Николай Викторович Медведев родился в 1954 г., окончил в 1977 г. МГТУ им. Н.Э. Баумана. Канд. техн. наук, зав. кафедрой "Информационная безопасность" МГТУ им. Н.Э. Баумана. Автор около 50 научных работ в области исследования и разработки защищенных систем автоматической обработки информации.

N.V. Medvedev (b. 1954) graduated from the Bauman Moscow Higher Technical School in 1977. Ph. D. (Eng.), head of "Information Security" department of the Bauman Moscow State Technical University. Author of about 50 publications in the field of study and development of protected systems of automated data processing.

Александр Юрьевич Быков родился в 1969 г., окончил в 1991 г. ВИКИ им. А.Ф. Можайского. Канд. техн. наук, доцент кафедры "Информационная безопасность" МГТУ им. Н.Э. Баумана. Автор около 20 научных работ в области информационной безопасности и исследования систем обработки информации и управления.

A.Yu. Bykov (b. 1969) graduated from the Military Institute for Engineering and Space n.a. A.F. Mozhaiskii in 1991. Ph. D. (Eng.), assoc. professor of "Information Security" department of the Bauman Moscow State Technical University. Author of about 20 publications in the field of study and development of data security and study of systems of data processing and control.

УДК 004.7

С. С. Баскаков

РАСПРЕДЕЛЕННЫЙ АЛГОРИТМ АВТОМАТИЧЕСКОГО ВЫБОРА ОПОРНЫХ УЗЛОВ В БЕСПРОВОДНЫХ МНОГОЯЧЕЙКОВЫХ (MESH) СЕТЯХ

Предложен алгоритм автоматического выбора произвольного числа опорных узлов с различными видами их распределения по площади покрытия беспроводной многоячейковой сети. Полученные оценки сложности по времени и памяти показали, что алгоритм является оптимальным при совместном использовании с протоколами географической маршрутизации по виртуальным координатам.

В последнее время активное развитие получили технологии беспроводных сенсорных сетей, которые имеют множество практических применений, связанных с распределенным сбором, анализом и передачей информации. Беспроводная сенсорная сеть (БСС) представляет собой распределенную, самоорганизующуюся и устойчивую к отказу сеть миниатюрных электронных устройств, обменивающихся информацией по беспроводному каналу связи. Предполагается, что такие сети будут иметь многоячейковую (mesh) топологию и состоять из большого числа (до нескольких десятков тысяч) узлов, которые способны ретранслировать сообщения друг друга.

Во многих задачах в области БСС необходимо наличие в сети определенного числа опорных узлов (beacon, landmark, anchor). Актуальной задачей является маршрутизация пакетов. Для решения этой задачи с помощью одного из наиболее современных методов — географической маршрутизации по виртуальным координатам узлов [1–4] — требуются опорные узлы.

Виртуальные координаты узла представляют собой вектор, элементами которого являются значения минимального числа элементарных передач (hop) между данным узлом и фиксированным набором опорных узлов. Опорные узлы отличаются от обычных только тем, что периодически начинают процесс широковещательной передачи специальных сигнальных пакетов, а обычные узлы эти пакеты только ретранслируют. Анализируя полученные сигнальные пакеты все узлы (как обычные, так и опорные) вычисляют набор значений длин кратчайших путей между ними и опорными узлами. Этот набор называется виртуальными координатами узла.

Виртуальные координаты используются для вычисления виртуального расстояния между узлами с помощью некоторой метрики (например, евклидовой нормы). Заметим, что виртуальное расстояние между узлами не имеет явной физической интерпретации из-за неортогональности системы виртуальных координат. Процесс географической маршрутизации заключается в последовательной минимизации виртуального расстояния до узла-назначения (получателя пакета). При обнаружении локального минимума (ситуация, при которой очередной узел не имеет узла-соседа, расположенного ближе к узлу-назначению) протоколы используют различные варианты бектрекинг-режима, который полностью или с высокой вероятностью гарантирует доставку пакета ценой увеличения длины пути.

В первую очередь на эффективность протокола маршрутизации по виртуальным координатам влияют выбранная метрика виртуальных расстояний, а также число и расположение (распределение) опорных узлов на площади покрытия сети. В настоящей работе рассматривается вопрос размещения опорных узлов.

