Труды МАИ. Выпуск № 119 http://trudymai.ru/

УДК 004.31 DOI: 10.34759/trd-2021-119-13

Математическая модель и аппаратно-ориентированный алгоритм планирования размещения программ в системах на кристалле

Масюков И.И.[1*], Борзов Д.Б.[1], Титов Д.В.[1***], Соколова Ю.В.[2****]**

[1]Юго-Западный государственный университет, ЮЗГУ,

ул. 50 лет Октября, 94, Курск, 305040, Россия

[2]Научно-производственное объединение им. С.А. Лавочкина,

ул. Ленинградская, 24, Химки, Московская область, 141400, Россия

**e-mail: ilmas46ru@gmail.com*

***e-mail: borzovdb@kursknet.ru*

**** e-mail: amazing2004@inbox.ru*

*****e-mail: jv.sokolova@mail.ru*

Статья поступила 15.06.2021

Аннотация

В данной работе представлена математическая модель и аппаратно-ориентированный алгоритм планирования программ в системах на кристалле, который позволит уменьшить временную задержку расчета новой топологии реконфигурируемой вычислительной системы и повысит ее отказоустойчивость. Производится программное моделирование разработанного алгоритма и алгоритма, основанного на идеях метода ветвей и границ для следующих типов графов: кольцевой, полносвязный, планарный, Кэли прямого произведения, звезда с применением построения Халина, двудольный.

Ключевые слова: система на кристалле, реконфигурация, архитектура, размещение, ПЛИС, конфигурация, математическая модель, алгоритм.

Введение

Реконфигурируемая вычислительная система (РВС) – это система, конфигурация которой может быть изменена после изготовления. РВС, в противовес другим вычислительным системам, за счет адаптации внутренней топологии под решаемую задачу, имеют более высокую производительность и энергоэффективность [1].

В общем виде РВС состоит из постоянной и реконфигурируемой части, которые возможно объединять для создания различных конфигураций. Многократное деление функциональных блоков в совокупности с гибкостью их соединений позволяет использовать множество вариантов параллелизма при выполнении программы. По сравнению с классической архитектурой набора команд, РВС позволяет изменять саму шину данных. Таким образом, РВС могут реализовать и программно-зависимые вычислительные структуры, при этом не нанося ущерба их гибкости. Эта особенность позволит в перспективе РВС занять место между архитектурой с набором команд и созданием специализированной архитектуре для конкретно реализуемой задачи[2].

Архитектура РВС представляет собой набор вычислительных структур (ВС), состоящих из множества вычислительных ячеек (ВЯ), соединённых между собой посредствам коммутаторов и распределенной памяти, управляемой по единой шине хост-процессором. Основной составной частью РВС является вычислительная ячейка (ВЯ), которая может быть настроена на выполнение какой-либо программы.

Одним из перспективных основ построения РВС является ПЛИС. [3-5] Построенные на ее базе РВС, реализующие вычислительно сложные задачи, имеют

следующие достоинства: высокую реальную и удельную производительность, высокую энергоэффективность, аппаратное конфигурирование, многократное использование и т. д.[6-7].

Одной из задач построения РВС в режиме реального времени является изменение организации внутренних модулей ПЛИС. [8-9] Однако это ведет к существенному росту времени коммутационной задержки. Снижение этого времени достигается путем перераспределения внутренних модулей. Это позволяет добиться максимально возможного быстродействия РВС в режиме ее основной работы. В следствии этого возникает необходимость изменения конфигурации ПЛИС, что приводит к снижению коэффициента готовности РВС. Последнее обстоятельство усугубляется тем, что существующие методы и алгоритмы решения задачи размещения имеют большую вычислительную сложность и решаются в основном программно [10].

Решение задач параллельных вычислений на одном вычислительном устройстве сложно осуществить, так как существует необходимость сложной программной реализации [11-13]. Также возникают дополнительные сложности, связанные с невозможностью в начале просчитать все возможные варианты решения размещения задач при проектировании РВС режима реального времени. Все вышесказанные ограничения не позволяют увеличить коэффициент готовности системы, что увеличивает время расчета планирования размещения внутренний модулей ПЛИС и данное решение становиться не эффективным способом повышения готовности РВС.

Следовательно, в настоящий момент есть противоречие между видимой необходимостью повышения производительности РВС режима реального времени и малого количества средств уменьшения времени конфигурирования и переконфигурирования внутренних модулей ПЛИС [14].

В соответствии с вышеизложенным научно-техническая задача разработки метода, алгоритма и аппаратных средств планирования и реконфигурирования вычислительных систем в режиме реального времени, обеспечивающих повышение надежности систем высокой готовности, является актуальной.

Постановка задачи

Под графом будем представлять взаимодействие задач [15].

$$X = \begin{Bmatrix} x_{1.1} & x_{1.2} & \cdots & x_{1.k} & \cdots & x_{1.n} \\ x_{2.1} & x_{2.2} & \cdots & x_{2.k} & \cdots & x_{2.n} \\ \cdots & & & & & \\ x_{q.1} & x_{q.1} & \cdots & x_{q.k} & \cdots & x_{q.n} \\ \cdots & & & & & \\ x_{n.1} & x_{n.2} & \cdots & x_{n.k} & \cdots & x_{n.n} \end{Bmatrix},$$

где множество вершин графа G, вершины $x_{qk} \in X$ которого соответствуют подпрограммам, а дуги $e_{ij} \in E$ связям между ними, которые передаются между подпрограммами и сведены в матрицу смежности (МС) $M = \left\| m_{ij} \right\|_{N \times |E|}$, где $N = |X|$.

Обозначим графом H размещение программ в ПЛИС:

$$H=\begin{Bmatrix} p_{1,1} & p_{1,2} & \cdots & p_{\varpi,\theta} & \cdots & p_{1,n} \\ p_{2,1} & p_{2,2} & \cdots & p_{\varpi,\theta} & & p_{2,n} \\ \cdots & & & & & \\ p_{m,1} & p_{m,2} & \cdots & p_{\varpi,\theta} & \cdots & p_{m,n} \end{Bmatrix},$$

где $P_{\varpi,\theta}$ – это отдельные модули ПЛИС, причем ($\varpi=\overline{1,m}$, $\theta=\overline{1,n}$).

Модуль $P_{\varpi,\theta}$ представляется в виду функции $F(O,X)$, т.е.

$$P_{\varpi,\theta}=F(O,X),$$

где O = o_1,ox_2,…, o_ξ - множество входных модуля, а X = x_1, x_2,…, x_ξ- множество выходов. Выводы модулей ПЛИС соединяется с выводами других модулей. Схематично модуль ПЛИС может быть представлено так, как показано на рисунке 1.

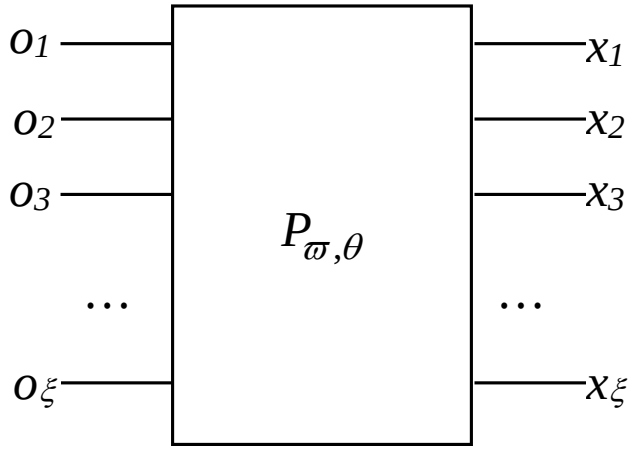

Рис. 1. Модуль ПЛИС

На рисунке 1 показано, что количество входов/выходов ПЛИС не известно заранее и зависит от конфигурации рассматриваемой системы.

Под смежными модулями (СМ) в ПЛИС будем понимать модули, которые соединены между собой межсоединениями одного вывода o_i или x_j ($(i=(1,\xi), j=(1,\xi))$ к другому.

Матрицей смежности модулей (МСМ) будем называть прямоугольную матрицу $M = (F(O,X), F(O,X))$, в ячейках которой заданы выводы, используемые для соединения модулей.

Граф смежности модулей (ГСМ) представляет собой граф, построенный по МСМ. Ребра графа описывают межсоединения между модулями, а вершины описывают сами модули. Вес ребра показывает количество межсоединений между модулями.

