ОБЪЕДИНЕННЫЙ ИНСТИТУТ ЯДЕРНЫХ ИССЛЕДОВАНИЙ
Лаборатория Вычислительной Техники и Автоматизации

На правах рукописи

Карначук Валерий Иванович

Управление данными в программах математической физики

Специальность 01.01.10 - математическое обеспечение вычислительных машин и систем

Автореферат диссертации на соискание ученой степени кандидата физико-математических наук.

Дубна, 1979.

Теоретической и прикладной

Научные руководители работы:
академик Н.Н.Яненко,
кандидат физико-математических наук А.Н.Коновалов.

Официальные оппоненты:
доктор физико-математических наук В.П.Шириков,
кандидат физико-математических наук В.Ф.Тюрин.

Ведущая организация
Институт Кибернетики АН УССР

Защита состоится "___"_____1979г.

в_____ часов на заседании Специализированного Совета Д047.01.04. Лаборатории Вычислительной Техники и Автоматизации ОИЯИ, г. Дубна, Московской области.

Автореферат разослан " ___ "_____1979г.

Ученый секретарь Специализированного Совета
к.ф-м.н. З.М.Иванченко

Актуальность проблемы. В программах для решения задач математической физики объем набора данных определяется количеством узлов сетки, в которых вычисляются значения искомых функций. В настоящее время актуальным является вопрос создания производственных программ, применяемых при математическом моделировании физических объектов и процессов. При этом требуется проведение расчетов на сетке большого размера, позволяющей более точно учесть геометрию области, в которой решается задача, или найти решение с более высокой точностью. Поэтому для хранения данных используются внешние запоминающие устройства (ВЗУ). При организации обменов между ОЗУ и ВЗУ время работы программы резко возрастает как из-за большого числа операций обмена, так и из-за того, что каждой операции обмена всегда сопутствует дополнительная работа (пересылки данных внутри ОЗУ, ведение служебных каталогов и пр.).

Предлагаемая диссертация посвящена вопросам построения систем управления данными в программах математической физики, основанных на понятии виртуальной памяти*). В диссертации рассматриваются: моделирование процесса обработки данных, сегментация набора данных, разработанные системы SYDAK и СВП, которые предназначены для управления данными в программах математической физики на ЭВМ БЭСМ-6, а также методика измерения эффективности этих систем и некоторые экспериментально полученные оценки эффективности.

Цель работы состоит в исследовании процесса управления данными в программах математической физики и в создании (на основа-

---

*) Л.Н.Королев. Структуры ЭВМ и их математическое обеспечение. М., "Наука", 1974, ч. II, гл. 2.

ними результатов исследования) системы управления данными для указанного класса программ. Эта система должна обладать высокой эффективностью, в частности малым расходом времени центрального процессора на организацию обменов, а также простотой сопровождения и использования.

Научная новизна работы заключается в следующем:

а) На уровне программы пользователя вводится однородная виртуальная память большого объема; в отличие от других подходов, вводящих виртуальную память на уровне операционной системы или системы программирования, достигается более высокая эффективность;

б) При разработке систем управления данными применяется проблемноориентированный подход, учитывающий специфику программ математической физики, что позволяет провести теоретическое и экспериментальное обоснование основных конструкторских решений, что как правило невозможно при разработке универсальных систем.

Практическая ценность заключается в разработке систем SYDAK и СНП, применяемых при создании производственных программ математической физики. Система SYDAK предназначена для программирования двумерных задач. СНП рассчитана на применение в двумерных и трехмерных программах, по сравнению с SYDAK она обладает дополнительными сервисными возможностями и более высокой эффективностью.

