Эвристический подход к адресации микрокоманд в композиционных микропрограммных устройствах управления с разделением кодов и кэш-памятью

Баркалов А.А., Ковалев С.А., Бабаков Р.М., Николаенко Д.В.
Университет Зеленогурский (Польша),
Государственный университет информатики и искусственного интеллекта
Донецкий национальный технический университет
a.barkalov@iie.uz.zgora.pl, cpld@mail.ru

Abstract

Barkalov A.A., Kovalev S.A., Babakov R.M., Nikolaenko D.V. The heuristic approach to addressing of microinstructions in compositional microprogram control units with division of codes and cache-memory. *The analytical heuristics of placement of operational linear chains of microinstructions in control memory of compositional microprogram control unit with division of codes and cache-memory of microinstructions are proposed. The heuristics are based on special features of structures of control units with division of codes.*

Введение

Одним из структурных элементов современных вычислительных систем является устройство управления (УУ), которое может быть реализовано в виде композиционного микропрограммного устройства управления (КМУУ) с разделением кодов, в котором достигается минимальное число выходов схемы адресации [1]. Использование элементного базиса ПЗУ при реализации управляющей памяти удешевляет схему устройства, однако в то же время приводит к значительному снижению быстродействия схемы [2]. Таким образом, для промышленности средств вычислительной техники актуальной научно-технической задачей является задача увеличения быстродействия схемы КМУУ. Решение данной задачи повлечет за собой увеличение быстродействия вычислительных систем, реализованных на базе КМУУ, и, как следствие, расширит область их применения.

Для увеличения быстродействия схемы КМУУ с разделением кодов в работе [3] предложен метод, заключающийся в использовании дополнительного модуля кэш-памяти микрокоманд (МК) для хранения наиболее часто используемых строк управляющей памяти. Использование кэш-памяти позволяет снизить среднее время доступа к управляющей памяти, что приводит к уменьшению средней длительности такта работы устройства и к увеличению его среднего быстродействия. При этом основным параметром, влияющей на эффективность использования кэш-памяти микрокоманд, является вероятность кэш-попаданий p_h, характеризующая отношение количества тактов, в которых произошло кэш-попадание, к общему количеству тактов работы устройства.

Характерной особенностью всех структур КМУУ является их аппаратная привязка к реализуемому алгоритму управления. Данная особенность позволяет провести оптимизацию схемы КМУУ для каждого конкретного случая реализации. В структуре КМУУ с разделением кодов и кэшированием сигналов, предложенной в [3], одним из факторов, влияющим на величину вероятности кэш-попаданий, является реализуемый алгоритм управления, традиционно представляемый в виде граф-схемы алгоритма (ГСА). Количество микрокоманд, операторных линейных цепей (ОЛЦ) и переходов между ОЛЦ, а также сама структура ГСА – все это оказывает влияние на величину p_h. При этом на сегодняшний день неисследованными остаются факторы, влияющие на увеличение значения p_h и, как следствие, на увеличение среднего быстродействия устройства управления.

Целью настоящей работы является решение научной задачи поиска факторов, влияющих на величину вероятности кэш-попаданий в структуре композиционного микропрограммного устройства управления с разделением кодов и кэш-памятью микрокоманд. Основная идея работы заключается в эвристическом подходе к адресации микрокоманд реализуемой ГСА при синтезе структуры КМУУ с разделением кодов и кэшированием сигналов.

В структурах КМУУ с разделением кодов имеют место естественная адресация микрокоманд внутри каждой ОЛЦ, а также тот факт, что в первой микрокоманде ОЛЦ адресные разряды, следующие после кода ОЛЦ, равны нулям [2]. По этим причинам процесс адресации

микрокоманд, являющийся одним из этапов синтеза КМУУ, в структурах с разделением кодов сводится по сути к адресации ОЛЦ.