Метод и алгоритм

Для достижения наименьшей суммарной длинны межсоединений, предложено использовать 2 критерия:

1) Вершины, имеющие максимальное число смежных вершин γ, должны иметь минимальное расстояние σ до смежных им вершин

$$\begin{cases} \gamma \to \max \\ \sigma \to 1 \end{cases} (1),$$

2) Смежные вершины, имеющие максимальный вес ребер λ, должны иметь минимальное расстояние σ

$$\begin{cases} \lambda \to \max \\ \sigma \to 1 \end{cases} (2).$$

Размещение задач в ПЛИС опишем как:

$$\beta_s = \begin{Bmatrix} x_{S_{1.1}} & x_{S_{1.2}} & \dots & x_{S_{1.k}} & \dots & x_{S_{1.n}} \\ x_{S_{2.1}} & x_{S_{2.2}} & \dots & x_{S_{2.k}} & \dots & x_{S_{2.n}} \\ \dots & & & & & \\ x_{S_{q.1}} & x_{S_{q.1}} & \dots & x_{S_{q.k}} & \dots & x_{S_{q.n}} \\ \dots & & & & & \\ x_{S_{n.1}} & x_{S_{n.2}} & \dots & x_{S_{n.k}} & \dots & x_{S_{n.n}} \end{Bmatrix} \rightarrow \begin{Bmatrix} p_{1,1} & p_{1,2} & \dots & p_{1,k} & \dots & p_{1,n} \\ p_{2,1} & p_{2,2} & \dots & p_{2,k} & \dots & p_{2,n} \\ \dots & & & & & \\ p_{q,1} & p_{q,2} & \dots & p_{q,k} & \dots & p_{q,n} \\ \dots & & & & & \\ p_{n,1} & p_{n,2} & \dots & p_{m,k} & \dots & p_{m,n} \end{Bmatrix}, (3)$$

где $S = \overline{1, N!}$. В формуле (3) символ «→» означает отображение одной из вершин графа $x_{S_{q,k}} \in X$ на один из модулей $p_{q,k} \in H$. Здесь s обозначает номер очередной перестановки, соответствующий *s*-му варианту размещения. Мощность множества $\psi = \{\beta_S\}$ всевозможных отображений (4) равна числу всевозможных перестановок подпрограмм $x_{q,k} \in X$ в матрице $M : |\psi| = N!$.

Пусть Ψ – множество всевозможных отображений вида (3). Тогда задачу размещения, можно сформулировать как поиск отображения $\beta^* \in \Psi$, такого, что

$$T_{\beta^*} = \min_{\Psi} \begin{Bmatrix} \max \gamma \\ \max \lambda \end{Bmatrix}, (4)$$

где $\min_{\Psi}$ соответствует поиску минимальных суммарных межсоединений σ, для условия max γ (максимальное число смежных вершин) и max λ (максимальный вес ребер λ).

Тогда, на содержательном уровне метод размещения модулей ПЛИС, согласно условию (3), заключается в поиске такого варианта размещения, при котором выполняется условие (1), (2), (4). [16]

На основании метода был разработан аппаратно-ориентированной алгоритм [16]:

1. Из множества $x_{q,k} \in X$ происходит выбор вершин с наибольшим количеством инцидентностей. Если количество выбранных вершин равно 1, то берем полученную вершину как опорную (вокруг нее будет строится конфигурация) и переходим к пункту 3. Если вершин нет, переходим к пункту 4.

2. Для выбранных вершин в пункте 1 используем следующие критерии выбора:

а) Производится выбор вершины с наибольшим весом ребра (при размещении находится координата для двух вершин, соединенных этим ребром).

б) Производится выбор вершин, которые имеют большее количество инцидентностей с уже установленными вершинами.

Вершины, полученные в пунктах а), б), располагаются на модули $p_{q,k} \in H$ в обратной последовательности от вершин, выбранных в пункте б) к вершинам в пункте а. Выставленные вершины не используются в дельнейшем поиске. Возвращаемся к пункту 1.

3. Производится выбор инцидентных вершин к вершине, полученной в пункте 1, и отсеиваем их по следующим критериям:

а) Производится выбор вершин с наибольшим числом инцидентных ребер.

б) Производится выбор вершин с наибольшим весом ребра (при размещении находится координата для двух вершин, соединенных этим ребром).

в) Производится выбор вершин, которые имеют большее количество уже установленных инцидентных вершин.

Вершины, полученные в пунктах а)-в), располагаются на модули $p_{q,k} \in H$ в обратной последовательности от вершин, выбранных в пункте б к вершинам в пункте а). Выставленные вершины не используются в дельнейшем поиске. Возвращаемся к пункту 1.

4. Восстановление межсоединений.

5. Конец алгоритма.

Основное преимущество представленного метода и алгоритма планирования конфигурации ПЛИС заключается в одновременной выборке и поиске расположения в конфигурации выбранных вершин, что уменьшает суммарную длину полученных межсоединений в рассчитываемой конфигурации и количества операций перебора.

Численное моделирование

Для оценки работоспособности и эффективности разработанного алгоритма планирования программ на ПЛИС была разработана его программная модель. Сравнение разработанного алгоритма производилось с алгоритмом, основанным на идеях метода ветвей и границ [17-20], так как этот метод находит оптимальное решение. Сравнение проводилось для таких типов графов: кольцевой, полносвязный, планарный, Кэли прямого произведения, звезда с применением построения Халина, двудольный.

Суммарная длинна межсоединений рассчитывается следующим образом:

1) Полученный в процессе работы алгоритма результат записывается в виде матрицы.

2) Перебирается МСМ (правая верхняя половина). Если есть ребро - запоминаются номера вершин, которые оно соединяет.

3) Происходит поиск сохраненных вершин в матрице конфигурации (находятся их координаты). Расчет расстояния между этими вершинами происходит по закону нахождения гипотенузы у прямоугольного треугольника: вычитаются координаты двух найденных вершин (высчитываются катеты), и затем находится корень из суммы квадратов данных расстояний. Если вершины на одной прямой, то одона из координат будет равна 0, а корень из квадрата числа равняется самому этому числу.

4) Полученное расстояние перемножается со значением рассчитанного ребра из МСМ и суммируется к общему результату.

5) Алгоритм повторяется, пока не будут перебраны все вершины

Результаты исследования были получены на ЭВМ с процессором Intel i7-2630QM, 8Гб ОЗУ, Windows 10 x64. Исследования алгоритма проводились для следующих типов графов: кольцевой, полносвязный, планарный, граф Кэли прямого произведения, звезда с применением построения Халина, двудольный.

Пример сравнения алгоритмов для кольцевого типа графа приведен на рисунке 2 и 3.