Апробация работы. Основные результаты докладывались на:

а) IУ Всесоюзном семинаре по комплексам программ математической физики (г. Таллин, I975);

б) конференции по программному обеспечению БЭСМ-6 (г. Москва, I975);

в) заседаниях Совета Секции пакетов прикладных программ (г. Миасс, I977, I978);

а также на семинарах отделения программирования ВЦ СО АН СССР, отдела численного решения задач математической физики ИТПМ СО АН СССР, факультета вычислительной математики и кибернетики МГУ, Лаборатории Вычислительной Техники и Автоматизации ОИЯИ (г. Дубна). Инструкция по системе SYDAK [I] включена в ФАП СО АН СССР.

Система SYDAK передана для внедрения в ВЦ СО АН СССР (Новосибирск), НИИ механики при ГГУ (Горький), НИИАР (Димитровград), ОИЯИ (Дубна).

Основные результаты работы опубликованы в [I-6].

На защиту выносятся:

а) метод создания систем виртуальной памяти, имеющий программную реализацию;

б) метод создания систем управления данными в программах математической физики, основанный на принципе виртуальной памяти.

Объем работы. Диссертация содержит 137 страниц текста, 28 рисунков (из них 7 графиков), 5 таблиц.

Диссертация состоит из введения, трех глав, заключения и приложений.

СОДЕРЖАНИЕ РАБОТЫ

В первой главе рассматриваются вопросы моделирования процесса обработки данных, а также математические постановки задачи сегментации и задачи оптимизации процесса обменов.

В первом параграфе описан способ построения модели процесса обработки данных в виде графа:

$$\Gamma = (\Omega, G),$$

где Ω – множество данных, а G –множество ребер, которое определяется отношением следования R, заданным в Ω. Вершины ω_i, ω_j связаны отношением R, если ω_i обрабатывается

непосредственно вслед за ω_j *).

На ребрах графа Γ заданы веса v_{ij}, которые указывают кратность отношения R. В общем случае модель этого вида является вероятностной из-за неоднозначного определения отношения R. Тогда, рассматривая процесс обработки как марковскую цепь с множеством состояний Ω, веса v_{ij} можно вычислить, используя матрицу переходов (p_{ij}).

Во втором параграфе приводится постановка задачи сегментации. Эта задача возникает когда объем набора данных превосходит размеры ОЗУ. Тогда Ω разбивается на некоторое число непересекающихся подмножеств-сегментов:

$$\Omega = \bigcup_{q=1}^{Q} C_q, \qquad (1)$$

$$C_q \cap C_r = \emptyset, \qquad (2)$$

При исполнении программы часть сегментов хранится в ОЗУ. Когда программе требуется обработать некоторый элемент ω, не находящийся в ОЗУ, то необходимо отыскать в ВЗУ сегмент, содержащий этот элемент, и считать его в ОЗУ. Для чтения сегмента нужно обеспечить для него место в ОЗУ, переписав один или несколько сегментов в ВЗУ. С точки зрения операций обмена сегменты являются неделимыми величинами. Всякому разбиению набора данных на сегменты можно поставить в соответствие цену разбиения, которая определяется как суммарный вес таких ребер графа Γ, концевые точки которых принадлежат различным сегментам.

Введем в рассмотрение набор векторов:

---

*) Модели программ такого вида рассматриваются, например, в работе: В.Н.Касьянов. К оценке частоты выполнения операторов и переходов в программе. "Программирование", № 5, 1975.

$$\vec{X}_1, \vec{X}_2, \ldots, \vec{X}_Q, \qquad (3)$$

компоненты которых имеют следующий смысл:

$$x_{qi} = \begin{cases} 1, & \text{если } \omega_i \in C_q, \\ 0, & \text{если } \omega_i \notin C_q. \end{cases}$$