Проблема выбора опорных узлов заключается в поиске такого их размещения, при котором как можно большая часть пакетов будет доставлена в режиме минимизации виртуального расстояния (greedy – “жадная” маршрутизация), так как найденные в этом режиме маршруты имеют длину, близкую к оптимальной (кратчайшей). При этом желательно задействовать как можно меньшее число опорных узлов для снижения накладных расходов на хранение и передачу служебной информации. Кроме того, в самоорганизующейся сети назначение

опорных узлов должно выполняться автоматически с помощью некоторого распределенного (децентрализованного) алгоритма, сложность (по памяти) которого не должна зависеть от общего числа узлов в сети для обеспечения масштабируемости системы.

В протоколе Beacon Vector Routing [1] в качестве опорных используется r узлов, случайно выбранных в сети. Далее в процессе маршрутизации задействуются только k из них ($k \leq r$), которые наиболее близко расположены к узлу-назначению. Такая схема отличается простотой, но протокол маршрутизации оказывается чувствительным к полученному случайным образом распределению опорных узлов, поэтому приходится применять слишком большое общее число опорных узлов r (порядка нескольких десятков), что увеличивает накладные расходы на хранение и передачу виртуальных координат.

В протоколе Virtual Coordinate Assignment Protocol [2] сделана попытка создать алгоритм, который автоматически будет выбирать в качестве опорных узлы, находящиеся на периферии сети. Однако предложенный алгоритм рассчитан на выбор только трех опорных узлов, в то время как на практике необходимо четыре и более.

В первоначальной версии протокола Logical Coordinate Routing (LCR) [3] предполагалось, что опорные узлы будут вручную размещены на границе (по периметру) сети на этапе развертывания системы. Ручное назначение опорных узлов усложняет настройку крупной сети, а в некоторых приложениях и вовсе невозможно, поэтому в последующей работе по LCR [4] разработан алгоритм автоматического выбора опорных узлов, распределяющий опорные узлы по периферии сети. Недостатком алгоритма [4] является зависимость сложности по памяти от масштаба сети.

Таким образом, в известных работах (кроме работы [4]) распределенные алгоритмы не позволяют выбрать произвольное наперед заданное число опорных узлов в беспроводной многоячейковой сети. В настоящей работе указанный недостаток устранен и предложен алгоритм для автоматического выбора опорных узлов как при начальной инициализации сети, так и при возможном выходе из строя одного или нескольких опорных узлов в процессе эксплуатации системы. В зависимости от используемой функции голосования результатом работы алгоритма являются равномерно распределенные по сети или находящиеся на периферии сети опорные узлы. При этом сложность алгоритма ниже по сравнению с известным аналогом из работы [4].

Постановка задачи. Рассмотрим беспроводную многоячейковую сеть из множества узлов V, общее число которых равно $|V| = n$. Считаем, что топология сети не изменяется, узлы случайно и равномерно распределены на двумерной плоскости, площадь покрытия сети равна

A, м^2, а дальность радиосвязи между узлами составляет r, м. Тогда плотность сетевого окружения (среднее число соседей у каждого узла) равна $\rho = n\pi r^2/A$ узлов. Обозначим через d диаметр сети — максимальное расстояние между всевозможными парами узлов, измеренное как длина кратчайшего пути между ними. Диаметр сети будем оценивать по упрощенной формуле

$$d \approx \frac{\sqrt{A}}{r} = \sqrt{\frac{\pi n}{\rho}}. \qquad (1)$$

Предположение о распределении узлов на плоскости выбрано ради наглядности. При имеющем место на практике распределении узлов в трехмерном пространстве изменяется только вид выражений, описывающих связь n, ρ и d, а остальные рассуждения остаются прежними.

Требуется разработать алгоритм выбора заданного числа опорных узлов n_b ($n_b \le n$) из исходного множества узлов V. Узлы из полученного в результате работы алгоритма множества опорных узлов V_b ($V_b \subseteq V$) должны быть распределены в пространстве по одному из двух вариантов:

- равномерно распределены по сети на максимальном расстоянии относительно друг друга;
- распределены по периферии (на границе) сети.

Нельзя утверждать, что указанные распределения опорных узлов являются наилучшими из всех возможных. Но в работах [4, 5] показано, что такие распределения снижают число совпадающих виртуальных координат и, следовательно, уменьшают вероятность появления локального минимума. Более подробное исследование влияния распределения опорных узлов на эффективность маршрутизации по виртуальным координатам планируется в дальнейших работах.

Алгоритм должен быть распределенным и иметь сложность по времени и по памяти $O(d)$ и $O(\rho)$ соответственно при условии, что $n_b \ll n$ и n_b не зависит от n. Под сложностью по времени понимается общее время выполнения алгоритма, а под сложностью по памяти — требование к объему памяти каждого из узлов сети.

