УДК 519.174.7, 004.75

Турсунбай к. Ы.

КГНУ

АЛГОРИТМЫ РАСКРАСКИ ГРАФОВ В РАСПРЕДЕЛЕННОЙ МОДЕЛИ ВЫЧИСЛЕНИЙ

Одним из способов улучшения выполнения распределенного алгоритма является представление стратегии раскраски в алгоритм, который, как известно, является эффективным в нераспределенных алгоритмах. В статье показано, что применение некоторых эвристик последовательного алгоритма раскраски, таких как наибольшие-первые (НП), наименьшие-последние (ПН) и наибольшие-первые насыщенности (НПН), для некоторых классов графов и для частных случаев вершинной раскраски в распределенных алгоритмах дают нам оптимальную или близкую к оптимальной раскраску.

Ключевые слова: раскраска графов, распределенный алгоритм, жадный алгоритм, w-совершенные графы, *T*-раскраска, суммирующая раскраска

Введение.

Постановка задачи. Рассмотрим задачу раскраски вершин графа в распределенной сети. Распределенные сети состоят из множества процессоров V и множества двусвязных каналов связи между парами процессоров - E. Данная сеть может быть смоделирована неориентированным графом G=(V, E). Обозначим через $n=|V|$, $m=|E|$ и для каждой вершины v определим открытую окрестность $N(v)=\{u:(u,v)\in E\}$, а также степень вершины $\deg_G v=|N(v)|$.

Раскраской вершин графа G называется такое приписывание цветов его вершинам, что никакие две смежные вершины не получают одинаковые цвета. Хроматическое число χ(G) графа G определяется как наименьшее количество цветов, необходимых для раскраски G, а раскраска графа χ (G) цветами называется оптимальной раскраской графа G. Хроматическое число графа ограничено сверху числом $\Delta+1$, где Δ обозначает максимальную степень вершины графа. Задача раскраски графа заключается в нахождении оптимальной раскраски. Данная распределенная модель раскраски может быть использована в распределенных беспроводных сетях для устранения столкновений пакетов путем назначения ортогональных кодов радиостанциям [1].

Модель вычислений. Распределенные вычисления на графах представляют собой такую организацию вычислений, при которой отсутствует всякая возможность использовать глобальные операции и механизмы, а также возможность получать информацию иначе, нежели использование информации из локальной памяти соседей. Это означает, что в основе любого распределенного вычисления на графах лежит организация взаимодействия с соседними вершинами и пересылка локальной информации.

Для измерения временной сложности используем число синхронизованных раундов, хотя обсуждаемые в работе алгоритмы могут быть адаптированы и для асинхронных моделей.

При оценке выполнения случайного распределенного алгоритма раскраски графа G имеется не менее двух случайных переменных, вызывающих наш интерес: $C_A(G)$ – количество цветов, используемых алгоритмом для раскраски графа G, и $T_A(G)$ – количество раундов, используемых для раскраски G.

Хорошим распределенным алгоритмом считается алгоритм, в котором $C_A(G)$ близок к χ(G) и $T_A(G)$ является маленьким относительно числа вершин в G. Разность $C_A(G)$ – χ(G) может рассматриваться как мера эффективности алгоритма. Хотя в общем случае аппроксимации χ(G) в пределах фактора $n^{1/7-\varepsilon}$ является NP-трудной задачей для всех $\varepsilon > 0$ [2].

Алгоритм жадной раскраски в распределенном контексте. Для заданного графа G и последовательности вершин $K=(v_1,v_2,...v_n)$ воспользуемся термином жадная раскраска для описания следующей процедуры назначения цветов:

алгоритм *Жадная Раскраска (G,K):*

для $v = v_1$ до v_n выполнить и назначить вершине v наименьший возможный цвет;

Последовательным алгоритмом раскраски называется алгоритм, который вычисляет последовательность K вершин графа G и красит его вершины, используя процедуру *Жадная Раскраска(G, K).*

Ниже мы коротко опишем основные принципы наиболее распространенных последовательных алгоритмов; более детальный анализ последовательных алгоритмов можно найти в [3, 4].

