

УДК 519.854

С.В. Листровой, С.В. Мощный

Украинский государственный университет железнодорожного транспорта, Харьков

ОПТИМИЗИРОВАННЫЙ МЕТОД РЕШЕНИЯ ЗАДАЧИ О НАИМЕНЬШЕМ ПОКРЫТИИ НА ОСНОВЕ НЕГАРАНТИРОВАННОГО ПРОГНОЗИРОВАНИЯ

В статье представлен оптимизированный метод решения задачи о наименьшем покрытии для произвольных графов, основанный на составлении и анализе пессимистического негарантированного прогнозирования наихудшего случая формирования выборки вершин, которые можно включить в покрытие. Рассмотрена эффективность работы данного алгоритма при использовании различных моделей построения графов. Проанализирована временная сложность, погрешность, а также рациональность использования данного метода в средах распараллеливания нагрузки и телекоммуникационных системах.

Ключевые слова: вершинное покрытие, временная сложность, булевы функции.

Введение

Задача о нахождении наименьшего покрытия имеет широкий спектр применений: в системах планирования, средах распараллеливания нагрузки, телекоммуникационных сетях и т.д. К примеру, при эксплуатации современных грид-систем [1] необходима более тонкая процедура планирования ресурсов, которая бы эффективно распределяла задания на свободные узлы кластерного пула с учетом возможных простоев и задержек. Это может быть реализовано путем использования эвристического алгоритма решения задачи о наименьшем покрытии, обеспечивающего оптимальную периодичность освоения пула заданий и перераспределение соответствующих ресурсов. Таким образом, время работы и эффективность планировщика напрямую зависят от временной сложности выбранного алгоритма.

До сих пор неизвестно ни одного полиномиального и точного алгоритма для решения данной задачи. На сегодняшний день существует два основных подхода: использование либо точного алгоритма экспоненциальной сложности при малых наборах входных значений, либо приближенного алгоритма, который бы эффективно находил решение за полиномиальное время. Поскольку первый вариант имеет слишком ограниченное применение, на практике в основном используются приближенные алгоритмы, которые, однако, не всегда могут обеспечить баланс между затребованными вычислительными ресурсами и достаточной оптимальностью решения, в связи с чем возникает необходимость в разработке нового оптимизированного алгоритма, который бы позволил удовлетворить перечисленным критериям на более широком диапазоне возможных экземпляров задачи.

Постановка задачи на исследование. Теория сложности вычислений, описывающая зависимость

объема необходимой работы некоторого алгоритма от количества входных переменных в той или иной абстрактной системе, занимает сегодня ведущее положение при построении и анализе эффективных методов решения задач постоянно возрастающей размерности и усложняющейся структуры [2]. Существуют различные подходы к решению данного вида задач с использованием специфических алгоритмов для каждой конкретной проблемы. В общем случае, под алгоритмом понимают четкую последовательность действий, которая приводит через конечное число шагов к определенному результату. В качестве примера решаемых задач могут выступать: дифференциальные уравнения, криптоанализ, задачи на графах, оптимизация систем диагностики и т.д.

Объем работы алгоритма охватывает вычислительные ресурсы исполняющей системы: процессорное время, оперативная память, пропускная способность сети и др. Ограниченность ресурсов в каждой подобной системе налагает жесткие требования к эффективности работы используемого алгоритма.

Для оценки порядка роста времени выполнения алгоритма вводится понятие асимптотической сложности. Чем меньшую асимптотическую сложность имеет алгоритм, тем более эффективные показатели он будет иметь при увеличении входных данных [3]. Для одной и той же задачи могут существовать алгоритмы с различной асимптотической сложностью, а также емкостной сложностью, характеризующей объем памяти, необходимый для работы алгоритма.