При любом протоколе маршрутизации по виртуальным координатам (независимо от способа размещения опорных узлов) требуется время $O(d)$ для вычисления узлами своих виртуальных координат и объем памяти $O(\rho)$ для хранения узлами координат своих ближайших соседей. Следовательно, если время выполнения некоторого алгоритма выбора опорных узлов составляет $O(d)$, используется информация только о соседних узлах в объеме $O(\rho)$ и при этом по завершении работы алгоритма все виртуальные координаты узлов вычислены, то

такой алгоритм для применения в протоколах маршрутизации по виртуальным координатам — оптимальный.

Описание алгоритма. Для простоты изложения сначала опишем общую схему алгоритма, состоящую из трех этапов.

Выбор узла-инициатора. На первом этапе случайным образом выбирается один узел-инициатор, который будет, как и обычный опорный узел, периодически передавать сигнальные пакеты, чтобы остальные узлы сети могли вычислить расстояние до него. Здесь и далее под расстоянием между узлами понимается длина кратчайшего пути в количестве элементарных передач.

Выбор узла-инициатора может быть выполнен с помощью следующей известной процедуры. Каждый узел при включении запускает таймер со случайным значением времени срабатывания. Если до истечения тайм-аута узел не получил информации о том, что в сети уже есть узел-инициатор, то он сам назначает себя узлом–инициатором. При этом включение узлов может происходить как одновременно, так и в разные моменты времени (например, последовательно).

Случайная величина тайм-аута необходима для уменьшения вероятности того, что несколько узлов одновременно назначат себя узлами–инициаторами. При возникновении такой ситуации конфликт может быть разрешен различными способами. Например, с учетом уникальности идентификаторов (адресов) узлов приоритет отдается узлу с наименьшим (или наибольшим) адресом.

Выполнение первого этапа продолжается в течение интервала времени t, которого должно быть достаточно для вычисления всеми узлами сети длины кратчайшего пути до узла-инициатора. Минимальное допустимое значение t равно задержке распространения сигнального пакета от узла-инициатора до остальных узлов сети, т.е.

$$t = dt_b, \tag{2}$$

где t_b — период передачи сигнальных пакетов.

Выбор первого опорного узла. По завершении предыдущего этапа алгоритма все узлы сети знают свое расстояние до узла-инициатора. Тогда первым опорным узлом b_1 назначается узел, находящийся на максимальном расстоянии от узла-инициатора, т.е.

$$b_1 = \operatorname*{argmax}_{v \in V} h_0(v),$$

где $h_0(v)$ — расстояние между узлом v и узлом-инициатором.

Выбранный узел начинает функционировать в качестве опорного, а узел-инициатор становится обычным узлом при получении первого сигнального пакета от b_1.

Выбор остальных опорных узлов. Выбор последующих опорных узлов основан на использовании функции голосования, значение которой для узла определяет его приоритет в получении статуса опорного узла. Функция голосования может иметь различный вид в зависимости от вида распределения, которое необходимо получить. В настоящей работе предложены две функции голосования, позволяющие получить указанные в постановке задачи распределения опорных узлов (равномерное и по границе соответственно),

$$f_{\min}(\vec{v}, B) = \min_{i \in B} v_i \tag{3}$$

и

$$f_{\text{prod}}(\vec{v}, B) = \prod_{i \in B} v_i, \tag{4}$$

где $\vec{v}$ — вектор виртуальных координат узла $v \in V$; v_i — i-я координата узла v (длина кратчайшего пути до i-го опорного узла); B — множество номеров (индексов) выбранных (активных) опорных узлов.

В функциях голосования (3) и (4) используются компоненты вектора виртуальных координат $\vec{v}$, соответствующие только активным опорным узлам на момент расчета числа "голосов".

Выбор второго и последующих опорных узлов выполняется последовательно с помощью итерационных вычислений функции голосования. Очередным k-м опорным узлом назначается узел, имеющий максимальное число "голосов",

$$b_k = \operatorname*{argmax}_{v \in V} f(\vec{v}, B_k), \quad 2 \le k \le n_b. \tag{5}$$

Множество номеров активных опорных узлов на k-й итерации алгоритма равно

$$B_k = \{1, 2, \ldots, k-1\}.$$

В результате работы алгоритма будет сформировано искомое множество опорных узлов $V_b = \{b_1, b_2, \ldots, b_{n_b}\}$.

Если в выражении (5) использовать функцию (3), то выбранные опорные узлы будут равномерно распределены по сети, если (4) — по границе сети.

Возможна ситуация, при которой для нескольких узлов функция голосования принимает максимальное значение. Этот конфликт также может быть разрешен на основе адресов узлов по описанному ранее принципу.