- НП-алгоритм: последовательность K образуется упорядочиванием вершин графа G в невозрастающем порядке их степеней;
- ПН-алгоритм: последовательность K образуется повторным удалением вершин с минимальной степенью из графа G и их помещением в конец K;
- НПН(DSATUR)-алгоритм: последовательность K образуется путем динамического упорядочения вершин графа G в невозрастающем порядке их степеней насыщения, где степень насыщения есть количество раскрашенных соседних вершин (в начале алгоритма цвета получают вершины, имеющие наибольшую степень).

Последовательные алгоритмы, обсуждаемые в данной работе, имеют практическое значение из-за числа класса графов, для которых они всегда дают оптимальные или почти оптимальные результаты. Поэтому естественно сформулировать задачу построения распределенных алгоритмов с аналогичными свойствами.

Основные результаты. Заметим, что не всегда легко добиться и скорости, и эффективности алгоритма. В работе [5] представлен распределенный алгоритм для раскраски графа в (Δ +1) цветов, где Δ – наибольшая степень вершины в графе. Время работы алгоритма O(log n). Будем называть этот алгоритм тривиальным. Данный алгоритм достаточно простой и быстрый, но не оптимальный. Действительно, количество цветов, используемых алгоритмом, близко к Δ, даже если граф двудольный. Не удивительно, что тривиальный алгоритм не имеет механизма экономии цветов. Дальнейшее усовершенствование тривиального алгоритма предложено в работе [6]. В ней приведен новый алгоритм для раскраски в O (Δ / log Δ) цвета, но этот алгоритм работает только на графах без триангуляторов.

Авторы работ [7, 8] представили детерминированный распределенный алгоритм для правильной раскраски кольца, использующий три цвета за время O (log*n). Обобщение данной техники может использоваться для раскраски корневых деревьев и произвольных графов с постоянной степенью в (Δ + 1) цветов за время O (log*n) [8]. В работе [9] показано, что время выполнения O (log*n) также является достаточным для получения (Δ + 1) раскраски в единичных дисковых графах, а его обобщение - и в единичных шаровых графах с постоянным удваивающимся размером. Для произвольных графов (Δ + 1) раскраска может быть вычислена за время O (log*n + Δ^2) [10] или O (Δ log n) [11]. Авторы [12] представили распределенные методы нахождения раскрасок в графах, допускающих раскраску в менее, чем Δ + 1 цветов. В работе [13] описан распределенный алгоритм для раскраски произвольных графов, использующий $O(\Delta^2 \log n)$ раундов. Наконец, экспериментальное исследование различных алгоритмов раскрасок вершин дано в [14].

Распределенное выполнение ПН-алгоритма для раскраски w-совершенных графов. Принципиальные трудности, которые возникают при раскраске графа и нахождении его хроматического числа, вынуждают, во-первых, найти и исследовать практически интересные классы графов, для которых задача раскраски полиномиально разрешима, и, во-вторых, вычислить или оценить хроматическое число графа с помощью других, более легко вычисляемых характеристик графа.

Одной из важных характеристик, связанных с хроматическим числом, является число Секереша-Вилфа $w(G) = \max_{G' \subset G} \delta(G') + 1$, где $\delta(G') = \min_{x \in V(G')} d_{G'}(x)$ – минимальная степень графа G', а $d_{G'}(x)$ – степень вершины x в G'. $w(G)$ в качестве верхней оценки хроматического числа впервые рассмотрена в работе Секереша и Вилфа [15].

Важность этой характеристики заключается в том, что, во-первых, $w(G)$ является довольно нетривиальной верхней оценкой для $\chi(G)$ [16], т.е. класс графов, для которых $w(G) = \chi(G)$ довольно большой и содержит в себе много практически интересных классов, и, во-вторых, она легко вычисляемая.

