Исследование одно/двунаправленных распределённых сетей конечным роботом

И.Б.Бурдонов

Изучение распределенных сетей обычно фокусируется на двунаправленных сетях, соответствующих неориентированным графам. Граф интерпретируется как сеть, вершины графа - это узлы сети, а рёбра - соединения. Однако, с однонаправленными сетями (ориентированные графы) приходится иметь дело чаще, чем это можно было бы ожидать. Во-первых, однонаправленные сети возникают в результате сбоев или разрушений в соединениях. Например, модемная БИС на одном конце соединения может перестать принимать (или посылать) данные, но продолжать работать в другом направлении. Во-вторых, односторонние соединения можно обнаружить в радиосетях с асимметричными матрицами передачи как результат различий в пропускной способности станций, в оптоволоконных сетях, и в СБИС.

Исследование сети, преследуя цели контроля, сбора статистики, тестирования или диагностики распределённых систем, основано на обходе графа сети. Обход - это маршрут, проходящий через все вершины и рёбра графа, причем, если ребро ориентированное (дуга), оно проходится только в направлении его ориентации. В терминах сети обход графа можно интерпретировать двумя способами. 1) Имеется единственный процесс, который перемещается по сети от узла к узлу, читая пометки в узлах и записывая новые пометки. Узлы сети пассивны и используются лишь для хранения пометок. 2) Наоборот, перемещается в сети единственное пассивное сообщение, пересылаемое от узла к узлу. Активными являются узлы сети, в которых находятся программы, которые срабатывают только при получении сообщения, отправляя, в свою очередь, новое сообщение по одному из выходящих соединений. Математически это описывается абстрактным автоматом, состояние которого, в зависимости от интерпретации, - это 1) сообщение в сети или 2) информация в узле, а входные/выходные символы - это 1) пометки в узлах или 2) принимаемое и посылаемое сообщение. Для определённости, мы будем опираться на первую интерпретацию.

В некоторых приложениях размер памяти в узлах и длина сообщения ограничены. В этом случае возникает проблема обхода конечным роботом, множества состояний и символов которого конечны. Такой робот аналогичен машине Тьюринга: лента заменена графом, а ее ячейки привязаны к вершинам и дугам графа. Следует подчеркнуть, что робот исследует сеть, ничего не зная о её топологии. Даже в том случае, когда «план сети» известен, для достаточно большой сети ограничения на память в узлах и длину сообщения не позволят роботу этот план использовать. Это означает, что робот обходит неизвестный ему граф. Единственная информация о графе, которая

задана первоначально и которая доступна роботу в вершине, - это перечисление дуг, выходящих из вершины (без указания, куда эти дуги ведут). Это аналогично обходу неизвестного города, все или некоторые улицы которого имеют одностороннее движение.

Для неориентированных графов задача обхода достаточно проста и соответствующие алгоритмы давно известны. В то же время, если алгоритм должен обходить любой граф, который можно обойти, деление дуг на однонаправленные и двунаправленные не имеет принципиального значения. Фактически, требуется алгоритм обхода любого конечного ориентированного графа, для которого существует обход. Наличие двунаправленных дуг (рёбер) используется лишь для оптимизации.

Обход с любой стартовой вершины существует только для сильно-связных графов, в которых каждая вершина достижима из каждой вершины. На классе таких графов длина обхода равна O(nm), где n - число вершин, а m - число дуг графа. Если робот - это «компьютер общего вида» без ограничения на число его состояний и символов, то известны алгоритмы обхода с той же оценкой [1]. Однако, для конечного робота нижняя оценка длины обхода неизвестна. Более того, хотя представляется сомнительным, чтобы конечный робот мог обходить граф за O(nm), тем не менее, это тоже не доказано.

Задача обхода конечным роботом была поставлена М.О.Рабином в 1967 г. [2]. В 1971 г. автор статьи предложил [3] робот с оценкой O(nm+(n**2)logn) (здесь и далее логарифм имеет основание 2). Алгоритм робота основан на построении максимального выходящего дерева графа и метода поиска в ширину (BFS) на этом дереве. В 1978 г. К.Кобаяши [4] представил алгоритм экспоненциальной сложности, основанный на идее DFS, а в 1988 г. С.Куттен [5] описал алгоритм с оценкой O(nm), но его робот не был конечным, так как использовал ячейки графа с logn битов памяти. В 1993 г. Y.Afek и E.Gafni [6] описали алгоритм с оценкой O(nm+(n**2)logn), также основанный на построении выходящего дерева, но использующий метод поиска в глубину (DFS). В обоих алгоритмах [3,6] возникает проблема отката по дереву: перебор всех вершин дерева в порядке их частичной упорядоченности от листьев к корню. Поэтому верхняя оценка отличается от оптимальной O(nm) на величину, требуемую для отката.

В 2004 г. автор статьи предложил алгоритм, совмещающий поиск в ширину (BFS) с методом отката, предложенным Y.Afek и E.Gafni, за счет чего достигается оценка O(nm+(n**2)loglogn) [7]. В процессе работы строится выходящее дерево с корнем в стартовой вершине и лес входящих деревьев с корнями в выходящем дереве. Каждая пройденная вершина принадлежит одному из входящих деревьев, а пройденное начало каждой непройденной дуги - выходящему дереву.