Замена опорного узла. При эксплуатации сети возможен выход из строя одного или нескольких опорных узлов. Тогда поиск замены для опорного узла с порядковым номером m также выполняется по выражению (5) при $B = \{1, 2, \ldots, m-1, m+1, \ldots, n_b\}$.

Распределенная реализация алгоритма. Согласно распределенной реализации алгоритма каждый узел $v \in V$ выполняет одинаковый набор действий, т.е. какого-либо централизованного управления не требуется. Именно этот вариант алгоритма предназначен для применения на практике.

Инициализация. После включения каждый узел $v \in V$ выполняет следующую последовательность операций:

- присвоение вектору виртуальных координат начального значения $\vec{v} = \{v_i = \infty\}_{i=1}^{n_b}$;
- запуск процесса «локальный обмен сигнальными пакетами» с периодом t_b;
- запуск процесса «выбор опорных узлов» с периодом t_{bs}.

Процесс "локальный обмен сигнальными пакетами" заключается в том, что каждый узел с периодом t_b передает широковещательный пакет, в котором содержатся его собственные координаты и координаты узлов из его таблицы сетевого окружения (таблица со служебной информацией о ближайших соседях узла). В ходе такого обмена узлы поддерживают в актуальном состоянии информацию о соседних узлах и обновляют свои виртуальные координаты по описанной далее процедуре.

Процесс "выбор опорных узлов" заключается в том, что с периодом t_{bs} узел проверяет необходимость назначения одного из опорных узлов. Значение t_{bs} должно быть порядка величины (2), чтобы хватило времени на вычисление всеми узлами сети расстояния до последнего выбранного опорного узла.

Если узел v уже является опорным или узлом-инициатором, то текущая итерация процесса прекращается.

Если $|B(v)| = n_b$, т.е. узлу v известны все опорные узлы и они находятся в активном состоянии, то данная итерация алгоритма также завершается.

Если узлу v не известно ни одного опорного узла, т.е. $|B(v)| = 0$, то он назначает себя узлом-инициатором процесса выбора опорных узлов. На этом текущая итерация алгоритма завершается.

Если узел-инициатор или один (или более) опорных узлов уже присутствуют в сети, то порядковый номер очередного выбираемого опорного узла равен

$$k = \min\{i : 1 \le i \le n_b, i \notin B(v)\}.$$

Далее узел v вычисляет свое число "голосов" на получение статуса опорного узла с номером k:

$$p(\vec{v}, B(v)) = \begin{cases} h_0(v) & \text{при } k = 1; \\ f(\vec{v}, B(v)) & \text{при } 1 < k \le n_b. \end{cases}$$

Аналогичным образом определяются "голоса" для подмножества соседних узлов $\{x : x \in C(v), B(x) = B(v)\}$, где $C(v)$ — множество всех узлов, находящихся в таблице сетевого окружения узла v, и среди них выбирается узел q с максимальным числом "голосов" $p\left(\vec{q}, B(q)\right)$.

При выполнении условия

$$\{p\left(\vec{v}, B(v)\right) > p\left(\vec{q}, B(q)\right)\} \vee \\ \vee \{[p\left(\vec{v}, B(v)\right) = p\left(\vec{q}, B(q)\right)] \wedge [a(v) < a(q)]\}, \quad (6)$$

где $a(v)$ и $a(q)$ — адреса узлов v и q соответственно, узел v назначает себя k-м опорным узлом.

Вторая часть условия (6) необходима для разрешения неоднозначности при равенстве числа "голосов" узлов v и q. Для упрощения записи в дальнейшем будем использовать только первую часть выражения (6).

В своих сигнальных пакетах опорный узел также сообщает число "голосов" и множество $B(v)$, которые действовали на момент его назначения опорным узлом. Обозначим эти величины как $\hat{p}(v)$ и $\hat{B}(v)$.

Обновление информации об опорных узлах. При приеме сигнального пакета от соседнего узла q узел v получает следующую информацию о q:

- адрес $a(q)$;
- вектор виртуальных координат $\vec{q} = \{q_i\}_{i=1}^{n_b}$;
- множество номеров известных опорных узлов $B(q)$;
- информацию об известных опорных узлах с номерами $i \in B(q)$:
  - адрес $a(b_i(q))$;
  - полученное число "голосов" при назначении $\hat{p}_i(q)$;
  - множество номеров опорных узлов при назначении $\hat{B}_i(q)$.

Узел v выполняет следующие действия для каждого i-го компонента вектора виртуальных координат.

