УДК 004.021; 519.687.1 doi 10.26089/NumMet.v18r105

ПРИБЛИЖЕННЫЙ АЛГОРИТМ ВЫБОРА ОПТИМАЛЬНОГО ПОДМНОЖЕСТВА УЗЛОВ В КОММУНИКАЦИОННОЙ СЕТИ АНГАРА С ОТКАЗАМИ

А. В. Мукосей[1], А. С. Семенов[2]

В Научно-исследовательском центре электронной вычислительной техники (НИЦЭВТ) разрабатывается высокоскоростная коммуникационная сеть Ангара с топологией "многомерный тор". При эксплуатации вычислительного кластера с сетью Ангара в условиях наличия занятых и отказавших узлов возникает задача поиска оптимального подмножества узлов сети для покрытия заданного числа узлов так, чтобы весь сетевой трафик лежал внутри этого подмножества узлов. В настоящей статье представлен приближенный полиномиальный алгоритм решения такой задачи.

Ключевые слова: отказоустойчивость, коммуникационные сети, многомерный тор, связность, детерминированная маршрутизация, маршрутизация с порядком направлений.

1. Введение. В Научно-исследовательском центре электронной вычислительной техники разрабатывается высокоскоростная коммуникационная сеть Ангара [1, 2] с топологией "многомерный тор". В маршрутизаторе сети реализована бездедлоковая детерминированная маршрутизация, основанная на правилах "пузырька" (bubble flow control) [4] и "порядка направлений" (Direction Ordered Routing, DOR) [5, 6] с использованием битов направлений [6]. Благодаря алгоритму First Step/Last Step "нестандартного первого и последнего шага" [6] аппаратно поддерживается обход отказавших узлов и линков. Эффективность этого метода по поддержанию связности в сети с отказами была показана в статье [3]. Применяемая маршрутизация позволяет избежать взаимных блокировок (дедлоков) из-за циклических зависимостей ожидания освобождения ресурсов в кольцах и между кольцами нескольких измерений, а также гарантирует сохранение порядка передачи пакетов между любыми двумя адресатами.

Для эффективного использования узлов системы необходимо уметь оптимально выделять ресурсы в зависимости от состояния кластера, которое может периодически изменяться по различным причинам (занятость отдельных узлов, неисправность оборудования и др.). Необходимо уметь за небольшое время принимать решения по выделению требуемого числа узлов. В настоящей статье предложен приближенный алгоритм выбора оптимального подмножества узлов в коммуникационной сети Ангара с отказами. Авторам статьи не известны подобные алгоритмы для сетей с топологией "многомерный тор" и для методов маршрутизации, используемых в сети Ангара.

Статья организована следующим образом. В разделе 2 приводятся необходимые формальные определения. В разделе 3 описывается маршрутизация в сети Ангара. В разделе 4 представлена постановка задачи. В разделе 5 рассматриваются алгоритмы решения поставленной задачи: вспомогательный алгоритм определения существования маршрутов из каждого узла множества в каждый другой узел этого множества, приближенный алгоритм расчета оптимальных таблиц маршрутизации для заданного множества узлов, алгоритмы выбора множеств узлов перебором всевозможных n-мерных прямоугольников и равномерным расширением. Исследование разработанных алгоритмов проводится в разделе 6.

2. Определения. В этом разделе вводятся некоторые формальные определения, которые в дальнейшем будут использоваться в статье.

Рассмотрим коммуникационную сеть с топологией многомерный тор. Множество всех узлов сети обозначим N, размерности тора обозначим $(d_1, d_2, \ldots, d_n)$, где n — число измерений тора. Общее число узлов обозначим через $|N|$. Каждый узел u имеет координаты $(u_1, u_2, \ldots, u_n)$, где $0 \leqslant u_i < d_i$.

Направлением D_j будем называть набор $D_j = (0, \ldots, \underbrace{\pm 1}_{j \bmod n}, \ldots, 0)$ длины n, где на позиции $j \bmod n$ будет стоять $+1$, а на остальных нули, если $1 \leqslant j \leqslant n$, или -1, если $n+1 \leqslant j \leqslant 2n$. Направления с

[1] Научно-исследовательский центр электронной вычислительной техники (НИЦЭВТ), Варшавское шоссе, 125, 117587, Москва; мл. науч. сотр., e-mail: mukav@mail.ru

[2] Научно-исследовательский центр электронной вычислительной техники (НИЦЭВТ), Варшавское шоссе, 125, 117587, Москва; начальник сектора, e-mail: semenov@nicevt.ru

номерами $1, \ldots, n$ будем называть *положительными*, а с номерами $n+1, \ldots, 2n$ — *отрицательными*. Соседними в рамках тороидальной топологии в направлении D_j будем называть узлы $u = (u_1, u_2, \ldots, u_n)$ и $v = u + D_j = \big(u_1, \ldots, (u_{j \bmod n} \pm 1) \bmod d_{j \bmod n}, \ldots, u_n\big)$ для любого индекса $1 \leqslant j \leqslant 2n$. В дальнейшем будем употреблять выражение "узел v находится в направлении D_j от узла u".

Множество направлений обозначим $\mathcal{D}$: $\mathcal{D} = \{D_j\}_{j=\overline{1,2n}}$. На множестве направлений $\mathcal{D}$ введем порядок в соответствии с указанной нумерацией: $D_i < D_j$, если $i < j$.

Определение 1. *Каналом связи* (*линком*) будем называть пару (u, D), где $u \in N$, $D \in \mathcal{D}$. Множество всех каналов связи обозначим $\mathcal{E} = N \times \mathcal{D}$.

Определение 2. Каждый узел сети может выступать в роли *транзитного* — узел, необходимый для поддержания связности системы, или *активного* — узел, поддерживающий связность системы и выполняющий инжекцию/эжекцию пакетов в сеть. Множество активных узлов будем помечать индексом a, транзитных — индексом t.

Определение 3. *Путем* $\mathcal{P}$, соединяющим два узла сети u^0 и u^l, назовем последовательность вида $u^0, s_1, u^1, s_2, \ldots, s_l, u^l$, где u_i — узел сети, $s_i \in \mathcal{D}$ — направление, по которому выполняется *шаг* между узлами u^{i-1} и u^i, и l — длина пути. При этом $u^1, \ldots, u^{l-1}$ — *транзитные узлы пути*. Так как транзитные узлы пути могут быть получены из соответствующих шагов, то их можно опустить, тогда подобный путь будет записываться в виде $u^0, s_1, s_2, \ldots, s_l$.

Определение 4. Подмножество $M_{(a,t)} = M_a \cup M_t$, где M_a — множество активных узлов сети, а M_t — множество транзитных узлов сети, множества узлов N назовем *маршрутизируемым*, если для любых двух узлов u, v множества M_a существует путь $\mathcal{P}$ из u в v такой, что транзитные узлы этого пути принадлежат $M_{(a,t)}$.

Определение 5. *Таблицей маршрутизации* $\mathcal{R}$ маршрутизируемого множества $M_{(a,t)}$ назовем некоторый набор путей таких, что для любых двух узлов u, v множества M_a в $\mathcal{R}$ существует единственный путь из u в v такой, что транзитные узлы этого пути принадлежат $M_{(a,t)}$.

Определение 6. *Диаметром* маршрутизируемого множества $M_{(a,t)}$ с таблицей маршрутизации $\mathcal{R}$ назовем максимальную длину пути из набора путей $\mathcal{R}$.

Определение 7. Пусть для некоторого маршрутизируемого множества $M_{(a,t)} \subseteq N$ построена таблица маршрутизации $\mathcal{R}$. *Загруженностью канала связи* $G_{(u,D)}$ маршрутизируемого множества $M_{(a,t)}$ будем называть количество путей, которым принадлежит данный канал связи:

$$G_{(u,D)} = \Big|\big\{\mathcal{P}_{ij} \mid (u, D) \in \mathcal{P}_{ij}, \mathcal{P}_{ij} \in \mathcal{R}\big\}\Big|.$$

3. Маршрутизация в сети Ангара.

