УДК 519.711.74

РЕАЛИЗАЦИЯ РЕКУРСИВНЫХ ЗАПРОСОВ В ДИНАМИЧЕСКОЙ АССОЦИАТИВНОЙ РЕСУРСНОЙ СЕТИ

Л.Ю. Жилякова (*zhilyakov@aaanet.ru*)
Педагогический институт ЮФУ, г. Ростов-на-Дону

Ключевые слова: ассоциативная память, сеть, проводимость, ресурс, рекурсивный запрос

Предложенная модель представляет собой двусторонний взвешенный граф с петлями; вершины обозначают сущности предметной области, ребра отвечают за ассоциативные связи между ними. При обращении с запросом в некоторое множество вершин передается ресурс, распространяющийся по сети в дискретном времени. Веса ребер обозначают их пропускную способность. Для управления движением ресурса используются рекурсивные запросы.

Введение

В работе предложена структура памяти, названная динамической ассоциативной ресурсной сетью [Кузнецов, 2009]. Способ хранения информации в ней таков, что наиболее часто используемые данные оказываются и наиболее доступными. Эта особенность позволяет значительно упростить обработку частых запросов; запросы к мало востребованным данным требуют для обработки больше времени.

В основе ассоциативной ресурсной сети лежит несимметричная двусторонняя ресурсная сеть [Кузнецов, Жилякова, 2010], которая принципиально отличается от классических потоковых моделей, (см. например, [Форд, Фалкерсон, 1966]).

Быстрый доступ к часто используемым данным обеспечивается следующими особенностями сети: каждая вершина обладает способностью хранить неограниченное количество ресурса, отвечающего за ее яркость, которая повышает ее доступность при поиске; каждое ребро сети имеет проводимость, соответствующую силе ассоциативной связи между сущностями.

Разность между суммарной входной и выходной проводимостью вершины задает ее дивергенцию. Некоторые вершины с неотрицательной дивергенцией играют роль аттракторов, которые способны аккумулировать значительную часть яркости при выполнении запросов.

При ненаправленном распространении ресурса по сети во время поиска зона яркости увеличивается равномерно во все стороны. Чтобы локализовать распространение яркости и задать направление движения пятна яркости по сети, используются рекурсивные запросы.

1. Ресурсная сеть. Основные определения

Ресурсная сеть представляет собой ориентированный граф, вершинам v_i которого приписаны неотрицательные числа $q_i(t)$, называемые *ресурсами*, и изменяющиеся в дискретном времени t.

Сеть двусторонняя, т.е., если существует ребро (v_i, v_j), то существует и противоположно ориентированное ребро (v_j, v_i).

Ребра графа взвешены – каждому ребру (v_j, v_i) приписано неотрицательное число r_{ij}, называемое его *проводимостью*. Проводимость ребра отвечает за его способность передавать *ресурс* от вершины к вершине. Каждая вершина имеет петлю (v_i, v_i) с проводимостью, равной r_{ii}. Петля отвечает за ресурс, который будет возвращен в вершину в процессе его перераспределения. Вершины обладают способностью удерживать неограниченное количество ресурса.

Матрицей проводимости называется матрица $R = \|r_{ij}\|_{n \times n}$, где r_{ij} – проводимость ребра (v_i, v_j).

Если пары $\langle(v_i, v_j), (v_j, v_i)\rangle$ не существует, $r_{ij} = 0$ и $r_{ji} = 0$.

Из определения ресурсной сети вытекают следующие свойства матрицы R:

1. R – неотрицательная матрица: $\forall\ i, j\ \ r_{ij} \geq 0$
2. $\forall\ i\ \ r_{ii} > 0$
3. $\forall\ i, j\ \ (r_{ij} > 0 \Leftrightarrow r_{ji} > 0)$

Суммарной проводимостью сети, r_{sum}, называется сумма проводимостей всех ее ребер:

$$r_{sum} = \sum_{i=1}^{n}\sum_{j=1}^{n} r_{ij}\ .$$

Суммарную проводимость входных ребер вершины с номером i будем называть ее *входной проводимостью*; суммарную проводимость выходных ребер назовем *выходной проводимостью* и обозначим их через r_i^{in} и r_i^{out} соответственно. Проводимость петли входит в обе суммы.

$$r_i^{in} = \sum_{j=1}^{n} r_{ji}\ ;\ r_i^{out} = \sum_{j=1}^{n} r_{ij}\ .$$

Сеть функционирует в дискретном времени t.

