УДК 621.391

Трегубов Роман Борисович
ГКОУ ВПО «Академия Федеральной службы охраны Российской Федерации»
Россия, Орёл[1]
Кандидат технических наук
treba@list.ru

Лазарев Сергей Николаевич
ГКОУ ВПО «Академия Федеральной службы охраны Российской Федерации»
Россия, Орёл
Доцент
serg.orel@mail.ru

Андреев Сергей Юрьевич
ГКОУ ВПО «Академия Федеральной службы охраны Российской Федерации»
Россия, Орёл
Сотрудник
us12a@mail.ru

Алгоритм решения прикладной задачи теории графов о нахождении *k* минимальных остовных деревьев

[1] 302034, Орёл, ул. Приборостроительная, 35

Аннотация. В работе рассматривается приложение теории графов, для решения задачи маршрутизации в локальных вычислительных сетях Ethernet методом связующего дерева. Данный механизм позволяет исключить повторную передачу кадров (фреймов) в локальной вычислительной сети Ethernet по причине наличия в последней топологических петель. В отличие от существующих решений предлагается оригинальный алгоритм нахождения k минимальных остовных деревьев. Описываемый подход представляет собой комплексное применение в едином оптимизационном цикле алгоритма Прима (Краскала) и алгоритма построения усеченного дерева состояний в ширину. Следует отметить, что введение дополнительного ограничения (относительно допустимого ранга узлов остовного дерева) на одном из шагов алгоритма позволяет использовать данный подход и для решения задачи нахождения кратчайших гамильтоновых маршрутов (цепей). В такой постановке предлагаемый алгоритм может найти применение и в задачах резервирования, использующих коллективные (сетевые) механизмы введения избыточности. Суть которых заключается в выделении на топологической структуре сети связи цикла или цепи с предварительно рассчитанной резервной пропускной способностью, которая будет использоваться в случае возникновения отказа отдельных сетевых элементов.

Ключевые слова: локальная вычислительная сеть Ethernet; маршрутизация; коммутация; связный граф; матрица смежности; минимальное остовное дерево; алгоритм Прима; алгоритм Краскала; алгоритм поиска в ширину; гамильтонова цепь.

Введение

Практика планирования и проектирования инфокоммуникационных систем (ИКС) различного назначения в настоящее время остро нуждается в научно-методическом инструментарии, соответствующем современному уровню развития телекоммуникационных и информационных технологий, аппаратно-программных средств связи. При этом в качестве теоретического базиса разработки инновационного инструментария служат классические средства анализа и синтеза сетевых структур, в том числе методы теории графов.

В настоящей работе рассматривается приложение теории графов для решения задач проектирования ИКС, в которых предусматриваются соединения типа "точка-многоточка". В ИКС такие соединения используются как при организации конференций различного типа, так и в ходе функционирования подсистем маршрутизации, резервирования, синхронизации и др. Так, например, для управления процедурой маршрутизации в современных локальных сетях *Ethernet* используется протокол связующего дерева (*STP*) или его модификации *PVSTP, RSTP* и *MSTP* [1–3]. Основное назначение протокола *STP* – это удаления циклов (петель коммутации) из топологии локальных сетей *Ethernet* на канальном уровне. Необходимость устранения топологических петель в локальных сетях *Ethernet* следует из того, что их наличие приводит к бесконечным повторам передачи одних и тех же кадров, отчего имеющаяся пропускная способность локальной сети оказывается почти полностью занятой.

Протокол *STP* использует алгоритм *STA*, результатом работы которого является связный граф в виде дерева. Отметим, во-первых, связный граф, найденный с помощью алгоритма *STA*, не является минимальным, во-вторых, если необходимо организовать соединение типа "многоточка" с учетом дополнительного ограничения (например, на величину сетевой задержки) следует знать уже не одно дерево, а несколько конкурирующих деревьев, при этом выбор лучшего из них делается в соответствии с дополнительным ограничением.

Ниже предлагается алгоритм нахождения k минимальных остовных деревьев в порядке возрастания их длины, который позволяет решить данную задачу.

Алгоритм нахождения k минимальных остовных деревьев

В качестве примера целесообразно рассмотреть неориентированный граф (рис. 1), представленный с помощью двух матриц смежности, в одной из которых систематизированы номера узлов и ребер (в случае ориентированного графа – дуг), а в другой – их длины.

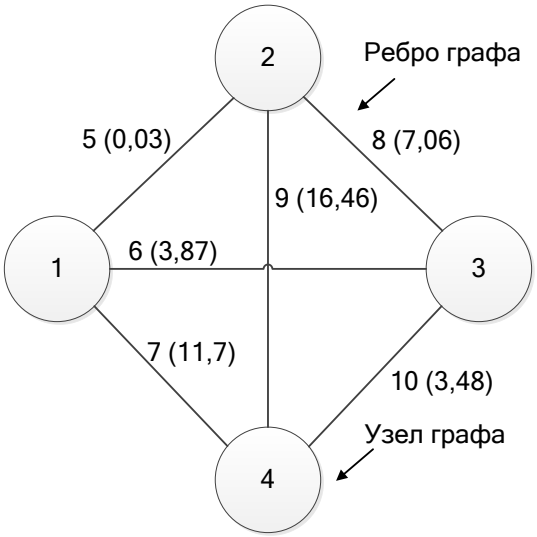

Рис. 1. *Неориентированный граф с пронумерованными элементами*

Матрица смежности с номерами узлов и ребер (дуг), их соединяющих (если элемента нет, тогда его номер равен нулю), имеет вид

$$M_{\text{номера}}^{\text{смежн}} = \begin{vmatrix} 1 & 5 & 6 & 7 \\ 5 & 2 & 8 & 9 \\ 6 & 8 & 3 & 10 \\ 7 & 9 & 10 & 4 \end{vmatrix}. \quad (1)$$