3.1. Правило порядка направлений с использованием битов направлений. Среди алгоритмов маршрутизации для многомерных торов можно выделить класс алгоритмов, соблюдающих *правило порядка направлений*: маршрут между любой парой узлов состоит из движения в направлениях, которые заданы в заранее определенном порядке. Эти алгоритмы обладают свойством отсутствия взаимных блокировок между кольцами нескольких измерений тора при любом количестве одновременных запросов на передачу данных по сети.

Определение 8. Путь $\mathcal{P} = u^0, s_1, s_2, \ldots, s_l$ из узла u в узел v удовлетворяет *правилу порядка направлений*, если $s_{i-1} \leqslant s_i$, $i = \overline{2,l}$, где l — длина пути; $s_i \in \mathcal{D}$ для всех i; $u \in N$ — стартовый узел пути; $v \in N$ — конечный узел пути.

В сети Ангара реализована маршрутизация с использованием *битов направлений*, которая вносит некоторые ограничения на маршрутизацию с правилом порядка направлений.

Заметим, что путь $\mathcal{P} = u^0, s_1, s_2, \ldots, s_l$ из узла u в узел v, удовлетворяющий правилу порядка направлений, можно записать в виде $u^0, S_1, \ldots, S_{2n}$, где S_j — набор шагов (возможно, пустой) в направлении $D_j \in \mathcal{D}$. Длина пути может быть получена следующим образом: $l = \sum\limits_{i=1}^{2n} |S_i|$, где $|S_i|$ — число шагов в наборе S_i.

Определение 9. Путь $\mathcal{P} = u^0, s_1, s_2, \ldots, s_l$ из узла u в узел v удовлетворяет маршрутизации с использованием *битов направлений*, если в пути $u^0, S_1, S_2, \ldots, S_{2n}$ множества S_i удовлетворяют одному из следующих условий для всех $i = \overline{1,n}$: $|S_i| > 0$, или $|S_{i+n}| > 0$, или $|S_{i+n}| = |S_i| = 0$.

Через $P_{dirbit} = u^0, s_1, s_2, \ldots, s_l$ обозначим путь, соответствующий маршрутизации с использованием битов направлений, а через D_{dirbit} — набор направлений $s_1, s_2, \ldots, s_l$.

3.2. First Step/Last Step. Метод First Step/Last Step [6] используется в сети Ангара как механизм обхода отказавших узлов. Этот механизм расширяет маршрутизацию с использованием битов направле-

ний путем добавления первого и последнего нестандартного шага.

Путь с использованием первого и последнего нестандартного шага будет записываться следующим образом: $u^0, D_{FS}, D_{dirbit}, D_{LS}$, где u^0 — стартовый узел, D_{FS} — первое положительное нестандартное направление, D_{LS} — последнее отрицательное нестандартное направление. При этом набор направлений $D_{FS}, D_{dirbit}, D_{LS}$ удовлетворяет правилу порядка направлений.

4. Постановка задачи. Во время эксплуатации разделяемого вычислительного кластера необходимо при любом состоянии системы уметь предоставлять требуемое число узлов, если это возможно, которое должно быть маршрутизируемо и не иметь транзитного трафика вне этого набора узлов. Состояние системы определяется набором отказавших линков и/или узлов и наличием занятых узлов. Занятый или отказавший узел можно интерпретировать как узел, у которого линки сломаны во всех направлениях.

Обозначим множество сломанных линков $F \subset \mathcal{E}$.

Так как физический канал связи между двумя узлами v и u представляет собой линки от узла v к узлу u и наоборот, то разумно предположить, что при неисправности одного из линков второй тоже неисправен. Таким образом, множество F будет включать в себя отказавшие каналы связи попарно.

Во введенных определениях задача будет формулироваться следующим образом. Пусть задан тор с размерностями $(d_1, \ldots, d_n)$ и набором отказавших линков F. Требуется построить алгоритм поиска маршрутизируемого множества $M_{(a,t)}$, такого, что $|M_a| = m$, где m — требуемое число узлов.

Так как различных систем $M_{(a,t)}$ может быть несколько, необходим критерий выбора оптимального маршрутизируемого множества. В нашей работе рассматривались следующие критерии:

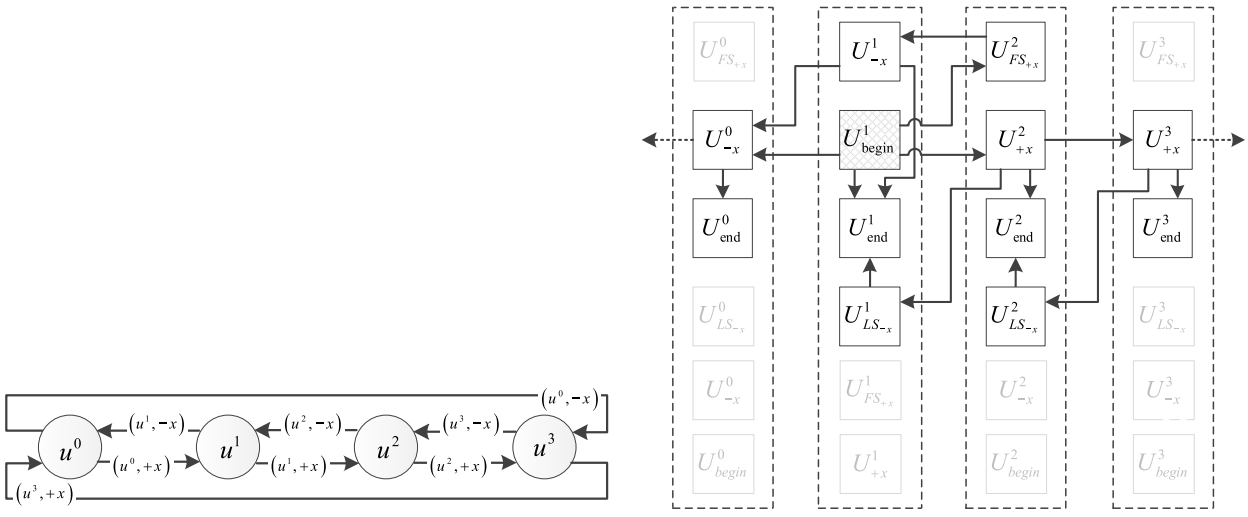
1) минимальный диаметр;

2) наименьшая максимальная загрузка линков;

3) наименьшее число транзитных узлов.

Первый критерий возникает из-за того, что в сети с минимальным диаметром задержка на передачу данных будет наименьшей. Второй критерий следует из стремления получить равномерно загруженную систему. Третий критерий — из необходимости эффективно использовать аппаратные ресурсы вычислительного кластера.

5. Алгоритмы решения задачи. Для решения поставленной задачи в настоящей статье предлагаются несколько алгоритмов. Основные алгоритмы выбора множества узлов равномерным расширением (см. подраздел 5.5) и перебором n-мерных прямоугольников (см. подраздел 5.4) приведены после используемых ими вспомогательных алгоритмов проверки множества на маршрутизируемость (см. подраздел 5.1) и приближенного алгоритма построения оптимальной таблицы маршрутизации (см. подраздел 5.2).



a) Кольцо длиной 4 b) Часть графа для кольца длиной 4

Рис. 1. Пример построения части графа для кольца длиной 4. Порядок направлений "$+x - x$"

5.1. Алгоритм определения маршрутизируемости множества. Сведем задачу определения маршрутизируемости множества к поиску пути в некотором ориентированном графе $G(V, E)$. Вершины графа будем обозначать U^i_X. Каждому вычислительному узлу сети соответствует несколько вершин графа. Верхний индекс i определяет, какому узлу сети соответствует данная вершина. Нижний индекс X определяет информацию о предыстории пути, которая в соответствии с правилами маршрутизации вносит ограничения на принятие решения о следующем шаге.