Пусть в заданной ГСА существует переход из ОЛЦ α_i в ОЛЦ α_j. При этом, если обе ОЛЦ в данный момент находятся в кэш-памяти, то возникнет ситуация кэш-попадания, в противном случае – ситуация кэш-промаха. В том случае, если обе ОЛЦ расположены в управляющей памяти последовательно и принадлежат одному блоку памяти, они окажутся вместе в одной строке кэш-памяти, и упомянутый переход из α_i в α_j всегда будет приводить к кэш-попаданию. Если же данные ОЛЦ находятся в различных блоках памяти, то при выполнении перехода $\alpha_i \rightarrow \alpha_j$ возможна ситуация кэш-промаха, и значение p_h для ГСА будет несколько ниже. Если же одновременное нахождение данных ОЛЦ в кэш-памяти невозможно (например, при использовании кэш-памяти с прямым отображением данных), то данный переход всегда будет приводить к кэш-промаху, что еще более снизит величину вероятности кэш-попаданий для заданной ГСА.

Пусть алгоритм управления задан ГСА G (рис. 1).

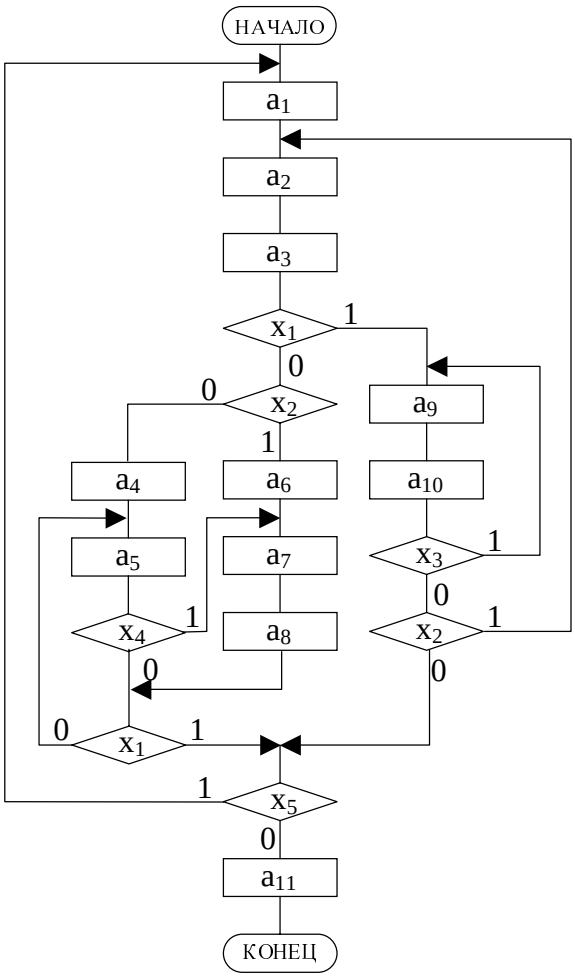

Рисунок 1 – Граф-схема алгоритма G

Очевидно, что ГСА G содержит $N_{ОЛЦ}=5$ операторных линейных цепей: $O_1=\{a_1, a_2, a_3\}$, $O_2=\{a_4, a_5\}$, $O_3=\{a_6, a_6, a_8\}$, $O_4=\{a_9, a_{10}\}$, $O_5=\{a_{11}\}$. Максимальный размер ОЛЦ $N_{max}=3$, следовательно, для кодирования микрокоманд внутри ОЛЦ требуется $R(T)=]\log_2(N_{max})[=2$ двоичных разряда. Пять ОЛЦ могут быть закодированы $R(\tau)=]\log_2 5[=3$ разрядами. Таким образом, общий размер адреса микрокоманды $R=R(T)+R(\tau)=5$ бит, а емкость ПЗУ схемы УП составит $2^R=32$ слова.

Согласно принципу пространственной локальности данных, в случае кэш-промаха в кэш из УП помещается не только запрашиваемая микрокоманда, но и несколько микрокоманд из ближайших к ней адресов памяти. При этом количество загружаемых команд определяется размером строки кэш-памяти, а для кодирования микрокоманд внутри строки используется часть младших разрядов адреса микрокоманды. С учетом этого содержимое управляющей памяти может быть представлено в виде *блоков микрокоманд*, имеющих размер, равный размеру строки кэш-памяти и располагающихся последовательно начиная с нулевого адреса. Деление УП на блоки является постоянным и не зависит от размещения микрокоманд в адресном пространстве УП. В нашем примере при размере строки кэш-памяти $N_C=8$ слов содержимое УП представляется состоящим из $2^R/ N_C=4$ блоков B_1-B_4.