Асимптотическая оценка сложности обозначается греческой буквой Θ (тета). Асимптотическое равенство (1) включает в себя одновременно верхнюю и нижнюю оценки сложности.

$$f(n) = \Theta(g(n)). \quad (1)$$

Данное равенство выполняется только при соблюдении ограничения (2).

$$\exists (c, c_0) > 0, n_0:$$

$$\forall (n > n_0) |c * g(n)| < |f(n)| < |c_0 * g(n)|. \quad (2)$$

Обозначение Θ символизирует множество функций, которые растут с той же скоростью, что и функция $g(n)$, т.е. с точностью до умножения на константу (c, c_0) , n – количество входных переменных.

На практике чаще всего пользуются верхней оценкой сложности (3), поскольку она характеризует временную составляющую реализации алгоритма. Другими словами, она показывает оценку того времени, за которое алгоритм гарантированно завершит работу, а не время, в пределах которого он точно не закончит выполнение инструкций.

Верхняя оценка сложности обозначается греческой буквой O (омикрон).

$$f(n) = O(g(n)). \quad (3)$$

Данная оценка является множеством функций, которые растут не быстрее, чем $g(n)$. При этом необходимо соблюдение ограничения (4).

$$\exists c > 0, n_0: \forall (n > n_0) |f(n)| \leq |c * g(n)|. \quad (4)$$

Были сделаны различные попытки классификации решаемых задач по тем или иным критериям (проблематика разрешимости, задачи оптимизации, анализа и т.д.). В случае алгоритмов, в зависимости от того, можно ли проверить решение на машине Тьюринга, выделяют несколько классов сложности: класс NP , который охватывает все переборные задачи; класс P , охватывающий те переборные задачи, которые можно решить за полиномиальное время [4]. Вопрос о том, можно ли свести вышеуказанные классы в один, до сих пор является открытым.

В переборных задачах необходимо найти решение среди конечного множества вариантов. Однако, при резком увеличении числа входных данных, задача становится чересчур трудоемкой, а иногда и вовсе неразрешимой. Поэтому для решения подобных задач необходима разработка полиномиального алгоритма, т.е. такого алгоритма, временная сложность которого равна $O(p(n))$, где $p(n)$ – полиномиальная функция, а n – входные параметры.

В классе NP выделяют универсальные проблемы, которые также называют NP -полными задачи. Считается, что любая задача из класса NP может быть полиномиально сведена к NP -полной задаче. Очевидно, что если удастся доказать принадлежность любой NP -полной задачи к классу P , также будет доказано равенство между классами P и NP , а значит, появится возможность разработки эффективных алгоритмов для широкого класса дискретных задач, не имеющих до сих пор точных методов решений за полиномиальное время.

Задача о нахождении наименьшего покрытия в графах может быть представлена в двух формах: оптимизационной и форме разрешимости. В случае оптимизационной формы предполагается нахождение наилучшего решения среди всех возможных. Форма

разрешимости может быть представлена в виде вопроса, требующего однозначного ответа «да или нет», в зависимости от значений входных параметров [5]. В такой форме задачу о нахождении наименьшего покрытия принято относить к классу NP -полных задач.

На основании вышеизложенного материала формализуем поставленную задачу. Пусть имеется произвольный ненаправленный граф $G(V, E)$, где V – множество вершин, E – множество ребер данного графа. Вершинным покрытием данного графа является такое подмножество вершин $V' \subseteq V$, что каждое ребро $(u, v) \in E$ удовлетворяет следующему условию: $u \in V'$ и/или $v \in V'$, соответственно. Под размером вершинного покрытия подразумевается количество вершин, входящих в него. Таким образом, задача сводится к нахождению вершинного покрытия наименьшего размера в заданном неориентированном графе G .

Анализ последних исследований и публикаций. Задача о нахождении наименьшего покрытия входит в известный список Карпа, состоящий из формулировки и доказательства NP -полноты 21 задачи. Однако данная задача привлекает многих исследователей не только принадлежностью к классу NP -полных проблем, но и тем, что к ней можно свести множество трудоемких задач из реальной жизни [6]. Сферы использования данной задачи охватывают электротехнику, биоинформатику, коммуникационные системы и другие отрасли.