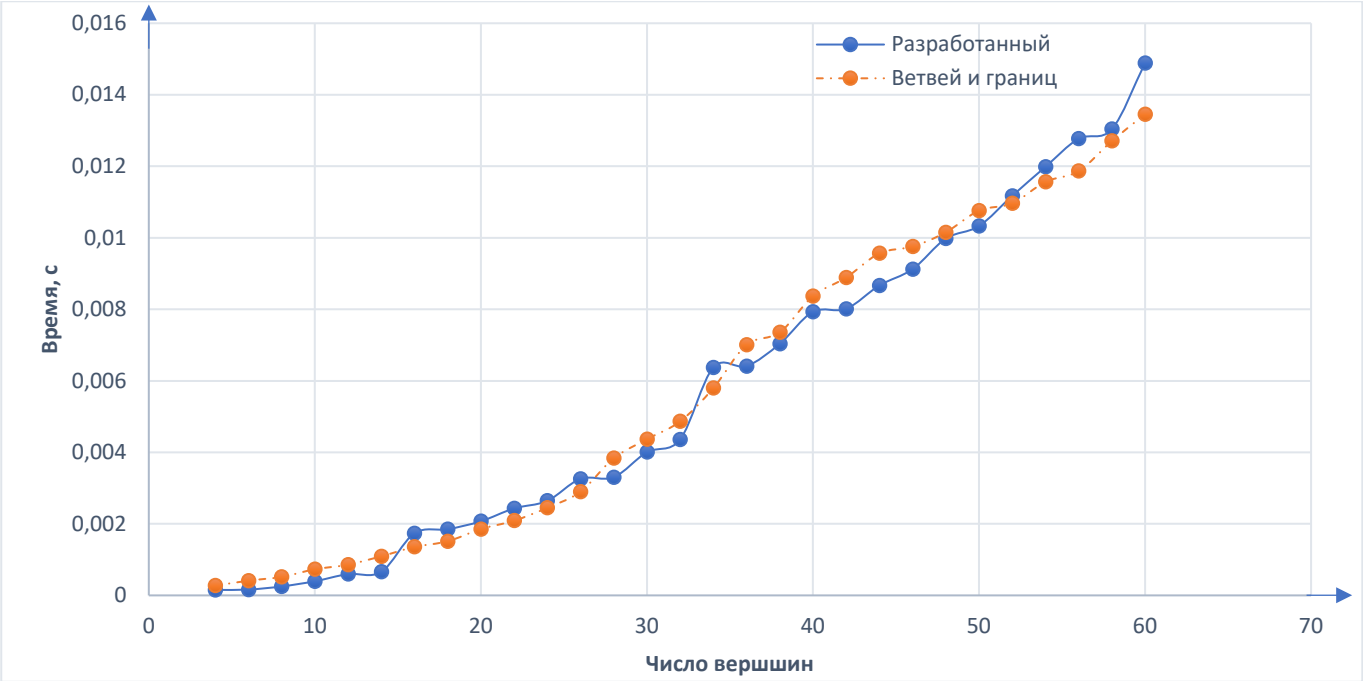

Рис. 2. График зависимости времени работы алгоритмов от количества вершин для графа кольцевого типа

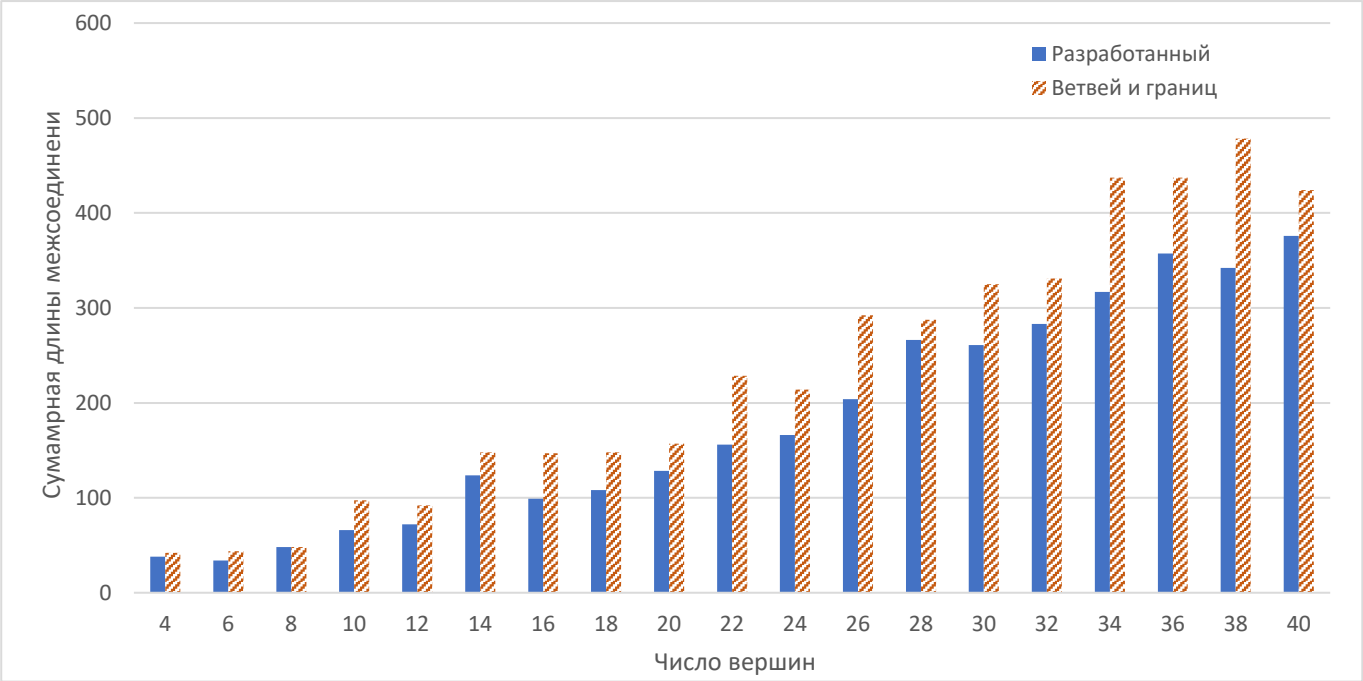

Рис. 3. Значения суммарных соединений для графа кольцевого типа

Из анализа полученных графиков можно сделать вывод, что разработанный алгоритм эффективнее для кольцевого типа графов по времени нахождения конфигурации на 5,42%, а по общей длине межсоединений на 10,88%.

Рассмотрим сравнение работы алгоритмов для полно связного графа (рисунок 4 и 5).

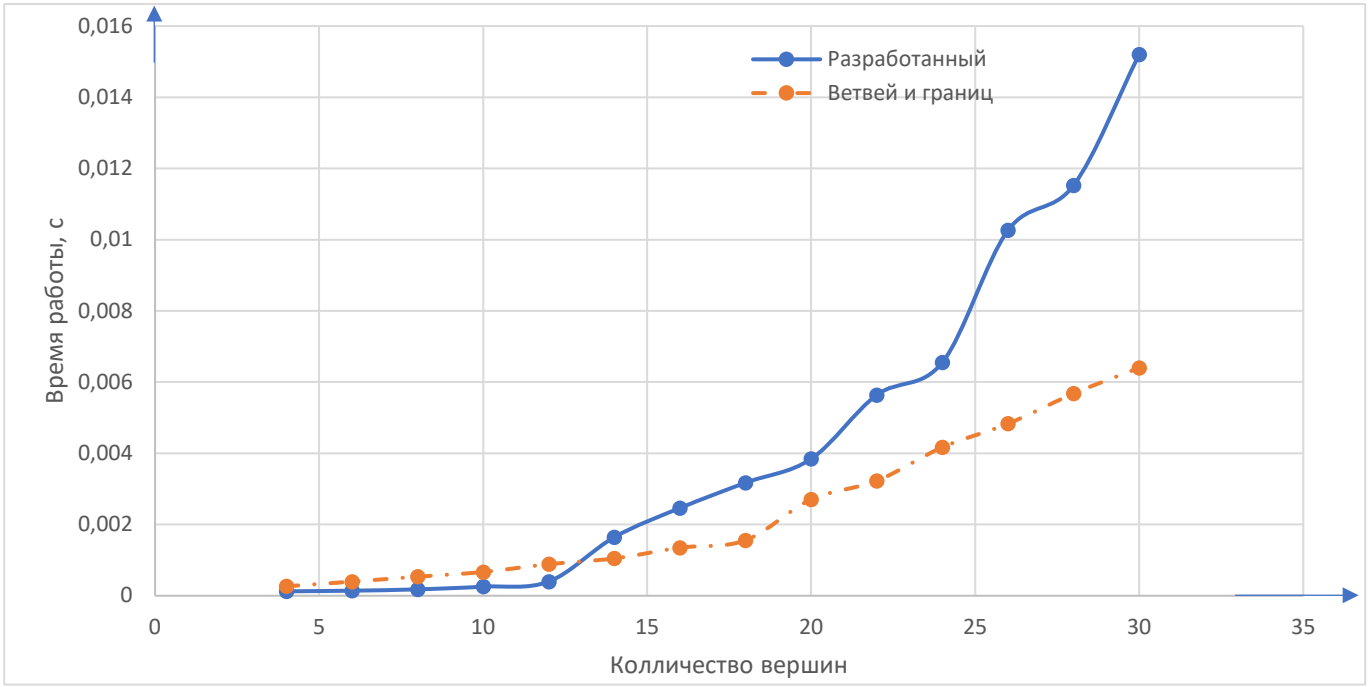

Рис. 4. График зависимости времени работы алгоритмов от количества вершин для полносвязного типа графа