Рассмотрим три варианта размещения микрокоманд в адресном пространстве ПЗУ управляющей памяти (рис. 2). Хотя значение адреса А лежит в диапазоне от 0 до 31 и разделяется на 4 блока по 8 адресов, последний блок в примере не используется и на рисунке не показан.

Подчеркнем, что для рассматриваемого примера использование различных вариантов размещения микрокоманд не оказывает существенного влияния на параметры логической схемы КМУУ, поскольку разрядность адреса микрокоманды во всех случаях остается неизменной.

Определим экспериментально вероятности кэш-попаданий для каждого размещения микрокоманд, для чего используем специально разработанную программную имитационно-аналитическую модель КМУУ с разделением кодов и кэш-памятью микрокоманд с кэш-памятью полностью ассоциативного типа и алгоритмом замещения данных Random. В табл. 1 представлены результаты экспериментов для следующих значений вероятностей логических условий: $p(x_1)=0.2$, $p(x_2)=0.7$, $p(x_3)=0.5$, $p(x_4)=0.1$, $p(x_5)=0.9$.

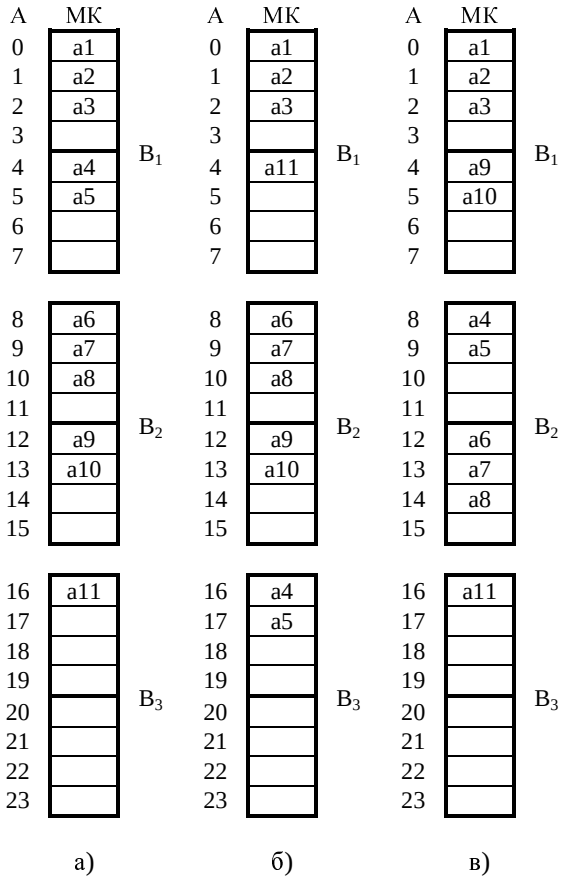

Рисунок 2 – Примеры вариантов размещения микрокоманд в управляющей памяти

Таблица 1. Значения вероятности кэш-попаданий для различных вариантов размещения микрокоманд

Вариант	p_h
а)	0.7596
б)	0.6781
в)	0.8273

Анализ содержимого столбца p_h позволяет сделать важное заключение: порядок размещения микрокоманд в управляющей памяти при прочих неизменных параметрах влияет на значение вероятности кэш-попаданий. Следовательно, увеличение эффективности использования модуля кэш-памяти в базовой структуре КМУУ с разделением кодов возможно за счет модификации размещения микрокоманд в управляющей памяти. В настоящей работе предлагается принцип повышения значения вероятности кэш-попаданий, основанный на оптимизации размещения операторных линейных цепей.

Рассмотрим два очевидных пути поиска размещения ОЛЦ, дающего максимально возможное значение вероятности кэш-попаданий для заданной ГСА.

1. Полный перебор возможных вариантов размещения.

Возможные размещения ОЛЦ в управляющей памяти для ГСА G не ограничиваются вариантами, представленными на рис. 2. Для произвольной ГСА количество возможных размещений может быть определено следующим образом.

