УДК 681.5.62–5+681.5.681.3

КОМПЛЕКСНАЯ СРАВНИТЕЛЬНАЯ ОЦЕНКА МЕТОДОВ ВЫБОРА РАЗБИЕНИЙ ПРИ ПРОЕКТИРОВАНИИ ЛОГИЧЕСКИХ МУЛЬТИКОНТРОЛЛЕРОВ

Э.И. Ватутин
Курский государственный технический университет
Россия, 305040, Курск, ул. 50 лет Октября, 94
E-mail: evatutin@rambler.ru

С.В. Волобуев
Курский государственный технический университет
Россия, 305040, Курск, ул. 50 лет Октября, 94
E-mail: magehunter@rambler.ru

И.В. Зотов
Курский государственный технический университет
Россия, 305040, Курск, ул. 50 лет Октября, 94
www: http://zotovigor.ru, E-mail: zotovigor@yandex.ru

Ключевые слова: параллельный алгоритм логического управления, проектирование логических мультиконтроллеров, разбиение, оптимизация
Key words: parallel logic control algorithm, logic multicontroller design, separation, optimization

В работе приведены предварительные результаты сравнения методов нахождения разбиений параллельных алгоритмов логического управления, применяемых в рамках проектировании логических мультиконтроллеров. Сравнение произведено на выборке случайных алгоритмов с использованием разработанной программной системы PAE. Описан состав частных показателей качества, влияющих на интегральный критерий качества разбиения, произведена оценка их значимости, выполнен анализ временных затрат на синтез разбиения. Продемонстрирована возможность получения разбиений с разной степенью приближения к оптимальному по различным показателям качества.

COMPREHENSIVE COMPARATIVE EVALUATION OF SEPARATION METHODS IN THE DESIGN OF LOGIC MULTICONTROLLERS / E.I. Vatutin (Kursk State Technical University, 94 50 Let Oktyabrya, Kursk 305040, Russia, E-mail: evatutin@rambler.ru), S.V. Volobuev (Kursk State Technical University, 94 50 Let Oktyabrya, Kursk 305040, Russia, E-mail: magehunter@rambler.ru), I.V. Zotov (Kursk State Technical University, 94 50 Let Oktyabrya, Kursk 305040, Russia, www: http://zotovigor.ru, E-mail: zotovigor@yandex.ru). The work presents preliminary results obtained in the comparison of parallel logic control algorithm separation methods employed in the design of logic multicontrollers. The comparative research is carried out using the developed PAE computer-aided separation environment based on the production of a comprehensive random sample of logic control algorithms. A set of partial quality coefficients affecting the integral separation quality is considered, their relative significance is examined, and separation synthesis time cost is analyzed. The potential of constructing separations differing in the deviation from the solution optimizing a given partial criterion is shown.

1. Введение

Одной из основных тенденций развития современных систем логического управления (СЛУ) является широкое внедрение принципов параллельности и модульности. Параллельные модульные СЛУ, часто называемые логическими мультиконтроллерами (ЛМК), способны выполнять комплексные управляющие алгоритмы теоретически неограниченной сложности за счет их декомпозиции на компоненты, распределяемые между модулями. Особый интерес представляют ЛМК, формируемые из однотипных микропрограммных модулей (контроллеров) и позволяющие достичь сверхвысокой производительности, обеспечивая при этом достаточную гибкость и хорошую контролепригодность.

Построение ЛМК сопряжено с необходимостью решения ряда оптимизационных задач на дискретных структурах. Особое место среди них занимает задача выбора разбиения алгоритмов управления, решаемая в процессе их декомпозиции. Качество ее решения напрямую влияет на аппаратную сложность мультиконтроллера и определяет время выполнения алгоритма управления.

В статье [1] был предложен параллельно-последовательный метод решения указанной задачи, основанный на серии эквивалентных редукционных преобразований и применимый к достаточно широкому классу управляющих алгоритмов. В монографии [2] показано применение данного метода при построении системы логического управления роботизированной сборочной ячейкой. Дальнейшее развитие разработанный метод получил в работе [3], а в [4] описана визуальная система декомпозиции алгоритмов логического управления PAE, содержащая его программную реализацию.

В настоящей статье приведены результаты сравнения параллельно-последовательного метода выбора разбиения [1–3] с наиболее близкими известными аналогами [5, 6]. Описана методика проведения сравнительной оценки, включающая вычислительный эксперимент со случайной выборкой управляющих алгоритмов в программной системе PAE. Рассмотрен способ генерации случайных алгоритмов управления с заданными предельными параметрами, использованный в экспериментальных исследованиях.

2. Постановка задачи

Задача выбора разбиения имеет ярко выраженный комбинаторный характер и заключается в оптимальном представлении исходного параллельного алгоритма в виде множества взаимосвязанных блоков (компонент) [1, 2]. Каждый такой блок в общем случае содержит набор нескольких несвязанных между собой фрагментов. Множество блоков разбиения в совокупности с соответствующими связями позволяет получить сеть взаимосвязанных компонентных алгоритмов, представляющую исходный управляющий алгоритм.

К получаемым блокам разбиения, а также к структуре сети компонентных алгоритмов предъявляется ряд требований, состав и содержание которых зависит от архитектуры ЛМК и его модулей, технологических ограничений, структурной организации мультиконтроллера, дисциплины межмодульного взаимодействия и т.д. К их числу относятся:

- минимальное число блоков разбиения;

- ограниченная сложность блоков, выражаемая максимально допустимым числом вершин (микрокоманд), микроопераций и логических условий;
- минимальная сложность сети межблочных связей;
- минимальная интенсивность межблочных взаимодействий.

Кроме того, к получаемым разбиениям предъявляется ряд дополнительных требований [2], наиболее важным из которых является отсутствие параллельных вершин в составе одного блока.

Формализованное представление задачи выбора разбиения имеет следующий вид. Пусть H – число блоков разбиения; $W_{\max}$ – емкость участка памяти микропрограмм контроллера, выделяемого для размещения микрокоманд блока (без учета дополнительных команд, обеспечивающих межмодульное взаимодействие); $n_{\text{ЛУ}}$ – число выводов контроллера, используемых для приема сигналов логических условий от объекта управления. Требуется получить разбиение множества вершин A^0 исходного управляющего алгоритма $Sep\left(A^0\right)=\left\{A_1, A_2, ..., A_H\right\}$, удовлетворяющее следующим условиям:

$$
\begin{aligned}
&\bigcup_{i=1}^{H} A_i = A^0, \quad A_i \neq \varnothing, \quad A_i \cap A_j = \varnothing, \quad i, j = \overline{1, H}, \quad i \neq j, \\
&\neg\left(a_i \omega a_j\right) \forall a_i, a_j \in A_k, \quad i \neq j, \quad k = \overline{1, H}, \\
&W\left(A_i\right) \leq W_{\max}, \quad \left|X\left(A_i\right)\right| \leq n_{\text{ЛУ}}, \quad i = \overline{1, H},
\end{aligned}
\tag{1}
$$

где $W\left(A_i\right)=\sum_{a_j \in A_i} W\left(a_j\right)$ – суммарный вес вершин в составе i-го блока, $X\left(A_i\right)=\bigcup_{a_j \in A_i} X\left(a_j\right)$ – множество логических условий, входящих в вершины i-го блока, такое что

$$
\begin{aligned}
&H \to \min; \\
&Z_1 = \sum_{i=1}^{H-1} \sum_{j=i+1}^{H} \alpha\left(A_i, A_j\right) \to \min; \\
&Z_2 = \sum_{i=1}^{H-1} \sum_{j=i+1}^{H} \delta\left(A_i, A_j\right) \to \min; \\
&Z_3 = \max_{i, j=\overline{1, H}, \; i \neq j} \left\{E\left(A_i\right) - E\left(A_j\right)\right\} \to \min,
\end{aligned}
$$

где Z_1 – сложность сети межблочных связей, порождаемая разбиением $Sep\left(A^0\right)$; $\alpha\left(A_i, A_j\right)$ – коэффициент связи блоков (он равен 1, если блоки разбиения связаны, и 0 в противном случае); Z_2 – суммарное число (интенсивность) межблочных взаимодействий; $\delta\left(A_i, A_j\right)$ – интенсивность взаимодействия блоков; Z_3 – дискриминант блоков по сложности; $E\left(A_i\right)$ – функция, характеризующая сложность блока разбиения.

3. Оценка качества получаемых разбиений

Всякое разбиение параллельного алгоритма управления на блоки, полученное произвольным методом в соответствии с условиями (1), можно оценить по ряду критериев, к которым относятся:

- количество блоков разбиения;
- количество дублирующихся микроопераций и логических условий в составе различных блоков;
- интенсивность межблочного взаимодействия;
- сложность сети межблочных связей;
- разность алгоритмов по сложности.

(для каждого из критериев меньшее значение является лучшим).