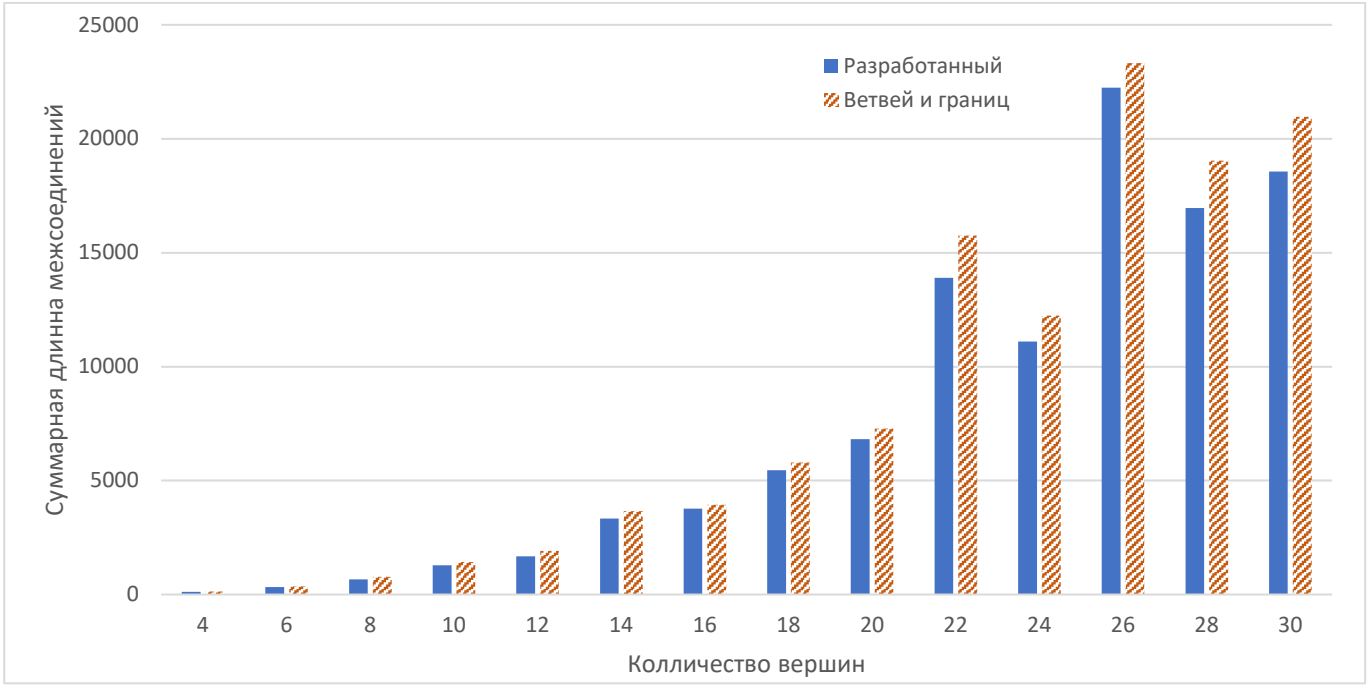

Рис. 5. Значения суммарных соединений на каждом шаге для полносвязного типа графа

Из анализа полученных графиков на рисунке 4 и 5 можно сделать вывод, что разработанный алгоритм менее эффективен по времени нахождения конфигурации – скорость поиска меньше на 33,54%, но более эффективен в нахождении конфигурации - общая длина межсоединений меньше на 8,37%.

На рисунках 6-7 приведены графики работы алгоритмов для планарного типа графа

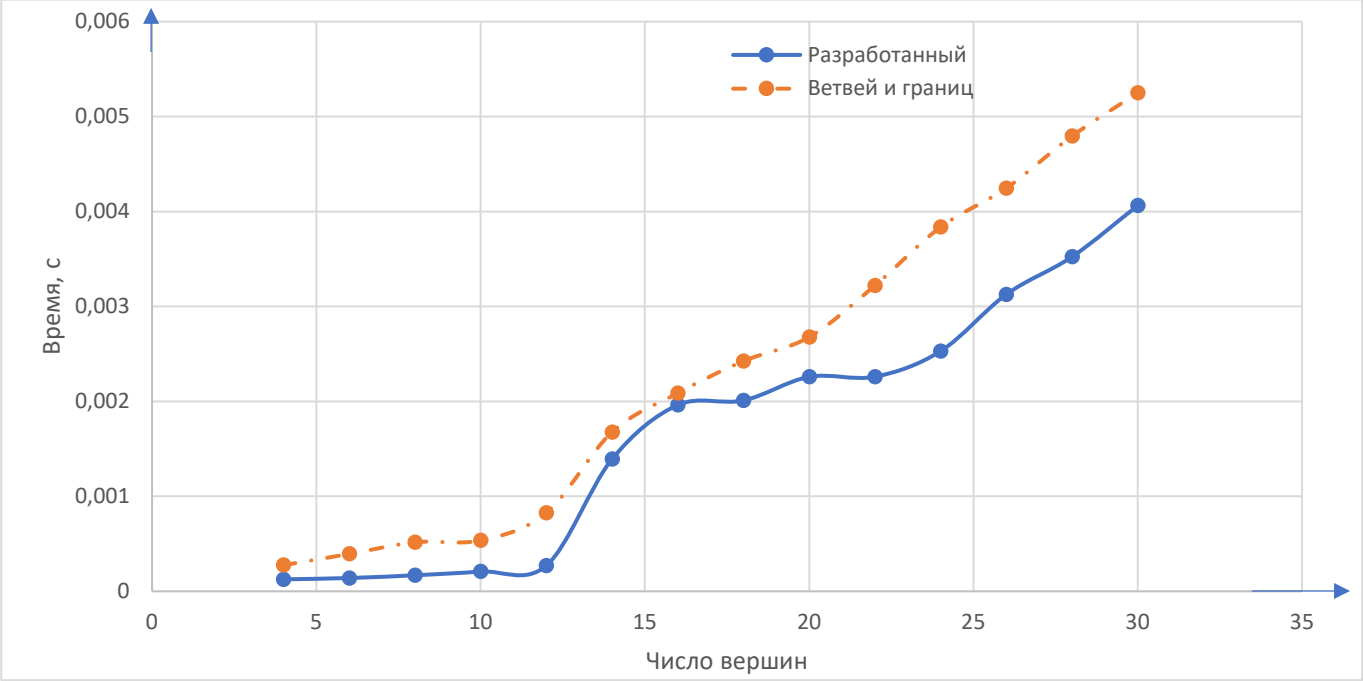

Рис. 6. График зависимости времени работы алгоритмов от количества вершин для планарного типа графа