Пусть микрокоманды заданной ГСА образуют $N_{ОЛЦ}$ операторных линейных цепей. Поскольку ОЛЦ могут быть размещены в УП в произвольной последовательности, число возможных перестановок $P_{ОЛЦ}$ определяется функцией факториала:

$$P_{ОЛЦ} = (N_{ОЛЦ})! \qquad (1)$$

Так, для ГСА, содержащей 50 ОЛЦ, количество возможных размещений ОЛЦ в управляющей памяти определяется числом с 64 десятичным порядком.

Упомянутая выше программная имитационно-аналитическая модель позволяет собрать статистическую информацию о количестве кэш-попаданий и кэш-промахов за один цикл моделирования, включающий большое количество повторений (от 10 000 до 10 000 000 в зависимости от размера и сложности ГСА) пошагового выполнения микрокоманд, и получить экспериментальное значение вероятности кэш-попаданий. При этом среднее время выполнения цикла моделирования составляет одну минуту на компьютере с процессором Pentium-IV частотой 3 ГГц.

Поскольку для каждого варианта размещения требуется выполнение цикла моделирования с целью определения значения p_h, полный перебор вариантов с учетом производительности современных средств моделирования не представляется возможным.

2. Аналитическое определение оптимального размещения данных в управляющей памяти.

Предположим, что существует метод аналитического определения оптимального размещения ОЛЦ в управляющей памяти, позволяющий найти абсолютно оптимальный с точки зрения p_h вариант размещения. Подобный метод должен учитывать тип архитектуры кэш-памяти, приводящий к соответствующим модификациям метода. Архитектура кэш-памяти может быть с прямым отображением, полностью ассоциативной или смешанной. В двух последних случаях в качестве стратегий замещения данных могут использоваться различные ассоциативные алгоритмы: LRU, Random, Timer, FIFO, MRU и

др., в результате чего возникает не менее десяти «архитектурных» модификаций гипотетического метода. Таким образом, разработка метода аналитического определения оптимального размещения микрокоманд сводится к разработке минимум десяти аналогичных методов, что удесятеряет сложность решения данной задачи. По этой причине метод аналитического определения абсолютно оптимального варианта размещения микрокоманд в настоящей работе не рассматривается.

Основываясь на сказанном выше, можно сделать вывод: наиболее эффективным мог бы быть алгоритм оптимизации размещения данных, обладающий следующими свойствами.

1. Множество всех возможных вариантов размещения ОЛЦ должно сводиться к некоторому подмножеству, позволяющему рассмотреть включенные в него варианты размещения за приемлемое время. Таким образом, алгоритм не должен носить полностью экспериментальный характер.

2. Алгоритм не должен учитывать особенностей какой-либо архитектуры кэш-памяти, но должен опираться на экспериментально подтвержденные свойства, общие для всех архитектур. Таким образом, алгоритм не должен носить полностью аналитический характер.

3. Не имея привязки к архитектуре кэш-памяти, алгоритм не должен определять абсолютно оптимальный вариант размещения данных. Тем не менее, результат работы алгоритма (или множество результатов) должен в общем случае быть близок к результатам, которые могли бы быть получены при полном переборе или точном аналитическом расчете.

Вышеперечисленным требованиям может соответствовать алгоритм, в основе которого лежат эвристические правила (эвристики) – общие рекомендации, основанные на статистической очевидности.

Сформулируем и обоснуем эвристики, отражающие влияние особенностей размещения ОЛЦ в управляющей памяти на значение вероятности кэш-попаданий.

Эвристика 1. *Вероятность кэш-попаданий не зависит от того, какому по порядку блоку памяти принадлежит ОЛЦ.* Иными словами, конкретные значения адресов памяти, занимаемые ОЛЦ, не оказывают влияния на вероятность кэш-попаданий. Подтверждением являются примеры размещений на рис. 3, дающие одинаковые значения вероятности кэш-попаданий.

Отметим, что в варианте «б» микрокоманда a_1, являющаяся стартовой микрокомандой ГСА, расположена не по нулевому адресу. При использовании подобного размещения данный факт должен быть учтен в функциональной схеме КМУУ.