Учитывая потребность в выявлении методов, дающих разбиения лучшего качества, необходимо определение интегрального показателя качества разбиения, объединяющего в себе все перечисленные выше критерии. Поскольку разные критерии качества разбиения имеют отличающиеся диапазоны значений и различную значимость, необходимо введение нормирующих коэффициентов с целью приведения диапазонов изменения критериев к интервалу [0, 1], а также весовых коэффициентов, отражающих значимость параметров. С учетом представленных требований интегральный показатель качества (оценочную функцию), как было предложено в [7], можно записать в виде

$$
\begin{aligned}
f\left(Sep\left(A^0\right)\right) &= \frac{K_H}{\omega_{\max}} H + \frac{K_X}{\left|X\left(A^0\right)\right|}\left(\sum_{i=1}^{H}\left|X\left(A_i\right)\right| - \left|X\left(A^0\right)\right|\right) + \\
&+ \frac{K_Y}{\left|Y\left(A^0\right)\right|}\left(\sum_{i=1}^{H}\left|Y\left(A_i\right)\right| - \left|Y\left(A^0\right)\right|\right) + \frac{K_\delta}{\delta\left(A^0\right)}\sum_{i=1}^{H-1}\sum_{j=i+1}^{H}\delta\left(A_i, A_j\right) + \\
&+ \frac{K_\alpha}{\omega_{\max}\left(\omega_{\max}-1\right)}\sum_{i=1}^{H-1}\sum_{j=i+1}^{H}\alpha\left(A_i, A_j\right) + K_W \Delta W, \\
\Delta W &= \max_{i=1,H} W\left(A_i\right) - \min_{i=1,H} W\left(A_i\right).
\end{aligned}
\qquad (2)
$$

При этом весовые коэффициенты K_i, $i = \{H, Y, X, \delta, \alpha, W\}$ располагаются в числителях дробей, а нормирующие коэффициенты – в знаменателях. Лучшим будем считать разбиение, у которого значение оценочной функции минимально:

$$f\left(Sep\left(A^0\right)\right) \to \min.$$

При задании значений весовых коэффициентов K_i необходимо учитывать значимость ранжируемых критериев (таблица 1). С целью получения разбиений с различными свойствами, а также проверки правильности ранжирования критериев, возможно изменение значений весовых коэффициентов с последующей субъективной оценкой их влияния на качество разбиения и его соотнесение с поставленными требованиями. Например, в случае неудовлетворительно большой сложности сети межблочных связей или высокой интенсивности межблочных взаимодействий имеет смысл увеличить значения коэффициентов K_α и K_δ.

Таблица 1. Значимость критериев качества разбиения.

Критерий (весовой коэффициент)	**Значимость**	**Значение весового коэффициента**	**Примечание**
Число блоков разбиения (K_H)	высокая	1,00	–
Дублирующиеся логические условия (K_X)	высокая	0,40	Желательно выполнения условия $K_X > K_Y$, т.к. ограничение на количество логических условий является критичным, в то время как ограничение на количество микроопераций может быть ликвидировано путем параллельного объединения нескольких контроллеров
Дублирующиеся микрооперации (K_Y)	высокая	0,30	
Интенсивность межблочного взаимодействия (K_δ)	высокая или низкая	0,60 0,10	Значимость зависит от структурной организации мультиконтроллера
Сложность сети межблочных связей (K_α)	высокая	0,60	Желательно выполнение условия $K_H > K_\alpha$, т.к. рост числа блоков ведет к росту сложности сети межблочных связей
Различие алгоритмов по сложности (K_W)	низкая	0,05	–

Введение нормирующих коэффициентов обусловлено необходимостью сохранения равномерного влияния компонентов весовой функции на ее результирующее значение при обработке алгоритмов различной размерности и сложности.

4. Особенности подсчета значения интенсивности межблочного взаимодействия

Необходимость в определении значения интенсивности межблочных взаимодействий Z_2, как показано в работе [10], возникает в двух случаях:

- при расчете приращения ΔZ_2 на этапе вычисления значения весовой функции при распределении субсечений по блокам [2, 3, 11] (в параллельно-последовательном методе);
- при окончательной оценке качества разбиения (в программной системе PAE [4, 7]).

Самым простым способом подсчета, обладающим линейной временной сложностью от количества дуг, является выделение множества дуг, отвечающих за передачу управления между блоками, и нахождение суммы их интенсивностей:

$$Z_2^* = \sum_{i=1}^{H-1} \sum_{j=i+1}^{H} \sum_{k=1}^{n_{ij}} \delta_k \left(A_i, A_j \right),$$

где n_{ij} – количество дуг, связывающих блоки A_i и A_j; $\delta_k\left(A_i, A_j\right)$ – интенсивность k-й дуги, связывающей блоки A_i и A_j. Данный способ расчета интенсивности дает существенно завышенную оценку, т.к. далеко не все дуги передачи управления между блоками могут быть активированы одновременно.

Попытка преодолеть указанное несоответствие была предпринята в [2] путем введения понятия совместимости дуг и деления множества всех дуг, отвечающих за передачу управления между блоками, на два подмножества: совместимых $D_{\text{совм}}$ (могут быть активированы одновременно) и несовместимых $D_{\text{несовм}}$ (соответственно не могут). При этом оценку суммарной интенсивности, так же как и в предыдущем случае, предполагалось вычислять за линейное время по формуле

$$Z_2^{**} = \sum_{v_i \in D_{\text{совм}}} \delta\left(v_i\right) + \max_{v_j \in D_{\text{несовм}}} \delta\left(v_j\right).$$

Однако, как показали дальнейшие исследования [11], в общем случае невозможно однозначно провести грань между совместимыми и несовместимыми дугами. В [11] как альтернативный вариант оценки было предложено выделение всех максимальных по включению подмножеств дуг, состоящих в отношении совместимости, расчет по ним значений интенсивности подмножества $\delta_k\left(D^k_{\text{совм}}\right) = \sum_{v_i \in D^k_{\text{совм}}} \delta\left(v_i\right)$ и затем выбор максимальной интенсивности $Z_2^{***} = \max_{k=\overline{1,M}} \delta_k\left(D^k_{\text{совм}}\right)$ (M – количество подмножеств) в качестве результирующего значения.

Но и у этого способа есть несколько существенных недостатков. Прежде всего, выделение максимальных по включению подмножеств совместимых дуг фактически является задачей выделения полных подграфов (клик), а эта задача, как известно, является NP-трудной. Уже при рассмотрении множества из 16 дуг время выделения всех подграфов составляет около 40 секунд (для б**о**льших размерностей подмножеств наблюдается стремительный рост времени, не позволяющий практически реализовать данный метод подсчета интенсивности).

Кроме того, получаемая оценка также является слегка завышенной (хотя и в меньшей степени, чем получаемая простым суммированием интенсивностей дуг), т.к. представляет собой худший вариант сочетания активации дуг, который на практике на самом деле может встречаться редко. Попытка рассмотрения всех вариантов как взвешенной суммы значений интенсивности $\delta_k\left(D^k_{\text{совм}}\right)$ k-го подмножества с вероятностью его появления на практике $p_k\left(D^k_{\text{совм}}\right)$ невозможна из-за априорной неопределенности самих значений вероятностей.

Третий недостаток данного способа относится к определению приращения ΔZ_2 при расчете значения весовой функции на этапе распределения субсечений по блокам. Он заключается в том, что вычисление приращения производится только для дуг, соединяющих пару выбранных блоков A_i и A_j. При этом фактически предполагается, что дуги, соединяющие блоки A_i и A_j, совместимы с дугами, соединяющими любую другую пару подмножеств, что в общем случае неверно (данный прием уменьшает размерность рассматриваемых подмножеств в среднем в H раз ценой небольшого завышения суммарной оценки).

Фактически можно сделать вывод о том, что получение точной аналитической оценки интенсивности межблочного взаимодействия, в должной мере отражающей поведение реальной системы логического управления, невозможно для реальных алгоритмов управления размерностью более 10–15 вершин. Как выход из сложившейся ситуации можно использовать моделирование динамики алгоритма сетью Петри, однако при этом возникает целый ряд дополнительных вопросов: сколько тактов времени занимает исполнение операторных и условных вершин, как реализуются механизмы синхронизации, сколько тактов времени уходит на передачу управления между модулями, как реализуется подобная передача и т.д. Кроме того, для получения достоверных оценок интенсивности межблочных взаимодействий необходимо неоднократное моделирование процесса выполнения алгоритма, что может потребовать значительных временных затрат (особенно при наличии в алгоритме вложенных циклов). Т.е. задача моделирования процесса выполнения алгоритма может оказаться сложнее исходной задачи по нахождению разбиения как в вычислительном плане, так и в плане трудоемкости реализации.

Для иллюстрации изложенных выше соображений приведем пример параллельного алгоритма логического управления и два варианта разбиения, для которых проведем подсчет интенсивности межблочных взаимодействий.

