Оптимальное размещение в памяти одного уровня n стеков и/или очередей[1]

А. В. Драц, А. В. Соколов, д. ф.-м. н.[2]
Петрозаводский универ-т, ИПМИ КарНЦ РАН, Петрозаводск
adeon88@mail.ru, avs@krc.karelia.ru

В работе рассмотрена проблема размещения n стеков и/или очередей в памяти одного уровня размера m для последовательного и связного способов представления. Предложены алгоритмы нумерации состояний и алгоритмы построения соответствующих поглощающих цепей Маркова и вычисления среднего времени до переполнения памяти для связного представления и нахождения оптимального (с точки зрения максимизации среднего времени до переполнения памяти) разбиения памяти для последовательных представлений. Приводятся результаты численных экспериментов.

Ключевые слова: LIFO-стеки, FIFO-очереди, связанные списки, цепи Маркова, динамические структуры данных.

1. Введение

Во многих приложениях возникает проблема размещения нескольких стеков и/или очередей [1] в памяти одного уровня. В ряде работ [2-4] и других были предложены математические модели и алгоритмы оптимального управления этими структурами данных. В данной работе предложены более общие модели и алгоритмы. Существуют 2 принципиально разных способа организации работы с динамическими структурами данных: последовательный и связанный [1]. (В дальнейшем под структурой данных будем понимать либо стек, либо очередь.)

Для стеков при последовательном представлении вся память делится на несколько частей и каждой паре стеков выделяется определенный участок памяти. Если количество стеков нечетно, то последний стек будет находиться один в отведенной ему памяти. При работе с очередями каждой из них выделяется своя область памяти.

[1]Работа выполнена при финансовой поддержке РФФИ (грант №06-01-00303).

[2]©А. В. Драц, А. В. Соколов, 2008

В этом случае могут возникнуть потери памяти когда какая-нибудь структура данных полностью исчерпает свою часть памяти.

При связанном представлении структура данных хранится в виде списка. Когда нужно добавить элемент, то необходимо найти незанятую область памяти и записать в нее новый элемент и указатель на предыдущий. В этом случае часть памяти тратится на указатели, но переполнение возникнет только в том случае, когда быстрая память будет полностью заполнена.

В данной работе рассматривается связанное и последовательное представление стеков и очередей, вычисляется среднее время работы с ними до переполнения, а в случае последовательного представления находится оптимальное разбиение быстрой памяти.

2. Обозначения

В работе мы будем придерживаться следующих обозначений:

- m – размер быстрой памяти
- n – количество стеков и/или очередей в быстрой памяти
- T – среднее время работы со структурами данных до переполнения
- p_i – вероятность включения элемента в i-ю структуру данных
- q_i – вероятность извлечения элемента из i-той структуры данных
- r – вероятность того что не произойдет операции включения или извлечения
- k_i – размер памяти выделенной для очереди с номером i или пары стеков с номерами $2i$ и $2i + 1$ при последовательном представлении
- x_i – текущая длина структуры данных с номером i
- l – отношение размера узла к размеру указателя (для связанного представления).

Будем считать, что время дискретно и вероятностные характеристики p_i, q_i, r известны. Предположим также, что не происходит завершения работы в случае исключения элемента из пустой структуры данных, и ,что работа начинается с пустых структур данных.

3. Связанное представление

3.1. Постановка задачи

При связанном представлении нет различий в поведении стеков и очередей. Различия будут лишь в расположении указателей внутри этих структур данных. Поэтому в случае связанного представления можно не рассматривать отдельно 2 случая для стеков и очередей, а ограничиться одним.

Рассмотрим n структур данных расположенных в m единицах памяти. Пусть на связи тратится $1/l$-я часть памяти Обозначим $m(1-1/l) = M$.

Требуется найти среднее время работы до переполнения.

3.2. Марковская модель

В качестве математической модели рассмотрим блуждание по целочисленной n-мерной пирамиде с ребрами

$$0 \le x_1 \le M, 0 \le x_2 \le M, \ldots, 0 \le x_n \le M$$

и основанием

$$x_1 + x_2 + \cdots + x_n = M.$$

Обозначим состояние в виде набора чисел $(x_1, x_2, \ldots, x_n)$. Покажем, что количество состояний равно $\frac{(M+n)!}{M!n!}$.

Для этого проведем аналогию с размещением объектов по ячейкам [5]. Число способов размещения n одинаковых объектов по m различным ячейкам равно C^n_{n+m-1}.

У нас в качестве объектов выступают единицы данных, а вместо ячеек — стеки. Всего единиц данных будет M, а стеков — $n+1$ (Будем считать, если ячейка памяти пустая, то она принадлежит

$n+1$-у стеку). Следовательно, общее количество состояний будет равно $C^M_{(n+1)+M-1} = C^M_{n+M} = \frac{(M+n)!}{M!n!}$.