Например, рассмотрим граф путей $G(V,E)$ (рис. 1b), построенный для одномерной топологии с длиной кольца 4 (рис. 1a). Пунктиром выделены группы вершин с одинаковым верхним индексом, т.е. соответствующих одному узлу сети. На рисунке для наглядности опущена бóльшая часть ребер.

Из вершины U^1_{begin}, соответствующей инжекции в сеть из узла u^1, возможны следующие шаги: в направлении "$+x$" в вершины U^2_{+x} и $U^2_{FS_{+x}}$, в направлении "$-x$" в вершину U^0_{-x} и на эжекцию в вершину U^1_{end}. Вершина U^2_{+x} соответствует узлу сети u^2. Нижний индекс $+x$ означает, что в эту вершину совершено движение по направлению "$+x$"; дальнейшее движение с учетом всех правил маршрутизации возможно только в вершину U^3_{+x} в направлении "$+x$", или в вершину $U^1_{LS_{-x}}$ в направлении "$-x$", совершив последний нестандартный шаг, или на эжекцию в вершину U^2_{end}. Вершина U^1_{+x-x} отсутствует, так как маршрут "$+x-x$" не соответствует правилам маршрутизации. Остальные ребра и вершины строятся аналогично.

Перейдем к формальному описанию. Рассмотрим движение по некоторому пути в торе в направлениях $\{S_1\},\dots,\{S_i\}$. Этот путь можно продолжить в том же направлении D_i или в таком направлении D_{i+1}, что набор направлений $\{S_1\},\dots,\{S_i\},D_{i+1}$ удовлетворяет правилам маршрутизации, где $i \leqslant 2n-1$. Возможность выбора D_{i+1} зависит только от набора $\{S_1\},\dots,\{S_i\}$.

В этой связи, для описания каждого узла сети $u^i \in N$ в графе $G(V,E)$ построим множество U^i вершин, которые будут характеризовать предысторию путей, которые проходят через узел u^i. Множество U^i состоит из следующих вершин:

1) U^i_{begin} — вершина, из которой начинается движение (инжекция пакета в сеть);

2) $U^i_{FS_j}$, $j=1,\dots,n$, — вершины, в которые возможно попасть, совершив первый нестандартный положительный шаг из соседнего узла в направлении D_j;

3) $U^i_{dirbit_j}$, где индекс $dirbit_j$ соответствует некоторому набору направлений, удовлетворяющему правилу с использованием битов направлений, двигаясь по которым, можно попасть в данную вершину;

4) $U^i_{LS_j}$, $j=n+1,\dots,2n$, — вершины, в которые возможно попасть, совершив последний нестандартный отрицательный шаг из соседнего узла в направлении D_j;

5) U^i_{end} — вершина, в которой заканчивается движение (эжекция пакета из сети).

Посчитать количество вершин U^i_{dirbit} можно следующим образом. Закодируем набор n-мерным числом, где на j-м месте стоит -1 в случае движения в отрицательном направлении, $+1$ — в случае движения в положительном направлении и 0 — в случае отсутствия движения в j-м измерении тора. Всего таких наборов 3^n-1, так как нулевому набору соответствует вершина U_{begin}.

Таким образом,

$$U^i = \left\{ \left(\bigcup_{j=1}^{n} U^i_{FS_j} \right) \bigcup \left(\bigcup_{j=n+1}^{2n} U^i_{LS_j} \right) \bigcup \left(\bigcup_{j=1}^{3^n-1} U^i_{dirbit_j} \right) \bigcup U^i_{begin} \bigcup U^i_{end} \right\};$$

$$\left|U^i\right| = n + n + 3^n - 1 + 2 = 3^n + 2n + 1;$$

$$V = \bigcup_{i=1}^{|N|} U^i;$$

$$|V| = |N| * \left|U^i\right| = |N| * \left(3^n + 2n + 1\right) \quad \forall u^i \in N.$$

Вершины в графе соединяются таким образом, чтобы переход от одной вершины графа к другой соответствовал путям, проходящим через соответствующие узлы сети и удовлетворяющим правилам маршрутизации. Опишем всевозможные ребра в графе $G(V,E)$.

1. Рассмотрим вершину U^i_{begin}, соответствующую узлу u^i, из которой начинается движение. Первым шагом в пути из узла u^i может быть

   - движение в направлении первого нестандартного положительного шага D_k в вершину $U^j_{FS_k}$, соответствующую узлу $u^j = u^i + D_k$;
   - движение в любом направлении D_k в вершину $U^j_{dirbit_l}$, где $dirbit_l$ в данном случае соответствует D_k; вершина $U^j_{dirbit_l}$ соответствует узлу $u^j = u^i + D_k$;
   - движение в вершину U^i_{end} для завершения движения.

2. Рассмотрим вершины $U^i_{FS_l}$, соответствующие узлу u^i. Из этих вершин возможно движение в вершины $U^j_{dirbit_t}$, соответствующие узлам $u^j = u^i + D_k$ в направлениях D_k, таких, что $D_{FS_l} < D_k$. Аналогично предыдущему случаю, индекс $dirbit_t$ в данном случае соответствует D_k.

3. Рассмотрим вершины $U^i_{dirbit_l}$, соответствующие узлу u^i, через который проходят пути с набором направлений $dirbit_l$. Из этих вершин возможно движение

   - в некоторую вершину $U^j_{dirbit_t}$ в таком направлении D_k, что последнее направление в $dirbit_l \leqslant D_k$ и $dirbit_t = \{dirbit_l, D_k\}$, при этом $u^j = u^i + D_k$;
   - в некоторую вершину $U^j_{LS_k}$ в таком направлении D_k, что последнее направление в $dirbit_l < D_k$, при этом $u^j = u^i + D_k$;
   - в вершину U^i_{end} для завершения движения.

4. Рассмотрим вершины $U^i_{LS_k}$, соответствующие узлу u^i, через который проходят пути с последними нестандартными отрицательными направлениями D_k. Из этих вершин возможно движение только в вершину U^i_{end} для завершения движения.

Посчитаем, сколько ребер графа приходится на один набор U^i вершин. В первом случае вершины U^i_{begin} имеют $n+2n+1$ ребер. Во втором случае для каждого первого нестандартного шага D_j существует $2n-j$ вариантов, итого получим $\sum\limits_{j=1}^{n}(2n-j) = 2n*n - \dfrac{(1+n)n}{2} = \dfrac{3n^2-n}{2}$ ребер. В третьем случае для каждой вершины $U^i_{dirbit_l}$ из 3^n-1 вершин будет не больше, чем $2n$ соседей. В четвертом случае — n ребер.

Просуммировав все, получим $n+2n+1+\dfrac{3n^2-n}{2}+\big(3^n-1\big)2n+n = 2n3^n+1.5n^2+1.5n+1$ — оценку числа ребер на каждый узел тора u^i.

Таким образом, общее количество ребер в графе $|E| = O\Big(\big(2n3^n+1.5n^2+1.5n+1\big)|N|\Big)$.

Утверждение. *Из узла $u^i \in N$ в узел $u^j \in N$ существует путь $\mathcal{P}$ тогда и только тогда, когда в графе $G(V,E)$ существует путь из вершины U^i_{begin} в вершину U^j_{end}.*

Доказательство. Для доказательства приведем взаимно однозначное соответствие между множеством путей в сети и множеством путей в графе G. Рассмотрим путь

$$P = u^0,\, D_{FS},\, D_{dirbit},\, D_{LS} = u^0,\, D_{FS},\, s_1, \ldots, s_l - 1,\, D_{LS} = u^0,\, \{S_1\}, \ldots, \{S_{2n}\},\, D_{LS}.$$

Построим соответствующий путь в графе G. Для простоты номера узлов, соответствующих данным переходам, будем обозначать некоторой функцией $\delta(j)$, где j — номер шага. Первый переход в первом нестандартном положительном направлении D_{FS} соответствует переходу из вершины U^0_{begin} в вершину $U^{\delta(1)}_{D_{FS}}$.