Матрица смежности, в которой представлены длины узлов и ребер (дуг), их соединяющих (если элемента нет или он находится в неисправном состоянии, тогда его длина равна бесконечности), имеет вид

$$M_{\text{длина}}^{\text{смежн}} = \begin{vmatrix} 0 & 0{,}03 & 3{,}87 & 11{,}7 \\ 0{,}03 & 0 & 7{,}06 & 16{,}46 \\ 3{,}87 & 7{,}06 & 0 & 3{,}48 \\ 11{,}7 & 16{,}46 & 3{,}48 & 0 \end{vmatrix}. \quad (2)$$

Для удобства описания алгоритма введем ряд определений и одну процедуру.

Определение 1. Минимальное остовное дерево (МОД) – это подмножество рёбер (дуг) графа и всех его вершин, таких, что из любой вершины графа можно попасть в любую другую вершину, двигаясь по этим рёбрам (дугам), и в нём нет циклов, то есть из любой вершины нельзя попасть в саму себя, не пройдя какое-то ребро дважды [4, 5].

Определение 2. Ветвь дерева состояний графа – это последовательность элементов графа (узлов и ребер), которая описывает группу состояний графа (последний элемент в ветви это длина минимального МОД для данное группы состояний, если такового МОД не существует, тогда длина равна бесконечности).

Заметим, что часть элементов графа могут находиться в исправном состоянии (положительные значения), а часть элементов – в неисправном состоянии (отрицательные значения). Элементы графа, которые не участвуют в формировании ветви дерева состояний, могут находиться в любом состоянии, как в исправном, так и в неисправном (рис. 2).

Пример ветви дерева состояний графа

$$vet_i = [5; 8; -6; 10; 10{,}57]. \quad (3)$$

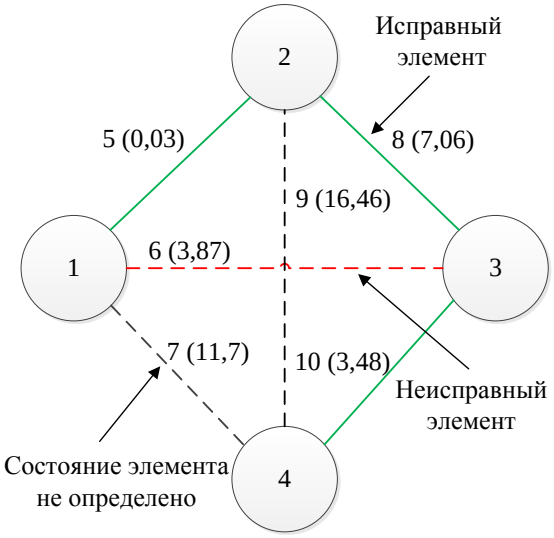

Рис. 2. *Граф, соответствующий ветви дерева состояний (выражение (*3*))*

Определение 3. Ответвлением от ветви дерева состояний графа будем называть новую ветвь дерева состояний, которая получается путем перевода одного из исправных элементов исходной ветви (кроме элемента, в котором хранится длина МОД) в неисправное состояние (отрицательное значение), при этом все последующие элементы, которые идут после него в исходной ветви, из новой ветви исключаются. Последний элемент новой ветви дерева состояний графа равен бесконечности.

Пример формирования ответвления от ветви дерева состояний (см. выражение (3))

$$otk_i = [5; 8; -6; -10; \infty]. \quad (4)$$

Определение 4. Список ответвлений от ветви дерева состояний графа получается путем получения всех возможных ответвлений от исправных элементов исходной ветви, начиная с предпоследнего (последний элемент – это длина МОД), и так далее вплоть до первого неисправного элемента исходной ветви состояний, если неисправных элементов в исходной ветви дерева состояний нет, тогда вплоть до первого элемента исходной ветви.

Исходная ветвь дерева состояний графа

$$vet_j = [5; 6; 10; 7{,}38], \quad (5)$$

Список возможных ответвлений от исходной ветви

$$otk_1 = [5; 6; -10; \infty], \quad (6)$$

$$otk_2 = [5; -6; \infty], \quad (7)$$

$$otk_2 = [-5; \infty]. \quad (8)$$

Процедура. Под формированием новых конкурирующих ветвей дерева состояний графа, на основе списка ответвлений будем понимать следующую процедуру. Для каждого ответвления из списка необходимо найти МОД, используя алгоритм Прима (Краскала) [6–9]. При этом необходимо учитывать, что часть элементов графа может находиться в неисправном состоянии – это отрицательные номера элементов, для таких элементов нужно заменить в исходной матрице $M^{\text{смежн}}_{\text{длина}}$ их реальную длину значением бесконечность (после того, как МОД будет найдено или же не найден, снова восстановить исходные значения длин для соответствующих элементов). Если МОД найдено, тогда с помощью его элементов

необходимо дописать соответствующую строку списка ответвлений, при этом повторяющиеся элементы не пишутся, а в качестве последнего элемента ветви следует записать длину найденного МОД. Если дерево состояний графа не найдено, тогда невозможно сформировать конкурирующую ветвь для текущего состояния графа.