Распределение ресурса в сети происходит по одному из двух правил, выбор которых зависит от величины ресурса в вершинах. В момент $t + 1$ вершина v_i в ребро, соединяющее ее с вершиной v_k, отдаст:

правило 1: r_{ik} единиц ресурса, если $q_i(t) > r_i^{out}$;

правило 2: $\frac{r_{ik}}{r_i^{out}} q_i(t)$ в противном случае.

По правилу 1 вершина отдаст за такт работы всего: $r_i^{out} = \sum_{j=1}^{n} r_{ij}$ ресурса. По правилу 2 вершина отдает весь свой ресурс. Если ресурс в вершине равен выходной проводимости вершины: $q_i(t) = r_i^{out}$, – то применение обоих правил даст один и тот же результат.

Суммарный ресурс, находящийся во всех вершинах, обозначим через W. В сети выполняется *закон сохранения*: при ее функционировании ресурс не поступает извне и не расходуется: $\forall t \sum_{i=1}^{n} q_i(t) = W$.

Ресурсная сеть называется *однородной*, если проводимости всех ребер одинаковы. В противном случае сеть называется *неоднородной*.

Ресурсная сеть называется *симметричной*, если симметрична ее матрица проводимости.

В симметричных сетях у каждой вершины входная и выходная проводимости совпадают. Ресурсная сеть называется *квазисимметричной*, если

$$\forall i: \; r_i^{in} = r_i^{out} \qquad (1)$$

Если сеть не квазисимметрична, в ней существует хотя бы пара вершин, для которых выполнится: $r_i^{in} - r_i^{out} \neq 0$. Для произвольной вершины v_i обозначим эту разность через Δr_i: $\Delta r_i = r_i^{in} - r_i^{out}$. Тогда все вершины сети можно разделить на три класса:

1) *вершины-приемники*, для которых $\Delta r_i > 0$;
2) *вершины-источники*, для которых $\Delta r_i < 0$;
3) *нейтральные вершины*, для которых $\Delta r_i = 0$.

В симметричных и квазисимметричных сетях все вершины *нейтральны*.

Сеть будем называть *несимметричной*, если она не удовлетворяет условию квазисимметричности (1). Несимметричная сеть обладает как минимум одним источником и одним приемником.

Состоянием $Q(t)$ сети в момент t будем считать вектор $Q(t) = (q_1(t), \ldots, q_n(t))$, состоящий из значений ресурсов в каждой вершине.

Состояние $Q(t)$ называется *устойчивым*, если $Q(t) = Q(t + 1) = Q(t + 2) = Q(t + 3) = \ldots$

Состояние $Q^* = (q_1^*, \ldots, q_n^*)$ называется *асимптотически достижимым* из состояния $Q(0)$, если для любого $\varepsilon > 0$ существует t_ε такое, что для всех $t > t_\varepsilon$

$|q_i^* - q_i(t)| < \varepsilon$, $i = 1, 2, \ldots, n$.

Состояние сети называется *предельным*, если оно либо устойчиво и достижимо из $Q(0)$ за конечное время, либо асимптотически достижимо из $Q(0)$.

Распределение ресурса в сети представляет собой Марковский процесс. На основании свойств регулярных стохастических матриц [Гантмахер, 2004] доказана сходимость процесса распределения ресурса для любой конфигурации сети.

2. Динамическое изменение топологии в ассоциативной ресурсной сети

Динамическая ассоциативная ресурсная сеть представляет собой ресурсную сеть, каждая вершина которой обозначает некоторую сущность: объект, атрибут или класс, – и имеет имя v_i из множества имен V. Рёбра сети обозначают ассоциации между сущностями.

Двусторонность сети отвечает за существование как прямой, так и обратной ассоциации (не всегда одинаковой силы).

Ресурс в ассоциативной сети отвечает за яркость понятия в памяти.

В неактивном состоянии сеть не имеет яркости. Яркость поступает в сеть только на стадии обращения к ней с запросом.

Запрос – это задание начального множества вершин с указанием количества ресурса (начальной яркости) в каждой из них.

Следуя правилам 1 и 2 из п.1, яркость начинает распространяться по сети от каждой вершины по всем инцидентным ребрам. Сеть функционирует в быстром времени t.