Далее следует серия переходов в направлениях $\{S_k\}$, где k — минимальный индекс, при котором $|S_k| \neq 0$. При первом переходе из вершины $U^{\delta(1)}_{D_{FS}}$ в вершину $U^{\delta(2)}_{D_k}$ в направлении D_k помимо смены верхнего индекса (номера узла) произойдет смена нижнего индекса с D_{FS} на D_k, тем самым зафиксировав предысторию пути. Последующие шаги в направлении D_k будут менять только верхний индекс. В результате $|S_k|$ переходов в направлении D_k будет произведен переход в вершину $U^{\delta(1+|S_k|)}_{D_k}$.

Аналогично строятся последующие переходы по множествам $\{S_i\}$, где $|S_i| \neq 0$ для всех i, $k < i \leqslant 2n$. Последним шагом совершается движение из вершины $U^{\delta\left(1+\sum_{j=1}^{2n}|S_j|\right)}_{D_{dirbit}}$ в направлении D_{LS}, а затем эжекция. Полный путь в графе имеет вид

$$U^0_{begin},\, U^{\delta(1)}_{FS},\, \underbrace{U^{\delta(1+1)}_{D_1}, \ldots, U^{\delta(1+|S_1|)}_{D_1}}_{|S_1|},\, \ldots,$$
$$\underbrace{U^{\delta\left(1+\sum_{j=1}^{2n-1}|S_j|+1\right)}_{D_{dirbit}}, \ldots, U^{\delta\left(1+\sum_{j=1}^{2n}|S_j|\right)}_{D_{dirbit}}}_{|S_{2n}|},\, U^{\delta\left(1+\sum_{j=1}^{2n}|S_j|+1\right)}_{LS},\, U^{\delta\left(1+\sum_{j=1}^{2n}|S_j|+1\right)}_{end}.$$

Таким образом, пути $\mathcal{P}$ в сети соответствует единственный путь в графе $G(V,E)$. Аналогичным образом по данному пути в графе G можно построить единственный путь в сети, удовлетворяющий правилам

маршрутизации. Это соответствие показывает, что существование пути в сети из u^i в u^j равносильно существованию пути в графе G из U^i_{begin} в U^j_{end}. Что и требовалось доказать.

Таким образом, задача определения маршрутизируемости множества $M \subseteq N$ сводится к определению связности множества вершин в графе G, соответствующих узлам множества M. Для этого можно применить метод поиска вширь. Для каждой из вершин U^i_{begin} определим множество достижимых из нее вершин вида U^j_{end}. При этом соответствующие узлы u^j будут достижимы из u^i, а u^j, u^i и транзитные узлы будут принадлежать M.

Алгоритм поиска вширь имеет сложность $T = O(V + E) = O\big(C_1|M| + C_2|M|\big) = O\big((C_1 + C_2)|M|\big)$. Так как алгоритм нужно выполнить для всех M вершин, то $T = O\big((C_1 + C_2)|M|^2\big) = O\big((C)|M|^2\big)$, где $C_1 = \big(3^n + 2n + 1\big)$ и $C_2 = \big(2n3^n + 1.5n^2 + 1.5n + 1\big)$.

5.2. Алгоритм построения таблицы маршрутизации. Пусть имеется маршрутизируемое множество $M_{(a,t)} \subset N$. Требуется найти оптимальную таблицу маршрутизации для узлов множества $M_{(a,t)}$ и вычислить загруженность каждого линка всех таких узлов $M_{(a,t)}$.

Допустим, что число путей между любыми двумя узлами ограничено некоторым числом N_{paths}, тогда существует $N_{paths}^{(|M_a|-1)*|M_a|}$ различных таблиц маршрутизаций. Даже при небольшом числе узлов сети и вариантов путей число различных таблиц маршрутизации очень велико, и требуется специальный алгоритм для создания таблиц маршрутизации.

Предложен следующий алгоритм построения таблицы маршрутизации. Предположим, что все линки узлов множества $M_{(a,t)}$ имеют нулевую загруженность. Для каждого узла u маршрутизируемого множества M_a в графе $G(V, E)$ запускается алгоритм поиска вширь. После окончания поиска из каждого узла множества M_a необходимо подняться по построенному дереву обратно вверх к узлу u, увеличивая при этом загруженность $G_{u,D}$ проходимых линков сети. Эвристически выяснено, что сбалансированная таблица маршрутизации получается, если в качестве следующего узла для запуска поиска вширь выбирать узел, максимально удаленный от узла u. Вторая эвристика, введенная для получения более равномерной загрузки линков, заключается в сортировке вершин на каждом новом слое поиска вширь по возрастанию загруженности линков, соответствующих вершинам.

Сортировку слоев в алгоритме можно оценить как

$$O\left(\sum_{i=1}^{l}\big(|V_i|\log_2|V_i|\big)\right) = O\left(\sum_{i=1}^{l}\big(|V_i|\log_2|V|\big)\right) = O\big(|V|\log_2|V|\big),$$

где l — число слоев в алгоритме, V_i — множество вершин на каждом слое. Сложность одного прохода этого алгоритма можно оценить как сумму трех слагаемых: $T_1 = O\Big((A+B)\big|M_{(a,t)}\big|^2\Big)$ для поиска вширь в графе $G(V, E)$, $T_2 = O\big(|V|\log_2|V|\big)$ — сортировка узлов на каждом шаге поиска, $T_3 = O\Big(L_{\max}\big|M_{(a,t)}\big|\Big)$ — вычисление загруженности линков. Итоговая сложность постройки таблицы маршрутизации составляет $T_R = T_1 + T_2 + T_3 = O\Big((A + B)\big|M_{(a,t)}\big|^2\Big)$.

5.3. Алгоритмы выбора подмножеств узлов полным перебором. Рассмотрим алгоритм решения задачи выбора оптимального подмножества узлов в сети с отказами полным перебором. Всего вариантов расположения m узлов в сети $\binom{\text{Nnodes}}{m}$. Для проверки маршрутизируемости каждого набора узлов требуется m^2 операций. Даже для небольших систем $\binom{\text{Nnodes}}{m} * m^2$ очень велико, эта функция растет очень быстро, поэтому такой алгоритм не может быть применен на практике. В связи с этим в работе предложено два других алгоритма. Они рассмотрены в следующих разделах.

5.4. Алгоритм выбора подмножеств узлов перебором n-мерных прямоугольников. Идея первого алгоритма заключается в ограничении полного перебора — вместо произвольного множества размера m ищется n-мерный прямоугольник с числом узлов меньшим или равным m. На вход алгоритму подается требуемое число узлов m.

Любой прямоугольник можно описать двумя параметрами: размер прямоугольника $P = (p_1, \ldots, p_n)$ и его расположение в сети. На первом этапе алгоритма происходит перебор всевозможных n-мерных прямоугольников, таких, что $|P| = \prod\limits_{i=1}^{n} p_i \leqslant m$ и $0 < p_i \leqslant d_i$ для всех $i = \overline{1, n}$. Сложность перебора таких прямоугольников $T = O\left(\prod\limits_{i=1}^{n} d_i\right) = O\big(|N|\big)$.

На втором этапе алгоритма для каждого найденного n-мерного прямоугольника подбирается его расположение в системе. В наихудшем случае таких расположений $|N|$. Так как прямоугольник может

покрывать число узлов больше требуемого, то необходимо выделить активные M_a и транзитные M_t узлы, такие, что $|M_a| = m$ и $|M_t| = |P| - m$. В текущей версии алгоритма они выбираются случайным образом.

Для каждого расположения каждого n-мерного прямоугольника проверяется маршрутизируемость узлов и строится таблица маршрутизации. Заметим, что этапы проверки маршрутизируемости и построения таблицы маршрутизации можно объединить, так как в основе этих алгоритмов лежит метод BFS (Breadth-First Search). Иными словами, для каждого n-мерного прямоугольника происходит попытка построения таблицы маршрутизации, при возникновении неразрешимой ситуации (во время применения метода BFS не была достигнута вершина из множества активных узлов) система помечается немаршрутизируемой и больше не рассматривается. В результате работы алгоритма получаем набор маршрутизируемых множеств с числом активных узлов $|M_a| = m$.