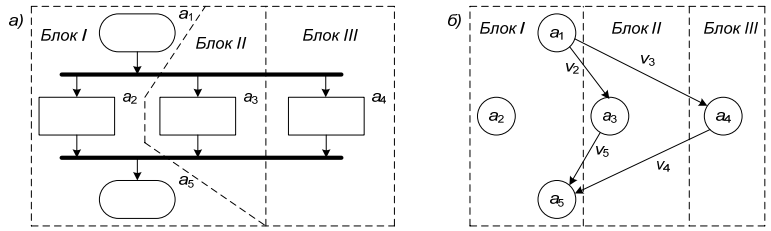

Рис. 1. Первый вариант разбиения.

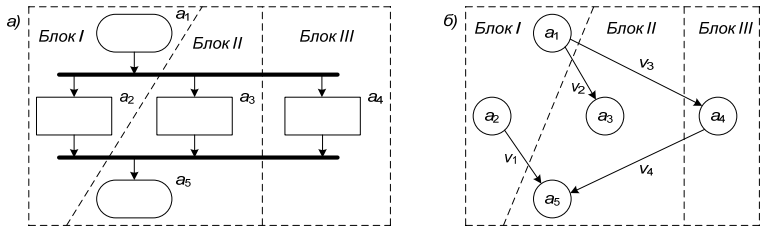

Рис. 2. Второй вариант разбиения.

Легко заметить, что и в первом, и во втором способе разбиения имеется по 4 дуги передачи управления между блоками, все дуги имеют интенсивность $\delta(v_i)=1$, расчет интенсивности межблочного взаимодействия как суммы интенсивностей приведет к значению $Z_2^{*}=4$. При расчете через подмножества совместимых дуг в первом примере максимальными по включению подмножествами совместимых дуг являются $\{v_2,v_3\}$, $\{v_2,v_4\}$, $\{v_5,v_3\}$, $\{v_5,v_4\}$ и максимальная интенсивность межблочных взаимодействий $Z_2^{***}=\max(2,2,2,2)=2$. Во втором случае также можно выделить подмножества совместимых дуг $\{v_1,v_2,v_3\}$, $\{v_1,v_2,v_4\}$ и максимальная интенсивность межблочных взаимодейст-

вий $Z_2^{***} = \max(3,3) = 3$. Как видно из примера, во-первых, оценка посредством отношения совместимости дуг является более точной $\left(Z_2^{***} \le Z_2^*\right)$, а во-вторых, при помощи оценки через совместимые дуги можно более детально классифицировать разбиения по качеству. Так, пример разбиения на рис. 1 на самом деле лучше разбиения на рис. 2, т.к. интенсивность межблочных взаимодействий меньше (чего не прослеживается при сравнении завышенных оценок Z_2^*).

5. Сравнение методов построения разбиений на выборке случайных алгоритмов

Для проведения объективного сравнения методов выбора разбиения между собой необходимо наличие достаточно объемного набора исходных данных (в данном случае – параллельных алгоритмов логического управления). Число доступных реальных примеров управляющих алгоритмов, так же как и число гипотетических алгоритмов, которые можно синтезировать за приемлемый для практики интервал времени, существенно ограничено. В связи с этим вызывает необходимость организации автоматической генерации гипотетических управляющих алгоритмов с заранее установленными предельными параметрами. Один из способов решения этой задачи предложен в [8].

В составе любого параллельного управляющего алгоритма рассматриваемого класса можно выделить набор образующих его типовых фрагментов. К ним относятся:

- начальный фрагмент (BE);
- линейный участок (LW);
- условное (альтернативное) ветвление (ALT);
- параллельный фрагмент (PAR);
- различные типы циклов – с пред- и постусловием, с прерыванием (LB, LA, LM).

(параллельные альтернативные ветвления и параллельные циклы [2] не рассматриваются, т.к. их можно заменить на перечисленные выше последовательные фрагменты в комбинации с параллельным фрагментом, как показано в [8]). Прямоугольниками с пунктирными границами (рис. 3) обозначены в общем случае произвольные дочерние фрагменты.

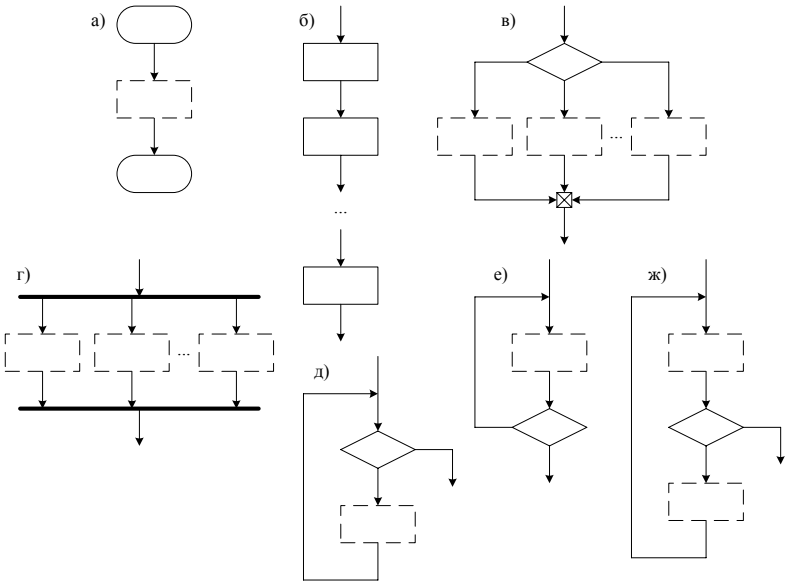

Рис. 3. Типовые фрагменты алгоритмов:
(а) – начальный фрагмент (BE);
(б) – линейный участок (LW);
(в) – альтернативное ветвление (ALT);
(г) – параллельный фрагмент (PAR);
(д) – цикл с предусловием (LB);
(е) – цикл с постусловием (LA);
(ж) – цикл с прерыванием (LM).

По аналогии с представленными типовыми фрагментами в случае необходимости можно синтезировать и более сложные фрагменты, встречающиеся в алгоритмах (рис. 4).

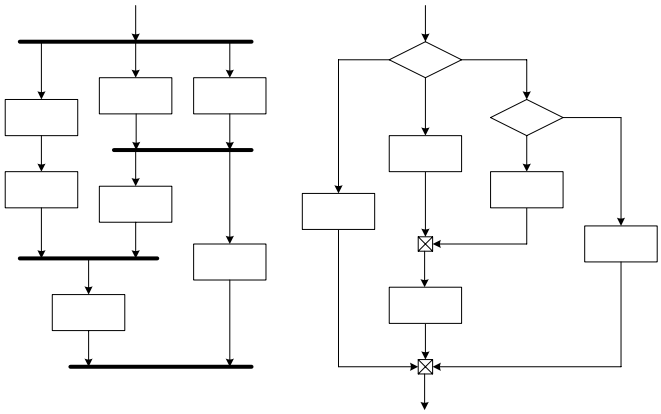

Рис. 4. Примеры более сложных типовых фрагментов.

Любой алгоритм (возможно с небольшими преобразованиями, не меняющими порядка выполнения операторов и его логики), изображенный в виде граф-схемы, может быть представлен в виде дерева, отражающего иерархию и вложение перечисленных выше типовых фрагментов. Пример подобного представления приведен на рис. 5.

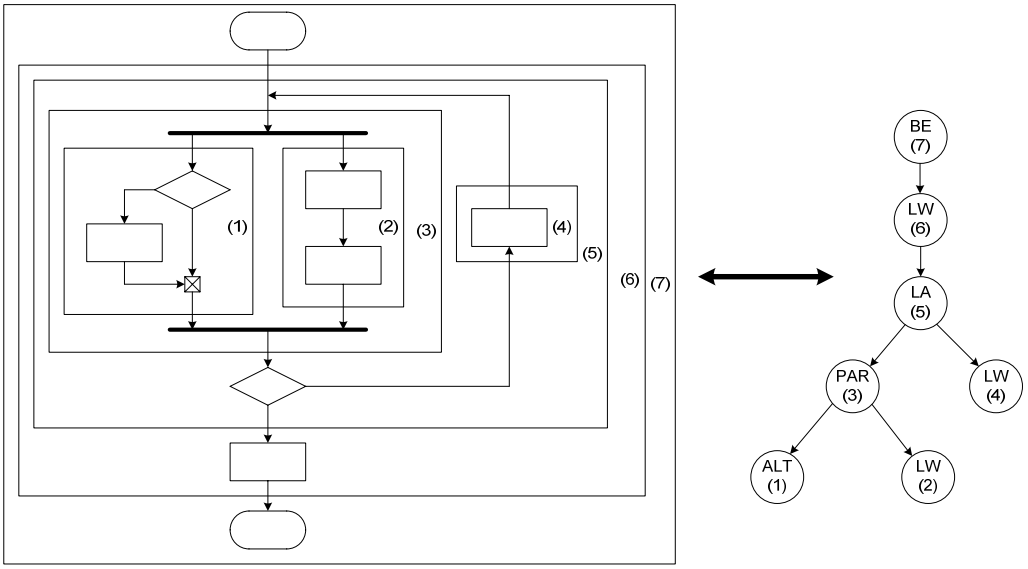

Рис. 5. Представление алгоритма в виде дерева фрагментов.