На сегодняшний день известно несколько приближенных алгоритмов решения данной задачи за полиномиальное время. Все они отличаются как погрешностью получаемых решений, так и эффективностью работы на различных плотностях исследуемых графов. Алгоритмы, которые обеспечивают точные решения на всем диапазоне входных данных, нецелесообразно использовать на практике ввиду их слишком высокой временной сложности.

Задача имеет коэффициент аппроксимации (5) равным $\rho(n)$, если для любого количества входных переменных n затраты c на решение алгоритма находятся в пределах, пропорциональных затратам c^* оптимального варианта.

$$\text{Max}(c/c^*, c^*/c) \leq \rho(n). \quad (5)$$

Популярным методом нахождения приближенных решений является жадный подход [7]. Данный метод основан на нахождении наилучших результатов на каждой стадии работы алгоритма с последующим сведением их в глобальное общее оптимальное решение. В случае задачи о наименьшем покрытии, на каждой стадии будет произведен поиск вершин с наибольшей степенью с последующим удалением всех покрытых ребер. Временная сложность данного метода равна $O(V+E)$ (V – вершины, E – ребра).

При решении задач оптимизации и моделирования часто пользуются генетическим алгоритмом [8].

Он представляет собой эвристический метод поиска с применением механизмов: аналогичных естественно-му отбору в природе. Первые попытки симуляции эволюции были проведены Нильсом Баричелли в 1954 году. Формализация задачи делается таким образом, чтобы её решение могло быть закодировано в виде вектора генов, где каждый ген может являться битом или числом. Затем случайным образом создаётся множество генотипов начальной популяции. Они оцениваются с использованием «функции приспособленности», в результате чего с каждым генотипом ассоциируется определённое значение. Временная сложность равна общему количеству кандидатов-решений из полученного «множества-поколения».

Еще одним популярным подходом является использование техники линейного программирования [9]. Линейная программа представляет собой совокупность линейных ограничений на диапазоне значений целевой функции. При наличии n переменных и m ограничений задача может быть решена за полиномиальное время $O(n \cdot m)$. В 1979 году Л. Хачияном был предложен первый полиномиальный алгоритм на основе линейного программирования.

Экспериментальные результаты и их обсуждение

Целью статьи является разработка оптимизированного алгоритма с гораздо меньшей погрешностью получаемых решений и улучшенной временной сложностью по сравнению с существующими подходами. Рассмотрим предлагаемую процедуру решения задачи о нахождении наименьшего покрытия на основе негарантированного прогнозирования для произвольных графов:

Шаг 1. Находим пары независимых вершин в заданном графе и составляем их суммированные характеристики.

Шаг 2. Среди найденных пар независимых вершин выбираем ту пару, которая имеет наибольшую общую степень (покрывает наибольшее количество ребер в соответствующем графе). Если таких пар несколько, выбираем ту, в состав которой входят вершины, встречающиеся в совокупности большее количество раз среди всех остальных независимых вершин. В случае если и таких пар окажется несколько – выбираем любую из них.

Шаг 3. Пусть x_1 и x_2 – выбранные независимые вершины. Используя технику пессимистического негарантированного прогнозирования, составляем три возможных варианта для последующего анализа:

- $x_1 = 0, x_2 = 0$, т. е. вершины x_1 и x_2 принадлежат минимальному вершинному покрытию;
- $x_1 = 0, x_2 = 1$, т. е. вершина x_1 принадлежит минимальному вершинному покрытию, а вершина x_2 нет;
- $x_1 = 1, x_2 = 0$, т. е. вершина x_2 принадлежит минимальному вершинному покрытию, а вершина x_1 нет.

Шаг 4. Для каждого варианта, описанного на предыдущем шаге, решаем соответствующую систему уравнений, подставляя значения выбранных независимых вершин рассматриваемого графа. Используя аксиомы булевой алгебры (метод поглощений и др.) определяем следующие характеристики: W – реальный размер покрытия, полученного в каждом варианте; P – прогнозируемое включение вершин в минимальное покрытие.