Обращение к сети с каждым новым запросом, так же как и наполнение ее новой информацией, изменяет топологию сети и, соответственно, влияет на результаты будущих запросов [Жилякова, 2010].

Занесение новой информации и обращение с запросами происходит в медленном времени τ. Одному такту τ соответствует выполнение одного запроса или занесение новой единицы информации.

Информация заносится в сеть минимальными структурными единицами. Они могут быть двух типов:

1) двусторонняя пара, связывающая две существующие вершины;
2) новая вершина с петлей и двусторонняя пара, связывающая эту

вершину с уже имеющейся.

На каждом такте τ все ребра, которые участвовали в передаче яркости, включая петли вершин, увеличивают проводимость на некоторую величину ρ(τ). Таким образом, чем чаще вершины участвуют в запросах, тем больше проводимость их петель и тем больше яркости они могут удержать. Чем выше проводимость ребер, ведущих к вершине, тем больше яркости она получит из других областей сети. Вершины, способные вблизи предельного состояния удержать или накопить яркость, превышающую их выходную проводимость, будем называть *потенциальными аттракторами*.

3. Управление движением ресурса

3.1. Реализация рекурсивных запросов

В больших сетях яркость может растекаться от каждой вершины неограниченно во все стороны. Чтобы локализовать область поиска и управлять движением «пятна яркости» в сети, используются рекурсивные запросы. Под рекурсивным запросом будем понимать многократный запрос, входное множество вершин которого изменяется в зависимости от полученного ответа по одному из наперед заданных правил. Глубина рекурсии может варьироваться от запроса к запросу.

Входным множеством вершин очередного запроса в рекурсии не обязательно должно быть выходное множество от предыдущего запроса, то есть множество вершин, входящих в ответ на предыдущий запрос. Каждое новое входное множество состоит из предыдущего входного множества, подвергнутого определенным изменениям. Эти изменения могут быть двух видов.

I. Добавление одной или нескольких вершин из выходного множества предыдущего запроса. Обозначим количество добавляемых вершин через k^+: $k^+ = 1, 2, \ldots l - l^*$, где l – количество вершин в ответе, l^* – мощность пересечения входного и выходного множества. Этот тип изменений соответствует ситуации, когда самый ожидаемый (вероятный) ответ на поставленный запрос является удовлетворительным, но его нужно расширить и/или уточнить.

В таком случае из множества вершин, входящих в ответ, выбираются некоторые дополнительные вершины (вместе со своим ресурсом) и добавляются в исходное множество вершин, участвующих в запросе. После чего снова происходит увеличение проводимостей соответствующих ребер, и весь процесс перераспределения ресурса повторяется заново – для нового начального состояния.

Количество новых запросов, которые возможно сгенерировать таким способом, составит: $N^+ = \sum_{i=1}^{l-l^*} C_i^{l-l^*}$

II. Удаление одной или нескольких вершин из входного множества предыдущего запроса. Количество удаляемых вершин обозначим через k^-: $k^- = 1, 2, \ldots m$.

Этот вид изменений входного множества может преследовать различные цели.

II a) Удаляются вершины из пересечения множеств вопрос-ответ, т.е. входного и выходного множеств предыдущего запроса. Такие изменения предназначены для отсекания самого очевидного ответа и поиска других, менее очевидных. То есть, чтобы получить заведомо «нетривиальный» ответ, сначала нужно узнать ответ тривиальный, и только затем его отсечь.

Количество удаляемых вершин может изменяться от 1 до l^*, где l^* – мощность пересечения множеств запроса и ответа.

Общее количество запросов, сгенерированных этим правилом, будет: $N_a^- = \sum_{i=1}^{l^*} C_i^{l^*}$

II b) Удаляются вершины из предыдущего входного множества, которых не оказалось в множестве выходном. Эти изменения производятся, если нужно создать длинные ассоциативные цепочки, – создать движение яркости сквозь сеть. Чем меньше вершин из предыдущего входного множества перейдет в следующее, тем быстрее будет передвигаться «пятно яркости» по сети, охватывая каждый раз новые участки. Если же основную массу из входного множества не трогать, пятно яркости будет блуждать около фиксированного центра. С помощью этого правила можно составить $N_b^- = \sum_{i=1}^{m-l^*} C_i^{m-l^*}$

Итоговые формулы для N^+ и N^-: $N^+ = \sum_{i=0}^{l-l^*} C_i^{l-l^*}$, $N^- = \sum_{i=0}^{m} C_i^m$

Комбинируя все возможные сочетания добавления и удаления вершин, получим $N = \sum_{i=0}^{l-l^*} C_i^{l-l^*} \cdot \sum_{i=0}^{m} C_i^m$ различных множеств, каждое из которых претендует на то, чтобы быть входным множеством запроса на следующем шаге рекурсии.