Ниже представлены новые ветви дерева состояний графа для списка ответвлений (см. выражения (6)–(8))

$$vet_k = [5; 6; -10; 7; 15{,}6], \quad (9)$$

$$vet_{k+1} = [5; -6; 10; 8; 10{,}57], \quad (10)$$

$$vet_{k+2} = [-5; 6; 10; 8; 14{,}41]. \quad (11)$$

Введенные выше определения и процедура позволяют сформулировать обобщенный алгоритм нахождения k минимальных остовных деревьев.

I. Для исходного графа найти минимальное остовное дерево, используя алгоритм Прима (Краскала).

II. Найденное остовное дерево необходимо записать в списки конкурирующих ветвей дерева состояний и конкурирующих деревьев. Списки ответвлений и МОД на этом шаге не содержат строк (эти списки пустые).

III. Подсчитать число строк в списке МОД, если оно равно числу k, тогда работа алгоритма заканчивается.

IV. Если список конкурирующих ветвей дерева состояний не содержит строк (список пустой), тогда работа алгоритма заканчивается. Иначе в списке конкурирующих ветвей необходимо найти строку с наименьшей длиной (минимальное значение последнего элемента ветви дерева состояний). На ее основе сформировать список ответвлений и удалить эту ветвь из списка конкурирующих ветвей дерева состояний графа. Соответствующее ей остовное дерево из списка конкурирующих деревьев необходимо сравнить с уже имеющимися деревьями в списке МОД и, если такового дерева нет, записать его в список МОД, а из списка конкурирующих деревьев соответствующую строку удалить. Если же такое остовное дерево уже есть, тогда нужно просто удалить соответствующую строку из списка конкурирующих деревьев.

V. Сформировать новые конкурирующие ветви дерева состояний графа на основе имеющегося списка ответвлений. При этом если для очередного ответвления остовное дерево найдено, тогда оно записывается последним в список конкурирующих деревьев, а полученная с его помощью новая ветвь дерева состояний графа записывается в список конкурирующих ветвей тоже в качестве последнего элемента, из списка ответвлений оно удаляется. Если остовное дерево не найдено, тогда соответствующая строка просто удаляется из списка ответвлений. По окончании работы данного шага алгоритма список ответвлений не должен содержать строк (список должен быть пустой).

VI. Переход на III шаг алгоритма.

Задача нахождения k гамильтоновых маршрутов (цепей) [10, 11] является частным случаем решения задачи нахождения k минимальных остовных деревьев. Для этого на IV шаге алгоритма вводится дополнительное ограничение: для того, чтобы записать остовное дерево в список МОД должно выполнятся условие $d_i \le 2$, где d_i – ранг i-го узла графа.

Далее работа предлагаемого алгоритма иллюстрируется на конкретном примере сети, представленной на рисунке 1. Пусть необходимо найти четыре МОД в графе в порядке возрастания их длины ($k = 4$).

Шаг 1. Список конкурирующих ветвей на первом шаге (рис. 3)