Упорядочим состояния следующим образом:
$(0,0,0,\ldots,0,0),(1,0,0,\ldots,0,0),\ldots,(M-1,0,0,\ldots,0,0),$
$(M,0,0,\ldots,0,0)$
$(0,1,0,\ldots,0,0),(1,1,0,\ldots,0,0),\ldots,(M-2,1,0,\ldots,0,0),$
$(M-1,1,0,\ldots,0,0)$
$\ldots$
$(0,M-1,0,\ldots,0,0),(1,M-1,0,\ldots,0,0)$
$(0,M,0,\ldots,0,0)$
$(0,0,1,\ldots,0,0),(1,0,1,\ldots,0,0),\ldots,(M-2,0,1,\ldots,0,0),$
$(M-1,0,1,\ldots,0,0)$
$(0,1,1,\ldots,0,0),(1,1,1,\ldots,0,0),\ldots,(M-3,1,1,\ldots,0,0),$
$(M-1,1,1,\ldots,0,0)$
$\ldots$
$(0,0,M-1,\ldots,0,0),(0,0,M,\ldots,0,0)$
$\ldots$
$(0,0,0,\ldots,0,1),(1,0,0,\ldots,0,1),\ldots,(M-2,0,0,\ldots,0,1),$
$(M-1,0,0,\ldots,0,1)$
$(0,1,0,\ldots,0,1),(1,1,0,\ldots,0,1),\ldots,(M-3,1,0,\ldots,0,1),$
$(M-2,1,0,\ldots,0,1)$
$\ldots$
$(0,M-2,0,\ldots,0,1),(1,M-2,0,\ldots,0,1)$
$(0,M-1,0,\ldots,0,1)$
$\ldots$
$(0,0,0,\ldots,0,M).$
Частный случай при $n=3$, $M=4$ представлен на Рис. 1.

Для того, чтобы построить матрицу переходных вероятностей, выпишем функцию $F(X)=I$, где $X=(x_1,x_2,\ldots,x_n)$ – текущие длины структур данных, I – номер состояния.

Будем искать ее в виде:

$$\begin{aligned}
&F(x_1,\ldots,x_n) = F(0,\ldots,0,x_n)+\\
&+(F(0,\ldots,x_{n-1},x_n) - F(0,\ldots,0,x_n)) + \cdots +\\
&+(F(0,x_2,x_3,\ldots,x_{n-1},x_n) - F(0,0,x_3\ldots,x_{n-1},x_n))+\\
&+(F(x_1,x_2,x_3,\ldots,x_{n-1},x_n) - F(0,x_2,x_3,\ldots,x_{n-1},x_n)).
\end{aligned}$$

То есть будем увеличивать значения аргументов, начиная с последнего, и вычислять на сколько увеличится значение функции. Увели-

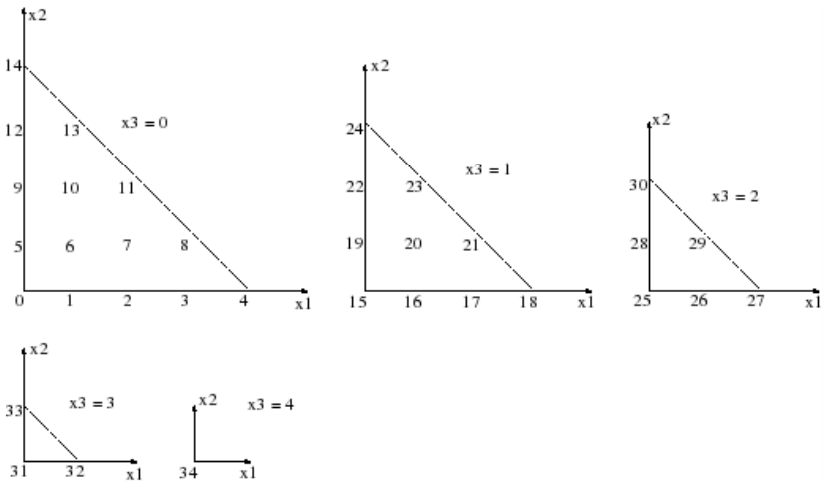

Рис. 1: Нумерация состояний при n=3, M=4.

чение значения функции ($F(0,0,\ldots,0,x_i,x_{i+1},\ldots,x_n)-F(0,0,\ldots,0,0,x_{i+1},\ldots,x_n)$) зависит от значения x_i и от суммы $x_{i+1}+\cdots+x_n$, то есть от количества уже занятых ячеек памяти. Поскольку в процессе работы программы будет многократно вычисляться значение функции F, то имеет смысл в начале работы программы вычислить таблицу $A[0..n][0..M][0..M]$, которая показывает изменение значения функции (первый индекс показывает номер структуры данных, второй – длину, а третий – количество уже занятых ячеек памяти структурами данных с более высокими индексами), а потом вычислять значение функции с ее помощью.

