УДК 519.178

Время поиска мостов и точек сочленения в сети с неизвестной топологией и синхронизированным временем

Вялый М.Н., Хузиев И.М.

Московский физико-технический институт (государственный университет)

Вычислительный центр им. Дороницына РАН2

Введение: нумерация в дереве

Что рассматриваем: сеть с неизвестной топологией, но в котором уже выбран лидер. То, как был выбран лидер, какие там ключи у узлов, нас на самом деле интересовать не будет.

Действительно, если есть лидер, а все остальные узлы знаю, что они не лидер, то за $O(ext_G(leeder))$ тактов времени можно построить остовное дерево, в корне которого будет лидер. Эксцентриситет ext — максимальное из расстояний по графу до остальных вершин.

Поэтому можно принять, что у нас есть сеть, в которой выделено остовное дерево с корнем R (он же лидер).

Следующий логичный этап — сделать "локальную"нумерацию узлов, то есть присвоить каждой вершине в сети уникальный номер из промежутка $1 \dots f(|V|, |E|)$. Мы построим наилучшую из возможных нумераций: $1 \dots |V|$, причём за время $O(|V|)$. Более того, мы одновременно передадим во все вершины структуру остовного дерева и для каждой — её место в этом дереве. И на самом деле ещё больше: наша нумерация будет задавать однозначно по дереву, а значит любой узел, зная номер другой вершины, сможет определить её положение в дереве.

Отступление. Если говорить про задачу нумерации: возможно, можно её решать быстрее - за $log|V| + diam(G)$, то есть нет априорной уверенности в точности оценки, которую даст наш протокол. Но с точки зрения плотности $(1 \dots |V|)$ — лучше нельзя. Если говорить про задачу распространения структуры дерева, то быстрее её решить нельзя ($\approx 4^V$ вариантов сверху, вроде снизу отличается только число в основании).

Отступление. Комбинаторные и информационные оценки для построения нижних оценок. Для построения простейшей верхней нужно использовать схему: отправить информацию в корень, вычислить ответ, распространить информацию назад. Конечно, до построения локальной нумерации делать это практически бесполезно — придётся переда-

вать ключи узлов. Ну а вообще, если длина ключей в узлах не превосходит L (для случая локальной нумерации можно применять с $L = \log_2 V$), то:

1. L шагов на задание нумерации рёбер. То есть каждый узел знает для смежных с ним рёбер некоторый идентификатор. За таковой можно принять, например, кодированную пару номеров вершин. В случаи мультирёбер можно использовать тройки, приписывая номер ребра в пучке.

2. $VL + E(2L)$ информации нужно передать в корень.

3. Соответственно, оценка $(VL + E \cdot (2L))/deg(R)$

Обратно к повествованию. Протокол построения столь замечательной нумерации прост: нам нужно во все вершины передать протокол выполнения поиска в ширину из R, пометить при этом для каждой вершины точку её появления в протоколе.

Протокол поиска в ширину - строится так: R пишет 1 столько раз, сколько у него детей. Затем приписывается 0. После этого пишется 1 столько раз, сколько детей у первого из потомков R, потом 0, повторяется для всех вершин на расстоянии один от R. (**Замечание**. Так как мы выполнили поиск в ширину из корня, то расстояние в дереве до корня совпадает с расстоянием до корня в графе.) Повторяем процедуру для всех слоёв.

Например, для полного 2-дерева на 7 вершинах получаем: 110 110110. Можно также заметить, что число символов в протоколе равно $|List(G)| + 2(|V| - |List(G)| - 1)$. Вычли 1 из-за корня, который не пораждает символов в сертификате. $List(G)$ — множество листовых вершин.

Однозначное соответствие между протоколами поиска в ширину из корня и укоренёнными деревьями очевидна. Чтобы не путать слово “протокол” относительно поиска в ширину и сетевой части, для первого будем использовать слово **“сертификат”**.

Обычно, для описания укоренённых деревьев используют протоколы поиска в глубину, но в распределённом случае, как это обычно и случается, удобнее поиск в ширину.