Сумму $\sum_{i=1}^{N} x_{qi}$ назовем объемом q-го сегмента и будем обозначать $|\vec{X}_q|$. Задание значений набора (3) определяет некоторую сегментацию набора данных, если выполняются условия:

$$\sum_{q=1}^{Q} x_{qi} = 1, \qquad (4)$$

$$|\vec{X}_q| \le P, \qquad (5)$$

где P – размер листа ОЗУ, Условие (4) соответствует (I) и (2), а (5) вводится для учета конструктивных особенностей ЭВМ БЭСМ-6, в которой операция обмена выполняет пересылку между ОЗУ и ВЗУ одного листа памяти. Определение сегментации, обладающей наименьшей ценой сводится к поиску значений набора (3), минимизирующего функционал:

$$\Phi(\vec{X}_1, \vec{X}_2, \ldots, \vec{X}_Q) = \sum_{q=1}^{Q} \sum_{i,j=1}^{N} v_{ij}\, x_{qi}(1 - x_{qj}) \qquad (6)$$

с учетом ограничений (4) и (5).

<u>В третьем параграфе</u> рассматривается задача минимизации числа обменов. Пусть задана сегментация набора данных:

$$\Omega = \bigcup_{q=1}^{Q} C_q.$$

Реализованный в программе процесс обработки данных порождает последовательность запросов на сегменты:

$$S = (s_1, s_2, \ldots, s_T). \qquad (7)$$

В каждый момент времени в ОЗУ может находиться некоторое число сегментов, не более $v\ (v < Q)$. Неупорядоченное подмножество

множества сегментов $P \subset \{C_1, C_2, \ldots, C_Q\}$, находящихся в ОЗУ в момент времени t, назовем состоянием ОЗУ и будем обозначать P_t. Число сегментов в P_t будем обозначать $|P_t|$. Очевидно, всегда $|P_t| \leqslant v$. Начальное состояние ОЗУ определим как пустое множество:

$$P_0 = \varnothing.$$

Обслуживание запросов из (7) заключается в пересылке запрошенного сегмента в ОЗУ, поэтому:

$$s_t \in P_t.$$

Процесс изменения состояния ОЗУ, который реализуется при исполнении программы, описывается следующим образом:

$$P_t = \begin{cases} P_{t-1}, & s_t \in P_{t-1}, \\ P_{t-1} \cup \{s_t\}, & s_t \notin P_{t-1}, \ |P_{t-1}| < v, \\ P_{t-1} \cup \{s_t\} \setminus \{y_t\}, & s_t \notin P_{t-1}, \ |P_{t-1}| = v, \ y_t \in P_{t-1}. \end{cases} \qquad (8)$$

Здесь сегмент $y_t \in P_{t-1}$ переписывается из ОЗУ, в ВЗУ, чтобы освободить место для сегмента s_t. Процесс, описываемый соотношением (8), называется процессом замещения по запросу, а алгоритм A, определяющий y_t, – алгоритмом замещения по запросу:

$$y_t = A(P_{t-1}, s_t).$$

Алгоритмы замещения рассматриваются обычно в литературе по вопросам операционных систем*). Существует оценка эффективности алгоритма замещения, имеющая вид:

$$H(A, S) = \sum_{t=1}^{T} h(P_{t-1}, s_t),$$

---

*) Укажем обзор О.И.Авен, Б.Н.Кимельфельд, Я.А.Коган. Управление многоуровневой памятью вычислительных систем (обзор). Автоматика и Телемеханика, № II, 1972.

$$h(P_{t-1}, s_t) = \begin{cases} 0, & s_t \in P_{t-1}, \\ 1, & s_t \notin P_{t-1}. \end{cases} \qquad (9)$$

Приведем несколько классических алгоритмов замещения, которые для определения y_t используют простое эвристическое правило:

LRU – из ОЗУ выбрасывается наиболее давно использованный сегмент;

MRU – из ОЗУ выбрасывается последний использованный сегмент;

FIFO – из ОЗУ выбрасывается тот сегмент, который первым был считан из ВЗУ;

RAND – из ОЗУ с равной вероятностью выбрасывается любой из v сегментов.