Для построения матрицы переходных вероятностей нужно в цикле перебрать все состояния (от $(0,0,\ldots,0,0)$ до $(0,0,\ldots,0,M)$) и вычислить вероятности перехода из I-го состояния в остальные ($I=F(x_1,x_2\ldots,x_n)$):
$Q[F(x_1,x_2,\ldots,x_n)][F(x_1+1,x_2,\ldots,x_n)]=p_1$
$Q[F(x_1,x_2,\ldots,x_n)][F(x_1-1,x_2,\ldots,x_n)]=q_1$
$\ldots$
$Q[F(x_1,x_2,\ldots,x_n)][F(x_1,x_2,\ldots,x_n+1)]=p_n$
$Q[F(x_1,x_2,\ldots,x_n)][F(x_1,x_2,\ldots,x_n-1)]=q_n$
$Q[F(x_1,x_2,\ldots,x_n)][F(x_1,x_2,\ldots,x_n)]=r.$

Для вычисления среднего времени блуждания нужно вычислить фундаментальную матрицу $N = (E - Q)^{-1}$ [6], где E – единичная матрица, и просуммировать элементы первой строки, поскольку она соответствует начальному состоянию.

4. Последовательное представление очередей

4.1. Постановка задачи

При последовательном представлении очередей необходимо каждой из них выделить свою область быстрой памяти. Рассмотрим n FIFO очередей, расположенных в m единицах памяти. Требуется найти разбиение памяти между очередями так, чтобы среднее время работы было наибольшим и вычислить это время.

4.2. Марковская модель

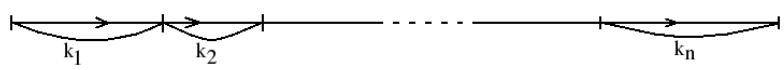

Рис. 2: Последовательное представление очередей.

Рассмотрим фиксированное разбиение памяти $k_1, \ldots, k_n$ между очередями. В качестве математической модели рассмотрим блуждание по целочисленному n-мерному параллелепипеду с вершиной в начале координат, ребрами параллельными осям координат, и длинами ребер $k_1, \ldots, k_n$.
Количество состояний равно $\prod_{i=1}^{n}(k_i + 1)$.
Перенумеруем состояния в соответствии с лексикографическим порядком:
$(0, 0, 0, \ldots, 0, 0), (0, 0, 0, \ldots, 0, 1), \ldots, (0, 0, 0, \ldots, 0, k_n - 1),$
$(0, 0, 0, \ldots, 0, k_n)$
$(0, 0, 0, \ldots, 1, 0), (0, 0, 0, \ldots, 1, 1), \ldots, (0, 0, 0, \ldots, 1, k_n - 1),$
$(0, 0, 0, \ldots, 1, k_n)$
$\ldots$
$(0, 0, 0, \ldots, k_{n-1} - 1, 0), (0, 0, 0, \ldots, k_{n-1} - 1, 1), \ldots,$
$(0, 0, 0, \ldots, k_{n-1} - 1, k_n - 1), (0, 0, 0, \ldots, k_{n-1} - 1, k_n)$

$(0,0,0,\ldots,k_{n-1},0),(0,0,0,\ldots,k_{n-1},1),\ldots,$
$(0,0,0,\ldots,k_{n-1},k_n-1),(0,0,0,\ldots,k_{n-1},k_n)$
$\ldots$
$(1,0,0,\ldots,0,0),(1,0,0,\ldots,0,1),\ldots,(1,0,0,\ldots,0,k_n-1),$
$(1,0,0,\ldots,0,k_n)$
$\ldots$
$(k_1,0,0,\ldots,0,0),(k_1,0,0,\ldots,0,1),\ldots,(k_1,0,0,\ldots,0,k_n-1),$
$(k_1,0,0,\ldots,0,k_n)$
$\ldots$
$(k_1,k_2,k_3,\ldots,k_{n-1},0),(k_1,k_2,k_3,\ldots,k_{n-1},1),\ldots,$
$(k_1,k_2,k_3,\ldots,k_{n-1},k_n-1),(k_1,k_2,k_3,\ldots,k_{n-1},k_n).$
Пример нумерации состояний представлен на Рис. 3.

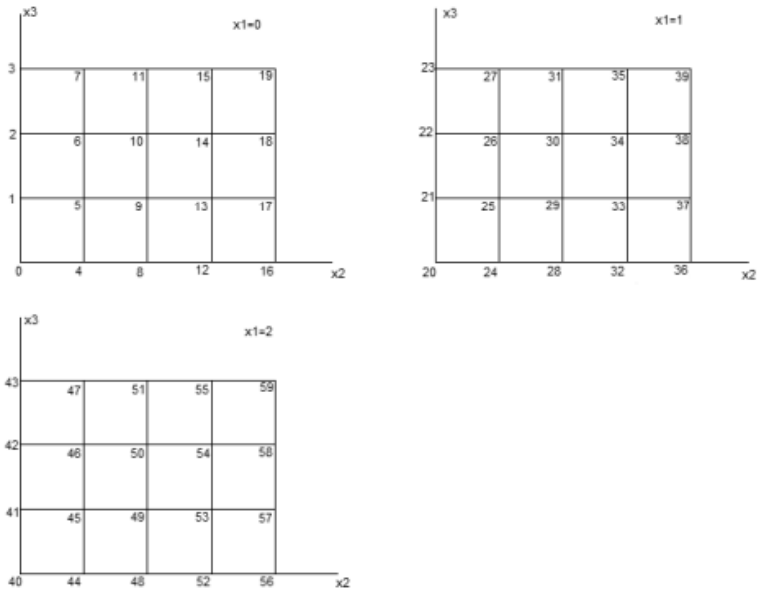

Рис. 3: Пример нумерации при $m=8$, $n=3$, $k_1=2$, $k_2=3$, $k_3=4$.

