) Дифференцирование полиномов. Сами полиномы задаются в функциональной скобочной структуре [3.3].

Замеры времени работы программ проводились на ССП (макет) с установленной системой лисп-лира и на БЭСМ-6 в системе "Пульт" лисп-ВЦ.

Для соблюдения чистоты эксперимента с точки зрения исследования сопоставимых программ и условий их функционирования была выполнена следующая подготовительная работа:

I. Написаны дополнительные функции, которые обеспечивают I0-20 кратное обращение к испытуемым программам.

2. Измерения проводились без подключения программ печати

(функций **PRINT**).

3. Были записаны на лиспе те стандартные функции, которые не являются базовыми для лисп-лира:

EQUAL, MEMBER, CADR, CADDR

4. По каждому представленному примеру в процессе работы лисп-программ через рефал была получена на БЭСМ-6 максимальная длина занимаемого поля зрения. Т.о. была определена длина потребляемой списковой памяти.

При вызове лисп-ВЦ заказывалась соответствующая списковая память под отдельные примеры, Тем самым делалась попытка поставить испытуемые программы в равные условия по допустимой свободной памяти. Окончательные итоги замеров сведены в **таблицу** табл.I.

Анализ результатов (табл.I) показывает, что в настоящее время лисп-ВЦ (**компилятор**) работает в 5-I0 раз быстрее, чем лисп-ЛИРА на макете.

Время исполнения произвольной лисп-программы на гипотетической системе для ЕС-I030 можно представить так:

$$T_{EC} = T_{66} \cdot K = T_{66} \cdot K_S \cdot I/K_a$$

где T_{66} - время исполнения той же программы на лисп-системе для БЭСМ-6;

$K_S = \frac{P_{66}}{P_{EC}} = II.7$ - отношение среднего быстродействия ЭВМ БЭСМ-6 и ЕС-I030;

K_a - коэффициент эффективности системы команд.

Коэффициент K_a определен экспериментально и для программной реализации рефала на БЭСМ-6 и ЕС-I030 составляет I.6. Полагая, что требования к системе команд для реализации лиспа не должны существенно отличаться от требований для реализации рефала, примем $K_a = I.6$. Теперь рассчитаем коэффициент $K = K_S/K_a = 7.4$, на который нужно увеличть времена выполнения тестовых программ, полученные на БЭСМ-6, чтобы определить соответствующие оценки для гипотетической лисп-системы на ЕС-I030. Времена выполнения рассматриваемых тестовых примеров на макете символьного процессора и гипотетической лисп-системе для ЕС-I030 приведены в табл.2. Там же представлены соотношения времен исполнения программ для этих двух реализаций. В среднем **исполнение** лисп-программы на макете требует на 8% меньше времени, чем

ПРИМЕРЫ	ЛИСП-ВЦ БЭСМ-6 (компилятор) $T_{вц.к}$ (сек.)	Лисп-Лира для МАКЕТа ССП $T_м$ (сек)	ОТНОШЕНИЕ $T_м/T_{вц.к}$
Перестановки: **PER(4)**	0.23	2.3I7	I0.I
PER(5)	I.45	I4.59	I0.06
Док.теорем: TI	0.0225	0,09I7	4,075
T2	0,075	0,443	5,906
Дифференц,полиномов	0,I5	0,6	4,0

Табл.I. Таблица сопоставления времени работы ЛИСП систем (ЛИСП-ВЦ БЭСМ-6, ЛИСП-ЛИРА МАКЕТ) при сопоставимых объемах списковой памяти

ПРИМЕРЫ	Гипотетическая лисп-система для ЕС-I030 $T_{ЕС}$ (сек)	Лисп-ЛИРА для макета ССП $T_м$ (сек)	Отношение $\frac{T_м}{T_{ЕС}}$
Переостановки: **PER(4)**	I,702	2,3I7	I,36I
PER(5)	I0,73	I4,59	I,360
Док.теорем: TI	0,I67	0,09I7	0,549
T2	0,555	0.443	0,798
Диффер.полиномов	I,I	0,6	0,55
		Среднее	**0.92**

Табл.2. Сопоставление времен работы гипотетической лисп-системы для ЕС-I030 и лисп-ЛИРА для макета ССП.