Граф, обладающий таким свойством, что хроматическое число и вырожденность (число Секереша-Вилфа) равны не только у самого графа, но и у каждого его порожденного подграфа называется *w-совершенным графом*.

Важным подклассом w-совершенных графов являются хордальные графы [17].

Граф называется *хордальным (триангулированным)*, если каждый его цикл длины > 3 содержит хорду, т.е. ребро, соединяющее несмежные вершины простого цикла.

Более подробные определения и свойства w-совершенных графов приведены в работах [17, 18].

Для вычисления $w(G)$ вводится понятие упорядочения по наименьшему последнему (*ПН - упорядочение*) {3, 19} графа G. Процедура определения ПН - упорядочения вершин и нахождения по нему раскраски называется *ПН – алгоритмом* [3]. По своему строению ПН - алгоритм приводит к раскраске не более, чем $w(G)$ цветами.

Описание алгоритма. *Параллельно-последовательным вычислительным алгоритмом* называется локальный алгоритм, каждый шаг которого состоит в обработке параллельно и независимо всех (или некоторых) вершин, смежных с вершинами, обработанными на предыдущем шаге; на первом шаге обрабатывается фиксированное множество начальных вершин [20].

Теорема 1. Задача раскраски w-совершенных графов ПН-алгоритмом разрешима в классе распределенных параллельно последовательных вычислительных алгоритмов.

Лемма 4. *Распределенный динамический ПН-алгоритм оптимально или почти оптимально красит графы из класса w-совершенных графов не боле, чем $w(G)$ цветами за время $O(\Delta^2 \log n)$.*

Отметим, что время выполнения распределенного динамического ПН-алгоритма уменьшается до $O(\log n)$ для раскраски r-регулярных графов с n вершинами. Использованные оценки не являются строгими, и граница для сложности не эффективна при $\Delta = \Theta(n)$. Однако графы, которые возникают в приложениях модели раскраски распределенного графа, обычно являются разреженными. Например, если рассмотрим графы, для которых $\Delta = O(\log n)$, тогда сложность алгоритма будет равна $O(\log^3 n)$.

Разновидности раскраски графов.

***T*-раскраска графов.** *T*-раскраска графа является обобщением вершинной раскраски графа. Она была введена в качестве модели для назначения передатчикам радиочастот [21, 22, 23]. Предположим, что G - простой граф и *T* - конечное множество неотрицательных целых чисел, включая 0. *T*-раскраска графа G - это некоторая функция c, которая назначает целое число (цвет) каждой вершине G так, что если $\{u,v\} \in E$, тогда $c(u) - c(v) \notin T$. Назовем множество *T* множеством запрещенных расстояний. Более подробно о *T*-раскраске представлено в работах [24, 25].

Отметим, что вершинная раскраска и T-раскраска графа G взаимосвязаны: каждая T-раскраска G является вершинной раскраской G и каждая вершинная раскраска G является его {0}-раскраской. Легко увидеть, что каждый граф G имеет бесконечное число T-раскрасок и, по меньшей мере, только одна из них является оптимальной (e-оптимальной).

Алгоритм *T*-DSATUR - это применение хорошо известного последовательного НПН-алгоритма (DSATUR) для *T*-раскраски графов. Алгоритм *T*-DSATUR начинается назначением цвета 1 вершине с максимальной степенью. Затем последовательно окрашиваются вершины с наибольшей степенью насыщения $\deg_s(v)$. Коста первым обратился к *T*-раскраске графов посредством эвристики и предложил обобщение эвристики DSATUR для раскраски графа [26].

Теорема 2. Задача *T*-раскраски графов НПН (DSATUR)-алгоритмом разрешима в классе распределенных параллельно последовательных вычислительных алгоритмов.

