Исследование возможности применения перколяционных методов в задаче декомпозиции графа

Н.И. Дубровин, М.Ю. Звягин, П.Ю. Шамин

Владимирский государственный университет
ndubrovin@rambler.ru, muz1953@ya.ru, trace83@mail.ru

Введение

При разработке средств компьютерного моделирования для решения различных инженерных задач разработчики по мере развития возможностей создаваемой системы рано или поздно сталкиваются с ситуацией, когда ресурсов вычислительной системы, на которой должны функционировать создаваемые средства имитационного моделирования, оказывается недостаточно. Очевидным выходом из этой ситуации является использование технологий параллельных вычислений, особенно с ориентацией на системы с распределённой памятью (например, кластерные системы), что позволяет достичь значительно более высокой масштабируемости по быстродействию и объёму данных, которыми может оперировать программа (о принципах осуществления такого перехода см., например, [1]). Однако на пути перехода от обычной непараллельной программы к параллельной, работающей на системе с распределённой памятью, имеются свои «подводные камни», которые осложняют такой переход. Одним из них является необходимость обеспечения как можно меньшей связности по данным между параллельно работающими процессами. Дело в том, что для систем с распределённой памятью сетевой интерфейс, объединяющий отдельные узлы в единую вычислительную систему является узким местом, и программа, требующая интенсивного обмена данными между процессами, с высокой вероятностью будет работать очень медленно, несмотря на огромную вычислительную мощность кластера, обладающего сотнями, а то и тысячами вычислительных ядер. С подобной проблемой столкнулись и мы при разработке параллельного сетевого симулятора (ПСС), предназначенного для моделирования компьютерных сетей (в том числе мобильных и гетерогенных), а также других сетевых структур, в том числе нетехнического характера. В системе ПСС (см. [2]) моделируемая сеть представляется в виде набора узлов, соединённых линиями связи, по которым они могут обмениваться пакетами данных. При этом различные узлы моделируемой сети могут моделироваться различными MPI-процессами, выполняющимися на различных узлах кластерной системы. Очевидно, что от того, как именно будут распределены узлы моделируемой сети по различным MPI-процессам, в существенной степени зависит величина сетевого трафика между узлами кластера в ходе моделирования. Поэтому для того, чтобы обеспечить эффективную работу симулятора требуется распределить узлы моделируемой сети по моделирующим MPI-процессам таким образом, чтобы средняя величина суммарного трафика между узлами моделируемой сети, обрабатываемыми разными MPI-процессами была как можно меньше. Очевидно, что в данном случае мы сталкиваемся с задачей об оптимальной декомпозиции взвешенного графа, вершинами которого являются узлы моделируемой сети, а весами рёбер – средний трафик между парами узлов в единицу времени. Оптимальной декомпозицией будет та, для которой суммарный вес связей, по которым прошло сечение, будет минимальным. В дальнейшем для простоты изложения будем считать, что декомпозицию требуется осуществить на две доли, состоящие из равного количества вершин. Вышеописанную задачу в случае малой размерности сети (графа), можно осуществить уже известными алгоритмами [3, 4]; это не является проблемой. Нас же, поскольку ПСС рассчитан на моделирование сетей с большим количеством узлов, интересуют сети больших размеров (10^4 узлов и более). При таких размерах сети меняются требования к алгоритму декомпозиции, например, процедура набора узлов при формировании доли должна опираться только на локальную информацию. Это означает, в частности, что такого алгоритма как сортировка всего массива данных сле-

дует избегать. Кроме того придется отказаться от самой идеи отыскания оптимального разбиения и как-нибудь по иному оценивать качество работы предлагаемого алгоритма. На изучаемые сети наложим следующее существенное ограничение. Будем полагать, что степени связности узлов k, на период моделирования, ограничены некоторой константой $k_{\max}$, не зависящей от N – количества узлов в сети. Это ограничение, по крайней мере, приводит к двум существенным заключениям: «затраты» на осуществление подсчета веса разбиения и любой версии лавины, оцениваемые по числу элементарных итераций, ограничиваются количеством связей сети. Следовательно, они растут не быстрее $k_{\max} N$.