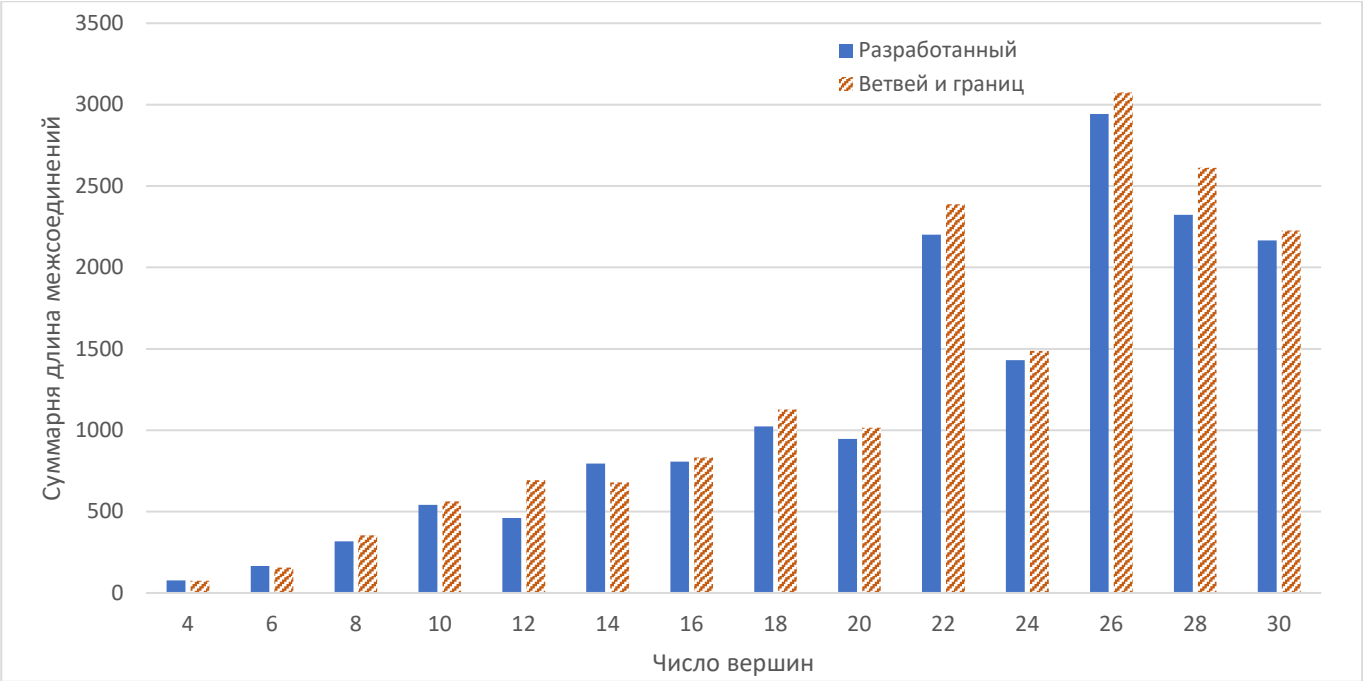

Рис. 7. Значения суммарных соединений на каждом шаге для планарного типа графа

Из анализа полученных графиков на рисунке 6 и 7 можно сделать вывод, что разработанный алгоритм эффективен по времени нахождения конфигурации на 36,4%, а общая длина межсоединений меньше на 5%.

На рисунке 8 и 9 приведены графики работы алгоритмов от количества вершин для графа Кэли прямого произведения.

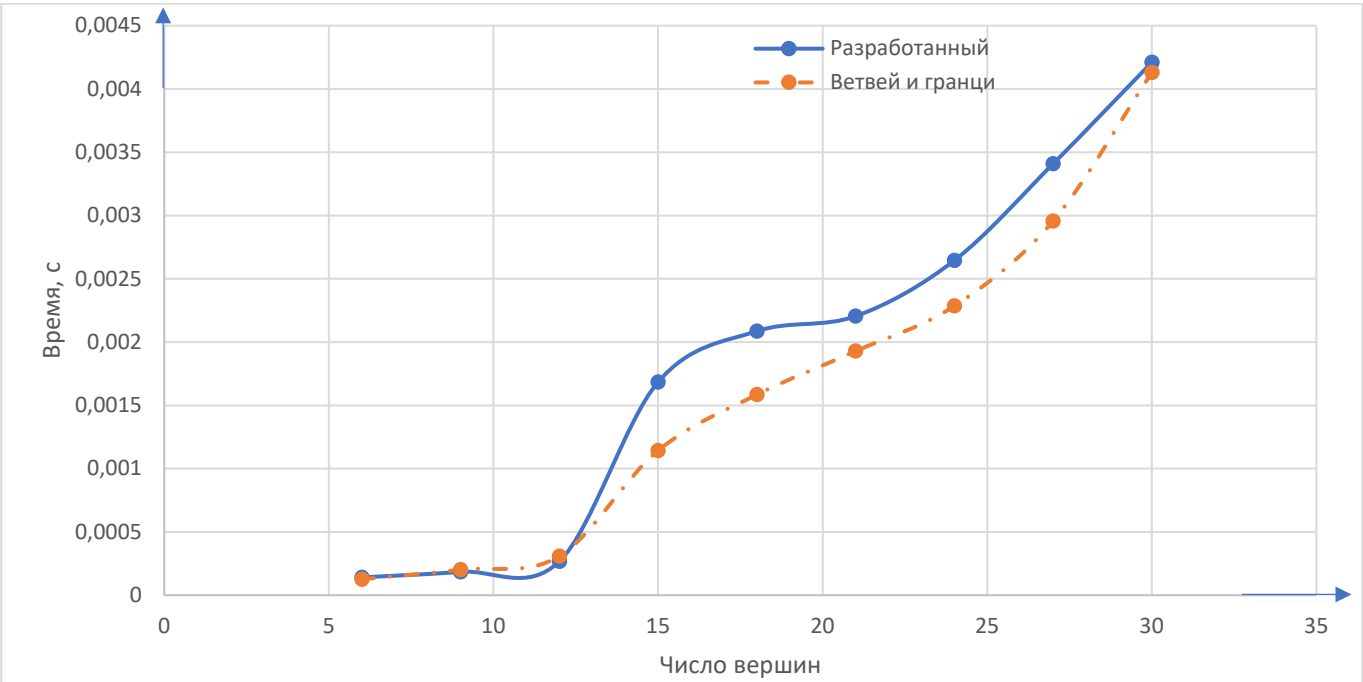

Рис. 8. График зависимости времени работы алгоритмов от количества вершин для графа Кэли прямого произведения

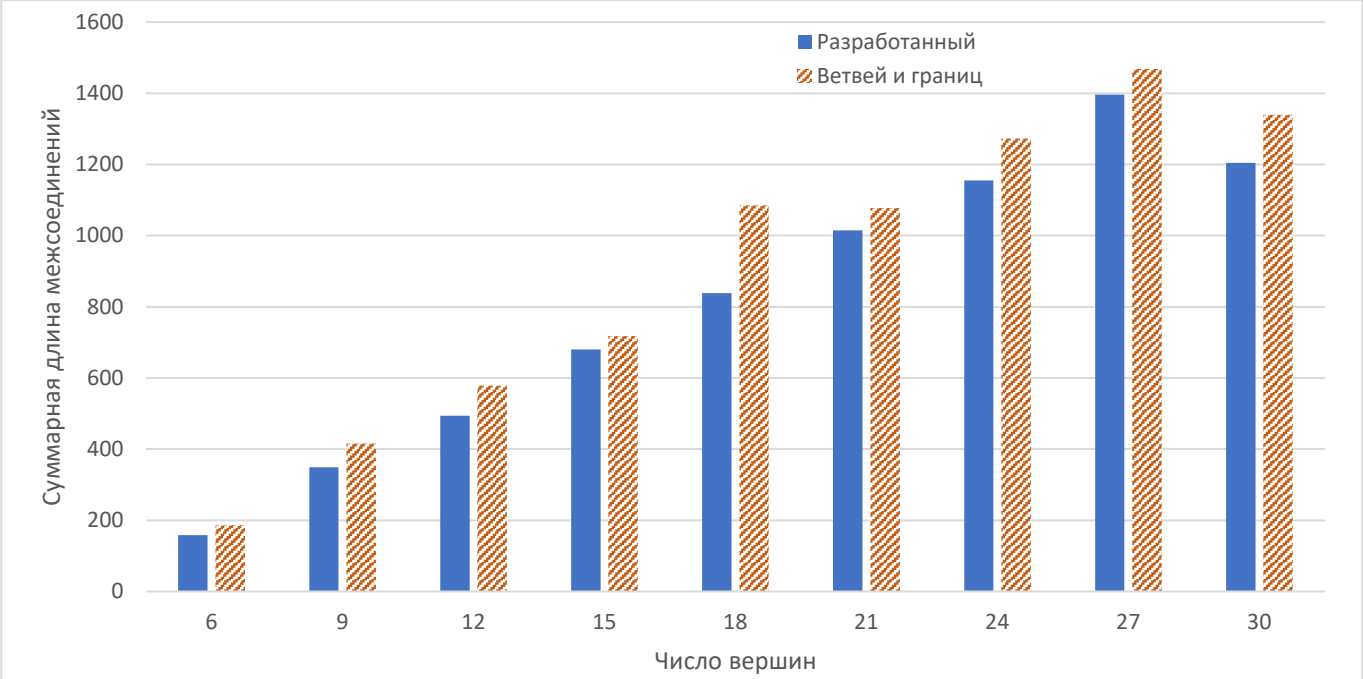

Рис. 9. Значения суммарных соединений на каждом шаге для графа Кэли прямого произведения

Из анализа полученных графиков на рисунке 8 и 9 можно сделать вывод, что разработанный алгоритм менее эффективен по времени нахождения конфигурации – скорость поиска меньше на 12,9%, но более эффективен в нахождении конфигурации - общая длина межсоединений меньше на 11,5%.

На рисунке 10 и 11 приведены графики работы алгоритмов для типа графа звезда с применением построения Халина.

