Об использовании параллельных вычислений для решателя NP-полной задачи

А.Д. Плотников

Восточноукраинский национальный университет им. В. Даля, Луганск, Украина

a.plotnikov@list.ru

Аннотация. *В статье обсуждается возможность использования параллельных или распределенных вычислений при реализации эвристического алгоритма решения задачи поиска наибольшего независимого множества вершин графа. Теоретическая оценка сложности алгоритма равна $O(n^8)$, где n есть число вершин графа. Анализируются основные алгоритмы решателя и устанавливается, что они хорошо поддаются распараллеливанию. В случае использования методов параллельных и распределительных вычислений, теоретическая временная оценка сложности решателя может быть уменьшена по крайней мере на два порядка.*

Ключевые слова

Параллельные вычисления, решатель, NP-полная задача, граф, орграф, независимое множество.

1 Постановка задачи

В работах [1, 2] был предложен экспериментальный алгоритм для поиска наибольшего независимого множества вершин в произвольном неориентированном графе $G = (V, E)$ без петель и кратных ребер, которая является NP-полной. Время работы такого алгоритма равно $O(n^8)$, где n – число вершин графа.

В соответствии с разработанным алгоритмом автором была написана программа на Паскале и на ее основе Томас Карбе из Берлинского технического университета написал программу решателя на языке Ява. Решатель расположен по следующим адресам [3, 4].

Естественно, что для практического использования эта оценка сложности алгоритма является высокой. Один из путей уменьшения сложности решателя, реализующего предложенный алгоритм, является применение параллельных и распределенных вычислений.

Цель данной работы — обсудить возможность распараллеливания основных составляющих частей алгоритма предложенного решателя.

2 Теоретические основы решателя

Кратко напомним основные этапы работы эвристического алгоритма.

Идея алгоритма основана на использовании частично упорядоченного множества (ч.у.м.), в котором согласно методологии Форда-Фалкерсона [5] находится минимальное цепное разбиение (МЦР) данного ч.у.м. и его наибольшая антицепь. Поиск такого разбиения производится как нахождение максимального паросочетания в двудольном графе и требует $O(n^{5/2})$ единиц времени.

Чтобы реализовать такую идею, по данному неориентированному графу $G = (V, E)$ мы строим ациклический ориентированный граф. Ясно, что граф транзитивного замыкания (ГТЗ) для построенного орграфа определяет некоторое ч.у.м., в котором и находится МЦР и наибольшая антицепь.

В ориентированном графе мы определяем множество всех вершин, имеющих только исходящие дуги и нет входящих. Множество всех таких вершин образует инициирующее множество $V^0 \subset V$. Орграф, имеющий инициирующее множество V^0, обозначаем: $\vec{G}(V^0)$.

Максимальная длина $\rho(v)$ ориентированной цепи, соединяющей вершину $v \in V$ и инициирующее множество V^0, называется *рангом* v. Ясно, что если $v \in V^0$, то $\rho(v) = 0$. Множество всех вершин, имеющих один и тот же ранг $\rho(v) = k$ образует k-й слой V^k орграфа $\vec{G}(V^0)$. Таким образом, имеем $V = V^0 \cup V^1 \cup \cdots \cup V^m$, где m – слой наиболее старшего ранга.

Слой V^K порождает ориентированный подграф $\vec{G}(V^k)$, если в орграфе $\vec{G}(V^0)$ удалить все слои V^S $(s < k)$ вместе с инцидентными им дугами. Ясно, что в таком подграфе слой V^k является инициирующим. Граф транзитивного замыкания $\vec{G}_t(V^k)$, очевидно, есть ч.у.м., в котором можно найти наибольшую антицепь $U(v)$, содержащие каждую из вершин $v \in V^k \cup \cdots \cup V^m$, когда такая антицепь существует.

Ориентированный подграф $\vec{G}(V^k)$ назовем насыщенным относительно своего инициирующего множества V^k, если этот слой является максимальным независимым множеством (МНМ) ориентированного подграфа $\vec{G}(V^k)$ и все наибольшие антицепи $U(v)$, когда они существуют, также являются МНМ подграфа $\vec{G}(V^k)$. Орграф $\vec{G}(V^0)$ назовем вершинно-насыщенным (ВН-орграфом), если в нем каждый подграф $\vec{G}(V^k)$ является насыщенным относительно своего инициирующего множества V^k.

Дугу $(v_i, v_j) \in V$ графа транзитивного замыкания $\vec{G}_t(V^0)$ назовем фиктивной, если она отсутствует в орграфе $\vec{G}(V^0)$ и обозначаем $v_i \gg v_j$. Если такая дуга имеется в орграфе $\vec{G}(V^0)$, то ее назовем существенной и обозначим $v_i > v_j$.