3.2. Операции над графами

При описании входных множеств рекурсивных запросов фрагменты сети рассматривались как множества вершин, имеющих яркость. Т.е. нигде не учитывалась структура графов, содержащих эти вершины.

Перейдем к графам и опишем последовательность действий, которые необходимо произвести, чтобы добавить новые вершины или удалить уже существующие из подграфов с известной топологией.

Ответом на единичный запрос к сети является некоторый фрагмент сети с предельным распределением яркостей. Иными словами, это подграф, каждой вершине которого сопоставлено некоторое число, обозначающее находящееся в ней количество ресурса.

Эти числовые значения задают частичный порядок на множестве вершин. Пронумеруем вершины графа, независимо от его конфигурации, от более ярких к менее ярким. Если фрагмент содержит несколько вершин с одинаковой яркостью, упорядочим это подмножество в соответствии с некоторым заданным правилом, например, в лексикографическом порядке имен. Таким образом, получим линейный порядок на множествах вершин входного и выходного подграфов.

Входной граф запроса на шаге с номером i обозначим $G_{In}(i)$. Выходной граф этого же запроса обозначим $G_{Out}(i)$. Вершины этих графов представляют собой пары: имя+яркость.

Если в качестве ответа на запрос брать весь подграф, то будет выполняться следующее вложение: $G_{In}(i) \subseteq G_{Out}(i)$.

Если же отсекать вершины с самой низкой яркостью (порог яркости может быть относительной величиной и зависеть от суммарной яркости вершин фрагмента), то вложение уже, возможно, не будет иметь места. Более того, в таком случае множества вершин графов $G_{In}(i)$ и $G_{Out}(i)$ могут даже не пересекаться.

Введем оператор T, действующий на графах. $T(G)$ – транзитивное замыкание графа G. Действует он следующим образом: для любого графа G $T(G)$ – такой граф, что для любых двух вершин верно: если есть путь любой длины из вершины v_i в вершину v_j, то есть и двусторонняя пара $<(v_i, v_j),(v_j, v_i)>$, связывающая эти вершины напрямую.

Если граф G состоит из одной компоненты связности, то $T(G)$ – полный граф.

Как только на шаге i по входному графу $G_{In}(i)$ получается выходной граф $G_{Out}(i)$, к объединению графов $G_{In}(i) \cup G_{Out}(i)$ применяется оператор T.

Получается граф $G_{InOut}(i) = T(G_{In}(i) \cup G_{Out}(i))$. Множество его вершин – это по-прежнему вершины графа $G_{In}(i) \cup G_{Out}(i)$.

Проводимость каждого вновь созданного ребра рассчитывается как среднее геометрическое проводимостей ребер, составляющих цепочку.

Если из вершины в вершину существует несколько путей, выбирается максимальное из средних геометрических их проводимостей.

При построении замыкания проводимости рассчитываются только по «опорным ребрам», т.е. тем, которые существовали в графе $G_{In}(i) \cup G_{Out}(i)$.

Введем еще два оператора, действующие на графах: A – добавление вершин к графу; E – удаление вершин из графа.

Добавление и удаление вершин, разумеется, производится со всеми инцидентными ребрами, соединяющими вершину со всеми смежными вершинами рассматриваемого графа.

Оператор *A* применяется к двум подграфам, – возможно, лежащим в одной компоненте связности. Запись: $A_{j_1,\ldots,j_k}(G_1, G_2)$ означает, что из графа G_2 в граф G_1 будет добавлено k вершин с номерами $j_1, \ldots, j_k$ вместе со всеми ребрами, соединяющими эти вершины с вершинами G_1.