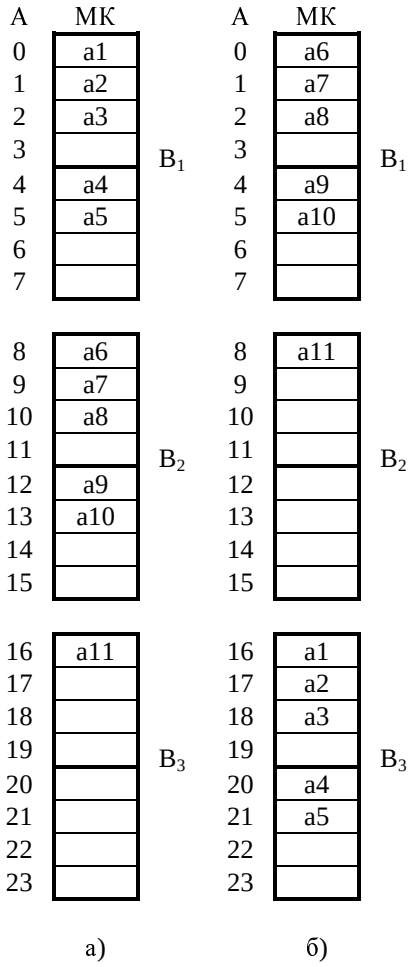

Рисунок 3 – Варианты размещения, эквивалентные в силу эвристики 1

Из рассмотренного правила вытекает следующий вывод: *если содержимое блока памяти сместить в адресном пространстве УП вперед или назад на количество ячеек, кратное размеру блока, то значение вероятности кэш-попаданий не изменится*.

Эвристика 2. *Вероятность кэш-попаданий не зависит от порядка следования ОЛЦ в блоке*. Пример размещений, иллюстрирующий данную ситуацию, показан на рис. 4. Эвристика действует и в том случае, если блок памяти содержит более двух ОЛЦ.

Вывод, следующий из данного правила, может быть сформулирован следующим образом: *если несколько ОЛЦ могут быть размещены в одном блоке УП, то порядок помещения ОЛЦ в блок не влияет на результирующее значение вероятности кэш-попаданий.*

Для того, чтобы сформулировать следующую эвристику, введем ряд понятий:

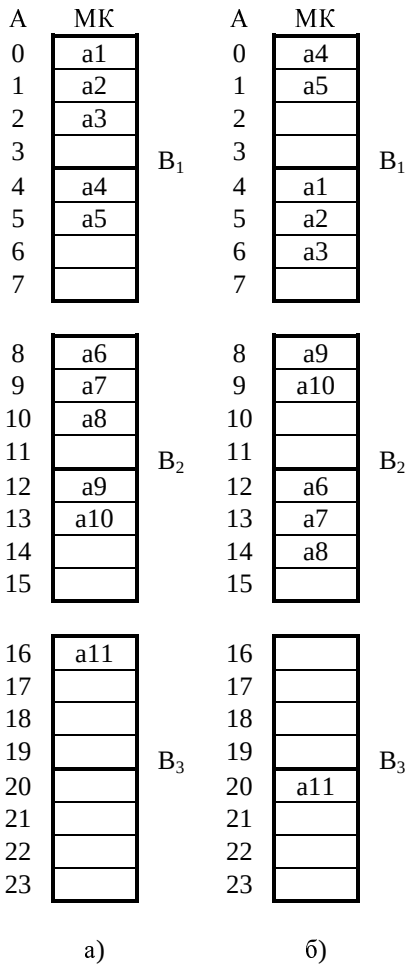

Рисунок 4 – Варианты размещения, эквивалентные в силу эвристики 2

– «опасные» и «безопасные» МК;
– вес микрокоманды;
– «опасный» вес микрокоманды и ОЛЦ.

«Опасные» и «безопасные» микрокоманды.

Рассмотрим вариант «а» размещения микрокоманд ГСА G на рис. 2.

Переход в МК a_1 может выполняться из других ОЛЦ. При переходе в a_1 из другой ОЛЦ блок B_1 может отсутствовать в кэш-памяти, и произойдет кэш-промах. Поскольку при выполнении a_1 кэш-промах возможен (хотя и не обязателен), будет считать a_1 «опасной» МК. По аналогии отнесем к категории «опасных» все МК, являющиеся входами ОЛЦ. Для нашего примера это a_2, a_4, a_6, a_9, a_{11}.

