ПРОГРАММНОЕ СРЕДСТВО ДЛЯ ПОИСКА ОДИНАКОВЫХ ПОДГРАФОВ В ГРАФЕ ВЫЧИСЛИТЕЛЬНОГО ПРОЦЕССА

Бурко Л.А.

Белорусский государственный университет информатики и радиоэлектроники
г. Минск, Республика Беларусь

Иванюк А.А. – профессор кафедры информатики

Представлено программное средство для нахождения одинаковых подграфов в ориентированном графе с проверкой значений в вершинах. Предлагаемый алгоритм использует эффективный подход, базируемый на хешировании в процессе сопоставления вершин и ребер между ними, который позволяет быстро находить и сравнивать идентичные подграфы в больших графах. Приведены результаты экспериментов с реальными графическими данными, демонстрирующие эффективность и точность предложенного алгоритма.

Сумматоры и умножители являются основными арифметическими блоками во многих цифровых системах. Их оптимальное использование может существенно повысить эффективность проектирования цифровых схем. Поэтому вопрос переиспользования ресурсов является актуальной задачей.

Классические алгоритмы, такие как алгоритм Ульманна [1], VF2, Subgraph Isomorphism направлены на решение задачи поиска изоморфных подграфов, где входными данными являются два неориентированных графа. Поставлена задача – отыскать в графе вычислительного процесса повторяющиеся комбинации, состоящие из одинаковых блоков. Для ее реализации требуется найти максимальное количество различных подграфов с одинаковыми значениями в вершинах для дальнейшего объединения их в блочные структуры.

Однако, чтобы учитывать значения в вершинах, необходимо изменить алгоритмы проверки графов на изоморфизм. Например, к проверке наличия изоморфизма между вершинами двух графов нужно добавить проверку равенство значений в этих вершинах. Для проверки структуры графов на изоморфизм возьмем существующий алгоритм VF2[2] и модифицируем его.

В общем случае, поиск одинаковых подграфов с учетом значений в вершинах является вычислительно сложной задачей, и время ее выполнения может зависеть от размера графа и количества подграфов, которые необходимо найти [3].

В графе вычислительного процесса существует два вида вершин. Вершины, хранящие операнды – обозначены на рисунке 1 прямоугольниками и знаки операций – обозначены кругами [4]. Для поиска общей структуры необходимы только вершины с знаками операций.

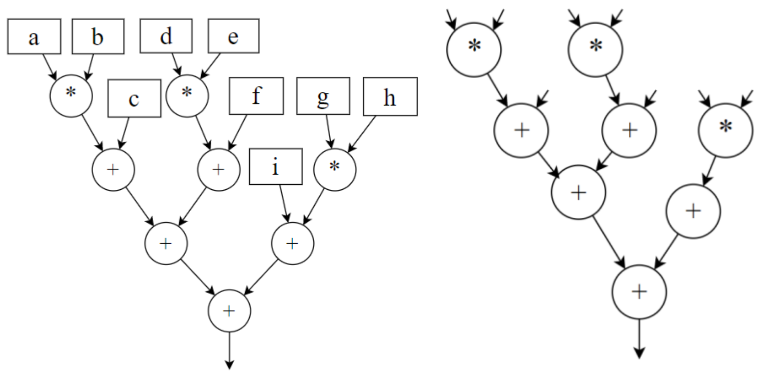

Рисунок 1 – Пример графа вычислительного процесса на 8 вершин (слева) и получившийся, из него граф *G* (справа).

Для описания графа будут использованы обозначения *M = len(V(G))* и *N = len(E(G))* для описания вершин и ребер графа *G*, где *V(G)* – множество вершин заданного граф, *M* – количество вершин, *E(G)* – множество ребер заданного графа, N – количество ребер. Каждой вершине графа присуще свойство (value) “sign”, которое хранит знак операции в данной вершине. Паттерном будет называться такой подграф *G’*, который выполняет условия *G’* ⊆ *G*, *V(G’)* ⊆ *V(G)*, *E(G’)* ⊆ *E(G)*, *values(G) = values(G’)* и встречается в графе *G* более двух раз.

Чтобы найти абсолютно одинаковые подграфы графа *G* являющиеся паттернами, нужно разбить граф на компоненты связности по имеющимся ребрам. Всего существует $2^M - 1$ вариантов графов на вершинах *M* на множествах *V(G)* и *E(G))*. Перебирать все $2^M - 1$ вариантов не оптимально, поэтому нужно выделить только те варианты, которые могут подходить для реализации задачи. Выделим некоторые условия:

1. Отсутствие изолированных вершин и наличие в подграфе более одного ребра.
2. Компонент связности должно быть больше одной.
3. Количество вершин в максимальной компоненте связности должно быть меньше *M/2*.
4. Количество вершин в компонентах связности должно быть одинаковым.
5. Значения в вершинах между компонентами связности должно совпадать.