Шаг 5. Сравнивая значения, полученные на шаге 4, в первую очередь выбираем вариант с $P = 0$. Если такого нет, выбираем тот, у которого больше параметр W . В случае если данные параметры одинаковы – выбираем любой из них. Возвращаемся на шаг 1 и решаем данную процедуру до тех пор, пока не встретится вариант с $P = 0$.

При решении вышеуказанной процедуры необходимо также учитывать два специфических для данного алгоритма момента:

1. Если в графе присутствует висющаяся вершина, то переменную, находящуюся с ней в паре, включаем в покрытие, а данную вершину приравниваем к нулю.

2. Если количество появления переменных в графе одинаково, то добавляем в прогноз возможного покрытия такое количество вершин: $(n+1)/2$ – когда суммарное число переменных нечетное, $n/2$ – при варианте с четным количеством.

Благодаря использованию техники негарантированного прогнозирования временная сложность не разрастается до экспоненциального уровня и остается в пределах $O(\log(V \cdot E))$.

Для детального тестирования и анализа работы алгоритма была написана программа на языке программирования C++.

С помощью данной программы был составлен график, представленный на рис. 1. Как видно на графике, данный алгоритм имеет низкий уровень погрешности, при этом он позволяет найти решение за полиномиальное время.

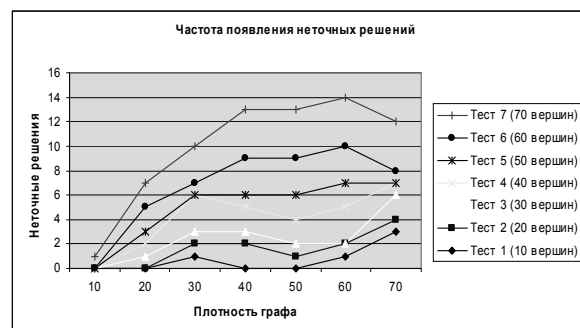


Рис. 1. График, отображающий эффективность работы предлагаемого алгоритма

На рис. 2 представлен график частоты появления неточных решений для алгоритма на основе частотного подхода, который позволяет провести

сравнительный анализ работы описываемой процедуры решения задачи о наименьшем покрытии. Как и на графике 1, по оси x отображается изменение плотности заданного графа. Ось y показывает, как быстро увеличивается частота появления неточных решений при возрастании количества вершин.

Для каждой точки графика, с помощью специально написанной программы проводилась серия тестов, позволяющая оценить точность полученного решения путем сравнения с результатом точного (экспоненциального) алгоритма.

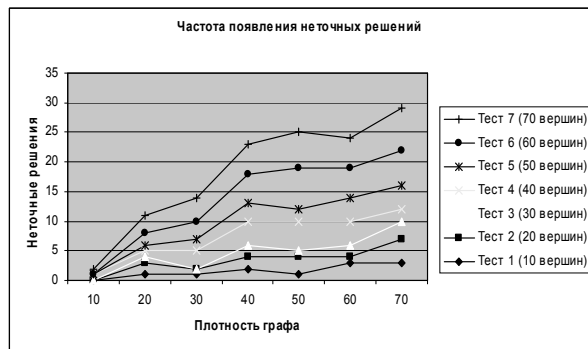


Рис. 2. График частоты появления неточных решений алгоритма на основе частотного подхода

Сопоставление числовых данных, полученных экспериментальным путем, позволяет составить суммированные характеристики разработанного алгоритма. Проведенные тесты говорят о том, что описанный алгоритм имеет хорошую точность решений по сравнению с другими полиномиальными приближенными алгоритмами и может быть использован как замена для менее эффективных процедур решения NP-трудных задач. В частности, данный алгоритм может быть использован в таких отраслях, как: диагностика и оптимизация работы телекоммуникационных сетей; рационализация установки систем обнаружения и предотвращения вторжений и др. Эффективность выполнения по сравнению с наиболее оптимальными существующими подходами в зависимости от выбранной модели построения графа может быть повышена на 30 – 35%.