Распределенный динамический НПН (DSATUR)-алгоритм оптимально красит все двудольные графы для произвольных множеств *T*. Рассматриваемый алгоритм дает наименьший промежуток (минимальную разницу между использованными цветами, для которой существует *T*-раскраска) и использует меньшее число цветов для раскраски произвольных графов, чем другие последовательные алгоритмы.

Покажем, что самые популярные последовательные алгоритмы раскраски налагают строгие нижние границы на ожидаемое время вычисления в распределенной среде. Рассмотрим НП и НПН (DSATUR) алгоритмы. Данные алгоритмы правильно раскрашивают пути и кольца, следовательно, распределенные выполнения данных алгоритмов имеют одинаковые свойства. Линиал [27] доказал, что точная раскраска кольца требует $\Omega(n)$ раундов, так что мы получаем следующее заключение.

Лемма 5. Любое распределенное выполнение НП или НПН (DSATUR)-алгоритма требует $\Omega(n)$ времени.

Суммирующая раскраска графов. Задача суммирующей раскраски состоит в нахождении правильной вершинной раскраски графа G, в которой минимизируется общая сумма цветов всех вершин графа G. Эта минимальная общая сумма называется хроматической суммой графа, $\sum(G)$, т.е.

$$\sum G := \min_c \sum(G,c),$$

где

$$\sum(G,c) = \sum_{v\in v(G)} c(v)$$

и $c := V(G) \to \mathrm{N}$ – правильная вершинная раскраска графа G. Раскраска графа, при которой достигнута минимальная хроматическая сумма, называется оптимальной раскраской. Существуют графы, для которых оптимальная суммирующая раскраска требует больше цветов, чем их хроматическое число.

Предположим, что P – оптимальная вершинная раскраска графа G с k цветами. Обозначим множество вершин, окрашенных в цвет i, через C_i, где $1 \le i \le k$. Поэтому, по определению, $\sum(G) = \sum_{i=1}^{k} i|C_i|$ и, следовательно, $|C_i| \le |C_{i+1}|$ для $1 \le i \le k$. Ясно, что $k \ge \chi(G)$. Можно предположить, что оптимальную вершинную раскраску можно получить, найдя правильную раскраску с $\chi(G)$ цветами и затем присваивая цвет 1 наибольшему цветному классу, цвет 2 – следующему наибольшему цветному классу и т.д. Однако это не всегда дает нам оптимальную вершинную раскраску, даже для деревьев.

Николоско и др. [28] связывают изучение задачи суммирующей раскраски с задачей проектирования СБИС, известной, как задача трассировки сверхячейки. Задано множество сетей с двумя терминалами, которые должны быть связаны электрической сетью. Существует основная линия, на которой имеются сети и несколько параллельных горизонтальных, равномерно распределенных каналов, расположенных на расстояниях d=1, 2, 3… от основной линии. Связь двух сетевых терминалов, чьи позиции задаются, должна быть составлена из двух вертикальных сегментов и одного горизонтального сегмента, где горизонтальный сегмент должен лежать на одном из каналов. Отметим, что перекрывающиеся сети не могут быть трассированы через один канал (рис. 1).

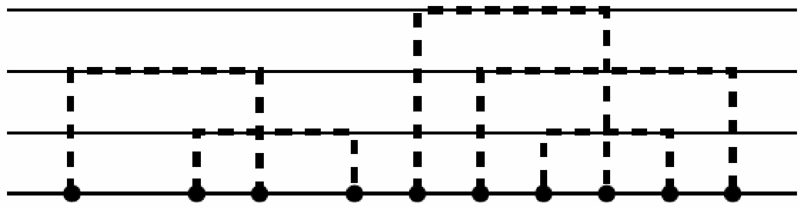

Рис.1. Пример задачи трассировки сверхячейки.

Цель задачи состоит в том, чтобы минимизировать полную длину проводов для соединения сетей. Полная длина горизонтальных проводов фиксирована и равна общей сумме расстояний двух терминалов сетей. Поэтому мы должны минимизировать сумму длин вертикальных проводов. Легко заметить, что эта задача эквивалентна задаче суммирующей раскраски ограниченного интервального графа.