1. Если узлы v и $b_i(q)$ являются опорными узлами с номером i, то узел v прекращает функционировать в качестве опорного узла при $\hat{p}(v) < \hat{p}_i(q)$. В противном случае выполняется переход к п. 5.
2. Если узлу v уже известна информация об i-м опорном узле, то узел v "соглашается" принять узел $b_i(q)$ в качестве нового i-го опорного узла только при $\hat{p}_i(v) < \hat{p}_i(q)$. Далее переход к п. 4.
3. Узел v принимает узел $b_i(q)$ в качестве опорного только при $p\left[\vec{v}, \hat{B}_i(q)\right] < \hat{p}_i(q)$.
4. Обновление i-го компонента вектора координат.
5. Выход.

В описанной процедуре п. 1–3 определяют правила приоритетов, если несколько узлов в сети назначат себя опорными узлами с одинаковыми номерами. Очевидно, что подобные ситуации будут возникать, так как узлы имеют только локальную информацию при принятии решения о назначении очередного опорного узла.

В п. 4 выполняется обновление соответствующего компонента вектора виртуальных координат по стандартной процедуре, принятой в протоколах маршрутизации рассматриваемого типа.

Оптимизированный вариант алгоритма. Из описания алгоритма следует, что при принятии решения о назначении себя опорным узел сравнивает свое число "голосов" с максимальным числом "голосов" среди его соседей. Следовательно, если нет необходимости хранить информацию о сетевом окружении для других задач (в частности, для маршрутизации пакетов), то узлу достаточно в ходе локального обмена сигнальными пакетами запоминать информацию только о соседнем узле, имеющем максимальное значение функции голосования. Тогда требования к объему памяти значительно сокращаются, что будет показано далее.

Примеры работы алгоритма. На рис. 1, *а, б* наглядно показан результат применения функций голосования $f_{\min}(\vec{v}, B)$ и $f_{\text{prod}}(\vec{v}, B)$ для выбора 16 опорных узлов в сети из примерно 2000 узлов со средней плотностью размещения 20 узлов. Опорные узлы обозначены с указанием порядка их выбора. На рис. 1, *в, г* приведен результат для сети примерно такого же масштаба, но с площадью покрытия в форме круга.

В первую очередь обе функции голосования стремятся выбрать максимально удаленные друг от друга узлы, т.е. находящиеся на периферии сети. Затем функция $f_{\min}(\vec{v}, B)$ задействует узлы из середины области покрытия, приводя в результате к равномерному распределению опорных узлов. При использовании функции $f_{\text{prod}}(\vec{v}, B)$ все опорные узлы выбраны среди граничных.

Пример выбора 8 опорных узлов в сети с низкой плотностью приведен на рис. 2.

Оценка сложности алгоритма. Выполним анализ сложности по времени и памяти распределенной реализации алгоритма.

Очевидно, что временные затраты на выполнение алгоритма будут зависеть от периода передачи сигнальных пакетов t_b (см. выражение (2)), который является параметром настройки протокола маршрутизации и определяет скорость его реакции на изменения в топологии сети, поэтому t_b следует рассматривать как постоянную величину, которая не зависит от размерности задачи.