Постановка задачи

Пусть задан граф с четным количеством узлов и нагруженными связями. Разбиваем его на две равные по количеству узлов доли и вычисляем суммарный вес W связей между долями. Классическая задача декомпозиции формулируется так: минимизировать значение W за счет выбора оптимального разбиения. В такой постановке она относится к классу NP-полных и не имеет удовлетворительного решения для больших графов (количество вершин не менее 10^4); а именно они и являются предметом нашего внимания. Чтобы решение задачи не было безнадежным, изменим её постановку. Для этого представим ситуацию следующим образом: элементарное событие ω это разбиение; множество всех разбиений $\Omega = \{\omega\}$ – пространство элементарных исходов; вес разреза $W = W(\omega)$ – случайная величина. Введем простейший алгоритм A_0 – «случайное разбиение графа на две равные доли и вычисление веса разбиения». Его выбор обусловлен тем, что он является безусловным лидером в таком показателе, как минимизация затрат на реализацию. Основной характеристикой алгоритма, этого и любого другого, является средний вес разбиения M_0 ($M_0 = M(W)$ – математическое ожидание случайной величины, введенной выше). Обозначим через T_0 – среднее время необходимое на его однократное применение (конечно, оно зависит от технических характеристик, например, от конкретной его программной реализации алгоритма). Итак, A_0 будем считать эталонным алгоритмом, все другие алгоритмы доставляющие разбиение, будем соразмерять именно с ним. Более конкретно, пусть A алгоритм именно такого типа. Обозначим через M_A средний вес разбиения, полученного при помощи алгоритма (условное математическое ожидание), соответственно T_A – среднее время на его реализацию. Качество A характеризуется парой чисел $\left(M_A/M_0 ; T_A/T_0\right)$, естественно, чем они меньше, тем лучше. Однако, скорее всего, уменьшение одного показателя будет достигаться за счет другого. Поэтому, в каждой конкретной ситуации следует применять целевую функцию, например, $C_1\left(1 - M_A/M_0\right) + C_2\left(1 - T_A/T_0\right)$. При этом, выбор коэффициентов C_1, C_2 зависит от рассматриваемого приложения и не является предметом исследования данной работы.

Обсудим кратко A_0. Может создаться впечатление, что он не имеет практического значения, но это не так. Действительно, если исходный граф полносвязный и веса связей одинаковы, то любое случайное разбиение является оптимальным. Более того, был проведен ряд компьютерных экспериментов на случайных графах (с количеством узлов не менее $4 \cdot 10^4$ и случайным распределением весов связей). Он показал, что при не слишком большом разбросе веса связей и достаточно однородной топологии сети, когда степени связности узлов отличаются друг от друга незначительно, веса разбиений (объем выборки до 10^5) практически не отличались (расхождение порядка 3%). Хотя вопрос о весе оптимального разбиения так и остается открытым, приходится сделать вывод, что именно A_0 и есть тот алгоритм, который следует применять в данных конкретных условиях.

Вместе с тем, легко привести пример, когда применение A_0 окажется совершенно неуместным. А именно, пусть исходный граф состоит из двух одинакового размера полносвязных подграфов, веса связей внутри которых превосходят некоторый уровень w_1. Пусть между подграфами установлены связи, веса которых не превосходят некоторый уровень w_0 $(w_0 << w_1)$. Параметры можно подобрать так, чтобы суммарный вес всех этих связей вообще не играл существенной роли. Очевидно, оптимальное разбиение – это разбиение на подграфы. Его практически невозможно найти при помощи A_0, вес же остальных разбиений на доли существенно больше веса оптимального. Вместе с тем, очевидно, как найти оптимальное разбиение при помощи алгоритма локального действия. Нужно просто случайно выбрать стартовый узел и запустить из него лавину по связям, вес которых не меньше w_1; узлы, которых она достигнет, и есть узлы одной из долей оптимального разбиения. Итак, предлагается перколяционный алгоритм A (простейший вариант); его принципиальная схема:

1)Выбираем уровень $w_{кр}$.