Полную сложность алгоритма можно оценить так: $T = O\Big(C\big|M_{(a,t)}\big|^2|N|^2\Big)$.

5.5. Алгоритм выбора подмножеств узлов равномерным расширением. В качестве второго решения задачи предлагается приближенный алгоритм выбора подмножеств узлов равномерным расширением. Идея алгоритма заключается в том, что из каждого узла тора происходит попытка построить n-мерный прямоугольник поочередным (равномерным) расширением в разные стороны. Ниже этот алгоритм будет называться алгоритмом равномерного расширения.

На вход алгоритму подается размер искомой системы m. Алгоритм состоит из трех этапов. На первом этапе из каждого узла сети производится расширение во все стороны с добавлением только тех узлов, у которых нет отказавших линков. На втором этапе проводится сортировка полученных систем и удаление одинаковых. На третьем этапе производится расширение получившихся систем с добавлением отказавших линков.

Рассмотрим алгоритм подробнее. На первом этапе алгоритм выделяет прямоугольники без сломанных линков. Такие прямоугольники хороши тем, что не требуют проверки множества на маршрутизируемость. Для каждого узла u^i сети строится многомерный прямоугольник следующим образом. Каждый прямоугольник можно задать двумя узлами в противоположных углах: Pa и Na. Первоначальный прямоугольник состоит из одной вершины: $Pa = Na = u^i$. Затем происходят попытки увеличения прямоугольника в каждом направлении по очереди с проверкой, что в захваченной области нет сломанных линков. Расширение прямоугольника происходит путем перемещения одного из его углов в выбранном направлении: Pa в случае расширения в положительном направлении, Na в случае отрицательного. После каждого расширения происходит проверка числа захваченных узлов: если это число больше m, то первый этап завершается. Кроме того, первый этап завершается, если отсутствуют направления, в которые можно расширить прямоугольник. На рис. 2 представлена схема работы первого этапа алгоритма для двухмерного случая.

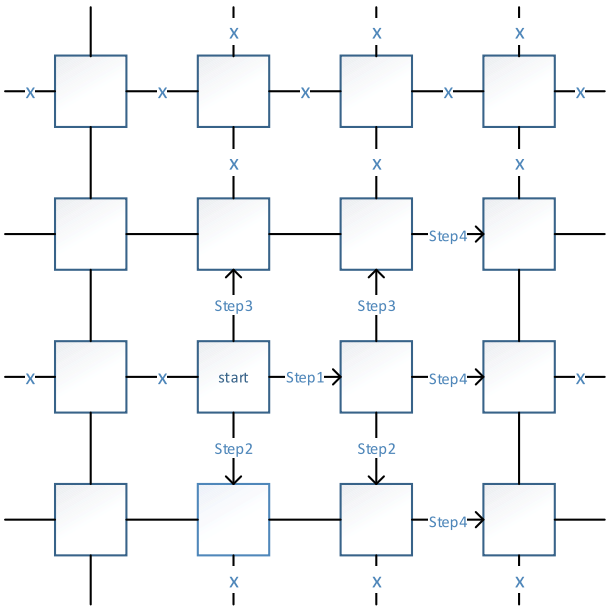


Рис. 2. Схема работы приближенного алгоритма равномерного расширения на примере двухмерного тора

Оценим сложность первого этапа алгоритма: за все время обхода нужно проверить $O\Big(2n\big|M_{(a,t)}\big| + \big|M_{(a,t)}\big|\Big) = O\Big((2n+1)\big|M_{(a,t)}\big|\Big)$ узлов и линков.

Этот алгоритм нужно применить ко всем узлам множества $|N|$. Получаем следующую оценку сложности: $T_{s_1} = O\Big((2n+1)\big|M_{(a,t)}\big|\,|N|\Big)$.

Для того чтобы сократить дальнейшую работу, на втором этапе происходит удаление одинаковых построенных прямоугольников при помощи предварительной сортировки. Если на втором этапе находятся прямоугольники нужного размера, то алгоритм заканчивается. Так как каждый прямоугольник задается с помощью двух узлов сети, то его можно описать набором $2n$ чисел. Чтобы сравнить все прямоугольники, необходимо $O\Big(|N|\log_2\big(|N|\big)\Big)$ сравнений. Таким образом, оценка сложности второго этапа $T_{s_2} = O\Big(|N|\log_2\big(|N|\big)\Big)$.

На третьем этапе происходят попытки расширить полученные прямоугольники в стороны со сломанными линками, на каждом шаге проверяется маршрутизируемость результирующего множества узлов. Заметим, чтобы проверить маршрутизируемость нового множества $M''_{(a,t)}$, которое получится из $M_{(a,t)}$ путем расширения в выбранном направлении, необходимо проверить маршрутизируемость добавленных узлов $M'_{(a,t)}$ со всеми узлами $M''_{(a,t)}$. Это можно сделать с помощью построенного графа $G(V,E)$. Для этого из каждой вершины множества $M'_{(a,t)}$ можно пройти поиском вширь по графу $G(V,E)$ и графу $G^T(V,E)$, полученному из графа $G(E,V)$ путем обращения связей (стартовая вершина теперь будет U^j_{end}, а конечная — U^i_{begin}). Таким образом, для каждой вершины u^i из $M'_{(a,t)}$ можно получить множество узлов u^j, для которых существует путь в одну сторону и обратно.

В наихудшем случае во всех расширениях прямоугольника присутствовали сломанные линки, а значит, для всех узлов множества $M_{(a,t)}$ пришлось выполнить поиск вширь по графу $G(V,E)$ и графу $G^T(V,E)$, поэтому сложность третьего этапа $T_{s_3} = O\Big(2(A+B)\big|M_{(a,t)}\big|^2|N|\Big)$.

Сложность алгоритма равномерного расширения можно оценить по формуле

$$T_{\text{expan}} = T_{s_1} + T_{s_2} + T_{s_3} = O\Big((2n+1)\big|M_{(a,t)}\big|\,|N| + |N|\log_2|N| + 2(A+B)\big|M_{(a,t)}\big|^2|N|\Big) = \\ = O\Big((A+B)\big|M_{(a,t)}\big|^2|N|\Big),$$

где $A = 3^n + 2n + 1$ и $B = 2n3^n + 1.5n^2 + 1.5n + 1$, $A + B = (2n+1)3^n + 1.5n^2 + 3.5n + 2$. Значение констант A и B довольно велико, для сети размерностью $n = 4$ значение выражения $A + B = 769$. Однако значение $A + B$ не меняется с ростом N, поэтому предложенный алгоритм является полиномиальным.

В результате работы алгоритма получается набор маршрутизируемых множеств размера больше или равного m. Аналогично предыдущему алгоритму выбираются активные и транзитные узлы и строится таблица маршрутизации.

5.6. Выбор оптимального решения. В результате работы обоих алгоритмов на выходе получается набор решений. Необходимо выбрать оптимальное множество. В разделе 4 были установлены критерии оптимальности. Для того чтобы выбрать наилучшее решение, производится сортировка решений сначала по диаметру, затем по максимальной загруженности и в конце по числу транзитных узлов.

6. Исследование. Исследование качества разработанных алгоритмов выбора оптимального подмножества узлов в сети с отказами проводились в сетях с топологией $5 \times 5 \times 5$, $6 \times 6 \times 6$ и $7 \times 7 \times 7$ в ситуациях со сломанными линками и без. Размеры искомых систем были выбраны следующие: $\frac{1}{6}, \ldots, \frac{5}{6}$ от общего числа узлов в сети $|N|$. Сломанные линки выбирались случайным образом. Число сломанных линков увеличивалось до тех пор, пока какой-либо из алгоритмов не перестанет находить решение. Шаг увеличения количества сломанных линков выбран равным 5.