Существует алгоритм Белэди, который при любых v и S доставляет минимум (9). По этому алгоритму из P_{t-1} выбрасывается такой сегмент, который в последовательности ($s_{t+1}, \ldots, s_T$) встречается позже, чем все другие сегменты из P_{t-1}. Заметим, что в обычных условиях алгоритм Белэди не может быть применен, поскольку при обслуживании запроса s_t следующие за ним запросы $s_{t+1}, \ldots, s_T$ неизвестны. В диссертации алгоритм Белэди применяется для сравнительной оценки эффективности других (приближенных) алгоритмов в моделирующих программах.

Вводится понятие рабочего графика алгоритма замещения, который характеризует зависимость H от величины коэффициента превышения $\mu = v/Q$. На примере нескольких рабочих графиков показывается, что при рассмотрении последовательностей запросов, возникающих в программах математической физики, величина H существенно зависит от выбора алгоритма замещения.

Во второй главе рассматриваются некоторые исходные предпо-

сылки создания системы SYDAK для управления данными в двумерных программах математической физики, предлагается методика экспериментального определения эффективности систем управления данными и приводятся некоторые полученные оценки.

В первом параграфе сравниваются два различных способа сегментации двумерных массивов: на полосы и на клетки. В качестве примера рассматривается схема продольно-поперечной прогонки для уравнения теплопроводности:

$$\frac{u^{n+1/2} - u^n}{\tau} = \Lambda_{11} u^{n+1/2} + \Lambda_{22} u^n, \qquad (10)$$
$$\frac{u^{n+1} - u^{n+1/2}}{\tau} = \Lambda_{11} u^{n+1/2} + \Lambda_{22} u^{n+1}$$

Двумерный массив значений неизвестной функции размером $N \times N$ разбивается или на μ полос по m строк в каждой или на $\varkappa^2$ клеток размером $K \times K$. Для обоих способов сегментации легко подсчитать число таких узлов сетки $(\mathcal{K})$, в которых для определения искомого значения требуются величины из нескольких сегментов.

При сегментации на полосы:

$$\mathcal{K}_{пол} = \frac{4(N-m)(N-2)}{m}, \qquad (11)$$

а на клетки:

$$\mathcal{K}_{кл} = 6\frac{(N-2K)}{K} + 2\frac{(N-2K)(5K-4)}{K} + 2\frac{(N-2K)(4K-2)}{K} + 12(K-1). \qquad (12)$$

Примем равные размеры сегментов ($mN = K^2$) и вычтем (12) из (11):

$$\Delta\mathcal{K} = \mathcal{K}_{пол} - \mathcal{K}_{кл} = \frac{2(N-2)}{K^2}(K^2 - 3NK + 2N^2). \qquad (13)$$

Трехчлен в (13) имеет корни $K_1 = N$ и $K_2 = 2N$, но т.к. в нашем случае K всегда меньше N, то $\mathcal{K}_{пол} > \mathcal{K}_{кл}$.

Рассматривается другая сравнительная оценка – число сег-

ментов, запрашиваемых при расчете одного шага по времени:

$$\Lambda_{пол} = 2\mu(N-1), \qquad (I4)$$

$$\Lambda_{кл} = 4\varkappa^2 \qquad (I5)$$

При равных размерах сегментов имеем $\varkappa^2 = \mu$ и (I5) можно также представить:

$$\Lambda_{кл} = 4\mu. \qquad (I6)$$

Р.А.Александровым был предложен*) прием программирования неявных разностных схем, основанных на методе расщепления, который заключается в предварительном расчете и запоминании коэффициентов поперечной прогонки. За счет этого при сегментации массивов на полосы число запросов сокращается до:

$$\Lambda_a = 6\mu \qquad (I7)$$

Далее приводится описание метода (предложенного Кернингемом**)) решения задачи сегментации. Этот метод позволяет быстро найти решение задачи (4), (5), (6), но с одним дополнительным ограничением: множество Ω считается пронумерованным и в каждый сегмент могут попасть только элементы со смежными номерами. Данное ограничение является очень сильным, однако, очевидно существование такой исходной нумерации, при которой решение, найденное методом Кернингема, совпадет с решением задачи (4), (5), (6). В диссертации предлагается использовать метод Кернингема для определения некоторого решения $\vec{X}_1, \vec{X}_2, \ldots, \vec{X}_Q$, которое затем может

---

*) Р.А.Александров. Численное решение на ЭВМ некоторых задач подземной гидромеханики. Автореферат диссертации на соискание ученой степени кандидата физико-математических наук, г. Казань, 1973.