Построим функцию $F(x_1,x_2,\ldots,x_n)=I$, где $x_1,x_2,\ldots,x_n$ – текущие длины очередей, а I – номер состояния.
Если бы у нас была только одна очередь, то функция имела бы

вид $F(x_1) = x_1$. Если появляется вторая очередь, то состояния (x_i) становятся эквивалентными $(x_i, 0)$. Между состояниями $(x_i, 0)$ и $(x_{i+1}, 0)$ нужно $k_2 - 1$ состояний: $(x_i, 1), (x_i, 2), \ldots, (x_i, k_{n-1} - 1)$, поэтому функция будет иметь вид $F(x_1, x_2) = k_2 x_1 + x_2$.
При появлении третьей очереди получаем функцию
$F(x_1, x_2, x_3) = k_3(k_2 x_1 + x_2) + x_3$.

Рассуждая таким образом, получим для n очередей функцию
$F(x_1, x_2, \ldots, x_n) = k_n(k_{n-1}(\ldots(k_3(k_2 x_1 + x_2) + x_3) + \cdots + x_{n-1}) + x_n$.
Для построения матрицы переходных вероятностей нужно перебрать все состояния (от $(0, 0, \ldots, 0, 0)$ до $(k_1, k_2, \ldots, k_{n-1}, k_n)$) и вычислить вероятности перехода из I-го состояния в остальные ($I = F(x_1, x_2 \ldots, x_n)$):
$Q[F(x_1, x_2 \ldots, x_n)][F(x_1 + 1, x_2, \ldots, x_n)] = p_1$
$Q[F(x_1, x_2 \ldots, x_n)][F(x_1 - 1, x_2, \ldots, x_n)] = q_1$
$\ldots$
$Q[F(x_1, x_2 \ldots, x_n)][F(x_1, x_2, \ldots, x_n + 1)] = p_n$
$Q[F(x_1, x_2 \ldots, x_n)][F(x_1, x_2, \ldots, x_n - 1)] = q_n$
$Q[F(x_1, x_2 \ldots, x_n)][F(x_1, x_2, \ldots, x_n)] = r$.
Для вычисления среднего времени блуждания при заданном разбиении памяти нужно вычислить фундаментальную матрицу $N = (E - Q)^{-1}$ [6], где E – единичная матрица, и просуммировать элементы первой строки, поскольку она соответствует начальному состоянию.

Особенностью данной оптимизационной задачи является то, что аналитический вид целевой функции неизвестен, а задан лишь алгоритм вычисления ее значений. Поэтому предложить какое-либо улучшение перебора типа “метода ветвей и границ” в общем случае не представляется возможным. Следовательно, для нахождения оптимального разбиения памяти необходимо перебрать все возможные разбиения памяти, для каждого из них найти среднее время работы до переполнения и выбрать наибольшее время. Всего для размера памяти m и n очередей существует $C_{m-1}^{n-1} = \frac{(m-1)!}{(n-1)!(m-n)!}$ способов разбиения. Но некоторые из них заведомо невыгодны, поэтому перед построение матрицы переходных вероятностей можно сделать очевидные проверки. Для 2-х очередей с номерами i и j должны быть выполнены условия:

1. Если ($q_i \leq q_j$ и $p_i > p_j$) или ($q_i < q_j$ и $p_i \geq p_j$), то должно выполняться $k_i \geq k_j$.

То есть, если в i-ю очередь чаще, чем в j-ю, происходят включения и реже исключения элементов, то и памяти i-ой очереди нужно выделить по крайней мере не меньше чем j-ой.

2. Если $q_i = q_j$ и $p_i = p_j$, то $|k_i - k_j| \leq 1$.
   То есть, если вероятности включения и исключения элементов в 2 очереди равны, то им нужно выделить примерно поровну памяти. (Условие $|k_i - k_j| \leq 1$ нельзя заменить на $k_i = k_j$, например для 2-х очередей и размера быстрой памяти 5 и вероятностных характеристик $p_1 = p_2 = q_1 = q_2 = 0.25$ оптимальным разбиением будет $k_1 = 2$, $k_2 = 3$ или $k_1 = 3$, $k_2 = 2$, но равенство $k_1 = k_2$ не выполняется).