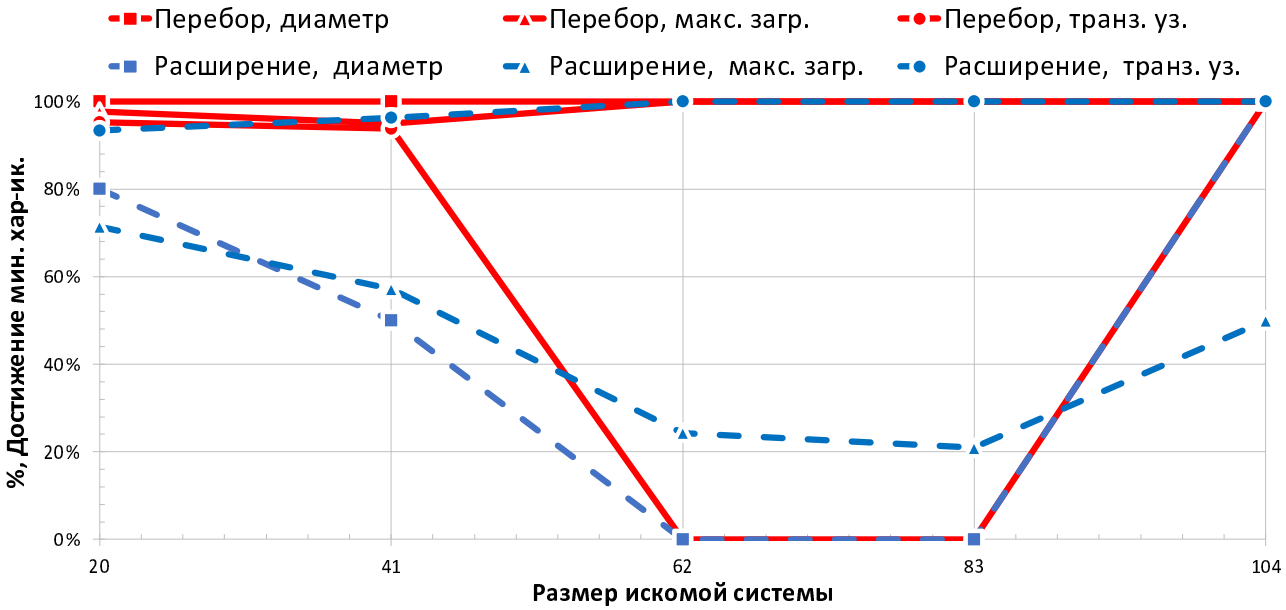


Рис. 3. Размер исходной системы $5 \times 5 \times 5$

На рис. 3–8 представлены оценки достижимости характеристики выбранного решения к наилучшим найденным значениям исследуемых алгоритмов в зависимости от размера искомой системы: в сети без отказов (рис. 3–5) и в сети с отказами (рис. 6–8). На графиках представлены значения диаметра, максимальной загрузки и числа транзитных узлов, полученных методом перебора и методом расширения.

На первом этапе исследование проводилось в сети без отказов. Размер искомых систем равнялся $\frac{1}{6}|N|, \ldots, \frac{5}{6}|N|$. На рис. 3–5 представлены значения отношений $\frac{c_{\max} - c}{c_{\max} - c_{\min}}$ (в процентах), характеризующих достижение характеристикой ее минимального значения, где значения $c_{\min}$ и $c_{\max}$ получены из множества всех решений исследуемых алгоритмов и c — характеристика выбранной в результате сортировок системы. Напомним, что в разделе 4 были выбраны следующие характеристики: диаметр, максимальная загрузка линка и число транзитных узлов. В ходе сортировки эти параметры минимизируются.

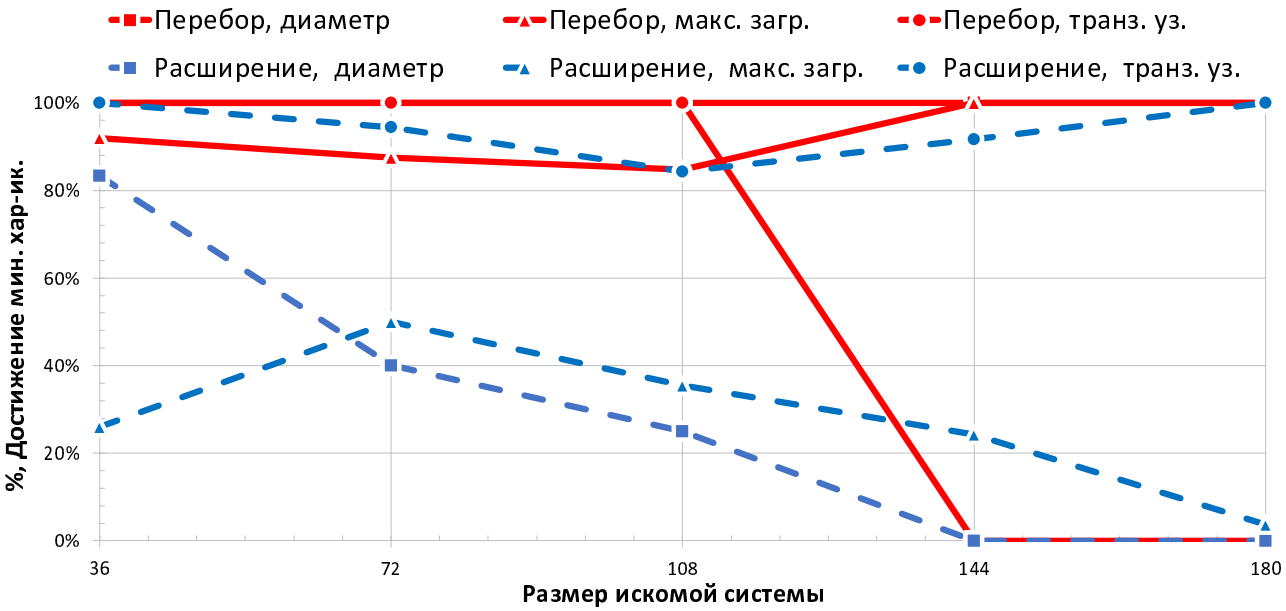


Рис. 4. Размер исходной системы $6 \times 6 \times 6$

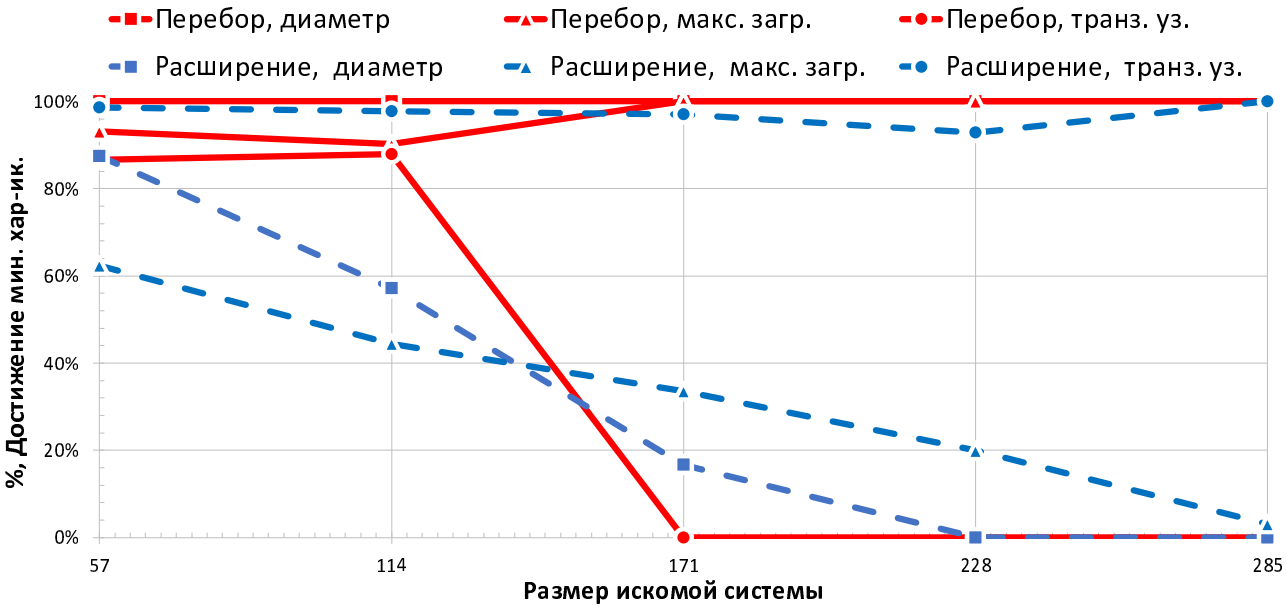


Рис. 5. Размер исходной системы $7 \times 7 \times 7$

На графиках видно, что алгоритм перебора n-мерных прямоугольников (алгоритм перебора) минимизирует диаметр лучше, чем алгоритм расширения. Это связано с тем, что алгоритм перебора не ограничен в количестве транзитных узлов (что видно на графике) и способен исследовать больше систем, когда

алгоритм расширения стремится к кубической системе. Кроме того, алгоритм перебора лучше находит системы, содержащие кольца, что сокращает диаметр и уменьшает загруженность линков.