2)Выбираем случайно стартовый узел. Запускаем лавину по связям, вес которых не менее $w_{кр}$ (перколяцию).

3)Каждый узел, достигаемый перколяцией, помещаем в долю 1. Процесс продолжается, пока в ней не окажется половина узлов сети. Доля 2 – оставшиеся узлы.

4)Если перколяция останавливается, а в доле 1 узлов не хватает, то повторяем пункт 2). При этом новый стартовый узел выбираем так, чтобы он не входил в число узлов, ранее помещенных в долю 1.

5)Вычисляем вес полученного разбиения.

Следует отметить, что если $w_{кр}$ меньше минимального веса связи графа, то A превращается в просто лавину. Если, наоборот, $w_{кр}$ велик, превосходит вес максимальной связи, алгоритм вырождается в A_0. Соображения, изложенные выше, приводят нас к следующей формулировке цели: Выявить класс графов, на которых A действует успешно (ясно, что это должны быть существенно неоднородные графы). Оценить поведение параметров эффективности, например, в зависимости от $w_{кр}$. По возможности, оценить практическую значимость тех классов графов, где применение перколяционного принципа при построении разбиения оправдано.

Эксперимент

Генерируется 50 случайных неориентированных графов по 200 вершин («малые» графы). Каждая вершина сгенерированных графов имеет степень 6. Производится объединение этих 50 графов в один граф из 10000 вершин (большой граф) путём добавления каждой вершине ещё двух рёбер, соединяющих её с вершинами других «малых» графов. В результате каждая вершина «большого» графа получает степень 8. Рёбрам графа придаются веса в диапазоне [1, 100). При этом рёбрам, соединяющим вершины одного «малого» графа, даются в среднем большие веса, чем рёбрам, соединяющим вершины различных «малых» графов. Конкретные диапазоны весов рёбер в ходе различных экспериментов приведены в табл. 1.

Таблица 1. Диапазоны весов рёбер в ходе экспериментов.

Номер эксперимента	Диапазон весов рёбер, соединяющих вершины из различных малых графов	Диапазон весов рёбер, соединяющих вершины из одного малого графа
1	[1, 10)	[10, 100)
2	[1,25)	[4, 100)
3	[1, 50)	[2, 100)

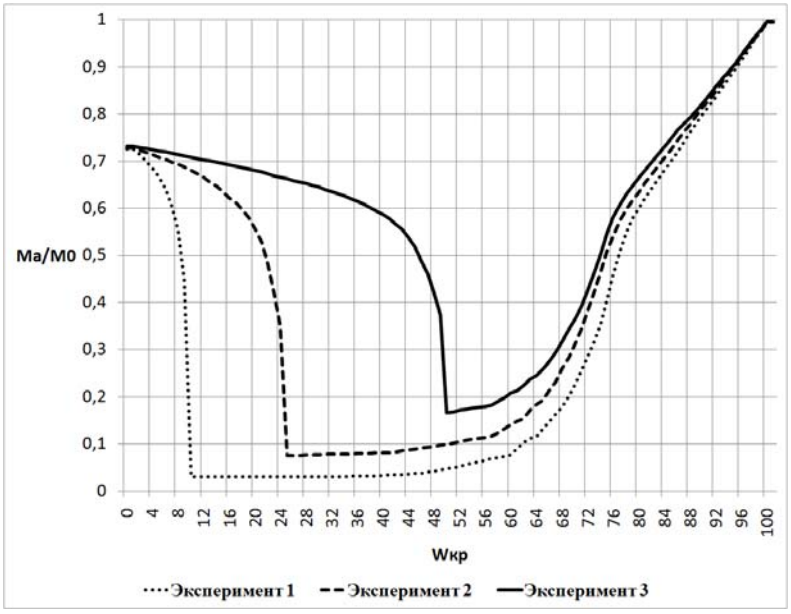

Рисунок 1. Отношение среднего веса перколяционного разбиения к среднему весу случайного разбиения в зависимости от выбранного значения .