Рассмотрим «опасную» МК a_2. Что бы ни произошло при ее выполнении (кэш-попадание или кэш-промах), сразу после выполнения a_2 блок B_1 будет находиться в кэш-памяти. Поскольку следующая МК a_3 принадлежит той же ОЛЦ, что и a_2, и находится с a_2 в одном блоке, можно утверждать, что на момент своего выполнения она всегда будет находиться в кэш-памяти. Таким образом, если микрокоманда не является входом ОЛЦ, то кэш-промах при ее выполнении невозможен. Условимся называть микрокоманды, для которых кэш-промах невозможен, «безопасными». В рассматриваемом примере к этой категории, помимо a_3, относятся МК a_5, a_7, a_8, a_{10}.

Таким образом, «опасными» (для которых кэш-промах возможен) будем считать МК, являющиеся входами ОЛЦ. Остальные МК (для которых кэш-промах невозможен) будем считать «безопасными». Отметим, что при разных размещениях микрокоманд в УП множества «опасных» и «безопасных» МК будут сохраняться.

Вероятность выполнения микрокоманды.

За время одного прохода ГСА каждая МК при заданных значениях вероятностей ЛУ выполняется в среднем некоторое количество раз, которое может быть определено путем программного моделирования. Это количество не обязательно является целым числом. Например, для ГСА G МК a_{11} всегда выполняется один раз, а среднее количество выполнений МК a_1 при вероятностях выполнения логических условий $p(x_1)=0.2$, $p(x_2)=0.7$, $p(x_3)=0.5$, $p(x_4)=0.1$, $p(x_5)=0.9$ равно 10.

Просуммировав количества выполнений всех МК, получим среднее количество МК, выполняющихся за один проход алгоритма. Для ГСА G при указанных выше вероятностях переходов это количество равно 113.84.

Вероятностью выполнения (или *весом*) $p(a_i)$ *микрокоманды* a_i будем называть отношение среднего количества выполнений $K(a_i)$ микрокоманды a_i за один проход алгоритма к среднему количеству микрокоманд K, выполняющихся за один проход:

$$p(a_i) = K(a_i) / K. \qquad (2)$$

Например, для ГСА G и $p(x_1)=0.2$, $p(x_2)=0.7$, $p(x_3)=0.5$, $p(x_4)=0.1$, $p(x_5)=0.9$ имеем $p(a_{11})=1/113.84=0.0088$.

Вероятность выполнения каждой микрокоманды складывается из вероятности кэш-попаданий и вероятности кэш-промахов:

$$p(a_i) = p_h(a_i) + p_m(a_i). \qquad (3)$$

При этом вероятность кэш-попаданий рассматриваемой ГСА равна сумме вероятностей кэш-попаданий каждой микрокоманды:

$$p_h(\text{ГСА}) = \sum_{i=1}^{N_{МК}} p_h(a_i). \qquad (4)$$

Следует подчеркнуть, что изменение способа размещения микрокоманд в адресном пространстве УП не влияет на значения $p(a_i)$, которые определяются лишь структурой ГСА и вероятностями выполнения ЛУ. Тем не менее,

изменение размещения микрокоманд позволяет в ряде случаев увеличить $p_h(a_i)$ с одновременным соответствующим уменьшением $p_m(a_i)$ и тем самым повысить значение p_h реализуемого алгоритма управления.

«Опасный» вес ОЛЦ.

«Опасный» вес $v(a_i)$ микрокоманды a_i равен сумме весов микрокоманд других ОЛЦ, из которых есть непосредственный переход в микрокоманду a_i, умноженных на вероятность перехода.