Приближение к максимальным значениям в некоторых случаях объясняется тем, что при отборе решений происходит сортировка сначала по диаметру, затем по значениям максимальной загруженности линка и затем по количеству транзитных узлов. Таким образом, выбрав все системы с минимальным диаметром, возможно получить не минимальное число транзитных узлов.

На следующем этапе испытание проводилось в системах со сломанными линками. Размер искомой системы был выбран $\frac{1}{6}|N|$. На рис. 6–8 аналогично предыдущему этапу представлены значения отношений $\frac{c_{\max} - c}{c_{\max} - c_{\min}}$ (в процентах), характеризующих достижение характеристикой ее минимального значения.

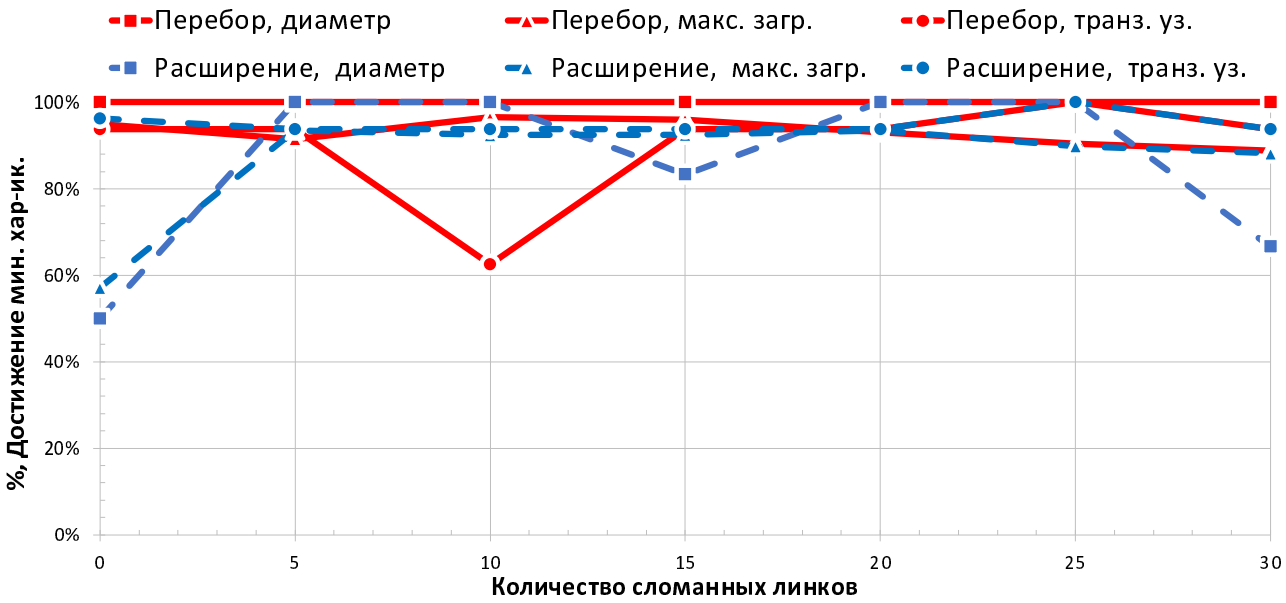


Рис. 6. Исходная система $5 \times 5 \times 5$, искомая система: 41

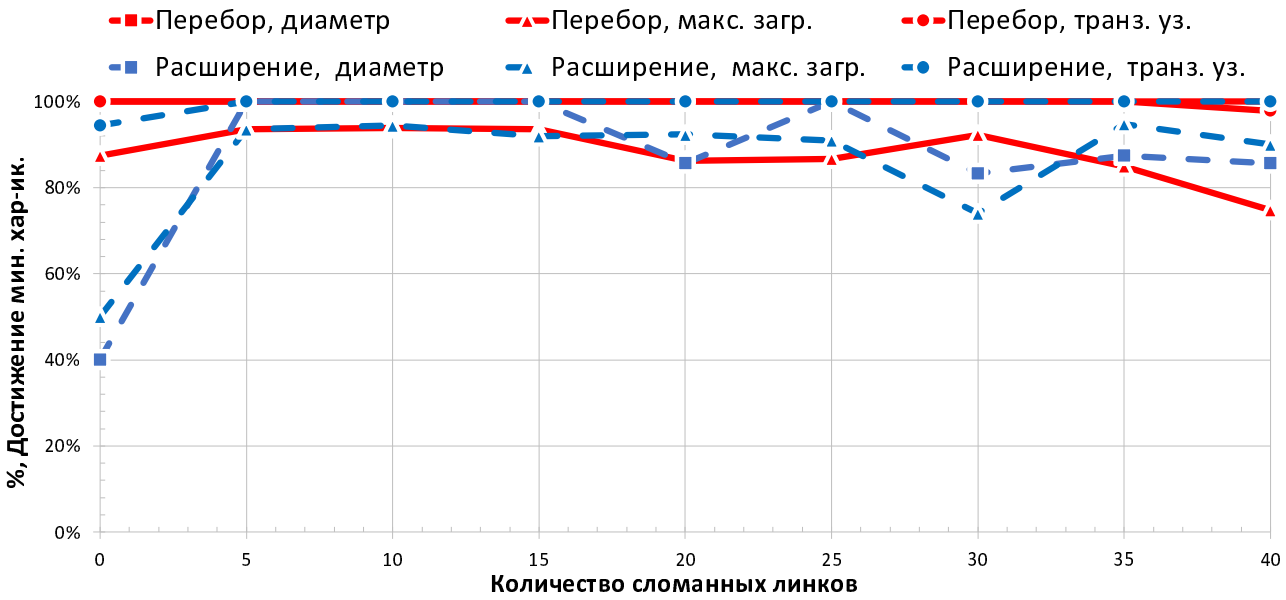


Рис. 7. Исходная система $6 \times 6 \times 6$, искомая система: 72

На графиках видно, что в сети со сломанными линками алгоритм расширения находит решения лучше, чем в системе без отказов. Это можно объяснить тем, что линки в системе разрушаются случайным образом. В результате получается сильно разреженная система, в которой становится меньше *хороших* решений, содержащих кольца.

Алгоритм перебора находит больше систем, чем алгоритм равномерного расширения. В таблице представлена медиана числа найденных систем и затраченного времени при поиске систем, описанных выше.

Алгоритм перебора n-мерных прямоугольников уже на системе $6 \times 6 \times 6$ работает недопустимо долго, но при этом находит гораздо больше систем по сравнению с алгоритмом равномерного расширения. Алгоритм равномерного расширения работает быстрее и показывает неплохой результат на системе из 343 узлов, но, учитывая рост времени работы алгоритма, на больших системах он будет уже неприменим в условиях задачи реального времени.

Все измерения проводились на системе с Intel E5620 2,4 ГГц.