$$vet_1^* = [5; 6; 10; 7{,}38]. \quad (12)$$

Список ответвлений на первом шаге не содержит элементов

$$otk_1 = [\varnothing]. \quad (13)$$

Список конкурирующих остовных деревьев на первом шаге

$$der_1 = [5; 6; 10; 7{,}38]. \quad (14)$$

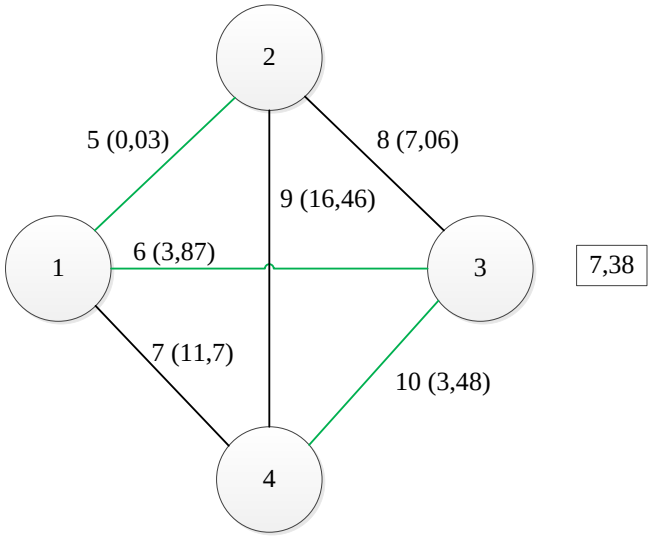

Рис. 3. *МОД, найденное на первом шаге*

Список МОД на первом шаге

$$mod_1 = [\varnothing]. \quad (15)$$

Шаг 2. Список конкурирующих ветвей на втором шаге

$$vet_1 = [\varnothing]. \quad (16)$$

Список ответвлений на втором шаге

$$otk_1 = [5; 6; -10; \infty], \quad (17)$$

$$otk_2 = [5; -6; \infty], \quad (18)$$

$$otk_3 = [-5; \infty]. \quad (19)$$

Список конкурирующих остовных деревьев на втором шаге

$$der_1 = [\varnothing]. \quad (20)$$

Список МОД на втором шаге

$$mod_1 = [5; 6; 10; 7{,}38]. \quad (21)$$

Шаг 3. Список конкурирующих ветвей на третьем шаге (рис. 4–6)