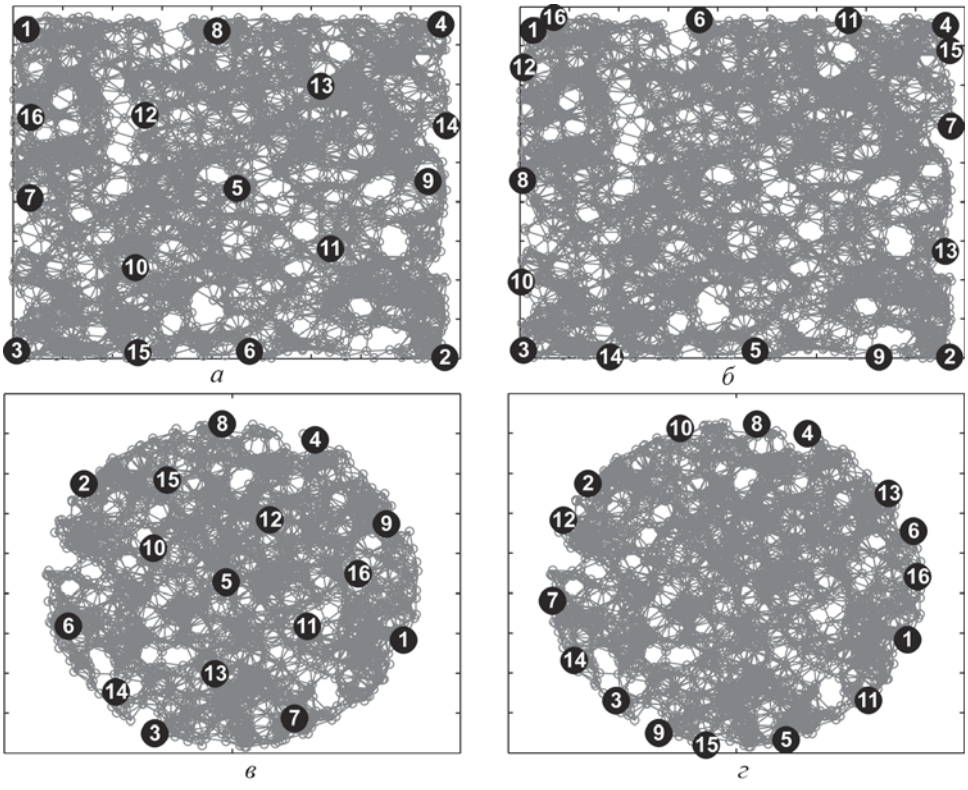

Рис. 1. Выбор 16 опорных узлов в крупной сети с высокой плотностью и площадью покрытия в форме квадрата (*а, б*) и круга (*в, г*):
а, в и *б, г* функции голосования $f_{\min}(\vec{v}, B)$ и $f_{\text{prod}}(\vec{v}, B)$ соответственно

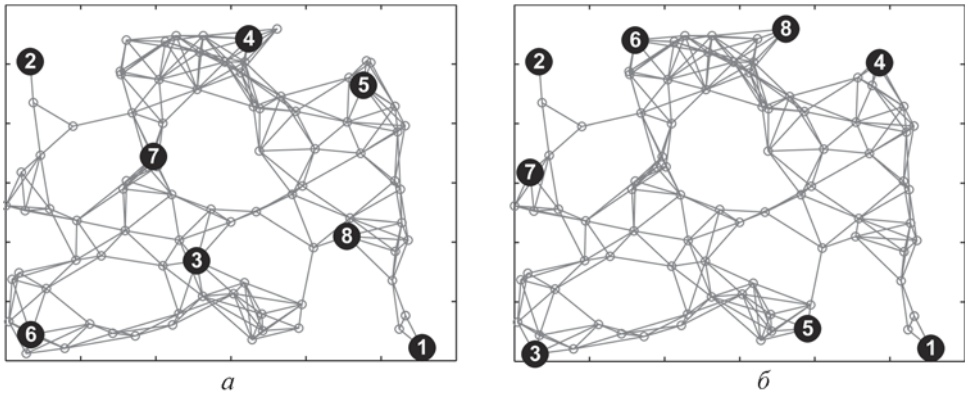

Рис. 2. Выбор восьми опорных узлов в сети с низкой плотностью:
а, б — функции голосования $f_{\min}(\vec{v}, B)$ и $f_{\text{prod}}(\vec{v}, B)$ соответственно

Полагаем, что затраты ресурсов на выбор узла-инициатора пренебрежимо малы.

На втором и третьем этапах последовательно выбираются опорные узлы. Для вычисления всеми узлами расстояния до узла-инициатора (или опорного узла) в наихудшем случае требуется время, пропорциональное диаметру сети d. Следовательно, оценка общих затрат времени на выполнение алгоритма составляет

$$T = O\left(n_b d\right).$$

Для принятия решения о назначении себя опорным узел должен хранить информацию о своих непосредственных соседях, число которых равно ρ. Служебная информация о каждом из них занимает объем $O\left(n_b\right)$, поэтому общие требования к памяти равны

$$M = O\left(n_b \rho\right).$$

В оптимизированной версии алгоритма узлу достаточно сохранять только свои собственные координаты, следовательно,

$$M_{opt} = O\left(n_b\right).$$

На практике число опорных узлов n_b задается постоянным и не зависит от общего числа узлов n, при этом $n_b \ll n$. Тогда n_b можно считать константой, не зависящей от размерности задачи, поэтому оценки сложности принимают вид

$$T = O\left(d\right); \qquad (7)$$

$$M = O\left(\rho\right)$$

и

$$M_{opt} = O\left(1\right).$$

Сравнение с алгоритмом Logical Coordinate Routing. В работе [4] предложен протокол маршрутизации Logical Coordinate Routing и дополнительно к нему алгоритм для автоматического выбора произвольного числа опорных узлов. Приведем краткое описание этого алгоритма и сравним его с предложенным в настоящей работе.