Рассмотрим ещё одну стратегию раскраски, последовательный НП-алгоритм для суммирующей раскраски в распределенной модели вычислений. Данный алгоритм раскрашивает вершины графа в порядке убывания их степеней. Наш алгоритм является обратным НП-алгоритму, т.е. вершины графа раскрашиваются в порядке возрастания их степеней.

Для рассматриваемого алгоритма также справедлива следующая теорема.

Теорема 3. Задача суммирующей раскраски графов обратным НП-алгоритмом разрешима в классе распределенных, параллельно последовательных вычислительных алгоритмов.

Доказательство теоремы проводится аналогично доказательствам теорем 1 и 2. Данный алгоритм схож с распределенным ПН-алгоритмом, лишь с тем отличием, что меняется приоритет вершин.

Далее будут рассмотрены некоторые классы графов, для которых распределенный обратный НП-алгоритм дает оптимальную суммирующую раскраску.

Теорема 4. Распределенный обратный НП-алгоритм оптимально красит гусеницу не более чем, за три раунда.

Гусеница – это дерево, листья которого присоединены к пути.

В первом раунде все висячие вершины получат цвет 1, во втором и в третьем раундах будут раскрашены вершины пути. Теорема доказана.

Теорема 5. Распределенный обратный НП-алгоритм оптимально красит двудольное колесо BW_k , для $k \geq 2$, не более, чем за три раунда.

Двудольное колесо – это граф, получаемый из четного цикла длиной 2k путем добавления центральной вершины, соединенной с каждой второй вершиной цикла.

В первом раунде цвет 1 получат вершины, не соединенные с центральной вершиной и имеющие наименьшие степени в графе, во втором раунде цвет 2 получат вершины, соединенные с центральной вершиной, и в третьем раунде центральная вершина без всяких конфликтов получит цвет 1. Теорема доказана.

Теорема 6. Распределенный обратный НП-алгоритм оптимально красит корону не более, чем за 3 раунда, при четном цикле и не более, чем за 4 раунда, при нечетном цикле.

Корона – это граф, получаемый из цикла путем присоединения висячих ребер.

В первом раунде все висячие вершины будут раскрашены в цвет 1. Неокрашенный цикл будет раскрашен в 2 цвета за два раунда, если цикл четный, и в 3 цвета за три раунда, если цикл нечетный. Теорема доказана.

Теорема 7. Распределенный обратный НП-алгоритм оптимально красит двойную звезду за три раунда.

Двойная звезда – это дерево с *n-2* листьями.

В первом раунде будут раскрашены все листья в цвет 1, во втором и в третьем раундах будут раскрашены оставшиеся две вершины в цвета 2 и 3. Теорема доказана.

Литература:

1. Battiti R., Bertossi A.A., Bonuccelli M.A. Assigning codes in wireless networks. Wireless Networks, 1999. - N.5. - P. 195-209.
2. Bellare M., Goldreich O., Sudan M. Free bits, PCPs and non-approximability - towards tight results. - SIAM J. Comp., 1998. - Vol.27. - P.804-915.
3. Евстигнеев В.А. Применение теории графов в программировании. - М.: Наука, 1985. - 352 с.
4. Емеличев В.А., Мельников О.И., Сарванов В.И., Тышкевич Р.И. Лекции по теории графов. - М.: Наука, 1990.
5. Johansson Ö. Simple distributed $\Delta+1$- coloring of graphs. - Inf. Process. Letters, 1999. - Vol. 70. - P. 229-232.
6. Grable D.A., Panconesi A. Fast distributed algorithms for Brooks-Vizing colorings. - J. Algorithms, - 2000. - Vol.37. - P. 85-120.
7. Cole R., Vishkin U. Deterministic Coin Tossing with Applications to Optimal Parallel List Ranking. - Inf. Control, 1986. - Vol.70. N.1. - P.32-53.
8. Goldberg A. V., Plotkin S. A. Parallel $\Delta+1$-Coloring of Constant-degree Graphs. - Information Processing Letters, 1987. - N.25. - P.241-245.
9. Kuhn F., Moscibroda T., Wattenhofer R. On the Locality of Bounded Growth. - Proc. of the 23rd ACM Symposium on Principles of Distributed Computing (PODC), 2005. - P. 60-68.
10. Panconesi A., Rizzi R. Some Simple Distributed Algorithms for Sparse Networks. - Distributed Computing, 2001. -Vol.14. N.2. - P.97-100.
11. Goldberg A., Plotkin S., Shannon G. Parallel symmetry-breaking in sparse graphs. - Proc. of the 19th Annual ACM Conference on Theory of Computing (STOC), 1987. - P. 315-324.