$$vet_1 = [5; 6; -10; 7; 15{,}6], \quad (22)$$

$$vet_2* = [5; -6; 8; 10; 10{,}57], \quad (23)$$

$$vet_3 = [-5; 6; 8; 10; 14{,}41]. \quad (24)$$

Список ответвлений на третьем шаге

$$otk_1 = [\varnothing]. \quad (25)$$

Список конкурирующих остовных деревьев на третьем шаге

$$der_1 = [5; 6; 7; 15{,}6], \quad (26)$$

$$der_2* = [5; 8; 10; 10{,}57], \quad (27)$$

$$der_3 = [6; 8; 10; 14{,}41]. \quad (28)$$

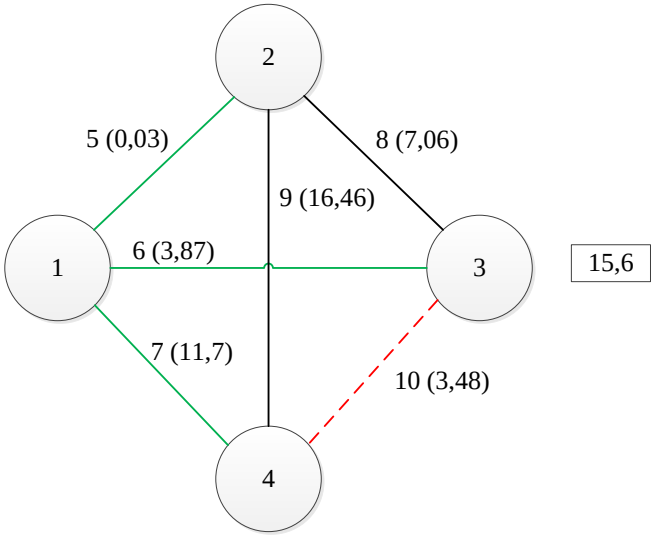

Рис. 4. *Первое МОД, найденное на третьем шаге*

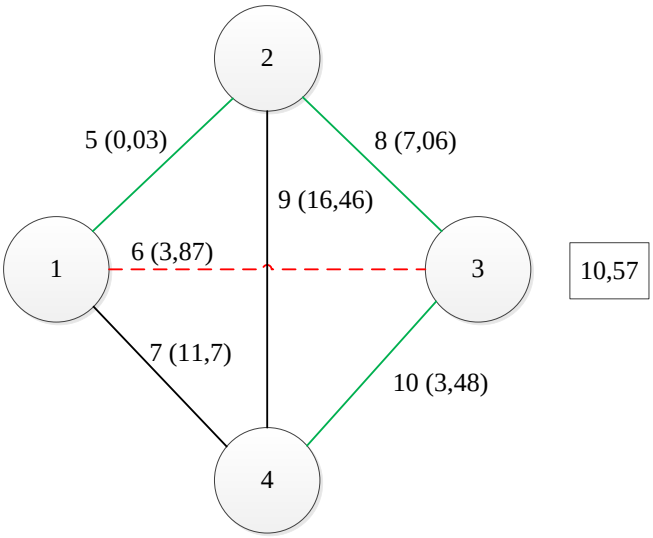

Рис. 5. *Второе МОД, найденное на третьем шаге*

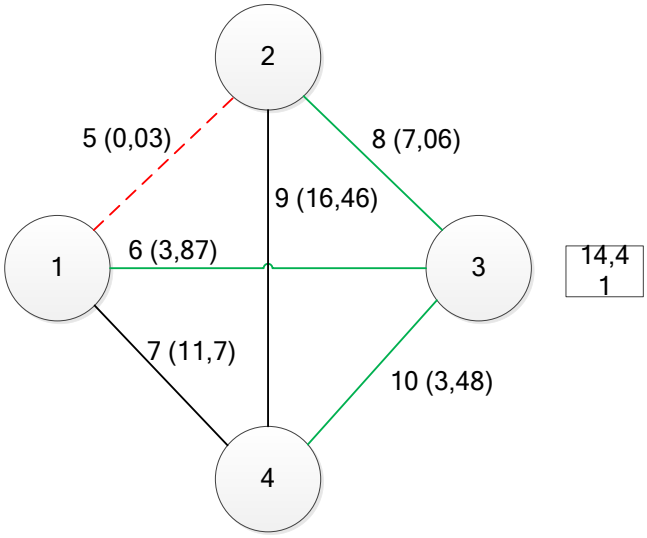

Рис. 6. *Третье МОД, найденное на третьем шаге*

Список МОД на третьем шаге

$$mod_1 = [5; 6; 10; 7{,}38]. \quad (29)$$

Шаг 4. Список конкурирующих ветвей на четвертом шаге

$$vet_1 = [5; 6; -10; 7; 15{,}6], \quad (30)$$

$$vet_2 = [-5; 6; 8; 10; 14{,}41]. \quad (31)$$

Список ответвлений на четвертом шаге

$$otk_1 = [5; -6; 8; -10; \infty], \quad (32)$$

$$otk_2 = [5; -6; -8; \infty]. \quad (33)$$

Список конкурирующих остовных деревьев на четвертом шаге

$$der_1 = [5; 6; 7; 15{,}6], \quad (34)$$

$$der_2 = [6; 8; 10; 14{,}41]. \quad (35)$$

Список МОД на четвертом шаге

$$mod_1 = [5; 6; 10; 7{,}38], \quad (36)$$

$$mod_2 = [5; 8; 10; 10{,}57] \quad .(37)$$

Шаг 5. Список конкурирующих ветвей на пятом шаге (рис. 7–8)