Пусть A – множество микрокоманд ГСА, $A(O_j)$ – множество микрокоманд ОЛЦ O_j , к которой принадлежит микрокоманда a_i. Тогда

$$v(a_i) = \sum_{k \in A \setminus A(O_j)} p(a_k) \cdot p(a_k \to a_i). \qquad (5)$$

В качестве примера рассмотрим микрокоманду a_{11} ГСА G. Существуют следующие непосредственные переходы в эту МК из других ОЛЦ:

– переход из a_5 весом $p(a_5)$ с вероятностью $(1–p(x_4))\cdot p(x_1)\cdot(1–p(x_5))$;

– переход из a_8 весом $p(a_8)$ с вероятностью $p(x_1)\cdot(1–p(x_5))$;

– переход из a_{10} весом $p(a_{10})$ с вероятностью $(1–p(x_3))\cdot(1–p(x_2))\cdot(1–p(x_5))$.

Тогда «опасный» вес определяется формулой

$$v(a_{11}) = p(a_5)\cdot(1–p(x_4))\cdot p(x_1)\cdot(1–p(x_5)) + p(a_8)\cdot p(x_1)\cdot(1–p(x_5)) + p(a_{10})\cdot(1–p(x_3))\cdot(1–p(x_2))\cdot(1–p(x_5)).$$

Отметим, что последнее выражение является сокращенной записью выражения (5), поскольку в нем отсутствуют непосредственные переходы, вероятность которых равна нулю (например, из МК a_1).

Вычислив «опасный» вес каждой микрокоманды, *можно определить «опасный» вес каждой ОЛЦ $v(O_i)$ как сумму «опасных» весов микрокоманд данной ОЛЦ*:

$$v(O_i) = \sum_{a_j \in O_i} v(a_j). \qquad (6)$$

Эвристика 3. *Если ОЛЦ O_i имеет переходы в ОЛЦ O_j, то размещение этих ОЛЦ в одном блоке памяти снижает «опасный» вес ОЛЦ O_j.*

Рассмотрим примеры размещений ОЛЦ ГСА G на рис. 5.

В варианте размещения «а» ОЛЦ $O_1=\{a_1, a_2, a_3\}$ является единственной в своем блоке. В варианте «б» в блок с ОЛЦ O_1 добавляется ОЛЦ O_2. Проанализируем данную ситуацию.

Из МК $a_5 \in O_2$ существует переход в МК $a_1 \in O_1$. Это приводит к невозможности возникновения кэш-промаха при выполнении данного перехода, поскольку обе ОЛЦ находятся в одном блоке УП, и при выполнении перехода будут находиться в одной строке кэш-памяти. Таким образом, данный переход оказывается «безопасным» и исключается из выражения (5), записанного для МК a_1. При этом «опасный» вес МК a_1 уменьшается, а следовательно, уменьшается «опасный» вес ОЛЦ O_1

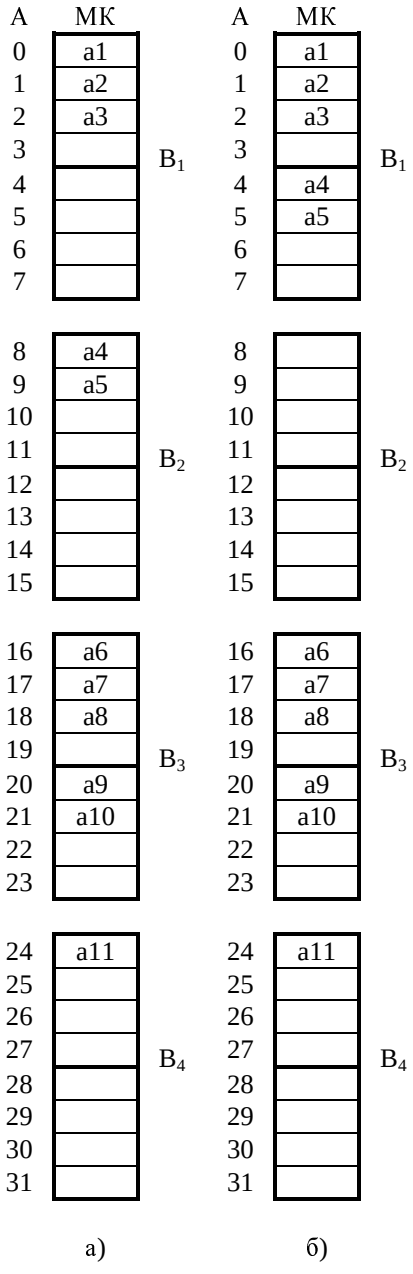

Рисунок 5 – Пример уменьшения опасного веса ОЛЦ

С другой стороны, существует переход из $a_3 \in O_1$ в $a_4 \in O_2$. Следовательно, при совмещении O_1 и O_2 в одном блоке уменьшается «опасный» вес a_4, а значит, и «опасный» вес ОЛЦ O_2.