Сначала робот, начиная с корня текущего входящего дерева, движется по выходящему дереву в «поиске непройденной дуги». Проходимые дуги помечаются так, чтобы при следующем поиске в каждой вершине выбирать дугу выходящего дерева, следующую после помеченной. Тем самым реализуется BFS-метод. Найдя непройденную дугу, робот «движется по непройденным дугам», добавляя их в выходящее дерево, пока не пройдёт хорду, оказавшись в уже пройденной вершине. Для «возврата по хорде» робот многократно проходит по циклу, составленному из двух путей во входящем и выходящем деревьях и хорды, корректируя входящие деревья. Найдя начало хорды, робот удаляет её из выходящего дерева и переходит в корень текущего входящего дерева для «поиска непройденной дуги». Если непройденная дуга не найдена, значит робот пришёл в листовую вершину выходящего дерева, все выходящие дуги которой пройдены (полностью пройденная вершина). Тогда применяется «однократный откат по выходящему дереву» для удаления из дерева листа и всего пути, ведущего в лист и содержащего полностью пройденные вершины. Для этого также используется цикл, составленный из пути во входящем и выходящем деревьях. Цикл длиной k робот проходит O(logk) раз; суммарно полный откат по выходящему дереву при BFS-методе даёт второе слагаемое оценки O((n**2)loglogn).

В качестве следующего шага предложен [8] алгоритм DFS-отката по выходящему дереву (с добавленными путями из листьев в корень) с оценкой O((n**2)LOG(n)). Функция LOG определяется как целочисленное решение неравенства 1<=(log**LOG)(n)<2, где log**t=loglog…log – t-ая композиционная степень логарифма. К сожалению, этот результат не означает, что можно построить робот для обхода с оценкой O(nm+(n**2)LOG(n)), поскольку в процессе обхода не всегда можно попасть из листа в корень выходящего дерева. Выделение роботом такого дерева определяется упорядочиванием графа – порядком выходящих дуг в каждой вершине, который задается извне. Тем не менее, для любого сильно связного графа существует такое упорядочивание, при котором стартовая вершина всегда достижима по пройденным дугам. Интересно, что такое упорядочивание может сделать и сам робот, совершая обход. Тем самым, можно построить робот, который дважды обходит граф, первый раз с оценкой O(nm+(n**2)loglogn), а второй раз – O(nm+(n**2)LOG(n)).

Алгоритм отката основан на следующей идее. Если граф – простой цикл, а число символов робота не ограничено, то полный откат можно выполнить за O(n**2). Робот нумерует вершины убывающими числами: k,k-1,...,2,1, далее ненумерованные вершины. На следующем проходе робот к каждому номеру прибавляет единицу, а номер 1 ставит в первой ненумерованной вершине. Когда все вершины будут занумерованы, робот начинает вычитать единицу из каждого номера на каждом проходе, «откатывая» вершину с номером 0, пока номер 0 не окажется в стартовой вершине. Конечный робот

выполняет этот же алгоритм, но только номер записывается не в одной, а в нескольких идущих подряд вершинах, используя позиционную систему записи. Поскольку откатываться теперь нужно не только по номерам, но и по вершинам внутри одного номера, приходится применять «вложенные» роботы, что даёт оценку O((n**2)(log**t)(n)) для любого наперёд заданного числа t. Заменяя рекурсию итерацией, получаем искомый робот с оценкой O((n**2)LOG(n)). Далее этот алгоритм расширяется с цикла на любое дерево с добавленными путями из листьев в корень.

В смешанных сетях, где есть как однонаправленные (дуги), так и двунаправленные (рёбра) соединения, для возврата по рёбрам не нужно многократно проходить по циклу, что оптимизирует алгоритм как при возврате по хорде, так и, что более важно, при откате по дереву. Фактически, ориентированно нужно обходить фактор-граф по отношению связности по рёбрам. Для k компонентов связности получаем оценку O(nm+nkloglogn), что для k=O(m/loglogn) совпадает с оптимальной O(nm).

Литература

1. И.Б.Бурдонов, А.С.Косачев, В.В.Кулямин. Неизбыточные алгоритмы обхода ориентированных графов. Детерминированнный случай. "Программирование", 2003, №5.

2. M.O.Rabin, Maze Threading Automata. MIT and UC Berkeley, 1967.

3. И.Б.Бурдонов. Изучение поведения автоматов на графах. Дипломная работа, МГУ, мех-мат., 1971г.

4. K.Kobayashi. The firing squad synchronization problem for a class of polyautomata networks. Journal of Computer and System Science, 17, 1978.

5. S. Kutten. Stepwise construction of an efficient distributed traversing algorithm for general strongly connected directed networks. Proc. of 9 International Conference on Computer Communication, 1988.

6. Y. Afek and E. Gafni, Distributed Algorithms for Unidirectional Networks, SIAM J. Comput., Vol.23, No.6, 1994.

7. И.Б.Бурдонов. Обход неизвестного ориентированного графа конечным роботом. "Программирование", 2004, №4.

8. И.Б.Бурдонов. Проблема отката по дереву при обходе неизвестного ориентированного графа конечным роботом. "Программирование", 2004, №6.