Тогда на шаге $i + 1$ добавление во входной подграф вершин с номерами $j_1, \ldots, j_k$, где $k \le l - l^*$ ($l - l^*$ – мощность множества вершин графа $G_{Out}(i) \backslash G_{In}(i)$), запишется в следующем виде:

$$G_{In}(i+1) = A_{j_1,\ldots,j_k}(G_{In}(i), G_{Out}(i)).$$

Ребра в полученный граф добавляются из транзитивного замыкания $G_{InOut}(i) = T(G_{In}(i) \cup G_{Out}(i))$, чтобы не были потеряны ассоциации, если прервались их цепочки.

Оператор *E* применяется к одному графу.

Запись: $E_{j_1',\ldots,j_h'}(G)$ означает, что из графа G будет удалено h вершин с номерами $j_1', \ldots, j_h'$ вместе с их инцидентными ребрами.

Тогда на шаге $i + 1$ удаление из графа $G_{In}(i)$ вершин с номерами $j_1', \ldots, j_h'$, где $h \le m$ (m – мощность множества вершин $G_{In}(i)$), запишется в следующем виде:

$$G_{In}(i+1) = E_{j_1',\ldots,j_h'}(G_{In}(i)).$$

Будем считать, что сначала к графу $G_{In}(i)$ применяется оператор E, а затем к результату – оператор A.

Операторы не коммутируют, порядок их применения важен.

Заметим, что индексы $j_1', \ldots, j_h'$ – это номера вершин графа $G_{In}(i)$, а $j_1, \ldots, j_k$ – номера вершин $G_{Out}(i)$. Поэтому если сначала применить оператор A, то в $G_{In}(i)$ добавится k вершин, после чего всё множество вершин переупорядочится. Тогда набор индексов $j_1', \ldots, j_h'$ оператора E будет обозначать совсем не те вершины, что были на этих местах до применения оператора A.

Таким образом, на шаге $i + 1$ входной граф запроса находится по следующей рекуррентной формуле:

$$G_{In}(i+1) = A_{j_1,\ldots,j_k}(E_{j_1',\ldots,j_h'}(G_{In}(i))).$$

Непосредственно из этой формулы вытекает, что каждый новый входной подграф однозначно определяется входным и выходным подграфами на предыдущем шаге и парой последовательностей натуральных чисел переменной длины: ($\{j_1', \ldots, j_h'\}$; $\{j_1, \ldots, j_k\}$).

Например, пара ({1, 3, 10}; {8, 12}) означает, что из графа, который был входным на прошлом шаге, нужно удалить вершины с номерами 1, 3 и 10 и присоединить вершины с номерами: 8, 12 из графа выходного. Затем заново перенумеровать вершины полученного графа.

Любой рекурсивный запрос будет однозначно идентифицироваться цепочкой таких пар множеств индексов. Длина цепочки совпадает с глубиной рекурсии.

Заключение

В работе описана структура и основные свойства динамической ассоциативной ресурсной сети, принципы и алгоритм ее построения; управление распространением ресурса, отвечающего за яркость понятий, при выполнении запросов.

Топология изменяется автоматически таким образом, что наиболее востребованная информация оказывается наиболее доступной. Ассоциативность сети заключается не только в адресации по содержанию, но и в структуре взаимосвязей моделируемой предметной области, в которой близость понятий определяется не только и не столько семантикой, сколько самим функционированием сети, т.е. пользовательскими запросами и ответами на них.

Управление движением ресурса сквозь сеть осуществляется как самой топологией сети, которая направляет ресурс по ребрам с большей проводимостью, так и рекурсивным заданием нового входного множества и продолжением поиска в заданном направлении.

Список литературы

[Гантмахер, 2004] Гантмахер Ф.Р. Теория матриц. – М.: Физматлит. 2004.

[Жилякова, 2010] Жилякова Л.Ю. Процессы изменения проводимости в ассоциативной ресурсной сети. // X международная конференция имени Т.А. Таран ИАИ-2010. Киев, «Просвіта», 2010.

[Кузнецов, Жилякова, 2010] Кузнецов О.П., Жилякова Л.Ю. Исследование эргодичности ресурсных сетей с произвольной проводимостью. // X международная конференция имени Т.А. Таран ИАИ-2010. Киев, «Просвіта», 2010.

[Кузнецов, 2009] Кузнецов О.П. Однородные ресурсные сети I. Полные графы. // Автоматика и телемеханика, 2009, № 11.

[Форд, Фалкерсон, 1966] Форд Л.Р., Фалкерсон Д. Потоки в сетях. – М.: Мир, 1966.