Теперь лишь заметим, что для того, чтобы указать вершине её место в дереве будем поступать очень просто: пусть у узла v родителем в дереве является u, у узла u есть нумерация дочерних по дереву узлов, пусть v в этом списке номер x; в протоколе есть подстрока, отвечающая перечислению дочерних узлов вершины u; наш протокол устроен так, что сертификат передаётся от родительских узлов дочерним; при передачи сертификата из узла u в узел v заменим 1 с номером x в определённой ранее подстроке на символ

2. То есть при передачи информации от родителя дочернему узлу в сертификате меняется ровно один символ, отвечающей дочерней вершине.

Заметим, что в таком способе передачи, локальный номер вершины = (число единиц до встречи двойки + 2): так как R имеет номер 1. Например, в приведённом выше примере от корня R первому своему дочернему узлу (с глобальным номером 2) будете передан сертификат: 210 110110. А самому "левому"листовому узлу будет передано: 110 210110

Теперь самое главное: как собирать и распространять сертификат. Тут срабатывает простая идея рекурсивного выполнения протокола.

Все вершины собирают свой “относительный сертификат”, то есть “сертификат” по поиску в ширину по их поддереву и отправляют наверх на каждом такте очередной символ из сертификата. **Главное:** если сертификат не отослан полностью, то очередной символ нам всегда известен (то есть мы знаем, что отсылать на текущей стадии). Действительно: базой является то, что узлы знают число своих детей. На каждом такте каждый из детей, который не закончил распространение, пришлёт по одному очередному символу своих локальных сертификатов (они могут это сделать в силу индукции с базой — листовой узел). Значит, число известных символов локального сертификата растёт хотя бы на 1 (до тех пор, пока локальный сертификат не собран полностью), а число отправленных увеличивается ровно на 1.

Это же рассуждение обеспечивает то, что на такте t от начала выполнения протокола нумерации R знает не менее $\min(Sert(G), deg(R) + 1 + t)$ префиксных символов сертификата, где $Sert(G)$ — длина полного сертификата. А отсылает корень по одному символу в такт своим прямым потомкам. Таким образом, начиная с такта номер 2, узлы на расстоянии 1 знают $t - 1$ префиксных символов сертификата (глобального).

Далее по индукции получаем, что узлы на расстоянии d от корня на такте $t > d + 1$ знают $t - d$ префиксных символов глобального сертификата.

Итого: $2|V| + ext_G(R)$

Замечание. Заметим, что в таком построении происходит трата времени, на распространение результата от корня. Второе слагаемое, кажется, можно убрать, если каждый узел будет сначала распространять локальный сертификат, потом сертификат более высокого уровня с указанием куда вклеить предыдущий, и так далее, пока не будет распространён главный сертификат. Тут наверное нужно додумать, вдруг ошибка. Ну и не факт что нужно, ибо второе слагаемое поглощается первым. Хотя в качестве упражнения и интересного факта — почему нет.

Замечание. $ext_G(R)$ в нашем случае есть высота дерева. Будем использовать обозначе-

ние $h(G)$ — высота фиксированного (у нас оно всегда одно) дерева относительно корневой вершины R.

1 Мосты

Задача ставится так: есть сеть с неизвестной топологией. Пусть в ней выбран лидер и построено остовное дерево от корня. Пусть более того, выполнена нумерация из первого пункта. (То есть стоимость этих предварительных стадий в расчёт мы брать пока не будем, если нужно, то их всегда можно прибавить). Требуется, чтобы каждая вершина знала, какие из смежных с ней рёбер являются мостами в графе.

Замечание Наличие дерева, и тем более дерева от корня, вообще ни как не влияет на наши оценки, разве что нужно либо таки выполнить поиск в ширину из корня в начале, заплатив $O(|V|)$, либо работать с другим (где расстояние по дереву до R и расстояние по графу до R могут не совпадать) деревом, заменяя $ext_G(R)$ на $diam(G)$. В общем всё это не важно с точки зрения асимптотики, так как укладывается в $O(|V|)$.

Замечание Ещё я пользуюсь тем, что протокол стартует одновременно. В случае стадий дерева и нумерации это может требовать дополнительной синхронизации. Но, вроде. на асимптотику тоже не влияет.

Не будем писать воду, почему эта задача интересна, а напомним оценку тривиального протокола “сбора информации в точку”: $O(E \log V)$ в худшем случае. Берёмся решить задачу за

$$O(h(G) \log h(G) + \log V) \leqslant O(V \log V)$$

тактов времени.