В работе [1] доказано, что если МЦР графа $\vec{G}_t(V^0)$ не имеет фиктивных дуг, то инициирующее множество V^0 является наибольшим независимым множеством (НБНМ) графа $G = (V, E)$. В общем случае, однако, МЦР ГТЗ $\vec{G}_t(V^0)$ может содержать фиктивные дуги. В частности, это может быть свидетельством того, что текущее инициирующее множество V^0 не является НБНМ графа $G = (V, E)$. Вершину v_j фиктивной дуги $v_i \gg v_j$, принадлежащей найденному МЦР графа $\vec{G}_t(V^0)$ назовем отмеченной.

Пусть $v_i \gg v_j$ есть некоторая фиктивная дуга графа транзитивного замыкания $\vec{G}_t(V^0)$ ВН-орграфа $\vec{G}(V^0)$. Удалим из орграфа $\vec{G}(V^0)$ вершины v_i и v_j. Из оставшегося орграфа построем некоторый ВН-орграф $\vec{G}(Z^0)$. Полученный орграф назовем ВН-орграфом, порожденным удалением дуги $v_i \gg v_j$ из орграфа $\vec{G}(Z^0)$.

Гипотеза, предложенная в работе [1] утверждает, что если в ВН-орграфе $\vec{G}(V^0)$ инициирующее множество V^0 не является НБНМ графа $G = (V, E)$, то в соответствующем ГТЗ $\vec{G}_t(V^0)$ найдется такая фиктивная дуга $v_i \gg v_j$, удаление которой из орграфа $\vec{G}(V^0)$ порождает орграф $\vec{G}(Z^0)$, в котором инициирующее множество Z^0 удовлетворяет соотношению: $Card(Z^0) \geq Card(V^0) - 1$.

Предложенная гипотеза позволяет построит полиномиально-временной решающий алгоритм для указанной выше NP-полной задаче.

3 Основные алгоритмы

Рассмотрим основные алгоритмы, реализуемые решателем.

Пусть имеется некоторый орграф $\vec{G}(V^0)$. Чтобы построить ВН-орграф, мы используем следующий алгоритм.

Вход алгоритма — некоторый орграф $\vec{G}(V^0)$. Выход алгоритма: ВН-орграф $\vec{G}(Y^0)$ и граф транзитивного замыкания $\vec{G}_t(Y^0)$.

Алгоритм ВН-орграф

Шаг 1. Положить $k := 0$ и $\alpha :=$ **false**.

Шаг 2. Найти граф транзитивного замыкания $\vec{G}_t(V^k)$.

Шаг 3. Построить МЦР графа $\vec{G}_t(V^k)$.

Шаг 4. Найти максимальную антицепь графа $\vec{G}_t(V^k)$ для каждой вершины $v \in V^k \cup \cdots \cup V^m$.

Шаг 5. Проверить, является ли каждая из найденных наибольших антицепей $\mathcal{U}(v)$ графа $\vec{G}_t(V^k)$ МНМ орграфа $\vec{G}(V^k)$ и $Card(\mathcal{U}(v)) = Card(V^k)$. Если это так, закончить построение орграфа $\vec{G}(V^k)$, насыщенного относительно инициирующего множества V^k. Перейти к Шагу 6.

В противном случае дополнить найденную антицепь $\mathcal{U}(v)$ (при необходимости) до МНМ W и построить новый ациклический орграф $\vec{G}(V^k)$ операцией сечения $\sigma_W(\vec{G}(V^0))$, положить $\alpha := \mathbf{true}$ и построить новый орграф $\vec{G}(V^0)$. Вернуться к Шагу 2.

Шаг 6. Вычислить $k := k + 1$. Если $k < m$, то выделить граф транзитивного замыкания $\vec{G}_t(X^k)$ из $\vec{G}_t(X^{k-1})$, сохраняя все цепи МЦР графа $\vec{G}_t(X^{k-1})$, которые инцидентны вершинам нового графа. Перейти к Шагу 4.

Если $k = m$ и $\alpha = \mathbf{true}$, то перейти к Шагу 1. Если $k = m$ и $\alpha = \mathbf{false}$, то перейти к Шагу 7.

Шаг 7. Закончить вычисления. ВН-орграф построен.

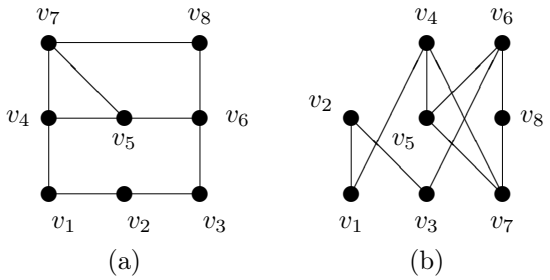

Рис. 1. Простой пример ВН-орграфа

Нетрудно видеть, что алгоритм построения ВН-орграфа содержит несколько этапов, которые могут быть выполнены одновременно (параллельно). Прежде всего это поиск наибольшей антицепи для каждой вершины $v \in V^k \cup \cdots \cup V^m$. Очевидно, что такой поиск может выполняться независимо для каждой вершины. Так как ранее однократное выполнение Шагов 1 – 6 требует $O(n_k^3)$ единиц времени, где n_k есть размер инициирующего множества в $\vec{G}(V^k)$ [1], то это время понизится на порядок и будет составлять $O(n_k^2)$. В результате параллельного нахождения антицепей время построения ВН-орграфа уменьшится на порядок, т.е. будет справедливо следующее утверждение.