Таким образом, можно разработать алгоритм, который, основываясь на заданных предельных параметрах, способен синтезировать алгоритмы логического управления произвольного размера и существенно различных топологий. Такой подход может быть использован не только для сравнения качества разбиений, но и для тестирования корректности программных реализаций методов выбора разбиений (в ходе заключительного тестирования программной системы PAE на данный момент выявлено и исправлено 58 ошибок, 31 из которых обнаружена с использованием случайных алгоритмов).

Алгоритм генерации случайных алгоритмов логического управления [8] может быть представлен в виде двух укрупненных этапов:

1) Построение дерева фрагментов сверху вниз (от корня к листьям).
2) Построение случайного алгоритма путем укрупнения фрагментов снизу вверх (от листьев к корню).

Такой специфический порядок обхода дерева обусловлен тем, что результирующий случайный алгоритм необходимо представлять не просто в виде графа, а в виде набора фигур [4], чтобы сохранить возможность графического отображения сгенерированного алгоритма.

Первый этап генерации алгоритма управления более детально можно представить в виде следующего алгоритма:

1) Добавить в дерево фрагментов начальный фрагмент (BE).
2) Выбрать среди имеющихся в дереве фрагментов родительский фрагмент случайным образом пропорционально количеству его свободных дуг. (Под свободными понимаются дуги, которые могут участвовать в процессе замены на дочерний фрагмент. Количество свободных дуг изначально определяется типом фрагмента и его параметрами и уменьшается на единицу при каждом добавлении дочернего фрагмента (рис. 6)).
3) Выбрать тип добавляемого фрагмента случайным образом пропорционально вероятностям встречаемости фрагментов различных типов, определить его параметры (количество ветвей, количество вершин, вероятности активации дуг и т.д.) случайным образом. Добавить фрагмент в дерево.

4) Если выполнено условие завершения генерации (достигнуто заранее заданное суммарное количество вершин или количество фрагментов), конец алгоритма, в противном случае перейти к п. 2.

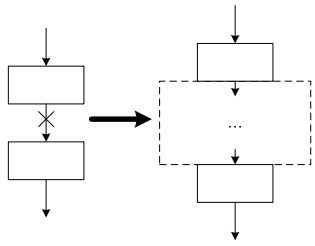

Рис. 6. Замена свободной дуги дочерним фрагментом.

В результате выполнения первого этапа образуется дерево фрагментов, которое может быть использовано непосредственно для синтеза алгоритма управления в виде графа. Алгоритм второго этапа выглядит следующим образом:

1) Пометить все фрагменты дерева как нерассмотренные.
2) Выбрать в дереве фрагмент, не имеющий нерассмотренных дочерних фрагментов. Сформировать его «геометрическую структуру»: подсчитать координаты вершин, дуг и дочерних фрагментов, определить геометрические размеры фрагмента (рис. 7). Настроить связи дуг дочерних фрагментов (если таковые имеются) с вершинами рассматриваемого фрагмента. Определить для дуг дочерних фрагментов значения вероятности активации и интенсивности передачи управления случайным образом. Определить случайным образом состав микроопераций и логических условий операторных и условных вершин. Пометить фрагмент как рассмотренный.
3) Если рассмотрены все фрагменты, конец алгоритма, в противном случае перейти к п.2.

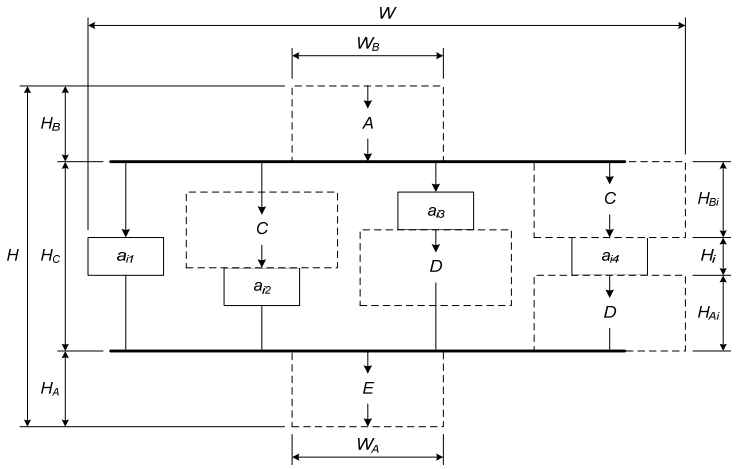

Рис. 7. Определение геометрической структуры типового фрагмента на примере параллельного фрагмента.

Следует отметить, что при определении геометрических параметров фрагментов (рис. 7) в общем случае геометрические параметры дочерних фрагментов не совпадают, что создает определенные затруднения при определении расположения ветвей, а также координат вершин и дуг.

В результате выполнения указанных выше действий получается гипотетический параллельный алгоритм логического управления, представленный в виде графа, причем все его вершины и дуги имеют координаты на плоскости и могут быть без особых затруднений изображены на экране.

Следует отметить, что в ходе подобной генерации алгоритма управления можно достаточно просто в вычислительном плане определить такую важную характеристику синтезированного алгоритма управления, как степень параллелизма.

С целью подтверждения состоятельности описанного выше алгоритма, а также демонстрации его возможностей, приведем несколько примеров сгенерированных ациклических алгоритмов управления, полученных с использованием программной реализации, входящей в состав программной системы PAE.

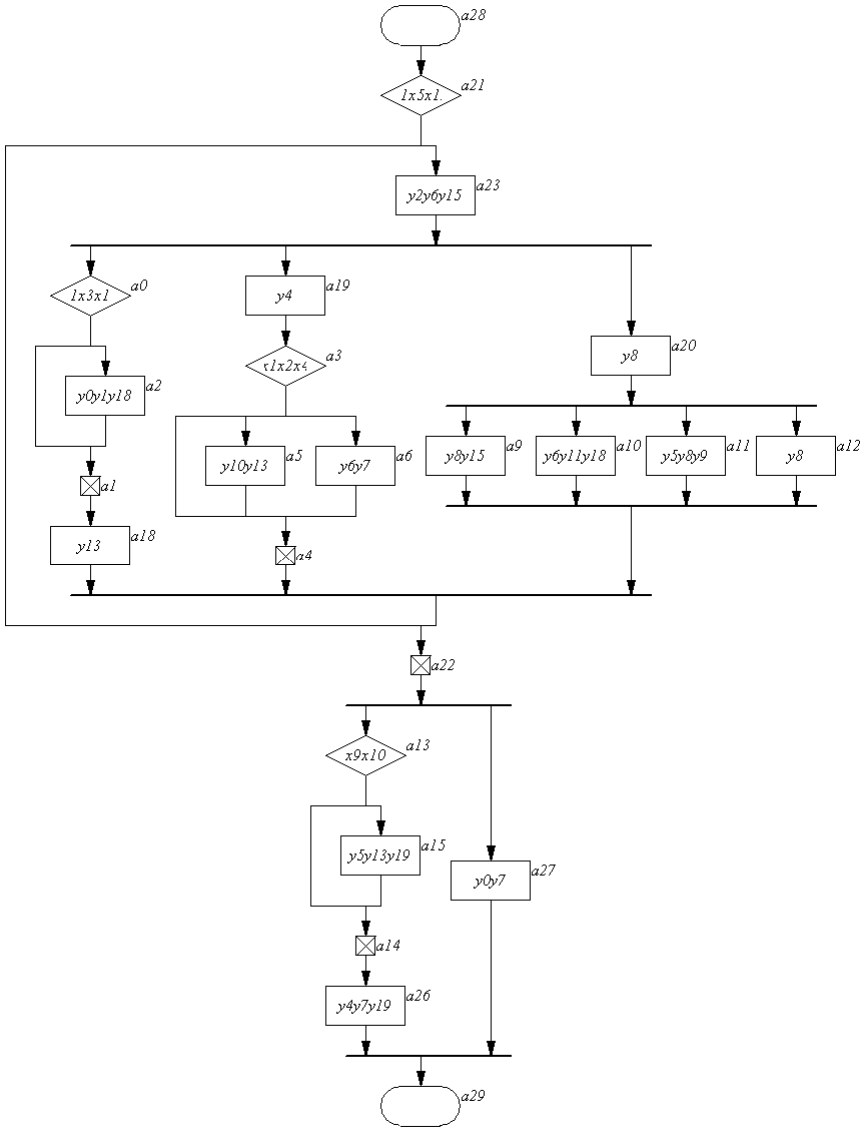

Рис. 8. Пример сгенерированного случайного алгоритма.