**) B.W.Kerningham. Optimal Sequential Partitions of Graphs. Journal of the ACM, v.18, n.1, 1971.

быть улучшено путем перестановки некоторых пар элементов (ω_i, ω_j), принадлежащих разным сегментам. Приводимый в диссертации пример показывает, что минимум (6) может быть достигнут и при строчной сегментации массивов.

Основной вывод данного параграфа заключается в том, что строчная и клеточная сегментация двумерных массивов являются основными способами сегментации, на которых может достигаться минимум (6). Метод строчной сегментации был использован в системе SYDAK .

Во втором параграфе рассматривается разработанный Г.В.Шустовым [I] алгоритм замещения, учитывающий специфику программ математической физики. Эта специфика состоит в особой формулировке двух общих свойств последовательностей запросов.

а) Цикличность. В программах математической физики последовательность запросов является циклической и имеет один из видов:

$$S = (\hat{S})^{\tau}, \qquad S = (S_1, (\hat{S})^{\tau_1}, S_2)^{\tau},$$

$$S = (S_1, (\hat{S}_1^1)^{\tau_1}, \ldots, (\hat{S}_k^1)^{\tau_k}, (\hat{S}_1^2)^{\tau_1}, \ldots, (\hat{S}_\ell^2)^{\tau_\ell}, S_2)^{\tau}.$$

б) Локальность. При обработке двумерного массива его строки или столбцы запрашиваются в порядке возрастания или убывания их номеров.

Эти два свойства играют роль эвристических правил, на которых основан алгоритм замещения. На первом шаге итерационного процесса ($\tau = 1$) проводится анализ запросов и построение неполной модели процесса обработки данных, в которой указывается только связность различных массивов. На последующих шагах ($\tau > 1$) алгоритм использует построенную модель и свойство локальности для определения сегмента y_t в (8), выбрасываемого из ОЗУ.

В третьем параграфе приводится краткое описание системы SYDAK , предназначенной для программирования двумерных задач математической физики. Основные операторы системы:

CALL GIVE('A',K,I) - запрос K -й строки массива A ,

CALL TAKE('A',K) - отказ от K -й строки массива A ,

CALL ANNUL('A',K) - аннулирование K -й строки массива A ,

CALL REGIMS - установка режима обучения,

CALL REGIMW - установка рабочего режима.

Система допускает использование памяти МБ объемом до 256 К. Быстродействие системы составляет от 40 до 200 запросов в секунду.

В четвертом параграфе описана методика измерения эффективности систем управления данными. Классический критерий (9) в данном случае неприменим, т.к. он оценивает качество самого алгоритма замещения, а не его программной реализации.

Предлагается для сравнения эффективности нескольких систем управления данными использовать три критерия:

C - число замещений,

n - число обменов с ВЗУ,

Δt - время, затраченное на обслуживание некоторой последовательности запросов.

Используются также средние оценки (T - длина последовательности запросов):

$$\tilde{C} = C/T , \qquad \tilde{n} = n/T , \qquad \tilde{t} = \Delta t/T$$

Эти величины измеряются экспериментально в процессе работы системы. Когда сравниваются оценки эффективности нескольких систем, то измерение должно производиться при одинаковом объеме ОЗУ, который выделяется как для программной части системы, так и для вспомогательной информации и для вызова сегментов.