Графы получившиеся после прошлого шага нужно проверить на изоморфизм. Для каждого графа вычисляется хеш для меток вершин и ребер. Если хеши равны, то графы - изоморфны. При дальнейшем переборе всех подходящих вариантов выбираются те, где количество компонент связности больше.

Для исследования результатов сгенерируем 3 графа с количеством вершин 8, 13 и 22 соответственно. Значения вершин данных графов может быть + или *. Граф с 8 вершинами представлен на рисунке 1, с 13 – на рисунке 2. Изменение количества подходящих графов в зависимости от добавления условий представлено в таблице 1.

Таблица 1 – изменения в количестве графов.

	Всего вариантов	Условие 1	Условие 2	Условие 3	Условие 4	Условие 5
Граф 1	255	120	87	75	27	8
Граф 2	8191	4083	3847	3478	420	151
Граф 3	4194303	2097130	2088129	2034983	20728	6601

Для просмотра результатов рассмотрим более подробно граф 2. При наличии нескольких паттернов объединяются между собой те, которые не будут пересекаться друг с другом, а именно паттерны [*, *] и [*, +, +]. Тем самым вместо использования 6 умножителей можно использовать трижды блок из двух. И конструкцию из двух сумматоров и умножителя так же можно использовать дважды.

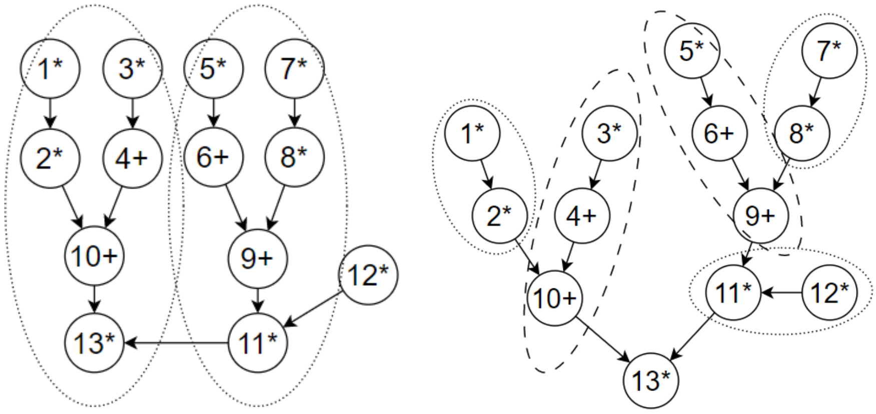

Рисунок 2 – Граф, содержащий 13 вершин, значения вершин ⊆ {+,∗}. Всего предложено 22 варианта замены. Варианты замены – 2 паттерна вида [*, *, *, +, +, *] или 3 паттерна вида [*, *] и 2 паттерна [*, +, +].

В графе на 22 вершины программным средством был найден 91 паттерн. Максимальное количество вершин в паттерне – 9, таким образом можно заменить 2 блока. Максимальное количество замен – 6, в данном случае можно заменить 2 вершины.

Таблица 2 – количество затрачиваемых ресурсов.

	Сумматоры до оптимизации	Сумматоры после оптимизации	Умножители до оптимизации	Умножители после оптимизации
Граф 1	5	3	3	1
Граф 2	4	2	9	4
Граф 3	12	5	10	7

Данное программное средство позволяет выделить множество повторяющихся подграфов в графе вычислительного процесса с целью переиспользования ресурсов, использующихся для реализации на RTL-уровне.

Список использованных источников:

1. Ullmann Julian R. An algorithm for Subgraph Isomorphism // Journal of the Association for Computing Machinery – 1976. – 23. – P. 31-4

2. Харари Ф. Теория графов / Ф. Харари – М.: Ленанд, 2018. – 304 с;
3. Crowe, J. E., Lynch, M. F., Town, W. G. Analysis of Structural Characteristics of Chemical Compounds in a Large Computer-Based File. I: Non-Cyclic Fragments.. J. Chem. Soc 1970, (C), 990-996;
4. H. Bunke, P. Foggia, A comparison of algorithms for maximum common subgraph on randomly connected graphs.