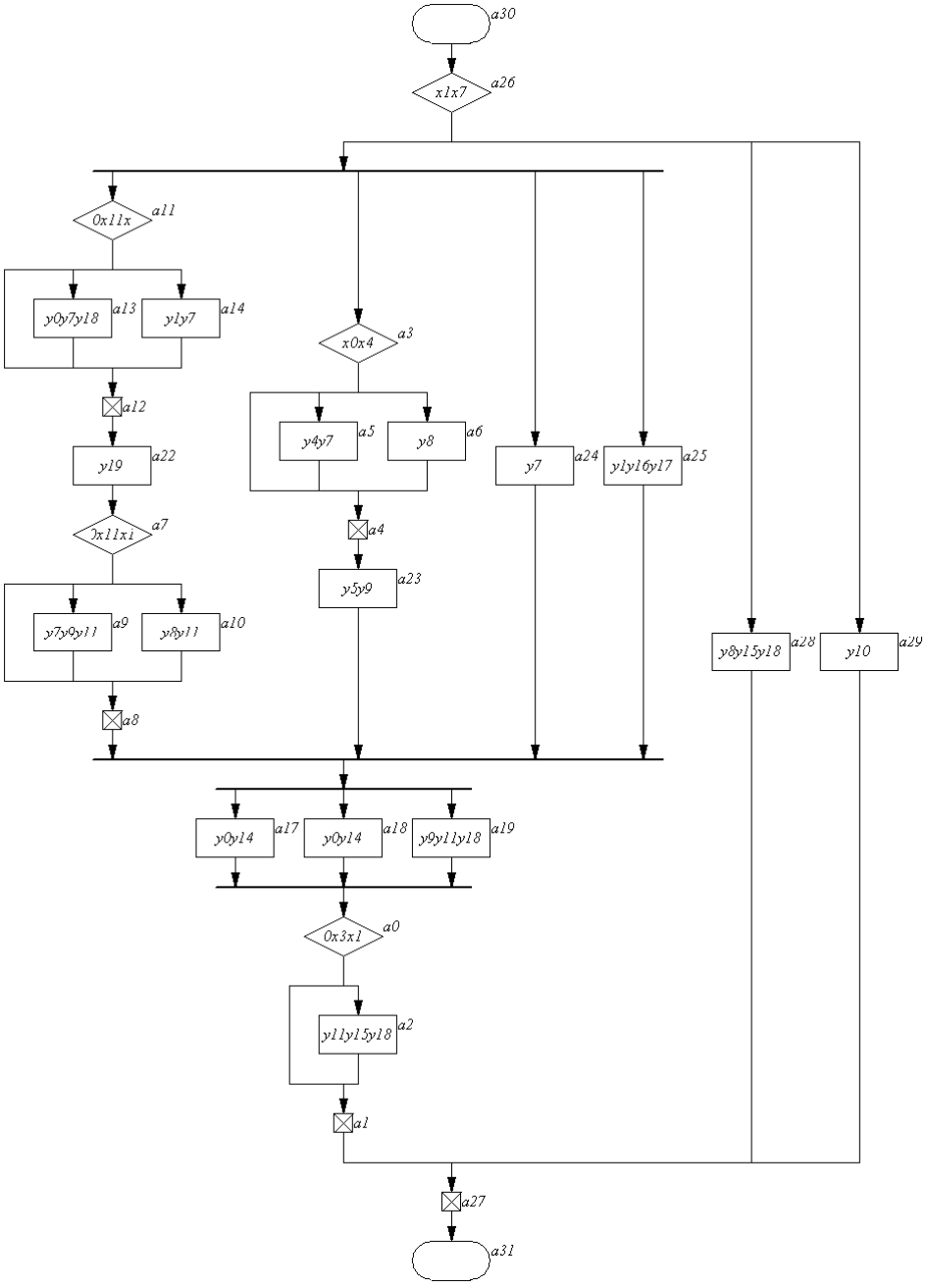

Рис. 9. Пример сгенерированного случайного алгоритма.

Алгоритмы, подобные приведенным на рис. 8 и 9, были использованы в составе выборки случайных алгоритмов при сравнении методов построения разбиений.

Необходимо заметить, что сгенерированные подобным образом алгоритмы управления слегка отличаются от реальных алгоритмов из-за некоторых допущений, полагаемых при генерации:

- используются только простые фрагменты (рис. 3);
- длины линейных участков равновероятны;
- количества ветвей в составе альтернативных и параллельных фрагментов равновероятны;

- количества микроопераций и логических условий в вершинных равновероятны;
- вероятности появления микроопераций и логических условий в вершинах равны.

Однако в целом принятые допущения не являются критичными и не должны оказать серьезного негативного влияния на объективность сравнения методов.

Скоростные характеристики алгоритма генерация случайных алгоритмов управления составляют приблизительно 1500 алгоритмов из 20 вершин (примерно) за 1 секунду (здесь и далее измерение временных характеристик программных реализаций методов и алгоритмов проводилось на компьютере с процессором Intel Celeron 850 МГц (CPUID=068Ah), материнской платой Gigabyte GA-6OXT (чипсет i815EP) и памятью SDR DIMM объемом 384 МБ).

6. Результаты сравнения методов выбора разбиений

6.1. Предварительное сравнение методов по составу критериев оптимизации

Краткий обзор возможностей методов выбора разбиений и основных оптимизируемых при этом критериев проведен в [9] и представлен в таблице 2.

Таблица 2. Состав критериев оптимизации различных методов выбора разбиений.

Название метода	**Критерии сравнения**				
	Количество блоков разбиения	**Распределение микроопераций**	**Распределение логических условий**	**Сложность сети межблочных связей**	**Интенсивность межблочных взаимодействий**
Параллельно-последовательный [1, 2, 3]	+	+	+	+	+
С.И. Баранова [4]	–	+	+	–	–
А.Д. Закревского [5]	+	–	–	–[1]	–
Полного перебора	+	+	+	+	+
Случайного перебора	+[2]	+	+	+	+

С целью обеспечения непротиворечивости сравнения методов необходимо обратить внимание на ряд замечаний.

Замечание 1. Метод А.Д. Закревского непосредственно не оптимизирует сложность сети межблочных связей, однако вследствие малого количества блоков зачастую наблюдается и невысокая сложность сети связей.

Замечание 2. Количество блоков минимизируется при задании маленькой вероятности попадания вершин в новый блок.

На основании представленной таблицы можно провести предварительное сопоставление методов. Метод С.И. Баранова является самым простым и изначально ориентирован на построение разбиений последовательных алгоритмов (нами рассматривается его обобщение на класс параллельных алгоритмов). Построение блоков разбиения проводится последовательно, при этом учитывается

распределение микроопераций, логических условий и «близость» расположения вершин к синтезируемому блоку, однако не оптимизируется ни количество блоков, ни сложность сети связей и интенсивность межблочных взаимодействий. Метод работает быстро в силу своей простоты. Метод А.Д. Закревского сводит решение данной задачи к нахождению минимальной раскраски специального графа, формируемого по описанию управляющего алгоритма на языке ПРАЛУ, качество разбиений в целом определяется качеством алгоритма раскраски. Метод при сравнительно низких временных затратах дает близкое к минимальному количество блоков и, как следствие, зачастую приводит к маленькой сложности сети межблочных связей (хотя непосредственно этот критерий не оптимизируется). Однако он не оптимизирует ни распределение микроопераций и логических условий, ни интенсивность межблочных взаимодействий. Разработанный авторами параллельно-последовательный метод является наиболее сложным (как в плане трудоемкости программной реализации, так и в плане временных затрат на построение разбиения), однако он оптимизирует все представленные в таблице критерии и работает с учетом технологических ограничений (что важно в случае ориентации на СБИС-реализацию контроллеров). Метод полного перебора по определению должен обладать наилучшими характеристиками, однако реально на практике он практически неприемлем из-за чрезмерных временных затрат на построение разбиения. Метод случайного перебора, как показывают предварительные исследования, требует перебора большого количества вариантов разбиений для достижения приемлемого качества, что трудновыполнимо на практике ввиду ограниченности временных затрат.

6.2. Практическое сравнение методов на выборке случайных алгоритмов

Практическое сравнение методов выбора разбиения проведем с использованием программной системы PAE на выборке из 1000 случайных алгоритмов со следующими предельными параметрами (таблица 3).

Таблица 3. Параметры генерации случайных алгоритмов.

Параметр	Значение
Объем выборки алгоритмов	1000
Частоты появления фрагментов	Линейный участок = 0,000 Условное ветвление = 1,000 Параллельный фрагмент = 1,000 Цикл с предусловием = 0,000 Цикл с прерыванием = 0,000 Цикл с постусловием = 0,000
Количество вершин (не более, примерно)	100
Количество фрагментов (не более)	8
Общее количество логических условий	20
Максимальное количество логических условий в вершине	3
Вероятность наличия у альтернативного ветвления "пустой" дуги [2]	0,900
Максимальное количество ветвей в условном ветвлении	3
Максимальное количество вершин в линейном участке	3
Общее количество микроопераций	20
Максимальное количество микроопераций в вершине	3
Максимальное количество параллельных ветвей	4

При проведении сравнения были использованы следующие программные реализации методов построения разбиений:

- параллельно-последовательный метод (разработчик Э.И. Ватутин) [13];
- метод С.И. Баранова (разработчик А.В. Тарловский) [14];
- метод А.Д. Закревского (разработчик С.В. Волобуев) [15].