Описание алгоритма LCR. На первом этапе из всего множества узлов случайным образом выбирается подмножество узлов-кандидатов $V_c \subset V$, среди которых затем будут назначены опорные узлы. При этом только один узел может быть узлом-кандидатом в любом сетевом окружении в пределах одной элементарной передачи.

Далее узлы-кандидаты начинают передавать сигнальные пакеты для измерения расстояния относительно друг друга. По истечении необходимого времени узел-кандидат, имеющий максимальную сумму

расстояний между ним и остальными узлами-кандидатами, назначает себя первым опорным узлом.

Пока число выбранных опорных узлов k меньше необходимого n_b, итерационно выполняются следующие действия:

- опорные узлы обмениваются друг с другом значениями расстояний между ними и оставшимися узлами-кандидатами;
- для каждого оставшегося узла-кандидата $c \in V_c$ выбранные опорные узлы b_i (при $1 \le i \le k$) вычисляют значение функции голосования вида

$$f_{LCR}(c) = \prod_{i=1}^{k} h(c, b_i), \tag{8}$$

  где $h(c, b_i)$ — расстояние между узлом-кандидатом c и i-м опорным узлом;
- опорные узлы назначают узел-кандидат с максимальным значением функции голосования (8) $(k+1)$-м опорным узлом.

Видно, что в алгоритме LCR используется функция голосования, аналогичная $f_{\text{prod}}(\vec{v}, B)$ (см. выражение (4)), поэтому полученные распределения опорных узлов примерно одинаковые. Отличие в расположении опорных узлов возможно из-за того, что в предложенном алгоритме опорным узлом может быть любой узел $v \in V$, а в алгоритме LCR — только узлы из множества $V_c \subset V$.

Таким образом, алгоритм LCR отличается от предложенного следующим:

- непосредственный выбор опорных узлов выполняется не из всего множества обычных узлов, а из их подмножества (среди узлов-кандидатов);
- решение о назначении очередного опорного узла принимают действующие опорные узлы;
- для принятия решения по следующему опорному узлу необходим обмен данными между выбранными опорными узлами.

Выполним сравнение алгоритмов по указанным критериям сложности.

Оценка сложности алгоритма LCR. К сожалению, в работе [4] не приведена оценка сложности алгоритма выбора опорных узлов в протоколе LCR, поэтому выполним оценку затрат времени и памяти для этого алгоритма. Предположим, что затраты на формирование множества узлов-кандидатов V_c и на передачу информации между опорными узлами в процессе выбора пренебрежимо малы.

Полученные оценки сложности равны

$$T^{LCR} = O(d)$$

и

$$M^{LCR} = O\left(\max(M_1, M_2)\right),$$

где

$$M_1 = n_c, \tag{9}$$

$$M_2 = \begin{cases} (n_b - 1)(n_c - n_b + 1) & \text{при } n_b < \dfrac{n_c}{2} + 1; \\ \dfrac{n_c^2}{4} & \text{при } n_b \geq \dfrac{n_c}{2} + 1. \end{cases} \tag{10}$$

Выражение (9) описывает требования к объему памяти узла-кандидата, а формула (10) — требования к памяти опорного узла. Необходимый запас памяти равен наибольшему значению среди них, так как в качестве кандидата или опорного может быть выбран любой узел сети.

С учетом формулы (1) число узлов-кандидатов $n_c = |V_c|$ в сети можно оценить следующим образом:

$$n_c = \frac{A}{\pi r^2} = \frac{n}{\rho} = \frac{1}{\pi} d^2. \tag{11}$$

При $n_b \ll n$ получаем

$$T^{LCR} = O(d) \tag{12}$$

и

$$M^{LCR} = O\left(\frac{n}{\rho}\right) = O\left(d^2\right). \tag{13}$$

Сравнение алгоритмов по критериям сложности. Согласно выражениям (7) и (12) временные затраты на выполнение автоматического выбора опорных узлов для обоих алгоритмов линейно зависят от диаметра сети d, т.е. размера сети в элементарных передачах. Следовательно, по времени выполнения оба алгоритма являются оптимальными для протоколов маршрутизации по виртуальным координатам.

По требованиям к ресурсам памяти предложенный в настоящей работе алгоритм полностью обладает свойством масштабируемости, так как затраты памяти зависят только от плотности сетевого окружения ρ, а не от общего числа узлов n или диаметра сети d. Алгоритм использует минимально необходимую для протокола маршрутизации информацию, поэтому по требованиям к памяти он также является оптимальным.

При использовании оптимизированного варианта алгоритма затраты памяти составляют $O(1)$, т.е. не зависят ни от плотности размещения, ни от общего числа узлов.