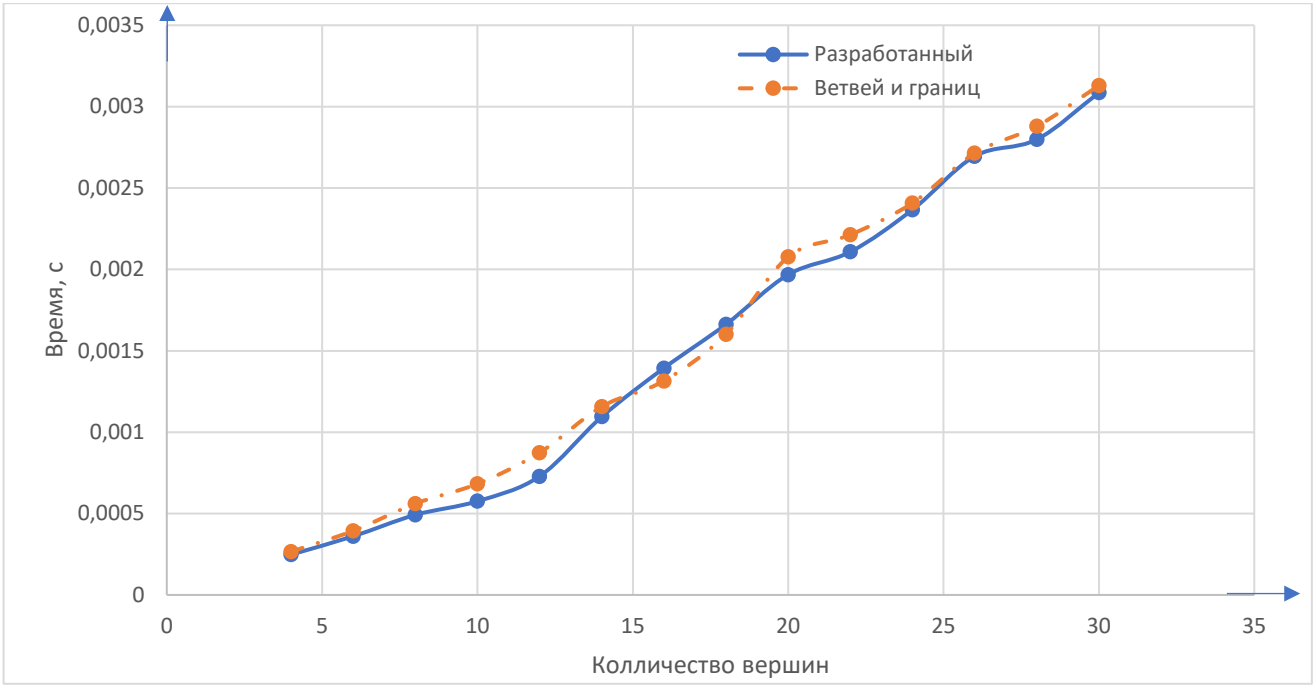

Рис. 10. График зависимости времени работы алгоритмов от количества вершин типа графа звезда с применением построения Халина

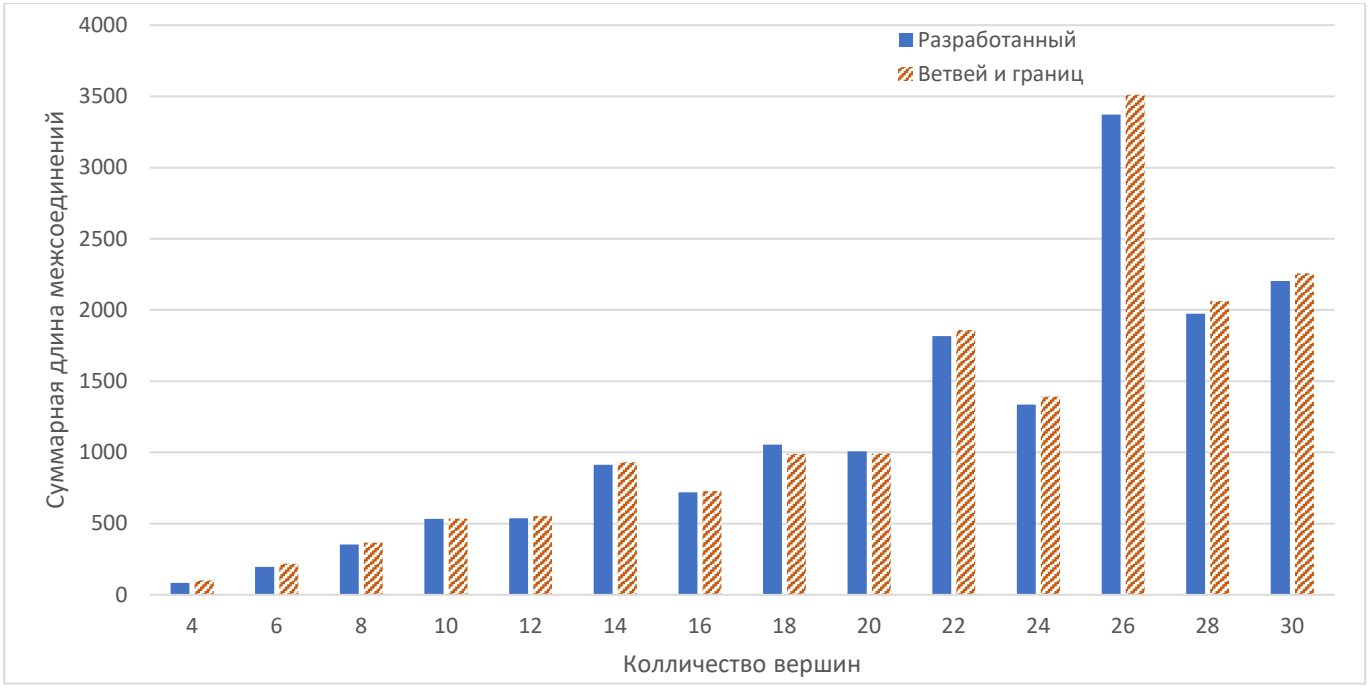

Рис. 11. Значения суммарных соединений на каждом шаге для типа графа звезда с применением построения Халина

Из анализа полученных графиков на рисунке 10 и 11 можно сделать вывод, что разработанный алгоритм эффективнее по времени нахождения конфигурации на 28,18%, а общая длина межсоединений меньше на 6%.

На рисунках 12 и 13 приведены график сравнения работы алгоритмов для двудольного типа графа.

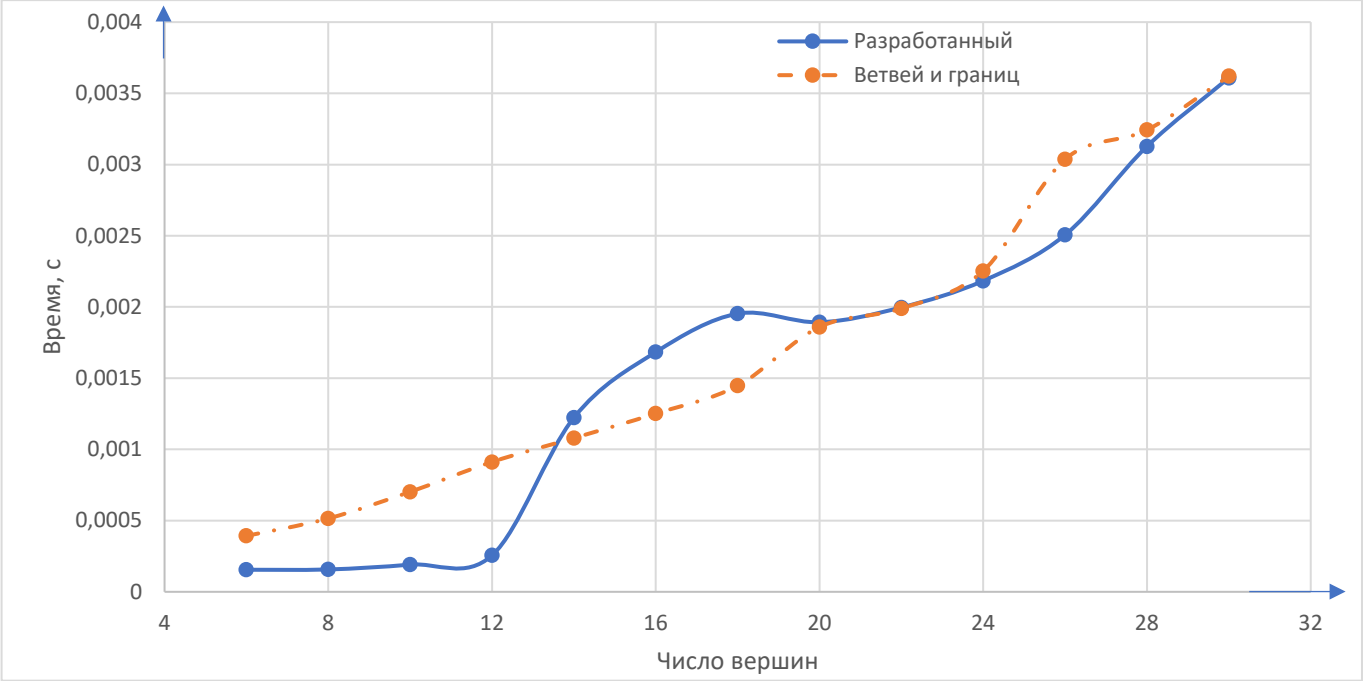

Рис. 12. График зависимости времени работы алгоритмов от количества вершин для двудольного типа графа