Проверка этих условий в некоторых случаях позволяет значительно сократить количество вычислений. (Например для 3-х очередей при размере быстрой памяти 30 и вероятностных характеристиках $p_1 = q_1 = p_2 = q_2 = p_3 = q_3 = 1/6$ только оптимальное разбиение $k_1 = k_2 = k_3 = 10$ удовлетворяет вышеописанным условиям, поэтому нужно будет вычислить время блуждания только для этого разбиения).

Получаем следующий алгоритм для решения задачи.

- Перебрать все возможные варианты разбиения быстрой памяти.
- Для каждого разбиения проверить выполнение (1) и (2).
  Если оно не удовлетворяет этим условиям, то перейти к следующему разбиению.
  Иначе построить матрицу переходных вероятностей и вычислить среднее время работы до переполнения
- Выбрать наибольшее время блуждания из всех найденных.

5. Последовательное представление стеков

5.1. Постановка задачи

При последовательном представлении стеков необходимо их разбить на пары, каждой паре выделить область памяти и направить

их расти навстречу друг другу. Задача со стеками несколько сложнее чем с очередями, т. к. требуется не только найти оптимальное разбиение быстрой памяти между парами стеков, но и оптимально разбить сами стеки на пары.

6. Марковская модель

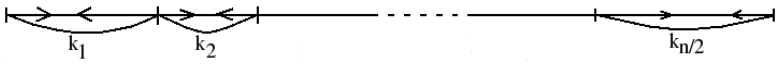

Рис. 4: Последовательное представление стеков.

Рассмотрим фиксированное разбиение памяти $k_1, \ldots, k_{\frac{n}{2}}$ между парами стеков. В качестве математической модели рассмотрим блуждание по целочисленному n-мерному многограннику, ограниченному гиперплоскостями $x_i \geq 0$, где $1 \leq i \leq n$ и $x_{2j-1} + x_{2j} \leq k_j$, где $1 \leq j \leq \frac{n}{2}$. Количество состояний равно $\prod\limits_{i=1}^{\frac{n}{2}} (\frac{(k_i+1)(k_i+2)}{2})$.
Перенумеруем состояния в соответствии с лексикографическим порядком:
$(0,0,0,\ldots,0,0,0), (0,0,0,\ldots,0,0,1), \ldots, (0,0,0,\ldots,0,0,k_{\frac{n}{2}}-1),$
$(0,0,0,\ldots,0,0,k_{\frac{n}{2}})$
$(0,0,0,\ldots,0,1,0), (0,0,0,\ldots,0,1,1), \ldots, (0,0,0,\ldots,0,1,k_{\frac{n}{2}}-2),$
$(0,0,0,\ldots,0,1,k_{\frac{n}{2}}-1)$
$\ldots$
$(0,0,0,\ldots,0,k_{\frac{n}{2}}-1,0), (0,0,0,\ldots,0,k_{\frac{n}{2}}-1,1)$
$(0,0,0,\ldots,0,k_{\frac{n}{2}},0)$
$(0,0,0,\ldots,1,0,0), (0,0,0,\ldots,1,0,1), \ldots, (0,0,0,\ldots,1,0,k_{\frac{n}{2}}-1),$
$(0,0,0,\ldots,1,0,k_{\frac{n}{2}})$
$(0,0,0,\ldots,1,1,0), (0,0,0,\ldots,1,1,1), \ldots, (0,0,0,\ldots,1,1,k_{\frac{n}{2}}-2),$
$(0,0,0,\ldots,1,1,k_{\frac{n}{2}}-1)$
$\ldots$
$(0,0,0,\ldots,1,k_{\frac{n}{2}}-1,0), (0,0,0,\ldots,1,k_{\frac{n}{2}}-1,1)$
$(0,0,0,\ldots,1,k_{\frac{n}{2}},0)$
$\ldots (k_1,0,k_2,\ldots,0,k_{\frac{n}{2}}-1,0), (k_1,0,k_2,\ldots,0,k_{\frac{n}{2}}-1,1)$
$(k_1,0,k_2,\ldots,0,k_{\frac{n}{2}},0).$
Пример нумерации состояний представлен на Рис. 5.