При оценке быстродействия систем используются две оценки, определяемые по текстам системы:

t_{min} – время обслуживания запроса, когда искомый сегмент находится в ОЗУ;

t_{max} – время обслуживания запроса, когда искомый сегмент находится в ВЗУ (включая время работы экстракода обмена).

Для измерения эффективности систем управления данными была создана система *STEND*, производящая измерение критериев эффективности Δt, c, n, $\tilde{t}$, $\tilde{c}$, $\tilde{n}$. Основными компонентами *STEND* являются: банк схем (программ, генерирующих различные последовательности запросов), банк систем управления данными, язык заданий, управляющая программа для измерения значений критериев и выдачи результатов в виде таблиц и графиков.

<u>В пятом параграфе</u> приводятся результаты ряда экспериментов, проведенных с системой *STEND*, на их основе делается ряд выводов.

I. Применение эвристического алгоритма системы *SYDAK* по сравнению с алгоритмом *MRU* сокращает число замещений на 6%.

2. При буферизованном обмене с ВЗУ число операций обмена слабо зависит от размеров буфера.

3. Возможное улучшение алгоритма замещения системы *SYDAK* почти не влияет на число замещений.

4. Число замещений сильно зависит от размера v области ОЗУ, в которой проводятся замещения.

5. Если улучшение алгоритма замещения связано с ростом объема вспомогательной информации, то в итоге это может привести к увеличению числа замещений.

6. Отказ от буферного обмена сокращает время обслуживания запроса примерно на 80%.

Результатом данного параграфа является вывод о том, что совершенствование систем управления данными должно быть основано не на улучшении алгоритма замещения, а на применении более экономичных приемов реализации систем.

В шестом параграфе излагается концепция виртуальной памяти. Это понятие возникло в начале 60-х годов при разработке машины "Атлас". Виртуальной называется воображаемая однородная память, реализуемая путем отображения в многоуровневую память ЭВМ. Это отображение с течением времени меняется, поскольку обрабатываемые части набора данных должны вызываться в ОЗУ. Для более простого отображения виртуальная и физическая память разбивается на листы одинакового размера. Предлагаемая диссертантом концепция состоит в том, что виртуальная память может быть реализована без соответствующего аппаратного обеспечения, на любом уровне системы математического обеспечения: в операционной системе*), в системе программирования**) или в программе пользователя.

Основные операции, выполняемые системой виртуальной памяти:

а) генерация виртуальной переменной,

б) запрос сегмента,

в) отказ от сегмента,

г) уничтожение виртуальной переменной.

---

*) Э.Диттрих, М.Зиберт. Иерархическая память для БЭСМ-6. В сб. "Проблемы повышения эффективности БЭСМ-6", Иркутск, 1976.

**) Н.А.Коновалов, В.А.Крюков, Э.З.Любимский. Управляемая виртуальная память. "Программирование", № 1, 1976.

В рамках этой концепции SYDAK является системой виртуальной памяти, вводимой на уровне программы пользователя.

В третьей главе описана система СВП, являющаяся дальнейшим развитием системы SYDAK . При её разработке были поставлены следующие задачи:

а) добиться более высокого быстродействия системы;

б) увеличить объем виртуальной памяти;

в) разработать несколько различных методов доступа;

г) ввести клеточную сегментацию массивов;

д) улучшить диагностические свойства системы;

е) включить в систему средства измерения эффективности.

В первом параграфе описана общая архитектура СВП. Эта система состоит из:

а) базиса, содержащего универсальную часть механизма виртуальной памяти;

б) средств доступа (B - доступа, S - доступа, V - доступа и U -доступа).

На уровне подпрограмм Базиса вводится виртуальная память, разбитая на блоки длиной по 1024 ячейки. Основная функция Базиса - это отображение множества виртуальных блоков в множество математических блоков (листов ОЗУ, трактов МБ, зон МД) и выполнение замещений по запросу. Для замещений используется алгоритм LRU . Существует три варианта Базиса:

- Базис 1; объем памяти 256 блоков на МБ первого направления;
- Базис 2; объем памяти 1251 блок на МБ первого направления и на одном МД;
- Базис 3; объем памяти 1024 блока; реализуется с помощью аппарата базовых возможностей ОС Диспак.