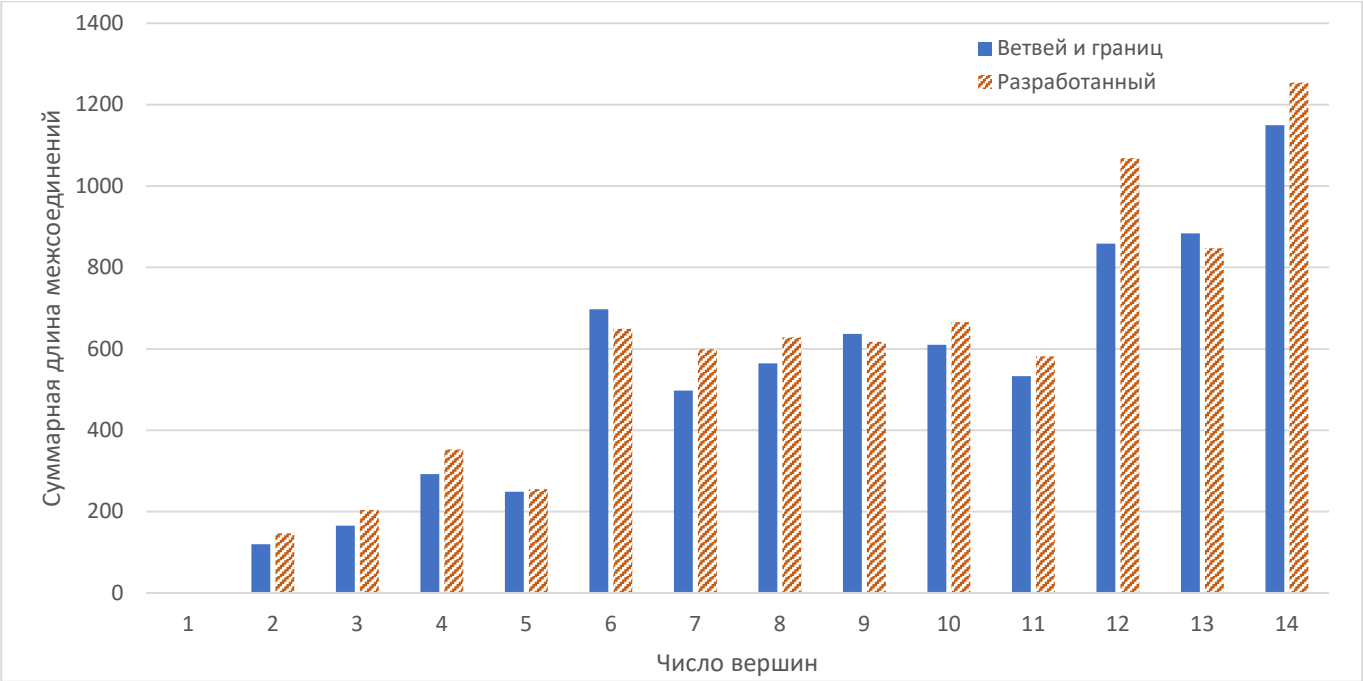

Рис. 13. Значения суммарных соединений на каждом шаге для двудольного типа графа

Из анализа полученных графиков на рисунке 12 и 13 можно сделать вывод, что разработанный алгоритм эффективнее по времени нахождения на 16,49%, а по общей длине межсоединений на 8,7%.

Вывод

Анализ полученных результатов говорит о том, что алгоритм планирования конфигурации ПЛИС работает эффективнее алгоритма, основанного на идеях ветвей и границ: в расчете конфигураций для всех представленных типов графов (в среднем суммарная длинна межсоединений меньше на 8,4%), а в скорости работы в 4 из 6 (в среднем быстрее на 6,6%).

Представленный метод и разработанный на его основе аппаратно-ориентированный алгоритм планирования конфигурации ПЛИС применим в

системах высокой готовности, таких как бортовая авиация, системы слежения, наблюдения, радиолокации, распознавания и т.д. Применение данного подхода позволит снизить как затраты времени на планирование или проектирование (составление) нового плана конфигурации ПЛИС, так и снизить коммутационные задержки системы.

Библиографический список

1. Каляев И.А., Левин И.И., Семерников Е.А., Шмойлов В.И. Реконфигурируемые мультиконвейерные вычислительные структуры. - Ростов на Дону: ЮНЦ РАН, 2008. - 320 с.

2. Воеводин В.В., Воеводин Вл.В. Параллельные вычисления. - СПб: БХВ-Петербург, 2004. – 608 с.

3. Суворова Е.А., Шейнин Ю.Е. Проектирование цифровых систем на VHDL. – СПб.: БХВ–Петербург, 2003. – 576 с.

4. Губанов Д.А., Стешенко В.Б., Храпов В.Ю., Шипулин С.Н. Перспективы реализации алгоритмов цифровой фильтрации на основе ПЛИС фирмы ALTERA // Chip News. 1997. № 9. С. 26 - 33.

5. Стешенко В.Б. Школа схемотехнического проектирования устройств обработки сигналов // Компоненты и технологии. 2000. № 3. С. 43 - 46.

6. Набатов А.Н., Веденяпин И.Э., Мухтаров А.Р. Применение онтологического подхода к процессу проектирования информационной системы // Труды МАИ. 2018. № 102. URL: http://trudymai.ru/published.php?ID=99177

7. Кондрашин М.А., Арсенов О.Ю., Козлов И.В. Применение технологии виртуализации и облачных вычислений при построении сложных распределенных моделирующих систем // Труды МАИ. 2016. № 89. URL http://trudymai.ru/published.php?ID=73411

8. Кудрявцев В.Б., Подколзин, А.С., Болотов А.А. Основы теории однородных структур. - М.: Наука, 1990. – 296 с.

9. Добряков В.А., Енгалычев А.Н., Назаров А.В. Начальное размещение базовых элементов комплементарных металл-окисел-полупроводниковых больших интегральных схем методом случайных назначений // Труды МАИ. 2014. № 72. URL: http://trudymai.ru/published.php?ID=47562

10. Панкратов А.В., Якимов В.Л., Маковский В.Н. Анализ избыточности битовой последовательности для проектов программируемых логических интегральных схем // Труды МАИ. 2015. № 82. URL: http://trudymai.ru/published.php?ID=58828

11. Борзов Д.Б., Басов Р.Г., Титов В.С., Соколова Ю.В. Устройство планирования загрузки процессоров в мультипроцессорных системах критического назначения // Труды МАИ. 2020. № 115. URL: http://trudymai.ru/published.php?ID=119942. DOI: 10.34759/trd-2020-115-14

12. Zhang L., Wong T.N. Solving integrated process planning and scheduling problem with constructive metaheuristics // Information Science, 2016, vol. 340 - 341, pp. 1 - 16. DOI: 10.1016/j.ins.2016.01.001

13. Zhang S., Wong, T.N. Integrated process planning and scheduling: An enhanced ant colony optimization heuristic with parameter tuning // Journal of Intelligent Manufacturing, 2018, vol. 29, pp. 585 – 601. URL: https://doi.org/10.1007/s10845-014-1023-3

14. Бобынцев Д.О., Борзов Д.Б. Минимаксиминный критерий оценки качества размещения параллельных подпрограмм в матричных мультиконтроллерах // Известия Юго-Западного государственного университета. Серия: Управление, вычислительная техника, информатика. Медицинское приборостроение. 2012. № 2-1. С. 27 – 31.