на гипотетической системе для ЕС-1030.

Приведенные здесь измерения выполнены на первом варианте реализации лиспа на макете символьного процессора. Настоящий вариант может быть значительно улучшен путем осуществления ряда оптимизаций программы конвертера и операторов символьного процессора.

Перечислим первоочередные резервы для совершенствования:

1. Перевод базовых функций в микрокод.

Приведенные измерения показали, что время работы базовых функций в лисп-ЛИРА составляет 35-49% от общего времени работы, а с учетом функций **EQUAL, MEMBER** и пр, время их работы составляет 43-60%.

2. Оптимизация результирующей рефал-программы:

- уменьшение числа шагов результирующей программы;
- ликвидация лишних скобок. (Работа операторов **BL, BR, СКОБ** в сумме составляет около 36% времени всей работы.) Результаты выполнения рефал-программы, прооптимизированной вручную по предполагаемым алгоритмам оптимизации, показывают, что это должно обеспечить ускорение как минимум в 2 раза.

Суммарный эффект от введения указанных выше усовершенствований может привести к сокращению в 4 раза времени выполнения лисп-программы на макете символьного процессора. Это означает, что лисп-программа будет выполняться на макете ССП примерно в 4 раза быстрее, чем на гипотетической программной лисп-системе для ЭВМ ЕС-1030.

К дополнительным резервам получения ускорения следует отнести:

1. Введение нового типа данных в рефал (ССП) - символа—ссылки на списки с непосредственным доступом (без лишнего шага). Это позволит ликвидировать значительные потери на операторы **MULE, TPLE** (13 - 22%).

Исследование и выбор оптимального способа трансформации структур данных лиспа в структуры данных рефала. При этом в целях достижения наиболее эффективной работы конкретных лисп-программ предполагается возможность осуществления различных способов представления объектных структур даже для одних и тех же структур данных исходного языка. Для организации как управляемого так и неуправляемого (автоматического) отображения

структур данных и функций актуальна разработка метаязыка спецификаций.

2. Введение новых операторов языка ССП, вызванных данной реализацией (вопрос исследуется в настоящее время).

3. Совершенствование архитектуры специализированного процессора в целях более эффективной реализации определенных операторов внутреннего языка ССП, определяющих суммарное время работы объектных программ представленной лисп-системы.

В настоящее время конвертер ЛИРА находится в стадии комплексной отладки. Его объем составляет примерно 400 строк на языке рефал. Время разработки и отладки программы составило приблизительно 3 чел.мес.

З А К Л Ю Ч Е Н И Е

I. В данной работе (на примере языка лисп) опробирован и предложен перспективный метод расширения класса входных языков специализированного символьного процессора путем построения конвертера "с самообеспечением".

Кроме достижения мобильной и открытой для пользователя реализации указанный подход позволяет:

- определить преимущества и недостатки базисного языка (в нашем случае рефала) ;
- отработать технологию получения единой виртуальной машины, предназначенной для символьной обработки (как верхний так и нижний ее уровень).

Эти вопросы являются предметом дальнейших исследований.

2. Перечисленные возможности, полученные результаты, а также интересные перспективы развития лисп-системы, по-видимому, характеризуют предлагаемую реализацию как достаточно удачную.

3. К первоочередным задачам по совершенствованию реализации авторы относят расширение и оптимизацию безисного варианта языка лисп.

При подготовке рукописи к печати была обнаружена интересная работа, посвященная реализации на языке ЛИСП интерпретатора с языка РЕФАЛ [4.I5]. Очевиден, однако, тот факт, что ведущая операция в языках подобного рода (отсутствующая в лиспе) - сопоставление с образцом выполняется в указанном интерпретаторе на уровне лисп-функций, а следовательно, чрезвычайно медленно, о чем упоминается в работе [4.I5]. Представляется более обоснованным выбор именно рефала (а не лиспа) в качестве базового языка для реализации других алгоритмических языков (в частности языков символьной обработки) в силу его значительно большей мощности и общности [4.I2, 4.I3, 5.I9, 6.5]. Рефал предоставляет программисту более тонкий инструмент и разрешает большую детализацию операций обработки символьных структур при тех же преимуществах, что и лисп (в частности, малой трудоемкости при подготовке программ). Указанное мнение укрепляет перспектива создания действительно мобильной реализации лиспа (не зависящей ни от особенностей конкретной машины, ни от характера используемой на ней операционной системы), а также накопленный опыт использования рефала в качестве базового языка специализированного символьного процессора.