Средства доступа выполняют следующие функции:

- динамическую генерацию и уничтожение виртуальных переменных;
- преобразование номера сегмента виртуальной переменной в номер виртуального блока;
- контроль правильности обращений пользователя.

Приводится краткая характеристика методов доступа.

B–доступ (базисный). При генерации виртуальной переменной можно задавать как строчную, так и клеточную сегментацию массивов. При запросе или отказе указываются имя переменной и индекс сегмента. Время обслуживания запроса: t_{min} = 0.22мсек, t_{max} = I3.I5 мсек.

S –доступ (совместимый с SYDAK). При генерации допускается только строчная сегментация массивов. Время обслуживания запроса: t_{min} = 0.28 мсек, t_{max} = I3.9 мсек.

V – доступ. При генерации допускается как строчная, так и клеточная сегментация массивов. Оператор запроса распространяется на строку, столбец или высоту массива и выполняется подобно оператору чтения: значения запрошенной части массива пересылаются в одномерный массив программы пользователя. Оператор отказа выполняет обратную пересылку. Среднее время обслуживания запроса строки длиной 300: $\tilde{t}$ = 6.6 мсек.

U – доступ (поэлементный). Этот метод доступа обладает наименьшим быстродействием, т.к. вход в систему и проверка присутствия нужного блока в ОЗУ происходит при обращении к каждому элементу виртуальной переменной.

Во втором параграфе подробно рассматривается работа программ Базиса I при обслуживании запроса, структура служебных таблиц. При обменах с МБ Базис I использует метод резервной страницы и такую организацию служебных таблиц, при которой число циклических просмотров сводится к минимуму: один цикл про-

смотра не более 8 ячеек для поиска свободного виртуального блока. Время обслуживания запроса: t_{min} = I4 μsec , t_{max} = 300 μsec (без учета времени выполнения обмена).

В третьем параграфе проводится ассимптотический анализ роста времени обслуживания запросов в зависимости от размеров сетки. Число запросов на одном шаге пропорционально размеру сетки (N):

$$T_1 = a N. \qquad (I8)$$

Число шагов по времени (K), необходимых для решения задачи, равно:

$$K = b N \qquad (I9)$$

или

$$K = b \ln(N). \qquad (20)$$

Отсюда суммарное время обслуживания запросов при увеличении N стремится к

$$\Delta t \longrightarrow t_{max} c N^2 \qquad (2I)$$

или

$$\Delta t \longrightarrow t_{max} c N \ln(N). \qquad (22)$$

Снизить скорость увеличения Δt при росте N можно, уменьшив t_{max} . Эта возможность реализуется на конфигурациях ЭВМ БЭСМ-6 с объемом ОЗУ I28 К.

Базис 3 имеет виртуальную память объемом I024 блока, которая отображается на математическую память четырех программ: главной, содержащей реализацию вычислительного алгоритма и основные подпрограммы СВП, и трех пассивных подчиненных программ. Каждая программа использует для организации виртуальной памяти по 8 МБ первого направления. Роль подчиненных программ сводится к обладанию ресурсами памяти и выполнению замещений блоков по запросам главной программы. Главная программа, обслуживая запро-

сы от программы пользователя, синхронизирует работу подчиненных программ и выполняет обмен листами.

При такой организации Базиса 3 физические ресурсы ОЗУ используются полнее и более высокая эффективность достигается за счет того, что взаимодействие главной программы с подчиненной и обмен листами между этими программами выполняется быстрее, чем экстракод обмена с ВЗУ.