15. Оре О. Теория графов. – М.: Наука, 1968. – 352 с.

16. Масюков И.И. Математическая модель и алгоритм устройства планирования программ в системах на кристалле // Бюллетень науки и практики. 2016. № 5. С. 40 - 44. DOI: 10.5281/zenodo.54825

17. Моисеев Д.В., Чинь В.М., Мозолев Л.А., Моисеева С.Г., Фам С.К. Маршрутизация полета легкого беспилотного летательного аппарата в поле постоянного ветра на основе решения разновидностей задачи коммивояжера // Труды МАИ. 2015. № 79. URL: http://trudymai.ru/published.php?ID=55782

18. Струченков В.И. Дискретная оптимизация. Модели, методы, алгоритмы решения прикладных задач. – М.: Солон-Пресс, 2016. - 192 с.

19. Головицына М.В., Литвинов В.П. Методы, модели и алгоритмы в автоматизированном проектировании промышленных изделий. – М.: Инфра-М, 2016. - 284 с.

20. Сигал И.Х., Иванова А.П. Введение в прикладное дискретное программирование: модели и вычислительные алгоритмы. – М.: ФИЗМАТЛИТ, 2003. 240 с.

Mathematical Model and Hardware-Oriented Algorithm for Programs Placement Planning in Systems on a Chip

Masyukov I.I.[1*], Borzov D.B.[1], Titov D.V.[1***], Sokolova Yu.V.[2****]**

[1]South-Western State University,

94, 50-let Oktyabrya str., Kursk, 305040, Russia

[2]Lavochkin Research and Production Association, NPO Lavochkin,

24, Leningradskay str., Khimki, Moscow region, 141400, Russia

**e-mail: ilmas46ru@gmail.com*

***e-mail: borzovdb@kursknet.ru*

**** e-mail: amazing2004@inbox.ru*

*****e-mail: jv.sokolova@mail.ru*

Abstract

A reconfigurable computing system () is a system that can be reconfigured after its manufacturing. A promising basis for such system constructing is an RCS–FPGA. One of the tasks of building an RCS in real time mode consists in changing the internal FPGA modules organization. However, this leads to the increase in the switching delay time. This time reduction is achieved by the internal modules redistributing, which allows increasing the RCS speed.

The article presents a mathematical model and a hardware-oriented algorithm for program scheduling in systems on a chip, which will allow reducing the time delay in computing the new topology of a reconfigurable computing system and increase its fault tolerance.

To achieve the smallest total length of interconnections, the article proposes to employ two criteria:

1) The programs with a maximum number of related programs must have a minimum distance to their adjacent programs;

2) Adjacent programs with maximum loading among themselves must have a minimum distance;

Thus, to achieve the minimum total interconnection length of the program configuration in the system-on-chip, it is necessary to find such a mapping that would satisfy the above presented conditions 1 and 2.

Based on the proposed criteria and method, a hardware-oriented algorithm was developed, which main advantages consist in the simultaneous selection and location search in the configuration of the selected programs. It reduces the total length of the obtained interconnects in the configuration being computed and the number of enumeration operations.

The software modeling of the developed hardware-oriented algorithm and its results comparison with the algorithm program model, based on the ideas of the branches and boundaries method was performed. Comparison was performed for the following types of graphs: ring, fully connected, planar, Cayley direct product, star using the Halin construction, bipartite. It revealed that the hardware-oriented algorithm works more efficiently for a configuration search for the most of the presented graphs types: in configurations computing for all presented types of graphs (on average, the total length of interconnections is 8.4% less), and in operation speed in four out of six (6.6% faster on average).

The presented method and the hardware-based system-on-chip configuration-planning algorithm developed on its basis are applicable in high-availability systems such as onboard aviation, tracking, surveillance, radar, recognition systems, etc. This approach application will reduce both the time spent on planning or design (compilation) of a new configuration plan, and reduce the switching delays of the system.

Keywords: system-on-a-chip, reconfiguration, architecture, placement, FPGA, configuration, mathematical model, algorithm.

References

1. Kalyaev I.A., Levin I.I., Semernikov E.A., Shmoilov V.I. *Rekonfiguriruemye mul'tikonveiernye vychislitel'nye struktury* (Reconfigurable multiconveyor computing structures), Rostov na Donu, YuNTs RAN, 2008, 320 p.

2. Voevodin V.V., Voevodin Vl.V. *Parallel'nye vychisleniya* (Parallel computing), Sankt Petersburg, BKhV-Petersburg, 2004, 608 p.

3. Suvorova E.A., Sheinin Yu.E. *Proektirovanie tsifrovykh sistem na VHDL* (Digital systems design with VHDL), Sankt Petersburg, BKhV-Petersburg, 2003, 576 p.

4. Gubanov D.A., Steshenko V.B., Khrapov V.Yu., Shipulin S.N. *Chip News*, 1997, no. 9, pp. 26 - 33.

5. Steshenko V.B. *Komponenty i tekhnologii*, 2000, no. 3, pp. 43 - 46.

6. Nabatov A.N., Vedenyapin I.E., Mukhtarov A.R. *Trudy MAI*, 2018, no. 102. URL: http://trudymai.ru/eng/published.php?ID=99177

7. Kondrashin M.A., Arsenov O.Yu., Kozlov I.V. *Trudy MAI*, 2016, no. 89. URL http://trudymai.ru/eng/published.php?ID=73411

8. Kudryavtsev V.B., Podkolzin, A.S., Bolotov A.A. *Osnovy teorii odnorodnykh struktur* (Fundamentals of the homogeneous structures theory), Moscow, Nauka, 1990, 296 p.

9. Dobryakov V.A., Engalychev A.N., Nazarov A.V. *Trudy MAI*, 2014, no. 72. URL: http://trudymai.ru/eng/published.php?ID=47562

10. Pankratov A.V., Yakimov V.L., Makovskii V.N. *Trudy MAI*, 2015, no. 82. URL: http://trudymai.ru/eng/published.php?ID=58828

11. Borzov D.B., Basov R.G., Titov V.S., Sokolova Yu.V. *Trudy MAI*, 2020, no. 115. URL: http://trudymai.ru/eng/published.php?ID=119942. DOI: 10.34759/trd-2020-115-14

12. Zhang L., Wong T.N. Solving integrated process planning and scheduling problem with constructive metaheuristics, *Information Science*, 2016, vol. 340 - 341, pp. 1 - 16. DOI: 10.1016/j.ins.2016.01.001

13. Zhang S., Wong, T.N. Integrated process planning and scheduling: An enhanced ant colony optimization heuristic with parameter tuning, *Journal of Intelligent Manufacturing*, 2018, vol. 29, pp. 585 – 601. URL: https://doi.org/10.1007/s10845-014-1023-3

14. Bobyntsev D.O., Borzov D.B. *Izvestiya Yugo-Zapadnogo gosudarstvennogo un iversiteta. Seriya: Upravlenie, vychislitel'naya tekhnika, informatika. Meditsinskoe priborostroenie*, 2012, no. 2-1, pp. 27 – 31.

15. Ore O. *Teoriya grafov* (Theory of graphs), Moscow, Nauka, 1968, 352 p.

16. Masyukov I.I. *Byulleten' nauki i praktiki*, 2016, no. 5, pp. 40 - 44. DOI: 10.5281/zenodo.54825

17. Moiseev D.V., Chin' V.M., Mozolev L.A., Moiseeva S.G., Fam S.K. *Trudy MAI*, 2015, no. 79. URL: http://trudymai.ru/eng/published.php?ID=55782

18. Struchenkov V.I. Diskretnaya optimizatsiya. *Modeli, metody, algoritmy resheniya prikladnykh zadach* (Discrete optimization. Models, methods, and algorithms for solving applied problems), Moscow, Slon-Press, 2016, p.

19. Golovitsyna M.V., Litvinov V.P. *Metody, modeli i algoritmy v avtomatizirovannom proektirovanii promyshlennykh izdelii* (Methods, models and algorithms in the computer-aided design of industrial products), Moscow, Infra-M, 2016, 284 p.

20. Sigal I.Kh., Ivanova A.P. *Vvedenie v prikladnoe diskretnoe programmirovanie: modeli i vychislitel'nye algoritmy* (Introduction to Applied Discrete Programming: Models and Computational Algorithms), Moscow, FIZMATLIT, 2003, 240 p.