ОПТИМІЗОВАНИЙ МЕТОД РІШЕННЯ ЗАДАЧІ ПРО НАЙМЕНШЕ ПОКРИТТЯ НА ОСНОВІ НЕГАРАНТОВАНОГО ПРОГНОЗУВАННЯ

С.В. Лістровий, С.В. Мотсний

У статті представлено оптимізований метод розв'язання задачі про найменше покриття для довільних графів, заснований на складанні та аналізі песимістичного негарантованого прогнозування найгіршого випадку формування вибірки вершин, які можна включити в покриття. Розглянуто ефективність роботи даного алгоритму при використанні різних моделей побудови графів. Проаналізована тимчасова складність, похибка, а також раціональність використання даного методу в середовищах розпаралелювання навантаження і телекомунікаційних системах.

Ключові слова: вершинне покриття, тимчасова складність, булеві функції.

THE OPTIMIZED METHOD FOR SOLVING THE MINIMUM VERTEX COVER PROBLEMS BASED ON A NON-GUARANTEED PREDICTION

S.V. Listrovoi, S.V. Motsnyi

In this article an optimized method for solving problems of finding the minimum vertex covers for the arbitrary graphs based on the analysis of an unwarranted pessimistic prediction of the worst-case sample of the vertices that can be included into the cover is discussed. The effectiveness of this algorithm is shown by testing the different graph models. The time complexity, measurement errors and the possible use of this technique in the distributed environments and the telecommunication systems have also been summarized.

Keywords: vertex coverage, temporal complication, functions are boole.

Заклучение

Разработанный алгоритм позволяет эффективно решать задачу о нахождении наименьшего покрытия, которая имеет широкий спектр использования в различных областях науки и техники. Свойства алгоритма были протестированы при помощи специально написанной программы.

Описанный в статье подход применим к различным моделям графов, что подтвердилось проведенным тщательным анализом.

Список литературы

1. Минухин С.В. Исследование эффективности методов планирования ресурсов для различных интенсивностей потоков заданий в Грид / С.В. Минухин, С.В. Знахур // *Радіоелектронні і комп'ютерні системи*. – 2012. – № 1 (53). – С. 165-171.
2. Goldreich O. *Introduction to Complexity Theory. – Lecture Notes*. [Електронний ресурс] / O. Goldreich. – Режим доступа: <http://www.wisdom.weizmann.ac.il/~oded/cc.html>.
3. Кормен Т. Алгоритмы. Построение и анализ / Т. Кормен, Ч. Лейзерсон, Р. Ривест. – М.: МЦНМО, 2001.
4. Michael Sipser. *Introduction to the Theory of Computation – the Class NP, NP-completeness, Additional NP-complete Problems*, 1997.
5. Dinur I. On the hardness of approximating minimum vertex cover / I. Dinur, S. Safra // *Annals of Mathematics*. – 2005. – 162(1). – P. 439-486.
6. Karp Richard M. *Reducibility Among Combinatorial Problems* / Richard M. Karp, 1972.
7. Garey M.R. *Computers and intractability: A guide to the Theory of NP-Completeness* / M.R. Garey, D.S. Johnson, 1978.
8. Dorit S. Hochbaum. *Approximation Algorithms for NP-hard problems* / Dorit S. Hochbaum, 2002.
9. Bar-Yehuda R. A linear time approximation algorithm for the weighted vertex cover problem / R. Bar-Yehuda, S. Even. // *IEEE Telecom*. – 1981. – Vol 2. – P. 198-203.

Поступила в редколлегию 10.12.2014

Рецензент: д-р техн. наук, проф. С.И. Приходько, Украинский государственный университет железнодорожного транспорта, Харьков.