Начнем рассмотрение результатов разбиения алгоритмов с анализа зависимостей, отражающих качество оптимизации отдельных критериев (рис. 10–14). На каждом из графиков приведены результаты качества оптимизации одного из критериев, при этом по оси абсцисс отложено отклонение в худшую сторону от лучшего значения рассматриваемого критерия, а по оси ординат – количество алгоритмов управления в предъявленной выборке, для которых данный метод обеспечил требуемое отклонение. Например, из рис. 10 видно, что в 77% случаев метод С.И. Баранова обеспечивает увеличение количества блоков разбиения не более чем на 2 по сравнению с лучшим решением, а метод А.Д. Закревского – в 94% случаев при том же увеличении количества блоков (глобальный оптимум в данном случае нам неизвестен из-за невозможности перебора всех возможных разбиений, поэтому в качестве его замены выбирается лучшее значение критерия среди разбиений, синтезированных сравниваемыми методами). Несложно заметить, что кривая наилучшего из методов на подобном графике должна быть расположена выше и левее кривых методов, дающих худшие результаты. Значение ординаты точки пересечения кривой с осью ординат (отклонение равно нулю) означает вероятность получения минимального значения оптимизируемого критерия (лучшего решения) в результате построения разбиения выбранным методом.

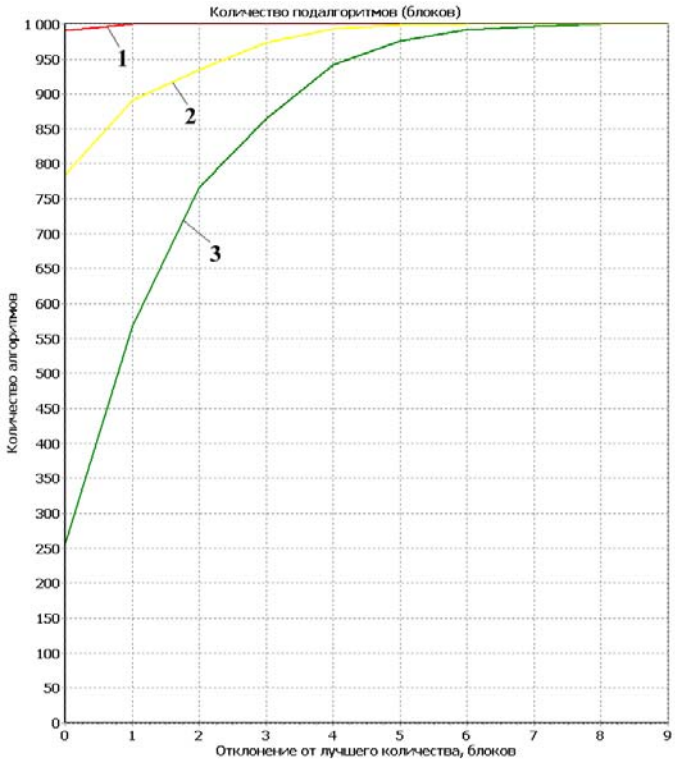

Рис. 10. Количество блоков разбиения (здесь и далее 1 – параллельно-последовательный метод, 2 – метод А.Д. Закревского, 3 – метод С.И. Баранова).

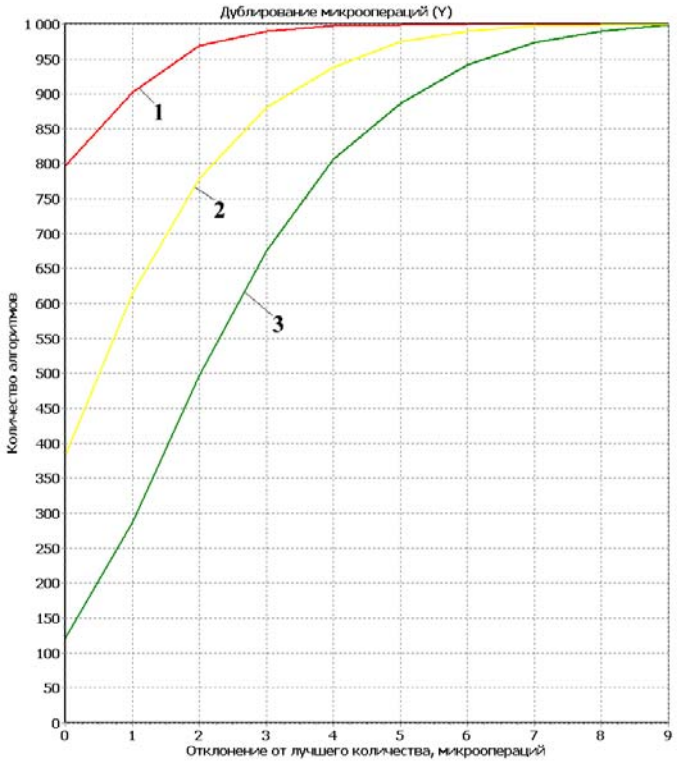

Рис. 11. Дублирование микроопераций.

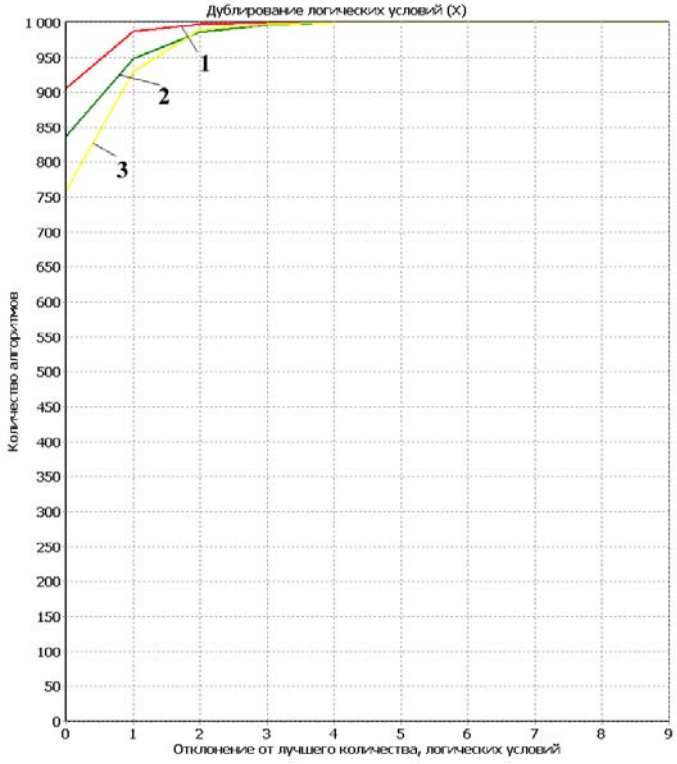

Рис. 12. Дублирование логических условий.

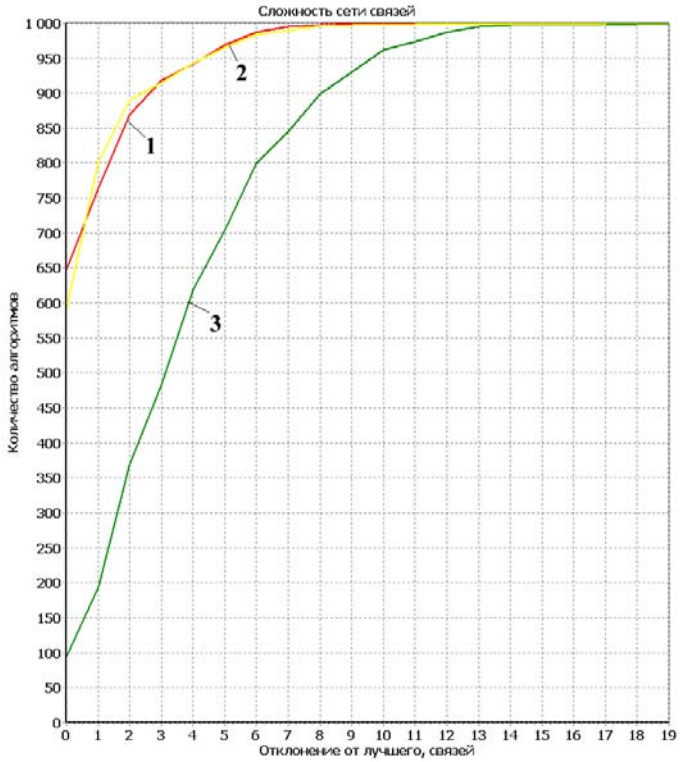

Рис. 13. Сложность сети межблочных связей.