Авторы выражают особую признательность д.т.н. А.Н.Мямлину за постоянное внимание к данной работе и содействие ее выполнению.

Л И Т Е Р А Т У Р А

Общие вопросы символьной обработки и программирование на языке ЛИСП

I.I. **McCarthy J., Bobrow D.G., Levin M.I. LISP I.5 Programmer's Manual. Cambridge., Mass., MIT Press., I962.**

I.2. Лавров С.С., Силагадзе Г.С. Автоматическая обработка данных. Язык ЛИСП и его реализация. М.: Наука, I978.

I.3. Семенова Е.Т. Язык программирования LISP I.5, М.: МЭИ, I977.

I.4. Пратт Т. Языки программирования: разработка и реализация. М.: Мир, I979.

I.5. Мауер У. Введение в программирование на языке ЛИСП. М.: Мир, I976г.

I.6. Хигман Б. Сравнительное изучение языков программирования. М.: Мир, I974.

I.7. Баррон Д. Рекурсивные методы в программировании. М.: Мир, I974.

I.8. Фостер Дж. Обработка списков, М.: Мир, I974.

I.9. Жиров В.Ф. Математическое обеспечение и проектирование структур ЭВМ. М., Наука, I979, с. 52-59.

I.I0. Языки программирования. Под ред.Ф.Женюи. М., Мир, I972, с. 278-343.

I.II. Грисуолд Р., Поудж Дж, Полонски И. Язык программирования СНОБОЛ-4. М., Мир, I980.

I.I2. Юдин С.Ф. Введение в алгоритмический язык ЛИСП, Л.: ЛПИ.. I977, 43 с.

I.I3. **Ribbens R. Programmation non numerique LISP I.5., Paris.,. I970.**

Вопросы реализации лисп-систем

2.I. Лавров С.С., Силагадзе Г.С. Входной язык и интерпретатор системы программирования на базе языка ЛИСП для машины БЭСМ-6, М., Изд-во АН СССР. I969.

2.2. Брябрин В.М., Ковалева В.А., Сафонов В.И., Филипов В.И., Юра В.М. Трансляция и отладка программ в режиме непосредственного доступа. Сообщения по выч.математике в. 9, М: ВЦ АН СССР, 1974, с. 7-12.

2.3. Юра В.М. Развитие системы программирования лисп БЭСМ-6- В сб.: Обработка символьной информации, вып. I., М.: ВЦ АН СССР, 1973, с. 5-16.

2.4. Юра В.М. О новых функциях в системе ЛИСП-БЭСМ-6. - В сб. Обработка символьной информации. вып. 4, М.: ВЦ АН СССР, 1978, с. 26-50.

2.5. Пантелеев А.Г. Об интерпретаторе с языка ЛИСП для ЕС-ЭВМ и SYSTEM 4 (ICL). Программирование, 1980. № 3, с. 86-87.

2.6. Закс М.Б., Посохин И.К.. **Интерпретация ЛИСП-ПЛ/I для ЕС ЭВМ.** см. (3.1), с. 42-45.

2.7. Волчков В.М., Грудцын С.Н., Калиниченко П.А., Ростовцев В.А. Адаптация системы **MLISP-REDUCE** на ЭВМ **ICL** - 1906А там же с. 36-41.

2.8. **Stoyan H. The TU-LISP system for DOS and OS.** там же с. 36-41.

2.9. **Griss M.L., Swanson M.R. MBALM/I700 : A Microcoded LISP Machine for the Burroughs BI726. Proc. of Micro-10., ACM., N.Y. (1977),15.**

2.10. **Griss M.L., Kessler R.R. REDUCE/I700 : A Microcoded Algebra System., Proc. of Micro-11., ACM., N.Y. (1978). 130-138.**

2.11. **Stoyan H. LISP - Anwendungsgebichte., Grundbegrifte, Geschichte. Berlin., 1980.**

2.12. Воронов С.В., Горбачик А.П., Гриб О.В., Дейнека В.К. Проблемы реализации систем программирования в рамках системы построения языковых процессоров. - сб.: Теоретические основы компиляции. Н., НГУ., 1980, 151-157.