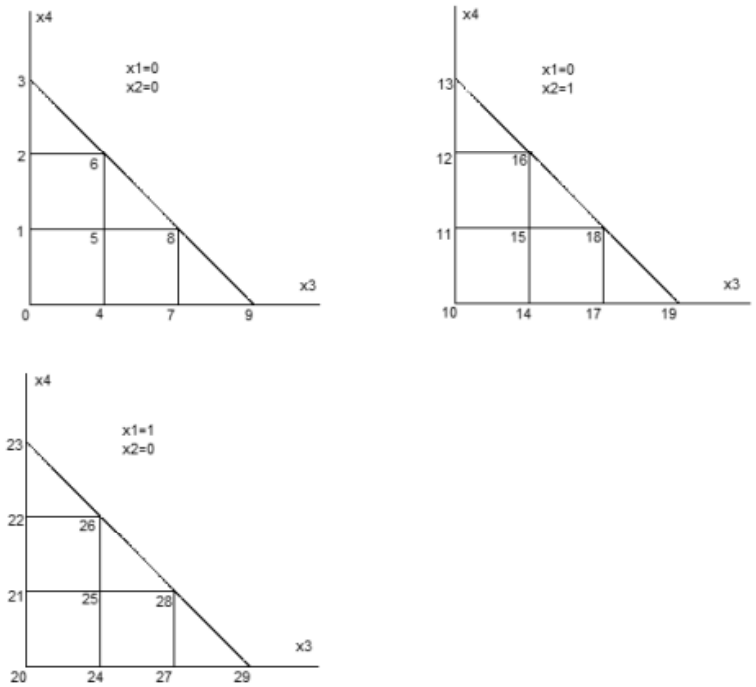

Рис. 5: Пример нумерации при $m = 4$, $n = 4$, $k_1 = 1$, $k_2 = 3$.

Как и в предыдущих случаях построим функцию $F(x_1, x_2, \ldots, x_n) = I$, где $x_1, x_2, \ldots, x_n$ – текущие длины стеков, а I – номер состояния. Найдем ее аналогичным способом как и в случае очередей.

Если бы у нас была только одна пара стеков, то функция имела бы вид $F(x_1, x_2) = a_2(x_2) + x_1$ (здесь и далее коэффициенты вида a_{2i} имеет тот же смысл что и в случае связанного представления стеков при $n = 2$). При появлении второй пары стеков получаем функцию
$F(x_1, x_2) = \frac{(k_2+1)(k_2+2)}{2}(a_2(x_2) + x_1) + a_4(x_4) + x_3$.
А при появлении $\frac{n}{2}$-ой пары стеков:

$F(x_1,\dots,x_n) = \Big(\frac{(k_{\frac{n}{2}}+1)(k_{\frac{n}{2}}+2)}{2}\dots\frac{(k_2+1)(k_2+2)}{2}(a_2(x_2)+x_1)+a_4(x_4)+$
$x_3)\dots\Big)+a_n(x_n)+x_{n-1}$.
Если количество стеков нечетно, то функция будет иметь отличие только в последнем шаге:
$F(x_1,\dots,x_n) = \Big(k_{\frac{n}{2}}\dots\frac{(k_2+1)(k_2+2)}{2}(a_2(x_2)+x_1)+a_4(x_4)+x_3)\dots\Big)+$
x_n.

Для построения матрицы переходных вероятностей нужно перебрать все состояния (от $(0,0,\dots,0,0)$ до $(k_1,0,k_3\dots,0,k_n,0)$) и вычислить вероятности перехода из I-го состояния в остальные ($I = F(x_1,x_2\dots,x_n)$):
$Q[F(x_1,x_2\dots,x_n)][F(x_1+1,x_2,\dots,x_n)] = p_1$
$Q[F(x_1,x_2\dots,x_n)][F(x_1-1,x_2,\dots,x_n)] = q_1$
$\dots$
$Q[F(x_1,x_2\dots,x_n)][F(x_1,x_2,\dots,x_n+1)] = p_n$
$Q[F(x_1,x_2\dots,x_n)][F(x_1,x_2,\dots,x_n-1)] = q_n$
$Q[F(x_1,x_2\dots,x_n)][F(x_1,x_2,\dots,x_n)] = r$.

Для вычисления среднего времени блуждания нужно вычислить фундаментальную матрицу $N = (E-Q)^{-1}$ [6], где E – единичная матрица, и просуммировать элементы первой строки, поскольку она соответствует начальному состоянию.

Как и в случае очередей, для вычисления оптимального времени блуждания нужно перебрать все возможные варианты разбиения памяти и для каждого из них найти среднее время работы до переполнения. Всего возможно $C_{m-1}^{\frac{n}{2}-1} = \frac{(m-1)!}{(m-\frac{n}{2})!(\frac{n}{2}-1)!}$ способов разбиения. Отличие здесь будет в том что нужно будет еще перебрать все возможные разбиения стеков на пары для каждого из разбиений памяти. Всего их можно разбить на пары $(n-1)!!$ способами при четном n и $n!!$ при нечетном. Поскольку при переходе от одного разбиения памяти к другому структура матрицы переходных вероятностей не меняется, а изменяются лишь некоторые ненулевые элементы, то нет необходимости снова ее вычислять, можно просто поменять их местами.

Также можно отсечь некоторые заведомо невыгодные разбиения стеков на пары, воспользовавшись следующими проверками перед вычислением матрицы переходных вероятностей.

1. Если ($q_{i_1} < q_{j_1}$ и $p_{i_1} > p_{j_1}$ и $q_{i_1} < q_{j_1}$ и $p_{i_1} > p_{j_1}$), то должно

выполняться $k_i \geq k_j$.
То есть, если включение в i-ю пару стеков происходит чаще чем в j-ую, то и памяти i-ой паре нужно выделить по крайней мере не меньше чем j-ой.