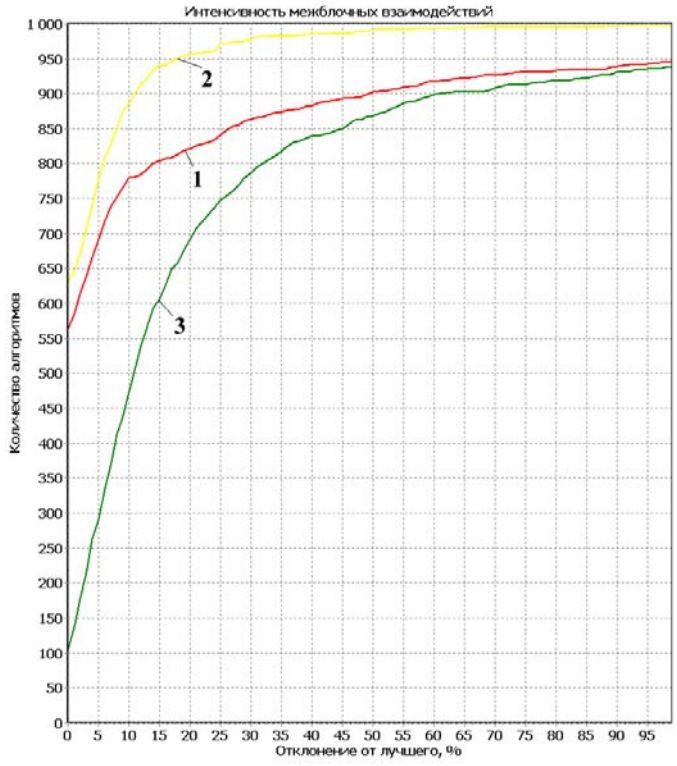

Рис. 14. Интенсивность межблочных взаимодействий.

Анализ представленных результатов показывает, что параллельно-последовательный метод проводит наилучшую оптимизацию количества блоков (минимальное количество в 99% случаев), а также распределения микроопера-

ций (минимальное дублирование в 80% случаев) и логических условий (минимальное дублирование в 91% случаев) при практически идентичной методу А.Д. Закревского оптимизации сложности сети связей (минимальное число межблочных связей в 65% случаев). Однако он уступает методу А.Д. Закревского по качеству оптимизации интенсивности межблочных взаимодействий (минимальное значение в 56% случаев).

С учетом того, что количество блоков разбиения и степень дублирования микроопераций и логических условий являются в большинстве случаев наиболее важными характеристиками, а вычисляемое значение интенсивности межблочных взаимодействий может не соответствовать реальному (значение является завышенным и оптимизируется с низким весовым коэффициентом), в целом качество разбиений, получаемых параллельно-последовательным методом, можно считать выше. Метод С.И. Баранова демонстрирует худшие результаты по всем параметрам, за исключением степени дублирования логических условий (даже несмотря на непосредственную оптимизацию распределения микроопераций и логических условий), что по видимому объясняется б**о**льшим средним количеством блоков в разбиениях (см. таблицу 4). Метод А.Д. Закревского, несмотря на непосредственную оптимизацию количества блоков разбиения, показывает «весьма средние» результаты (что может объясняться как особенностями метода, так и недостатками примененного алгоритма раскраски).

Кривые для интегрального критерия качества, рассчитанного по формуле (2), приведены на рис. 15.

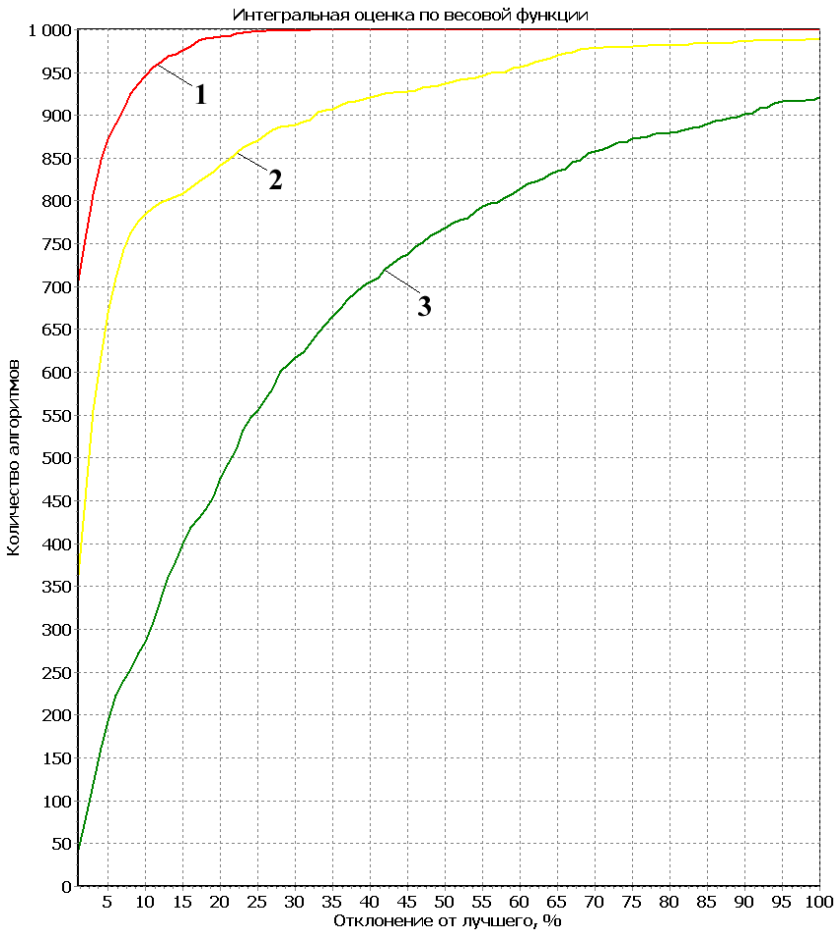

Рис. 15. Сравнение методов по интегральному критерию качества.

Из приведенных зависимостей видно, что параллельно-последовательный метод дает разбиения лучшего качества в 71% случаев по интегральному критерию. При допустимом ухудшении качества разбиения на 28% параллельно-последовательный метод дает 100% лучших разбиений. Метод А.Д. Закревского обеспечивает лучшее качество решений в 37% случаев, а метод С.И. Баранова – в 5% случаев.

Приведенные выше графики дают представление о вероятности получения лучшего разбиения каждым из методов, однако не несут количественной информации о средних числовых значениях оптимизируемых критериев для рассматриваемой выборки случайных алгоритмов, которые представлены в таблице 4.

Таблица 4. Средние значения параметров разбиений.

Метод	**Число блоков разбиения**	**Повторяющиеся сигналы логических условий**	**Повторяющиеся микрооперации**	**Сложность сети межблочных связей**	**Интенсивность межблочных взаимодействий**	**Значение оценочной функции**	**Время построения разбиения (мс)**
Параллельно-последовательный	6,249	0,378	9,363	13,382	14,916	1,841	118,507
Метод С.И. Баранова	7,878	0,502	11,842	16,628	16,008	2,517	12,100
Метод А.Д. Закревского	6,665	0,593	10,458	13,403	14,865	2,002	18,061

Таблица 5. Вероятность получения разбиения с минимальным значением критерия.

Метод	**Число блоков разбиения**	**Повторяющиеся сигналы логических условий**	**Повторяющиеся микрооперации**	**Сложность сети межблочных связей**	**Интенсивность межблочных взаимодействий**	**Значение оценочной функции**
Параллельно-последовательный	99%	91%	80%	65%	56%	71%
Метод С.И. Баранова	26%	84%	13%	10%	11%	4%
Метод А.Д. Закревского	79%	76%	38%	59%	64%	36%

Таблицы 4 и 5 дают общее представление о качестве оптимизации отдельных критериев и интегрального показателя качества и могут быть использованы, например, при оценке влияния тех или иных нововведений в методах (нововведение можно считать оправданным, если оно уменьшает среднее значение какого-либо критерия, либо увеличивает вероятность получения минимального значения критерия).

Анализ приведенных в таблице 4 средних значений оптимизируемых критериев показывает, что параллельно-последовательный метод обеспечивает ми-

нимальные средние значения числа блоков разбиения, степени дублирования микроопераций и логических условий, а также сложности сети межблочных связей и интегрального показателя качества при несколько худшем значении интенсивности межблочных взаимодействий (уступает методу А.Д. Закревского). Метод С.И. Баранова является худшим практически по всем критериям, за исключением степени дублирования сигналов логических условий.

Для полноты картины сравнения методов приведем также графики зависимости временных затрат на построение разбиения от среднего размера алгоритма, выражаемого в количестве вершин (рис. 16).

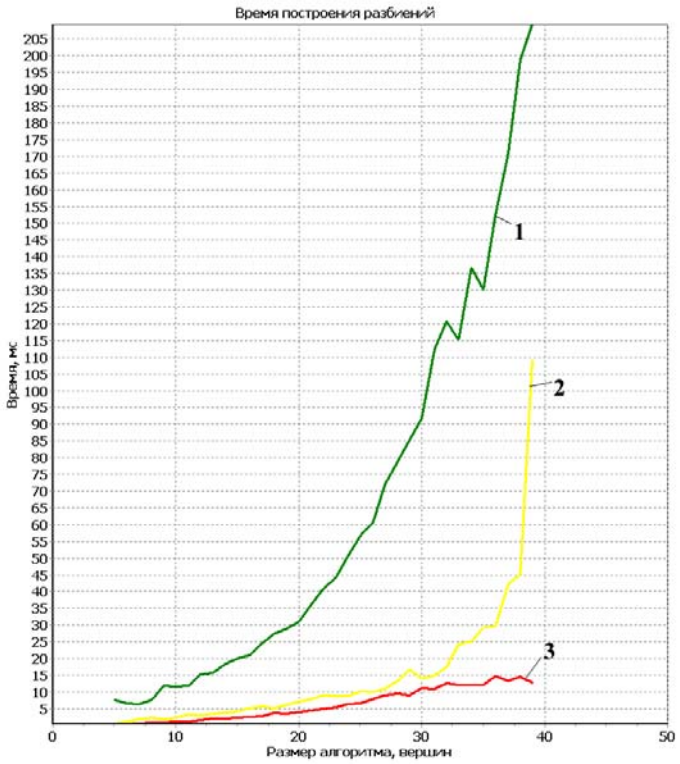

Рис. 16. Временные затраты на построение разбиений.