Сначала, для построенного «большого» графа, 1000 раз строится случайное разбиение. Вычисляется вес каждого разбиения и измеряется время его построения. Вычисляются M_0 и T_0 – средние значения веса и времени построения, соответственно. Далее, для каждого из 102 значений $w_{кр}$, взятых из интервала [0, 101] с шагом 1 проделывается то же самое. А именно: во-первых, строится по 1000 перколяционных разбиений, для каждого разбиения вычисляется вес и измеряется время его построения; во-вторых, по каждой выборке вычисляются средние значения M_A и T_A. После чего вычисляются отношения M_A/M_0 и T_A/T_0, полученные зависимости M_A/M_0 от $w_{кр}$ приведены на рис. 1, а T_A/T_0 от $w_{кр}$ – на рис. 2.

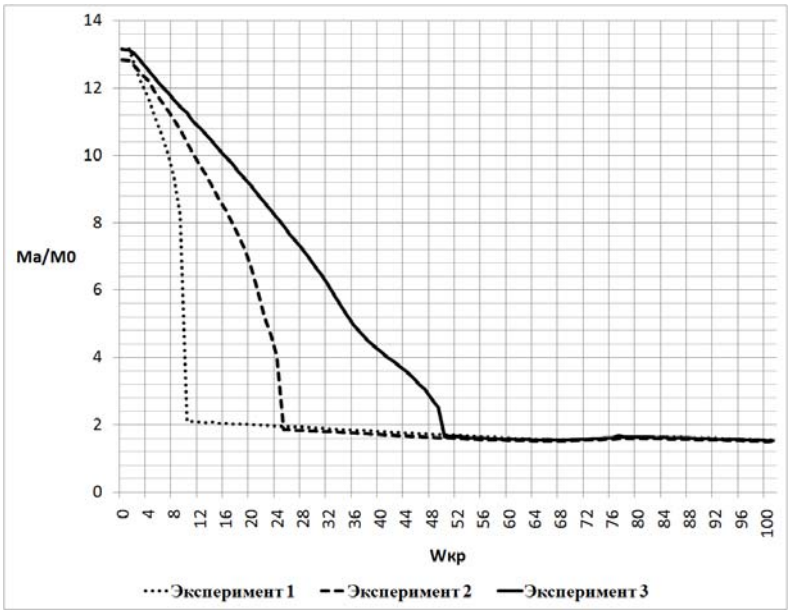

Рисунок 2. Отношение среднего времени построения перколяционного разбиения среднему времени построения случайного разбиения в зависимости от выбранного значения.

Выводы

Как и ожидалось, для неоднородных графов, которые, на наш взгляд, являются естественными моделями гетерогенных сетей и сетевой структуры технопарковых зон, предлагаемый подход оказался весьма эффективным. Представляется перспективным и практически значимым, наряду с другими методами, его дальнейшее развитие.

Литература

1. Воеводин, В.В. Параллельные вычисления / В.В Воеводин, Вл.В. Воеводин. – СПб.: БХВ-Петербург, 2004. – 608с. – ISBN 5-94157-160-7.
2. Шамин, П.Ю. Параллельный сетевой симулятор: концепция и перспективы развития / П.Ю. Шамин, А.С. Алексанян, В.В. Прокошев // Научно-технические ведомости Санкт-Петербургского государственного политехнического университета. — СПб, 2009. – № 3. – С. 18-24.
3. Батищев, Д.И. Применение генетических алгоритмов к решению задачи дихотомического разбиения графа. / Д.И. Батищев, Н.В. Старостин // Межвузовский сборник науч. трудов «Оптимизация и моделирование в автоматизированных системах». – Воронеж, 1998., С.3 – 10.
4. Chamberlain, B.L. Graph partitioning algorithms for distributing workloads of parallel computations / B.L. Chamberlain // Technical Report TR-98-10-03. – Univ. of Washington, Dept. of Computer Science & Engineering. – October 1998.