2. Если $q_{i_1} = q_{j_1}$ и $p_{i_1} = p_{j_1}$ и $q_{i_2} = q_{j_2}$ и $p_{i_2} = p_{j_2}$, то $|k_i - k_j| \leq 1$.
То есть, если вероятности включения и исключения элементов в 2 пары стеков равны, то им нужно выделить примерно поровну памяти.

Здесь стеки с номерами i_1, i_2 и j_1, j_2 объединены в пары.

Получаем следующий алгоритм для решения задачи.

- Перебрать все возможные варианты разбиения быстрой памяти.
- Для каждого разбиения проверить выполнение (1) и (2).
- Для каждого разбиения перебрать все возможные разбиения стеков на пары.
Если оно не удовлетворяет этим условиям, то перейти к следующему разбиению.
Иначе построить матрицу переходных вероятностей и вычислить среднее время работы до переполнения
- Выбрать наибольшее время блуждания из всех найденных.

7. Численные результаты

Были написаны программы для ЭВМ, которые реализуют вышеописанные алгоритмы нахождения среднего времени блуждания до переполнения. В этом разделе приведены некоторые численные результаты и сделаны выводы.

7.1. Сравнение последовательного и связанного представлений

В следующих таблицах сравнивается последовательное и связанное представление стеков и очередей.

В строке T указано оптимальное время для последовательного представления структур данных В строке l_1 указано среднее время работы со связанным представлением структур данных когда на связи тратится половина памяти, т. е. когда размер информационной части равен размеру указателя. В строке l_2 – когда на связи тратится 1/3 часть памяти, то есть размер указателя равен 1/2 размера информационной части. В строке l_4 – когда на связи тратится 1/5 часть памяти (размер указателя равен 1/4 информационной части). В строкеl_8 – когда тратится 1/9 часть памяти (размер указателя равен 1/8 информационной части).

Размер быстрой памяти равен 10, количество стеков равно 6 (Для последовательного представления приводится оптимальное время блуждания, но для экономии места не приводится оптимальное разбиение памяти и разбиение самих стеков на пары).

r	0.04	0.1	0.1	0	0	0
p_1	0.08	0.05	0.1	0.03	0.4	0.3
p_2	0.08	0.05	0.1	0.05	0.08	0.2
p_3	0.08	0.05	0.1	0.1	0.08	0.05
p_4	0.08	0.05	0.1	0.2	0.08	0.03
p_5	0.08	0.05	0.1	0.01	0.08	0.01
p_6	0.08	0.05	0.1	0.03	0.08	0.01
q_1	0.08	0.1	0.05	0.1	0.4	0.1
q_2	0.08	0.1	0.05	0.05	0.08	0.2
q_3	0.08	0.1	0.05	0.07	0.08	0.05
q_4	0.08	0.1	0.05	0.08	0.08	0.03
q_5	0.08	0.1	0.05	0.08	0.08	0.01
q_6	0.08	0.1	0.05	0.2	0.08	0.01
T	24.423	64.66	15.854	29.523	9.4	21.54
l_1	18.378	47.946	11.97	20.678	8.762	13.9
l_2	22.97	68.366	14.366	25.317	9.19	16.82
l_4	28	96.329	16.851	30.174	9.47	19.85
l_8	33.57	134.912	19.417	35.21	9.66	22.96

Размер быстрой памяти равен 12, количество очередей равно 5. (Для последовательного представления очередей не указывается оптимальное разбиения памяти для экономии места).

r	0	0	0	0	0	0
p_1	0.1	0.15	0.05	0.01	0.4	0.4
p_2	0.1	0.15	0.05	0.05	0.02	0.05
p_3	0.1	0.15	0.05	0.15	0.01	0.02
p_4	0.1	0.15	0.05	0.2	0.01	0.01
p_5	0.1	0.15	0.05	0.11	0.01	0.02
q_1	0.1	0.05	0.15	0.04	0.4	0.05
q_2	0.1	0.05	0.15	0.08	0.02	0.4
q_3	0.1	0.05	0.15	0.2	0.01	0.02
q_4	0.1	0.05	0.15	0.1	0.01	0.02
q_5	0.1	0.05	0.15	0.05	0.01	0.01
T	21.739	11.232	108.021	22.022	9.48	16.73
l_1	23.714	10.892	242.357	21.31	9.2	21.84
l_2	35.124	14.466	894.412	30.033	9.67	24.43
l_4	41.635	16.296	1798.367	34.681	9.78	27.015
l_8	48.684	18.145	3728.087	39.489	9.86	21.84

Из таблиц видно, что если на связи тратится $1/5$ часть или меньше памяти, то для стеков предпочтительнее использовать связанное представление.

Для очередей связанное представления выгоднее использовать при потере на связи $1/3$ части памяти. Кроме того если вероятности исключения у структур данных больше чем вероятности включения, то даже при потере половины памяти лучше использовать связанное представление.

В остальных случаях предпочтительнее последовательное представление.