Таким образом, при совмещении ОЛЦ O_1 и O_2 в одном блоке происходит уменьшение

«опасных» весов обеих ОЛЦ. Результатом является уменьшение количества возникновений кэш-промахов при переходах между данными ОЛЦ, что приводит к увеличению вероятности кэш-попаданий алгоритма: при $p(x_1)=0.2$, $p(x_2)=0.7$, $p(x_3)=0.5$, $p(x_4)=0.1$, $p(x_5)=0.9$ для варианта «а» $p_h=0.6781$, для варианта «б» $p_h=0.7596$.

Из эвристики 3 может быть сделан следующий вывод: *наиболее целесообразно добавить в текущий блок ту ОЛЦ, при которой «опасный» вес блока с учетом уже имеющихся в нем ОЛЦ будет минимальным.*

Эвристика 4. *Увеличение количества ОЛЦ, на которые разбито множество микрокоманд рассматриваемой ГСА, не влияет на вероятности выполнения микрокоманд, но в силу выражения (1) увеличивает количество вариантов размещения ОЛЦ.*

Необходимо подчеркнуть, что данная эвристика ничего не говорит об изменении значения вероятности кэш-попаданий в связи с увеличением количества ОЛЦ. Если максимальный размер ОЛЦ значительно превышает средний размер ОЛЦ в ГСА, то скорее всего, будет иметь место большой процент неиспользуемых ячеек памяти в схеме УП. Разделение подобных «длинных» ОЛЦ на несколько более коротких позволит более эффективно заполнить УП, а также в случае уменьшения R(T) размещать большее количество ОЛЦ в одном блоке памяти. Увеличение количества ОЛЦ, которые могут быть размещены в блоке памяти, позволяет в общем случае снизить «опасный» вес блока и увеличить величину p_h для заданной ГСА.

В то же время подобные разбиения увеличивают количество строк ПСТ и, как следствие, аппаратурные затраты в схеме адресации, что снижает положительный эффект от возможного увеличения p_h и в ряде случаев может оказаться неприемлемым.

Таким образом, эффект от использования эвристики 4 оказывается неопределенным. Тем не менее, данная эвристика может быть учтена на этапе формирования ОЛЦ: если аппаратурные затраты в логической схеме проектируемого устройства управления не являются критичными, использование большего количества ОЛЦ меньшей длины в некоторых случаях может способствовать увеличению вероятности кэш-попаданий рассматриваемой ГСА.

Заключение

В настоящей работе найдены принципы оптимизации размещения операторных линейных цепей в управляющей памяти КМУУ с разделением кодов и кэшированием сигналов, приводящие к увеличению эффективности использования кэш-памяти микрокоманд. Данные принципы основываются на экспериментальных исследованиях и сформулированы в виде эвристических правил, основанных на статистической очевидности результатов экспериментов.

Сформулированные эвристики могут быть использованы при разработке эвристического алгоритма оптимизации формирования содержимого управляющей памяти. Реализация подобного алгоритма позволит в дальнейшем решить задачу повышения быстродействия схем КМУУ с разделением кодов и кэш-памятью, что, в свою очередь, приведет к повышению быстродействия вычислительных систем, использующих данный класс устройств управления.

Литература

1. Баркалов А. А, Палагин А.В. Синтез микропрограммных устройств управления. – Киев: ИК НАН Украины, 1997. – 136 с.

2. Баркалов А.А. Синтез устройств управления на программируемы логических устройствах. – Донецк: ДНТУ, 2002. – 262 с.

3. Баркалов А.А., Ковалев С.А., Бабаков Р.М., Николаенко Д.В. Организация композиционных микропрограммных устройств управления с разделением кодов и кэш-памятью // Искусственный интеллект. – 2007. – №3. – С. 135-138.