12. Grable D. A., Panconesi A. Fast Distributed Algorithms for Brooks-Vizing Colorings. - Proc. of the 9th Annual ACM-SIAM Symposium on Discrete Al-gorithms (SODA), 1998. - P. 473-480.
13. Hansen J., Kubale M., Kuszner L., Nadolski A. Distributed Largest-first al-gorithm for graph coloring. - Proc. of EuroPar. - Lect. Notes Comput. Sci. - Springer-Verlag, 2004. - Vol. 3149. - P. 527-539.
14. Panconesi A., Silvestri R. Experimental Analysis of Simple, Distributed Vertex Coloring Algorithms. Proc. of the 13th ACM-SIAM Symposium on Discrete Algorithms (SODA), 2002. - P. 606-615.
15. Szekeres G., Wilf H.S. An inequality for the chromatic number of a graph. - J. Combin. Theory, 1964. - Vol.4. - P. 1-3.
16. Волошин В.И. Свойство триангулированных графов // Исслед. операций и программирования. Мат.наук. - Кишинев, 1982. - С.24-32.
17. Маркосян С.Е., Гаспарян Г.С. w-совершенные графы // Ученые записки. Ереван.гос.универ-т, 1987. - №3 - С.9-15.
18. Евстигнеев В.А. Хордальные графы и их свойства // Проблемы систем информатики и программирования. - Новосибирск, 1999. - С.33-64.
19. Matula D.W., Bleck L.L. Smallest-last ordering and dustering and graph coloring algorithms. - J. Assoc. Comput. Math, 1983. - Vol.30. N.3. - P.417-427.
20. Евстигнеев В.А. О некоторых свойствах локальных алгоритмов на графах // Комбинаторно-алгебраические методы в прикладной математике. - Горький: ГГУ, 1983. - С.72-105.
21. Hale W.K. Frequency assignment: theory and applications. - Proc. of the IEEE, 1980. - Vol.68. N.12. - P.1497-1514. 38.
22. Cozzens M. B., Roberts F. S. T-colorings of graphs and the channel assignment problem. - Congressus Numerantium, 1982. - Vol.35. - P.191-208.
23. Simon H. U. Approximation algorithms for channel assignment in cellular radio

networks. - In Fundamentals of Computation Theory. - Springer-Verlag, 1989. - Lect. Notes Comput. Sci. - Vol. 380. - P. 405-415.

24. Roberts F. S. T-colorings of graphs: Recent results and open problems. - Discrete Mathematics, 1991. - Vol.93. N.2-3. - P.229-245.

25. Tesman B. A. Set T-colorings. - Congressus Numerantium, 1990. - N.77. - P.229-242.

26. Costa D. On the use of some known methods for T-colorings of graphs. Annals of Operations Research, - 1993. - Vol.41. N.4. - P.343-358.

27. Linial N. Locality in distributed graph algorithms. SIAM J. Computing, 1992. - Vol.21. - P. 193-201.

28. Nicolosco S., Sarrafzadeh M., Song X. On the sum coloring problem on interval graphs. - Algorithmica, 1999. - Vol.23. - P. 109-126.