В четвертом параграфе описаны программы B -доступа и структура служебных таблиц. При разработке программ B -доступа также достигнута возможность проводить обслуживание запроса без циклического просмотра таблиц. При обслуживании отказа не удается избежать циклического просмотра таблицы, в которую заносятся сведения о запросах виртуальных блоков, которые считаны в ОЗУ. Этот просмотр необходим в связи с тем, что система допускает многократный запрос одного и того же блока.

В методах B-доступа, V- доступа и U-доступа виртуальные переменные подчиняются принципу локальности: доступ к переменной разрешается только из той подпрограммы, которая провела её генерацию. Если в операторе генерации виртуальной переменной её имя является значением общей переменной, то доступ к ней возможен из всех подпрограмм, в которых эта общая переменная определена.

В заключении приводятся сведения о внедрении системы SYDAK, перечисляются основные результаты работы.

I. Создана методика программной реализации систем виртуальной памяти, обладающая высокой эффективностью, простотой, использования и возможностью обращения к виртуальной памяти из языка высокого уровня. Эта методика апробирована для ЭВМ БЭСМ-6 и может применяться на других машинах, не имеющих аппаратно реализованной виртуальной памяти.

2. Разработаны системы управления данными SYDAK - для двумерных и СВП- для двумерных и трехмерных программ математической физики. В системе СВП на примере некоторых программ математической физики достигнуто высокое быстродействие, превосходящее быстродействие всех других известных систем.

3. Анализ процесса управления данными проводится на основе модели, из которой логически вытекают постановки задач сегментации набора данных и оптимального замещения сегментов. Преимуществом этого подхода является возможность проводить сравнение применяемого в системах метода управления данными с оптимальным методом, который обычно не может быть реализован на практике.

4. Предложена методика оценки эффективности систем управления данными. Разработана система STEND для экспериментального исследования эффективности систем. Средства измерения эффективности включены в систему СВП.

В приложениях к диссертации содержатся следующие материалы:

I. Сводка обращений к системе SYDAK ;

2. Пример программы, использующей SYDAK;

3. Сравнение SYDAK с операторами READ , WRITE;

4. Пример модели, использованной в системе STEND;

5. Сводка обращений к системе СВП с примерами программ для каждого метода доступа;

6. Сравнение эффективности системы SYDAK с S -доступом в СВП.

Основное содержание диссертации опубликовано в работах:

I. В.И.Карначук, Г.В.Шустов. Использование динамической памяти в языке Фортран. (Инструкция по системе SYDAK), препринт ВЦ СО АН СССР, Новосибирск, 1974.

2. В.И.Карначук, Г.В.Шустов. Работа с динамической памятью в мониторной системе "Дубна", в сб. "Развитие программного обеспе-

чения БЭСМ-6", ВЦ АН СССР, М., 1975, стр. 39-41.

3. В.И.Карначук, Г.В.Шустов. Запоминающая среда пакета прикладных программ математической физики. В сб. "Труды IУ Всесоюзного семинара по комплексам программ математической физики", ВЦ СО АН СССР, Новосибирск, 1976, стр. 194-200.
4. Н.Н.Яненко, В.И.Карначук, А.Н.Коновалов. Проблемы математической технологии. В сб. "Численные методы механики сплошной среды", т. 8, № 3, Новосибирск, 1977, стр. 129-157.
5. В.И.Карначук, Г.В.Шустов. Методика развертки программ по числовым данным. там же, стр. 91-97.
6. В.И.Карначук. О разработке систем виртуальной памяти для ЭВМ БЭСМ-6. В сб. "Численные методы механики сплошной среды", т.9,№7, Новосибирск,1978.

Карначук Валерий Иванович

Управление данными в программах
математической физики

Подписано к печати 16.03.79. МН 07396

Формат бумаги 60 х 90/16,Объем 1.3 усл.печ.л.,0.81 уч.изд.л.

Тираж 150 экз. Заказ 178 , Бесплатно.

Отпечатано на ротапринте ИТПМ СО АН СССР, 630090, Новосибирск, 90, Институтская, 4/1, Ротапринт.