2.13. Курбатов С.С., Байдун В.В., Сукумаран Наир Ч.Г. О реализации интерпретирующей системы с языка **LISP** на машинах серии ЕС ЭВМ. Труды МЭИ, в. 295, М., 1976.

2.14. **Griss M.L., Hearn A.C. A portable LISP Compiler. Software-Practice and Experience., Vol.11., 541-605., (1981).**

Аналитические преобразования на языке ЛИСП

3.I. Труды международного совещания по системам и методам аналитических вычислений на ЭВМ и их применению в теоретической физике, Дубна, I980.

3.2. Гердт В.П., Тарасов О.В., Ширков Д.В.. Аналитические вычисления на ЭВМ в приложении к физике и математике.
там же, стр. I22;
УФН, I980, I30, I, стр. II3-I47.

3.3. Закс М.Б. Язык ЛИСП и его применение к аналитическим преобразованиям на ЭВМ. Саратов, Саратовск Ун-т, I979, с. 96.

3.4. Абрамов С.А. **PR** - система для произведения действий над рациональными выражениями. В сб.Алгоритмы и алгоритмические языки, в.6, ВЦ АН СССР, I973, с. 88-I02.

3.5. Абрамов С.А. Система АПРЕФ, - сб. Обработка символьной информации, в. I, М.: ВЦ АН СССР, I973, с. I7-34.

Искусственный интеллект и проблемы машинного перевода

4.I. Уинстон П.. Искусственный интеллект, М.: Мир, I980.

4.2. Кузин Л.Т. Основы кибернетики т.2. М.: Энергия, I979.

4.3. Шенк Р. Обработка концептуальной информации. М.: Энергия, I980, с.360.

4.4. Вудс В.А. Сетевые грамматики для анализа естественных языков. Киб. сб., нов.сер., в. I3, М.: Мир, I976, с. I20-I58.

4.5. Кулагина О.С. Исследования по машинному переводу. М.: Наука, I979, с. 79-83.

4.6. Мальковский М.Г. Программы, понимающие естественный язык - Сб. - Обработка символьной информации, в. I, М.: ВЦ АН СССР, I973, с. 73-II5.

4.7. Мальковский М.Г. Программа **APRIL** , решающая арифметические задачи в словесной формулировке. В сб.Алгоритмы и алгоритмические языки, в.6, М.: ВЦ АН СССР, I973. с. II3-I59.

4.8. Виноград Т. Программа, понимающая естественный язык. М.: Мир, I976.

4.9. **Boyer R.S., Moore J.S. A computational logic. Ass. Press., I979.**

4.I0. Несговорова Г.П. Перевод формул исчисления предикатов первого порядка в префиксный вид. В сб.Выч.системы, вып.59, Новосибирск, I974, с. I39-I50.

4.II. Литвинцева Л.В. Временная логика в работах и диалоговых схемах. Тезисы докл. и сообщ. Всесоюзный конф. "Методы математической логики в проблемах искусственного интеллекта и системное программирование". Вильнюс, ИМиК АН Лит ССР, I980, с. I23-I24.

4.I2. Хорошевский В.Ф., Красовский А.Г., Флоренцев С.Н. Программное обеспечение моделей искусственного интеллекта. в кн.: Искусственный интеллект. Итоги и перспективы., М., МДНТП., I974.

4.I3. Кузин Л.Т., Храмов А.А. Вопросы использования метаязыков для программного обеспечения системы искусственного интеллекта. В сб. Инженерно-математические методы в физике и кибернетике, в.6., МИФИ., М., Атомиздат., I977.

4.I4. Труды IУ Международной объединенной конференции по искусственному интеллекту. М.: АН СССР, I975.

4.I5. Сукумаран Наир. Ч.Г. Исследование возможностей языка LISP при реализации языков искусственного интеллекта. Автореф. дисс. М.,: МЭИ, I977.

Применение языка ЛИСП в трансляции, управлении вычислениями, ситуационном управлении и других областях знаний

5.I. Брябрин В.М. Обработка синтаксических деревьев в системе автоматического конструирования трансляторов. - сб. Обработка символьной информации, в., I, М.: ВЦ АН СССР, I973, с. 35-57.