Базовое утверждение 1: мостовыми узлами могут быть те рёбра, что вошли в остовное дерево.

Базовое утверждение 2: если две вершины соединены ребром, не вошедшим в остовное дерево, то на пути (единственном!) в дереве между этими вершинами нет ни одного моста.

Базовое утверждение 3: если ребро дерева не является мостом, то найдутся две вершины графа, что ребро лежит на их пути, между этими двумя вершинами есть ребро (не входящее в остовное дерево).

Из перечисленных очевидных следствий определения моста мы получаем прото-алгоритм: надо взять все рёбра не из дерева, найти ближайшего общего предка (по рёбрам дерева), пути от вершин ребра до этого общего предка образуют две половинки пути из базового

утверждения 2. Все рёбра на этих половинках пути помечаем как не-мостовые. Все рёбра, которые не будут помечены как не мостовые после рассмотрения всех рёбер являются мостами.

Итоговый протокол:

1. По всем рёбрам не из дерева узлы передают свой номер (локальный, естественно). Тратим на это $\log V$

2. После того как вершина получила номера всех соседей, она по номеру может определить положение этих соседей в остовном дереве, а значит найти номер наиближайшего общего предка. Но заметим, что тут всё ещё проще: наибольший общий предок всегда является предком вершины, значит достаточно вычислить на сколько уровней от вершины нужно подняться вверх по родительским связям, чтобы достичь общего предка.

   Пусть вершина v с ребром $e = (u, v)$ получила номер. Далее, зная глобальный сертификат можно найти номер w_e — наибольшего общего предка узлов u, v в данном дереве. Далее v вычисляет $d_{v,e} = h(v) - h(w_e)$ - разницу в высотах вершин w и v.

   Далее вычисляется $d_v = \max\limits_{e:v \in e} d_{v,e}$.

   Из информации, полученной от рёбер, смежных с v следует, что нужно поднятся от вершины v на d_v уровней и пометить все пройденные рёбра как не являющиеся мостами.

3. Передаём родителю d_v за $\log h(G)$ тактов.

4. Вычисляем $d'(v)$ — максимум из d_v и $\{d_u - 1 | u \in U(v)\}$, где $U(v)$ — множество дочерних в дереве узлов для вершины v.

   Отправляем $d'(v)$ родителю.

   Повторяем эту стадию $h(G)$ раз.

Нужно всего лишь заметить, что последний пункт обеспечивает корректную передачу информации вверх по дереву.

Идеей унарной нумерации улучшаем оценку до

$$O(h(G) + \log V).$$

Вершина передаёт $h(G)$ унарной записью по одному символу за такт, агрегаций информации от нижних уровней: передаём от дочернего узла строку длины на один меньше (начинаем ретрансляцию после получения второй единицы).

Работа поддержана грантом РФФИ 14–01—93107

Работа поддержана грантом РФФИ 14–01–00641

Список литературы

[1] I. M. Khuziev, M. N. Vyalyi. Distributed communication complexity of spanning tree construction. Problems of Information Transmission, 2015, Vol. 51, No. 1, pp. 44–60.

[2] Dinitz Y., Moran Sh., Rajsbaum S. Bit complexity of breaking and achieving symmetry in chains and rings // Journal of the ACM, 2008. Vol. 55, no 1. P. 1–28.

[3] N.A. Lynch. Distributed algorithms. Morgan Kaufmann Publ., 1996.

[4] Perlman R. An algorithm for distributed computation of a spanning tree in an extended LAN // ACM SIGCOMM Computer Communication Review 15 (4), 1985. P. 44–53.

[5] Peleg D., Rubinovich V. A near-tight lower bound on the time complexity of distributed MST construction // SIAM J. on Computing, 2000. Vol. 30. P. 1427–1442.

[6] Itai A., Rodeh M. Symmetry breaking in distributed networks. // Information and Computation, 1990. Vol. 88, no 1. P. 60–87.

[7] Wattenhofer R. Principles of distributed computing. 2014.
`http://dcg.ethz.ch/lectures/podc_allstars/lecture/podc.pdf`