Теорема 1 *ВН-орграф может быть построен за время $O(n^4)$ при условии параллельного поиска наибольших антицепей.*

Далее, в принципе, построение подграфа $\vec{G}_t(V^k)$ насыщенного относительно своего инициирующего множества V^k может происходить параллельно для каждого k ($k = 0, 1, \ldots, m-1$). Конечно, это потребует от управляющей программы дополнительной пересылки информации в случае изменения исследуемого орграфа $\vec{G}(V^0)$.

Основной этап в работе решателя состоит в выполнении следующего алгоритма.

Входом алгоритма является неориентированный граф $G \in L_n$, а выход — наибольшее независимое множество $\hat{U}$ этого графа.

Алгоритм нахождения НБНМ.

Шаг 1. Выполнить первоначальную ориентацию ребер графа G так, чтобы получить ациклический орграф $G(V^0)$.

Шаг 2. Для орграфа $G(V^0)$ выполнить алгоритм ВН-орграф.

Шаг 3. В ГТЗ ВН-орграфа найти непомеченную фиктивную дугу $v_i \gg v_j$. Пометить найденную фиктивную дугу как рассмотренную. Если все фиктивные дуги помечены, то перейти к Шагу 7.

Шаг 4. Из ВН-орграфа $G(V^0)$ удалить вершины v_i, v_j, а также все смежные с ними вершины. В результате будет получен орграф $\vec{G}_1(V_1^0)$.

Шаг 5. Для орграфа $G_1(V_1^0)$ выполнить алгоритм ВН-орграф. В результате будет получен орграф $\vec{G}(Z^0)$.

Шаг 6. Если $Card(Z^0) \geq Card(V^0) - 1$, то построить множество $W = Z^0 \cup \{v_i, v_j\}$ и в насыщенном орграфе $G(V^0)$ выполнить операцию сечения $\sigma_W(\vec{G}(V^0))$. Перейти к Шагу 2. В противном случае вернуться к Шагу 3.

Шаг 7. Положить НБНМ $\hat{U} = V^0$.

В этом алгоритме необходимость параллельной обработки орграфов, полученных в результате удаления фиктивных дуг очевидна (Шаги 3–6). По-видимому, в этом случае целесообразно закрепить за отдельным процессором (компьютером в сети) одну или ряд определенных вершин орграфа, которые начинают свою работу в случае,

если эта вершина является концевой для фиктивной дуги. Было бы идеальной возможность при работе алгоритма нахождения НБНМ закрепить за каждой фиктивной дугой отдельный процессор или компьютер, если это возможно. В последнем случае общее время работы алгоритма можно было бы уменьшить по крайней мере на порядок. Таким образом, имеем следующее утверждение.

Теорема 2 *Время работы алгоритма нахождения НБНМ равно* $O(n^6)$*, если возможна параллельная обработка всех фиктивных дуг графа* $G_t(V^0)$.

Доказательство. В самом деле, однократное выполнение Шагов 3 – 6 требует $O(n_1^4)$ единиц времени, где n_1 есть число вершин в орграфе $\vec{G}(Z^0)$, порожденном удалением фиктивной дуги. В худшем случае, для выполнения Шагов 3 – 6 требуется $O(n^5)$ единиц времени. Если предположить, что после выполнения этих шагов, найденное независимое множество V^0 будет увеличино на единицу, то общее время выполнения шагов 2 – 6 равно $O(n^6)$. Q.E.D.

4 Заключение

В работе “Сложность комбинаторных алгоритмов” (сборник "Методы дискретного анализа в решении комбинаторных задач М.: Мир, том 17, 1980) Тарьян представил таблицу, которая показывает динамику улучшения оценки сложности для решения некоторых задач. Эта динамика улучшения не предполагала использования параллельных или распределительных вычислений. Откуда можно сделать вывод, что после появления первоначального алгоритма решения некоторой задачи, быстро находятся его улучшения за счет более глубокого изучения свойств решаемой задачи. Представляется, что дополнительное использование методов распараллеливания алгоритма позволит построить более эффективный решатель.

Список литературы

[1] *Плотников, А. Д.* Эвристический алгоритм для поиска наибольшего независимого множества / А. Д. Плотников // *Кибернетика и системный анализ*. — 2012. — № 5. — С. 41–48.

[2] *Plotnikov, A. D.* Problems of the class NP: Research and simulating / A. D. Plotnikov. — Saarbrucken: LAP Lambert Academic Publishing, 2011. — 145 pp.

[3] http://www.is.svitonline.com/plot/solver.html

[4] http://www.vinnica.ua/ aplot/solver.html

[5] *Форд, Л. Р.* Потоки в сетях: пер. с англ. / Л. Р. Форд, Д. Р. Фалкерсон. — М.: Мир, 1966. — 276 с.