5.2. Брябрин В.М. Система управления интеллектуальной базой данных на ЛИСПе. - Сб. Обработка символьной информации. в.4, М.: ВЦ АН СССР, I978, с. 75-83.

5.3. Голиков К.П., Педанов И.Е., Реализация языка ГЕОМАЛ, там же, с. I08-I28.

5.4. Серебряков В.А. Входной язык метапроцессора системы ЛОРД, там же с. II-25.

5.5. Абрамов С.А. Некоторые вопросы трансляции в ЛИСПе, там же с. 60-65.

5.6. Петренко А.К. Интерактивная систе а общения с задачей МИМОЗА. Тез. докл. I-й межд.конф. молодых ученых "Проблемы проектирования и применения дискретных систем в управлении" Минск, I977, п. 445-447.

5.7. Гнездилова Г.Г. Метод диалогового проектирования вычислительных процессов. Программирование., № 3, I980, с. 44-5I.

5.8. Лозовский В.С.. Ситуационная и дефиниторная семантика системы представления знаний. - Кибернетика, I979, № 2, с. 98-I0I.

5.9. Лозовский В.С. Задание реляционной базы данных в виде мультисети и реализации поиска по образцу. - В сб."Информационное программное обеспечение систем ситуационного управления". Препринт ИК АН УССР, № 78-I4, Киев, I978.

5.I0. Пильщиков В.Н. Строчные преобразования в языке ЛИСП. В сб.Алгоритмы и алгоритмические языки, в.6, М.: ВЦ АН СССР, I973, с. I60-I88.

5.II. Harrison M.G. BALM - an extendable list-processing language. AFIPS., I970., V.36., 507-5II.

5.I2. Griss M.L., Cowan R.M. Hashing -the key to rapid pattern matching. "Lect. Notes Comput. Sci.".,I979. 72. 266-278.

5.I3. Guzman A., McIntosh M.V., CACM., I966.,9,8.,pp.604-6I6.

5.I4. Ершов А.П. о сущности трансляции - Программирование, № 5, I975.

5.I5. Бабич Г.Х. Математическое обеспечение АСУ цветной металлургии при неполной информации М.: Металлургия, I975.

5.I6. The Programming Language LISP : Its Operation and Applications. Edited by Berkeley E.C. and Bobrow D.G., The MIT Press., Cambridge., Mass., I967.

5.I7. Программирование на ЛИСПе. М., ВЦ АН СССР, I978.

5.I8. Клещев А.С. Семантические структуры алгоритмических языков - в кн. Средства реализации систем искусственного интеллекта, В.: ИАПУ АН СССР., I978., с. 65-7I.

5.I9. Клыков Ю.И., Горьков Л.Н. Банки данных для принятия решений.М., Сов.Радио., I980, с.58-59.

5.20. Allen J. Anatomy of LISP. McGRAW-Hill Book Company. I978.

Вопросы реализации и применения специализированного символьного процессора

6.I. Базисный рефал и его реализация на вычислительных машинах (Методические рекомендации)., М., ЦНИПИАСС, I977, Вып.У-40.

6.2. Базисный рефал. Описание языка и основные приемы программирования (Методические рекомендации). М., ЦНИПИАСС, I974, вып. У-33.

6.3. Романенко С.А. Машинно-независимый компилятор с языка рекурсивных функций, Диссертация, М., I979.

6.4. Климов А.В., Романенко С.А. Рефал в мониторной системе "Дубна" БЭСМ-6, Интерфейс рефала и фортана. ИМП АН СССР. М.I975.

6.5. Задыхайло И.Б., Котов Е.И., Мямлин А.Н., Поздняков Л.А., Смирнов В.К.. Вычислительная система с внутренним языком повышенного уровня. ИПМ АН СССР, Препринт № 4I, М. I975.

6.6. Мямлин А.Н., Смирнов В.К., Ковалев Э.С., Меламед В.И., Рубин А.Г., Тульский В.П., Пржебловская С.К. Специализированный символьный процессор. Всесоюзная конференция. Технология программирования. Тезисы докладов.ИК АН УССР., Киев, I979.