$$vet_1 = [5; 6; -10; 7; 15{,}6], \quad (38)$$

$$vet_2^* = [-5; 6; 8; 10; 14{,}41], \quad (39)$$

$$vet_3 = [5; -6; 8; -10; 7; 18{,}79], \quad (40)$$

$$vet_4 = [5; -6; -8; 7; 10; 15{,}21]. \quad (41)$$

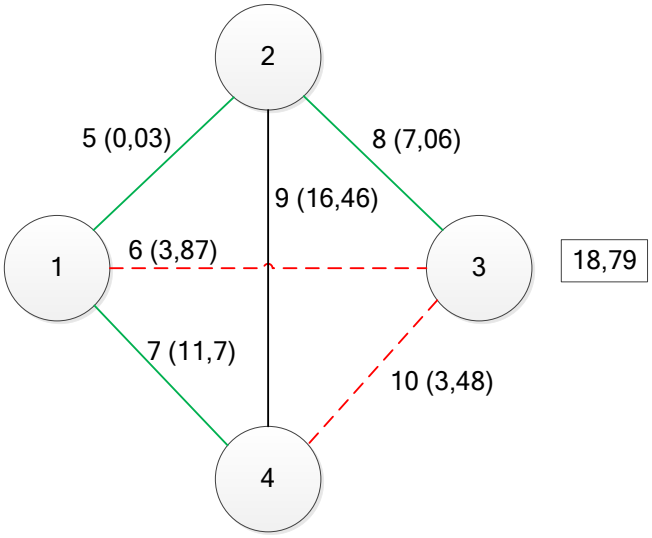

Рис. 7. *Первое МОД, найденное на пятом шаге*

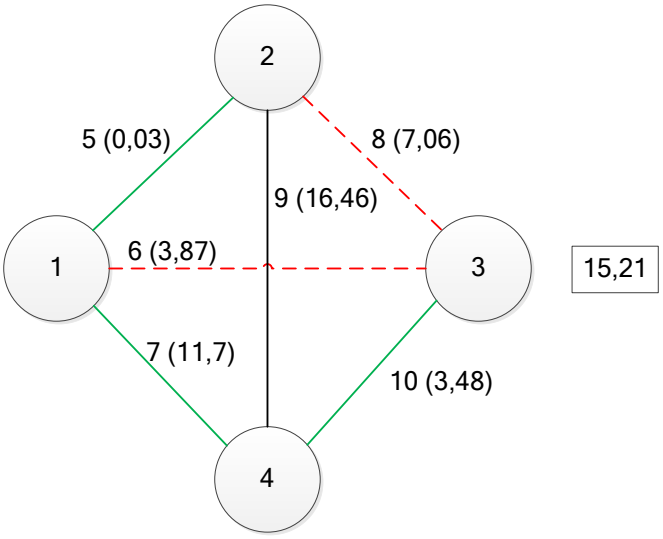

Рис. 8. *Второе МОД, найденное на пятом шаге*

Список ответвлений на пятом шаге

$$otk_1 = [\emptyset]. \quad (42)$$

Список конкурирующих остовных деревьев на пятом шаге

$$der_1 = [5; 6; 7; 15{,}6], \quad (43)$$

$$der_2^* = [6; 8; 10; 14{,}41], \quad (44)$$

$$der_3 = [5; 8; 7; 18{,}79], \quad (45)$$

$$der_4 = [5; 7; 10; 15{,}21]. \quad (46)$$

Список МОД на пятом шаге

$$mod_1 = [5; 6; 10; 7{,}38], \quad (47)$$

$$mod_2 = [5; 8; 10; 10{,}57]. \quad (48)$$

Шаг 6. Список конкурирующих ветвей на шестом шаге

$$vet_1 = [5; 6; -10; 7; 15{,}6], \quad (49)$$

$$vet_2 = [5; -6; 8; -10; 7; 18{,}79], \quad (50)$$

$$vet_3 = [5; -6; -8; 7; 10; 15{,}21]. \quad (51)$$

Список ответвлений на шестом шаге

$$otk_1 = [-5; 6; 8; -10; \infty], \quad (52)$$

$$otk_2 = [-5; 6; -8; \infty], \quad (53)$$

$$otk_3 = [-5; -6; \infty]. \quad (54)$$

Список конкурирующих остовных деревьев на шестом шаге

$$der_1 = [5; 6; 7; 15{,}6], \quad (55)$$

$$der_2 = [5; 8; 7; 18{,}79], \quad (56)$$

$$der_3 = [5; 7; 10; 15{,}21]. \quad (2.58)$$

Список МОД на шестом шаге

$$mod_1 = [5; 6; 10; 7{,}38], \quad (57)$$

$$mod_2 = [5; 8; 10; 10{,}57], \quad (58)$$

$$mod_3 = [6; 8; 10; 14{,}41]. \quad (59)$$

Шаг 7. Список конкурирующих ветвей на седьмом шаге (рис. 9–11)