Видно, что метод С.И. Баранова требует минимум временных затрат, однако при этом он синтезирует разбиения самого низкого качества. Метод А.Д. Закревского и в особенности параллельно-последовательный метод характеризуются б**о**льшими временными затратами, причем тенденция роста затрат времени при росте размера алгоритма выражена наиболее ярко. С учетом того, что реальные алгоритмы управления имеют размерность порядка нескольких сотен вершин, временн**ы**е затраты на построение высококачественного разбиения, например, параллельно-последовательным методом могут быть весьма высоки. В таком случае необходимо либо довольствоваться разбиениями более низкого качества, получаемыми, например, с использованием метода С.И. Баранова достаточно быстро, либо проводить оптимизацию и/или распараллеливание программной реализации параллельно-последовательного метода, либо синтезировать аппаратный ускоритель для параллельно-последовательного метода.

Параллельно-последовательный метод имеет возможность получать разбиения с различными свойствами путем подбора значений коэффициентов весовой функции, используемой на этапе синтеза блоков разбиения [1–3, 11].

Оценим влияние коэффициентов весовой функции на качество получаемых разбиений. Для этого будем производить увеличение значения весового коэффициента (или пары коэффициентов, соответствующих оптимизируемому критерию) исследуемого критерия в 1000 раз и оценивать влияние на результирующие значения критериев (средние по выборке случайных алгоритмов). Результаты исследований представлены в таблице 6. При этом в качестве исходных были взяты следующие значения коэффициентов:

$$K_1^X = 0,015;\ K_2^X = 0,025;$$
$$K_1^Y = 0,01;\ K_2^Y = 0,01;$$
$$K_1^Z = 0,2;\ K_2^Z = 0,001;$$
$$K_W = 0,01.$$

Таблица 6. Влияние изменения весовых коэффициентов параллельно-последовательного метода на качество разбиений.

Параметр	Влияние
распределение логических условий	дублирование логических условий уменьшается в 2 раза при ухудшении остальных характеристик на 0,7%–4,6%
распределение микроопераций	дублирование уменьшается на 10% при ухудшении остальных характеристик на 2,9%–4,6%
разность алгоритмов по сложности	становится в 1,9 раза меньше при ухудшении остальных характеристик в 1,3-2,3 раза (в т.ч. количества блоков на 9,6%)
сложность сети связей и интенсивность межблочных взаимодействий	сложность уменьшается на 3%, интенсивность на 1,3% при увеличении дублирования логических условий на 2,3% и микроопераций на 2,1%

Результаты, представленные в таблице 6, показывают, что путем изменения весов коэффициентов можно получать разбиения с различным качеством оптимизации отдельных критериев. В случае необходимости можно более качественно оптимизировать выбранный критерий (группу критериев) в ущерб качеству оптимизации остальных. Однако подобной возможностью необходимо пользоваться с осторожностью, что демонстрирует пример с изменением весового коэффициента разности алгоритмов по сложности. По сути данный критерий предназначен только для более равномерного формирования блоков разбиения с целью выравнивания загрузки контроллеров. Высокое значение весового коэффициента этого критерия, как показывает пример, может привести к существенному ухудшению более важных характеристик разбиения.

7. Выводы

Основные результаты работы можно кратко представить в следующем виде.

1) Проведено сравнение методов выбора разбиений, в ходе которого подтверждена состоятельность параллельно-последовательного подхода, обеспечивающего более высокое качество разбиений в 71% случаев по интегральной оценке.

2) В результате вычислительного эксперимента с использованием выборки случайных алгоритмов управления установлено, что параллельно-последовательный метод обеспечивает минимальное количество блоков разбиения в 99% случаев и производит более оптимальное распределение микроопераций и логических условий в 80% и 91% случаев соответственно при обеспечении минимальной средней сложности сети межблочных связей и некотором ухудшении (менее 1%) интенсивности межблочных взаимодействий.
3) Подтверждено, что в результате изменения значений коэффициентов весовой функции параллельно-последовательного метода возможно получение разбиений с различной степенью оптимизации отдельных критериев качества.
4) Установлено, что при использовании параллельно-последовательного метода требуются более существенные временные затраты на нахождение разбиения по сравнению с другими методами. Качество получаемых разбиений при этом также выше.

Учитывая представленные выше результаты исследований, авторы считают необходимым дальнейшее развитие параллельно-последовательного подхода к формированию разбиений. При этом целесообразна активизация усилий, нацеленных на проектирование средств аппаратной поддержки формирования блочных разбиений [12].

Исследования И.В. Зотова выполнены при поддержке гранта Президента Российской Федерации для молодых ученых МК-3073.2007.8.

Список литературы

1. Зотов И.В., Колосков В.А., Титов В.С. Выбор оптимальных разбиений алгоритмов при проектировании микроконтроллерных сетей // Автоматика и вычислительная техника. 1997. № 5. С. 51–62.
2. Архитектура и синтез параллельных логических мультимикроконтроллеров: Учебное пособие: в 2 ч. Ч.1 / Зотов И.В. и др.; Курск: Изд-во КурскГТУ, 2006. 200 с.
3. Ватутин Э.И., Зотов И.В. Метод формирования субоптимальных разбиений параллельных управляющих алгоритмов // Параллельные вычисления и задачи управления (PACO'04). М.: Институт проблем управления им. В.А. Трапезникова РАН. 2004. С. 884–917.
4. Ватутин Э.И., Зотов И.В. Программная система для построения разбиений параллельных управляющих алгоритмов // Идентификация систем и задачи управления (SICPRO'06). М.: Институт проблем управления им. В.А. Трапезникова РАН, 2006. С. 2239–2250.
5. Баранов С.И., Журавина Л.Н., Песчанский В.А. Метод представления параллельных граф-схем алгоритмов совокупностями последовательных граф-схем // Автоматика и вычислительная техника. 1984. № 5. С. 74–81.
6. Закревский А.Д. Параллельные алгоритмы логического управления. Минск.: ИТК НАН Б. 1999. 202 с.
7. Ватутин Э.И. Оценка качества разбиений параллельных управляющих алгоритмов на последовательные подалгоритмы с использованием весовой функции // Интеллектуальные и информационные системы (Интеллект-2005). Тула, 2005. С. 29–30.
8. Vatutin E.I. Constructing Random Sample Parallel Logic Control Algorithms // 11th International Student Olympiad on Automatic Control (Baltic Olympiad BOAC'06). Saint-Petersburg, 2006. P. 162–166.
9. E.I. Vatutin, J.N. Abdel-Jalil, M.H. Najajra, I.V. Zotov. Comparison of Methods for Getting Separation of Parallel Logic Control Algorithms // Information and Telecommunication Technologies in Intelligent Systems (ITT IS'06). Katania, Italy, 2006. P. 92–94.
10. Ватутин Э.И. Проблема оценки интенсивности межблочного взаимодействия в задаче нахождения субоптимальных разбиений параллельных управляющих алгоритмов // Образование, наука, производство. Белгород, 2006.

11. Ватутин Э.И., Зотов И.В. Построение блоков разбиения в задаче декомпозиции параллельных управляющих алгоритмов // Материалы и упрочняющие технологии. Курск, 2003. Т. 2. С. 38–42.
12. Борзов Д.Б., Ватутин Э.И., Зотов И.В., Титов В.С. К задаче субоптимального разбиения параллельных алгоритмов // Известия вузов. Приборостроение. 2004. Вып. 12. С. 34–39.
13. Ватутин Э.И., Зотов И.В. Параллельно-последовательный метод формирования субоптимальных разбиений параллельных управляющих алгоритмов // Свидетельство об официальной регистрации программы для ЭВМ № 2005613091 от 28.11.05.
14. Тарловский А.В., Зотов И.В. Библиотека функций для разбиения параллельных алгоритмов логического управления модифицированным методом Баранова // Свидетельство об официальной регистрации программы для ЭВМ № 2006612337 от 05.07.06.
15. Волобуев С.В., Евглевский К.О., Зотов И.В. Библиотека функций для разбиения параллельных управляющих алгоритмов методом Закревского // Свидетельство об официальной регистрации программы для ЭВМ № 2006613146 от 06.09.06.