Из выражения (13) следует, что требования к памяти на выполнение алгоритма выбора опорных узлов протокола LCR линейно зависят от общего числа узлов n и квадратично (кубично — при размещении узлов в трехмерном объеме) от диаметра сети d, поэтому требование масштабируемости не выполняется.

Выводы. Предложенный алгоритм предназначен для автоматического выбора произвольного числа опорных узлов как при начальной инициализации сети, так и при возможном выходе из строя опорных узлов в процессе эксплуатации. Указанные варианты функций голосования позволяют получить равномерное распределение опорных узлов по площади покрытия сети или распределение по периметру (границе) сети.

Применение одной из предложенных функций голосования дает результат, аналогичный полученному с помощью алгоритма автоматического выбора опорных узлов [4]. Главное отличие алгоритмов заключается в величине оценок сложности, которые показали:

- предложенный в настоящей работе алгоритм является оптимальным по времени и памяти;
- алгоритм из работы [4] является оптимальными по временно́й сложности, а сложность по памяти в значительной степени зависит от масштаба сети.

Оптимизированная версия алгоритма может использоваться в приложениях, в которых необходимо только выбрать опорные узлы с определенным распределением по площади покрытия сети.

Таким образом, предложенный алгоритм обладает гибкостью для получения различного вида распределений опорных узлов по сети и является полностью масштабируемым в отличие от существующих аналогов, поэтому может быть эффективно использован в беспроводных многоячейковых сетях крупных размеров.

СПИСОК ЛИТЕРАТУРЫ

1. F o n s e c a R. Beacon vector routing: scalable point-to-point routing in wireless sensornets // Proceedings of the 2nd Symposium on Networked Systems Design and Implementation. – Boston, Massachusetts, USA: 2005.
2. C a r u s o A. GPS free coordinate assignment and routing in wireless sensor networks // Proceedings of the 24th Annual Joint Conference of the IEEE Computer and Communications Societies INFOCOM 2005. – Vol. 1. – 2005. – P. 150–160.
3. C a o Q. A scalable logical coordinates framework for routing in wireless sensor networks // Proceedings of the 25th IEEE International Real-Time Systems Symposium. – 2004. – P. 349–358.
4. C a o Q., A b d e l z a h e r T. Scalable logical coordinates framework for routing in wireless sensor networks / // ACM Transactions on Sensor Networks. – 2006. – Vol. 2. No. 4. – P. 557–593.

4. Баскаков С. С. Алгоритм автоматического выбора опорных узлов в беспроводных сенсорных сетях // Тр. конф. “Информационные технологии и системы” (ИТиС-2007). – Звенигород: 2007. – С. 2–8.

Статья поступила в редакцию 21.01.2008

Сергей Сергеевич Баскаков родился в 1981 г., окончил МГТУ им. Н.Э. Баумана в 2006 г. Аспирант кафедры “Информационные системы и телекоммуникации” МГТУ им. Н.Э. Баумана. Автор 3 научных работ в области беспроводных сетей и систем связи.

S.S. Baskakov (b. 1981) graduated from the Bauman Moscow State Technical University in 2006. Post-graduate of “Information Systems and Telecommunications” department of the Bauman Moscow State Technical University. Author of 3 publications in the field of wireless networks and communication systems.

УДК 004.422.6

К. А. Пасечников, Г. С. Иванова

СИНТЕЗ ОПТИМАЛЬНЫХ СТРУКТУР ДАННЫХ ДЛЯ РЕШЕНИЯ ЗАДАЧ НА ГРАФАХ

Предложена модель, позволяющая адекватно отобразить характеристические особенности базовых структур данных, используемых для представления графовых моделей. Формально определена операция объединения базовых структур данных, что позволило автоматизировать расчет временны́х и емкостных параметров полученных комбинированных структур данных. Предложена формальная постановка задачи синтеза оптимальной (с точки зрения минимимизации вычислительной сложности выполнения заданного набора операций) одноуровневой комбинированной структуры данных при условии допустимой емкостной сложности этой структуры.

При решении задач структурного анализа и синтеза на графах применяется большое число различных структур данных. Они используются для хранения не только основной (информации о графе), но и различной вспомогательной информации. Как было показано в работах [1–4], способ организации элементов структуры данных существенно влияет на вычислительную и емкостную сложности программ решения задач на графах. В работах [2, 3] показано, что применение лишь базовых структур данных для решения задач на графах неэффективно в силу отсутствия такой базовой структуры данных, которая обеспечила бы минимальную вычислительную сложность для любой операции. Следовательно, необходим синтез комбинированных структур данных и выбор из них оптимальной с точки зрения вычислительной сложности.