$$vet_1 = [5; 6; -10; 7; 15{,}6], \quad (60)$$

$$vet_2 = [5; -6; 8; -10; 7; 18{,}79], \quad (61)$$

$$vet_3^* = [5; -6; -8; 7; 10; 15{,}21], \quad (62)$$

$$vet_4 = [-5; 6; 8; -10; 7; 22{,}63], \quad (63)$$

$$vet_5 = [-5; 6; -8; 10; 9; 23{,}81], \quad (64)$$

$$vet_6 = [-5; -6; 7; 10; 8; 22{,}24]. \quad (65)$$

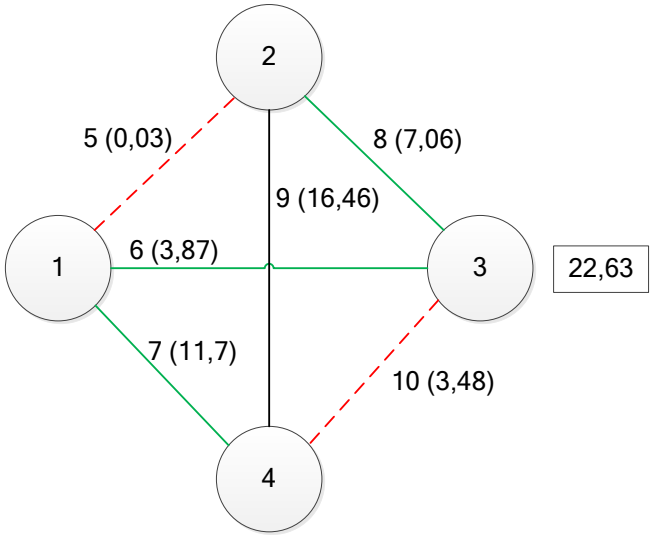

Рис. 9. *Первое МОД, найденное на седьмом шаге*

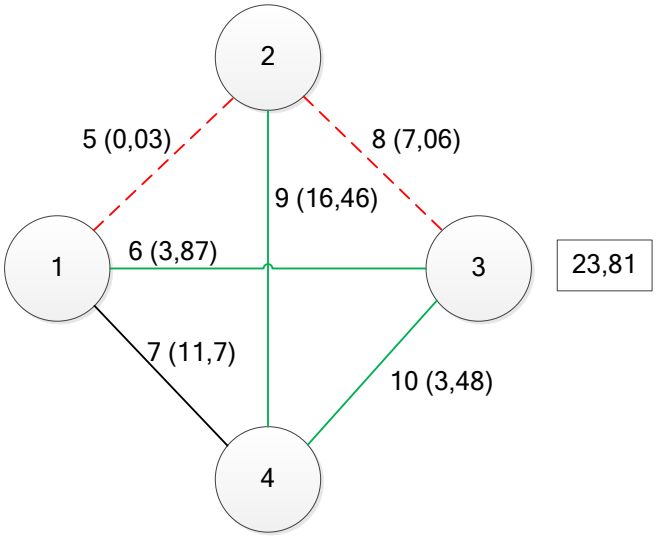

Рис. 10. *Второе МОД, найденное на седьмом шаге*

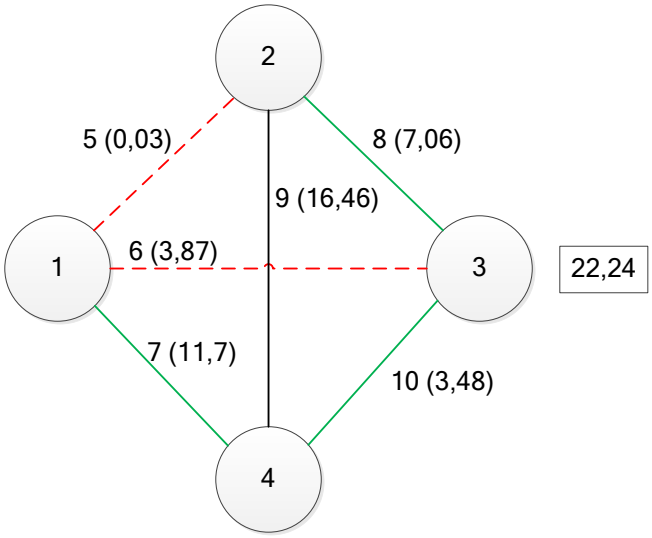

Рис. 11. *Третье МОД, найденное на седьмом шаге*

Список ответвлений на седьмом шаге

$$otk_1 = [\emptyset]. \quad (66)$$

Список конкурирующих остовных деревьев на седьмом шаге

$$der_1 = [5; 6; 7; 15{,}6], \quad (67)$$

$$der_2 = [5; 8; 7; 18{,}79], \quad (68)$$

$$der_3^* = [5; 7; 10; 15{,}21], \quad (69)$$

$$der_4 = [6\ ;8; 7; 22{,}63], \quad (70)$$

$$der_5 = [6; 10; 9; 23{,}81], \quad (71)$$

$$der_6 = [7; 10; 8; 22{,}24]. \quad (72)$$

Список МОД на седьмом шаге

$$mod_1 = [5; 6; 10; 7{,}38], \quad (73)$$

$$mod_2 = [5; 8; 10; 10{,}57], \quad (74)$$

$$mod_3 = [6\ ;8; 10; 14{,}41]. \quad (75)$$

Шаг 8. Список конкурирующих ветвей на восьмом шаге

$$vet_1 = [5; 6; -10; 7; 15{,}6], \quad (76)$$

$$vet_2 = [5; -6; 8; -10; 7; 18{,}79], \quad (77)$$

$$vet_3 = [-5; 6\ ;8; -10; 7; 22{,}63], \quad (78)$$

$$vet_4 = [-5; 6\ ;-8; 10; 9; 23{,}81], \quad (79)$$

$$vet_5 = [-5; -6; 7; 10; 8; 22{,}24]. \quad (80)$$

Список ответвлений на восьмом шаге

$$otk_1 = [5; -6; -8; 7; -10; \infty], \quad (81)$$

$$otk_2 = [5; -6; -8; -7; \infty]. \quad (82)$$

Список конкурирующих остовных деревьев на восьмом шаге

$$der_1 = [5; 6; 7; 15{,}6], \quad (83)$$

$$der_2 = [5; 8; 7; 18{,}79], \quad (84)$$

$$der_3 = [6\ ;8; 7; 22{,}63], \quad (85)$$

$$der_4 = [6; 10; 9; 23{,}81], \quad (86)$$

$$der_5 = [7; 10; 8; 22{,}24]. \quad (87)$$

Список МОД на восьмом шаге

$$mod_1 = [5; 6; 10; 7{,}38], \quad (88)$$

$$mod_2 = [5; 8; 10; 10{,}57], \quad (89)$$

$$mod_3 = [6; 8; 10; 14{,}41], \quad (90)$$

$$mod_4 = [5; 7; 10; 15{,}21]. \quad (91)$$

Шаг 9. Работа алгоритма заканчивается, так как были найдены четыре МОД для исходного графа в порядке возрастания их длины.

Обоснование эффективности предлагаемого в работе алгоритма основано на очевидном факте, длины новых ветвей дерева событий, получаемые на V шаге алгоритма, не могут быть меньше длины исходной ветви, ответвлениями от которой они являются.

Заключение

Предлагаемый в работе алгоритм поиска нескольких минимальных остовных деревьев целесообразно применять для решения задач проектирования информационных инфраструктур, в которых предусматриваются соединения типа "точка-многоточка" с учетом дополнительного ограничения (например, на величину сетевой задержки). В данной ситуации проектировщику необходимо определить несколько конкурирующих деревьев, при этом выбор лучшего из них будет делаться в соответствии с дополнительным ограничением.

Этот алгоритм может также применяться для анализа устойчивости алгоритмов Прима и Краскала, так как с его помощью можно определить насколько изменится оптимальное решение, полученное с помощью данных алгоритмов, в случае изменения некоторых входных параметров модели сети связи.

Также с помощью данного алгоритма было выявлено, что при организации процедуры ветвления предлагаемой в [5] (только для узлов графа с рангом три и более), в ряде случаев найденное решение (гамильтонов маршрут или цепь) не является оптимальным.