7.2. Сравнение последовательного и связанного представлений, если вероятности заранее неизвестны

Вероятности включения и исключения, которые считались известными, на практике не всегда могут быть вычислены. В этом случае будет логичным разделить память поровну между всеми структурами данных в случае последовательного представления. В этом разделе сравнивается связанное представление и последовательное представление в том случае если память разделена поровну между структурами данных.

В следующих таблицах приведены некоторые численные результаты (в строке T_1 указано среднее время работы до переполнения.

В строке T_2 указано среднее время до переполнения при оптимальном разбиении памяти, строки l_1 и l_2 такие же как и в предыдущем пункте).

Количество стеков равно 6, размер быстрой памяти равен 12 (каждой паре стеков выделено по 4 единицы памяти).

r	0	0	0	0.1	0
p_1	0.4	0.4	0.02	0.03	0.2
p_2	0.02	0.4	0.02	0.1	0.2
p_3	0.08	0.05	0.1	0.1	0.04
p_4	0.02	0.02	0.02	0.05	0.04
p_5	0.02	0.02	0.02	0.1	0.01
p_6	0.02	0.02	0.02	0.06	0.01
q_1	0.4	0.02	0.4	0.1	0.2
q_2	0.02	0.02	0.4	0.07	0.2
q_3	0.02	0.02	0.02	0.05	0.04
q_4	0.02	0.02	0.02	0.1	0.04
q_5	0.02	0.02	0.02	0.04	0.01
q_6	0.02	0.02	0.02	0.1	0.01
T_1	30.71	6.45	163.92	31.98	25.13
T_2	49.6	11.92	191.87	32.57	43.024
l_1	32.73	8.26	145.32	23.34	27.19
l_2	48.75	10.68	223.3	32.9	40.42

Количество очередей равно 6 размер быстрой памяти равен 12 (каждой очереди выделено по 2 единицы памяти).

r	0	0	0	0.1	0
p_1	0.4	0.4	0.02	0.03	0.3
p_2	0.02	0.4	0.02	0.1	0.1
p_3	0.08	0.05	0.1	0.1	0.05
p_4	0.02	0.02	0.02	0.05	0.01
p_5	0.02	0.02	0.02	0.1	0.02
p_6	0.02	0.02	0.02	0.06	0.02
q_1	0.4	0.02	0.4	0.1	0.1
q_2	0.02	0.02	0.4	0.07	0.3
q_3	0.02	0.02	0.02	0.05	0.01
q_4	0.02	0.02	0.02	0.1	0.05
q_5	0.02	0.02	0.02	0.04	0.02
q_6	0.02	0.02	0.02	0.1	0.02
T_1	14.57	5.27	104.32	19.12	11.85
T_2	33.86	9.4	112.77	20.715	21.45
l_1	32.73	8.26	145.32	23.34	20.24
l_2	48.75	10.68	223.3	32.9	27.32

Из приведенных таблиц видно, что если вероятности включения и исключения у разных очередей не равны между собой, то последовательное представление оказывается хуже связного даже при потере половины памяти на связи. Если вероятности включения и исключения у разных очередей равны (см. предыдущий пункт), то последовательное представление оказывается лучше связного при потере 1/3 части памяти на связи.

Исходя из полученных результатов, можно дать рекомендацию: при неизвестных заранее вероятностях стоит выбирать связанное представление, если нужен более предсказуемый результат для среднего времени работы. Можно рискнуть и выбрать последовательное представление, но, в случае неравных вероятностей, результат может оказаться значительно хуже.

Для стеков наблюдается аналогичная ситуация. В случае различных значений вероятностей среднее время работы при последовательном представлении примерно равно времени работы при связанном представлении при потере половины памяти. Если вероятности окажутся равными (см. предыдущий пункт), последовательное представление лучше связанного при потере 1/3 части памяти, но хуже при потере 1/5.

Исходя из этого, можно дать практическую рекомендацию: в случае неизвестных вероятностей лучше использовать последовательное представление при потере половины памяти на указатели. В случае потери 1/3 части памяти на указатели лучше использовать последовательное представление, так как это приведет к более предсказуемому результату.

Список литературы

[1] *Кнут Д.* Искусство программирования для ЭВМ. Т. 1. М.: Вильямс. 2001.

[2] *Соколов А. В.* Математические модели и алгоритмы оптимального управления динамическими структурами данных. – Петрозаводск. 2002.

[3] *Лазутина А. А.* Разработка и анализ математических моделей и алгоритмов динамического распределения памяти в за-

дачах управления стеками. Автореферат диссертации на соискание ученой степени кандидата физико-математических наук. – Петрозаводск. 2006. 19 с.

[4] *Аксенова Е. А.* Математические модели и оптимальные методы реализации динамических структур данных. Автореферат диссертации на соискание ученой степени кандидата физико-математических наук. – Петрозаводск. 2007. 21 с.

[5] *Риордан Дж.* Введение в комбинаторный анализ. – М.: Издательство иностранной литературы. 1963.

[6] *Кемени Дж., Снелл Дж.* Конечные цепи Маркова. – М.: Наука. 1970.