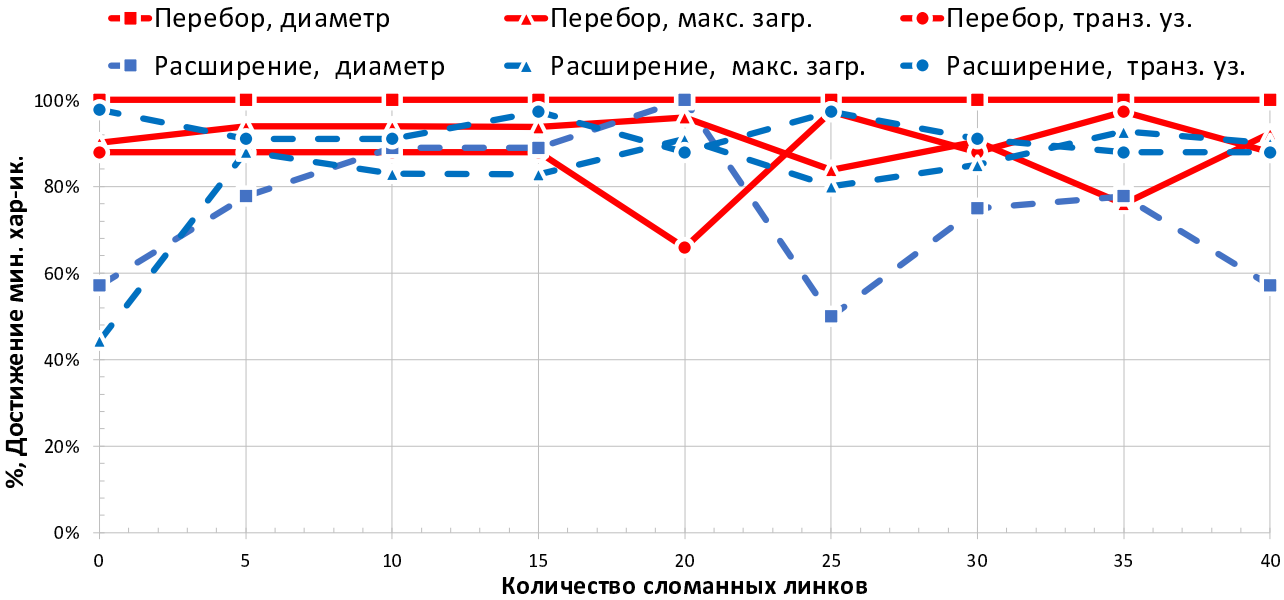


Рис. 8. Исходная система $7 \times 7 \times 7$, искомая система: 114

Медиана числа найденных систем и затраченного времени при поиске подмножеств, описанных в разделе 6 в системе с отказами

	Перебор n-мерных пр-ов			Равномерного расширения		
	$5 \times 5 \times 5$	$6 \times 6 \times 6$	$7 \times 7 \times 7$	$5 \times 5 \times 5$	$6 \times 6 \times 6$	$7 \times 7 \times 7$
Время, с	3,22	31,10	167	0,14	0,6	2,03
Число найденных решений	1964	9288	12364	27	25	40

7. Заключение. В настоящей статье описано два полиномиальных алгоритма построения маршрутизируемого подмножества узлов заданного размера в сети с отказами и проведено предварительное исследование.

Алгоритм перебора применим на сетях до 125 узлов, равномерного расширения — до 343 узлов. Оба алгоритма минимизируют исследуемые характеристики относительно всех найденных множеств.

Предварительное исследование результатов работы алгоритмов показало, что требуется их доработка для улучшения качества результатов и увеличения скорости его работы на больших системах.

В будущих работах планируется оптимизировать алгоритмы и выполнить более подробное исследование.

Статья рекомендована к публикации Программным комитетом Международной научной конференции “Параллельные вычислительные технологии” (ПаВТ–2016; http://agora.guru.ru/pavt2016).

СПИСОК ЛИТЕРАТУРЫ

1. *Жабин И.А., Макагон Д.В., Поляков Д.А., Симонов А.С. Сыромятников Е.Л., Щербак А.Н.* Первое поколение высокоскоростной коммуникационной сети “Ангара” // Наукоемкие технологии. 2014. № 1. 21–27.
2. *Агарков А.А., Исмагилов Т.Ф., Макагон Д.В., Семенов А.С., Симонов А.С.* Результаты оценочного тестирования отечественной высокоскоростной коммуникационной сети Ангара // Тр. Международной конференции “Суперкомпьютерные дни в России”. М.: Изд-во Моск. ун-та, 2016. 626–639.

3. *Пожилов И.А., Семенов А.С., Макагон Д.В.* Алгоритм определения связности сети с топологией "многомерный тор" с отказами для детерминированной маршрутизации // Программная инженерия. 2015. № 3. 13–19.
4. *Puente V., Beivide R., Gregorio J.A., Prellezo J.M., Duato J., Izu C.* Adaptive bubble router: a design to improve performance in torus networks // Proc. of International Conference on Parallel Processing. Washington, DC: IEEE Press, 1999. 58–67.
5. *Adiga N.R., Blumrich M.A., Chen D., et al.* Blue Gene/L torus interconnection network // IBM Journal of Research and Development. 2005. **49**, N 2/3. 265–276.
6. *Scott S.L., Thorson G.M.* The Cray T3E network: adaptive routing in a high performance 3D torus // Proc. IV Symp. on Hot Interconnects. Washington, DC: IEEE Press, 1996. 147–156.

Поступила в редакцию
30.12.2016

An Approximate Algorithm for Choosing the Optimal Subset of Nodes in the Angara Interconnect with Failures

A. V. Mukosey [1] **and A. S. Semenov** [2]

[1] *Scientific Research Center for Electronic Computer Technology; Varshavskoe shosse 125, Moscow, 117587, Russia; Junior Scientist, e-mail: mukav@mail.ru*

[2] *Scientific Research Center for Electronic Computer Technology; Varshavskoe shosse 125, Moscow, 117587, Russia; Ph.D., Head of Sector, e-mail: semenov@nicevt.ru*

Received December 30, 2016

Abstract: The Angara high-speed interconnect with multidimensional torus topology is under development in Scientific Research Center for Electronic Computer Technology. During the utilization of the Angara interconnect in cluster systems, there exist busy and failed nodes. Thus, there is a problem of finding an optimal cluster node subset such that the network traffic belongs to this node subset and the node subset size is not less than a given size. The paper presents an approximate algorithm for solving this problem.

Keywords: fault tolerance, interconnect, multidimensional torus, connectivity, deterministic routing, direction-order routing.

References

1. I. A. Zhabin, D. V. Makagon, D. A. Polyakov, et al., "First Generation of Angara High-Speed Interconnection Network," Naukoemkie Tekhnol., No. 1, 21–27 (2014).

2. A. A. Agarkov, T. F. Ismagilov, D. V. Makagon, et al., "Performance Evaluation of the Angara Interconnect," in *Proc. Int. Conf. on Russian Supercomputing Days, Moscow, Russia, September 26–27, 2016* (Mosk. Gos. Univ., Moscow, 2016), pp. 626–639.

3. I. A. Pozhilov, A. S. Semenov, and D. V. Makagon, "Connectivity Problem Solution for Direction Ordered Deterministic Routing in nD Torus," Programm. Inzhener., No. 3, 13–19 (2015).

4. V. Puente, R. Beivide, J. A. Gregorio, et al., "Adaptive Bubble Router: A Design to Improve Performance in Torus Networks," in *Proc. Int. Conf. on Parallel Processing, Aizu-Wakamatsu, Japan, September 21–24, 1999* (IEEE Press, Washington, DC, 1999), pp. 58–67.

5. N. R. Adiga, M. A. Blumrich, D. Chen, et al., "Blue Gene/L Torus Interconnection Network," IBM J. Res. Develop. **49** (2/3), 265–276 (2005).

6. S. L. Scott and G. M. Thorson, "The Cray T3E Network: Adaptive Routing in a High Performance 3D Torus," in *Proc. IV Symp. on Hot Interconnects, Palo Alto, USA August 15–17, 1996* (IEEE Press, Washington, DC, 1996), pp. 147–156.