6.7. Мямлин А.Н., Смирнов В.К., Задыхайло И.Б. Процессор для обработки текстовой информации. Доклады Всесоюзной конференции "Параллельное программирование и высокопроизводительные системы", Новосибирск, I980.

6.8. Озеркова В.Г., Рубин А.Г., Синицын В.Н. Подключение специализированного микропрограммного процессора к ЕС ЭВМ через стандартный интерфейс ввода-вывода. I-я Международная конфе-

ренция молодых ученых по проблемам проектирования и применения дискретных систем в управлении. Минск., 1977.

6.9. Рубин А.Г., Смирнов В.К., Каминская К.В. Запуск задач в пакет из диалогового редактора для алфавитно-цифровых дисплеев в ОС ЕС ЭВМ. ИПМ АН СССР. Препринт № 47, М., 1978.

6.10. Наумов Н.А., Смирнов В.К. О синтезе и отладке микропрограммных алгоритмов. Всесоюзная конф. Синтез, тестирование, верификация и отладка программ. Рига., 1981г.

6.11. Наумов Н.А. Некоторые вопросы структурного проектирования Препринт № 63, ИПМ АН СССР., М., 1980.

6.12. Задыхайло И.Б., Мямлин А.Н., Смирнов В.К., Эйсымонт Л.К. Об эффективной аппаратной реализации языка для описания объектов на уровне понятий и символьных преобразований. В кн.: Искусственный интеллект. Итоги и перспективы., М., МДНТП., 1974.

6.13. Задыхайло И.Б., Котов Е.И., Красовский А.Г., Мямлин А.Н., Смирнов В.К. О повышении эффективности символьных преобразований. Препринт № 15, ИПМ АН СССР, М., 1975.

6.14. Эйсымонт Л.К. О возможности параллельных схем реализации одного языка для программирования задач переработки текстовой информации, УСИМ, 1977, № 2, с. 56-64.

6.15. Задыхайло И.Б., Мямлин А.Н., Платонова Л.Н., Эйсымонт Л.К. Исследование процессов параллельного выполнения компилирующих программ некоторого типа, Препринт № 124, ИПМ АН СССР, М., 1980.

6.16. Проскурин М.И., Смирнов В.К., Юдина М.Л. Микропрограммный процессор. Препринт ИПМ АН СССР, № 27, 1976.

6.17. Кузин Л.Т., Флоренцев С.Н., Перминов О.Н. Вопросы программной и информационной совместимости ЭВМ Минск-32 и ЕС ЭВМ.В сб. "Обработка данных в системах управления"., М., МДНТП, 1973.

6.18. Ефимкин К.Н., Задыхайло И.Б. О верификации программ на одном языке. - Программирование., 1980., № 2, с.69.

Н.А. Наумов, А.Г. Рубин, В.К. Смирнов " Об одном способе реа – лизации входных языков для символьного процессора. "
Редактор В.И. Дрожжинов. Корректор С.М. Шаменко.

Подписано к печати 12.11.81 г. № Т- 29519. Заказ № 385.
Формат бумаги 60X90, 1/16. Тираж 200 экз.
Объем 1,4 уч.-изд.л. Цена 10 коп.

055 (02)2

Отпечатано на ротапринтах в Институте прикладной математики АН СССР
Москва, Миусская пл. 4.

Ссылки на издание рекомендуется делать по следующей форме: и.о., фамилия, название, препринт Ин. прикл. матем. им. М.В. Келдыша АН СССР, год, №.

Распространение: препринты института продаются в магазинах Академкниги г. Москвы, а также распространяются через Библиотеку АН СССР в порядке обмена.

Адрес: СССР, 125047, Москва-47, Миусская пл. 4, Институт прикладной математики им. М.В. Келдыша АН СССР, ОНТИ.

The references should be typed by the following form: initials, name, title, preprint, Inst.Appl.Mathem., the USSR Academy of Sciences, year, N(number).

Distribution. The preprints of the Keldysh Institute of Applied Mathematics, the USSR Academy of Sciences are sold in the bookstores "Academkniga", Moscow and are distributed by the USSR Academy of Sciences Library as an exchange.

Adress: USSR, I25047, Moscow A-47, Miusskaya Sq.4, the Keldysh Institute of Applied Mathematics, Ac.of Sc., the USSR, Information Bureau.

Цена 10 коп.