ЛИТЕРАТУРА

1. Маршрутизаторы Cisco. Пособие для самостоятельного изучения / пер. с англ. – П. Шера; под ред. А. Галунова. – СПб.: Символ-Плюс, 2003. – 512 с.
2. Программа сетевой академии Cisco CCNA 1 и 2. Вспомогательное руководство, 3-е изд., с испр. / пер. с англ. С. Балицкого, Г. Клапанова, А. Н. Крикуна; под ред. А. В. Мысника. – М.: Издательский дом "Вильямс", 2008. – 1168 с.
3. Уолрэнд, Дж. Телекоммуникационные и компьютерные сети. Вводный курс / Дж. Уолрэнд. – М.: Постмаркет, 2001. – 480 с.
4. Берж К. Теория графов и ее применение / К. Берж. – М.: Издательство иностранной литературы, 1962. – 319 с.
5. Кристофидес Н. Теория графов. Алгоритмический подход. / Н. Кристофидес – М.: Издательство "Мир", 1978 – 430 с.
6. Таха, Хэмди А. Введение в исследование операция. 6-е издание / Хэмди А. Таха; пер. с англ. – М.: Издательский дом "Вильямс", 2001. – 915 с.
7. Prim R. C. Shortest connection networks and some generalizations. In: Bell System Technical Journal, 36 (1957), P. 1389–1401.
8. Joseph. B. Kruskal. On the Shortest Spanning Subtree of a Graph and the Traveling Salesman Problem. // Proc. AMS. 1956. Vol 7, No. 1. P. 48–50.
9. Кормен, Т. Алгоритмы: построение и анализ. 3-е издание / Т. Кормен, Ч. Лейзерсон, Р. Ривест, К. Штайн. – М.: "Вильямс", 2013. – 1323 с.
10. Асанов М. О. Дискретная математика: графы, матроиды, алгоритмы / М. О. Асанов, В. А. Баранский, В. В. Расин. – Ижевск: НИЦ "Регулярная и хаотическая динамика", 2001.
11. Филлипс Д. Методы анализа сетей / Д. Филипс, А. Гарсиа-Диас – М.: Издательство "Мир", 1984. – 496 с.

Рецензент: Еременко Владимир Тарасович, заведующий кафедрой "Электроника, вычислительная техника и информационная безопасность", доктор технических наук, профессор, ФГБОУ ВПО "Государственный университет – УНПК", Россия, Орёл.

Roman Tregubov
The Academy of the Federal Guard Service of the Russian Federation
Russia, Orel
treba@list.ru

Sergej Lazarev
The Academy of the Federal Guard Service of the Russian Federation
Russia, Orel
serg.orel@mail.ru

Sergej Andreev
The Academy of the Federal Guard Service of the Russian Federation
Russia, Orel
us12a@mail.ru

Solution algorithm of the k-minimum spanning trees finding graph theory task applied

Abstract. In work the application of the graph theory for the solution of the task of routing on the Ethernet local computer networks by method of a span tree is considered. This mechanism allows to exclude repeated transmission of frames on the Ethernet local computer network because of existence in it topological loops. Unlike the existing decisions the original algorithm of finding of the k-minimum spanning trees is offered. The described approach represents complex application in a uniform optimization cycle of algorithm Prima (Kraskala) and algorithm of creation of a sectional tree of statuses in width. We will note that introduction of additional restriction (rather admissible rank of nodes of a spanning tree) on one of steps of algorithm allows to use this approach and for the solution of the task of finding of the shortest Hamilton routes (circuits). In such setting the offered algorithm can find application in the tasks of reservation, which using collective (network) mechanisms of introduction of redundancy. Which entity consists in separation on topological structure of a communication network of a cycle or circuit with beforehand the calculated reserve throughput, which will be used in case of failure occurrence of separate network elements.

Keywords: Ethernet local computer network, routing; switching; connected graph; connectivity matrix; minimum spanning tree; algorithm Prima; Kraskal's algorithm; algorithm of the search in width; Hamilton circuit.

REFERENCES

1. Marshrutizatory Cisco. Posobie dlja samostojatel'nogo izuchenija / per. s angl. – P. Shera; pod red. A. Galunova. – SPb.: Simvol-Pljus, 2003. – 512 s.
2. Programma setevoj akademii Cisco CCNA 1 i 2. Vspomogatel'noe rukovodstvo, 3-e izd., s ispr. / per. s angl. S. Balickogo, G. Klapanova, A. N. Krikuna; pod red. A. V. Mysnika. – M.: Izdatel'skij dom "Vil'jams", 2008. – 1168 s.
3. Uolrjend, Dzh. Telekommunikacionnye i komp'juternye seti. Vvodnyj kurs / Dzh. Uolrjend. – M.: Postmarket, 2001. – 480 s.
4. Berzh K. Teorija grafov i ee primenenie / K. Berzh. – M.: Izdatel'stvo inostrannoj literatury, 1962. – 319 s.
5. Kristofides N. Teorija grafov. Algoritmicheskij podhod. / N. Kristofides – M.: Izdatel'stvo "Mir", 1978 – 430 s.
6. Taha, Hjemdi A. Vvedenie v issledovanie operacija. 6-e izdanie / Hjemdi A. Taha; per. s angl. – M.: Izdatel'skij dom "Vil'jams", 2001. – 915 s.
7. Prim R. C. Shortest connection networks and some generalizations. In: Bell System Technical Journal, 36 (1957), P. 1389–1401.
8. Joseph. B. Kruskal. On the Shortest Spanning Subtree of a Graph and the Traveling Salesman Problem. // Proc. AMS. 1956. Vol 7, No. 1. P. 48-50.
9. Kormen, T. Algoritmy: postroenie i analiz. 3-e izdanie / T. Kormen, Ch. Lejzerson, R. Rivest, K. Shtajn. – M.: "Vil'jams", 2013. – 1323 s.
10. Asanov M. O. Diskretnaja matematika: grafy, matroidy, algoritmy / M. O. Asanov, V. A. Baranskij, V. V. Rasin. – Izhevsk: NNC "Reguljarnaja i haoticheskaja dinamika", 2001.
11. Fillips D. Metody analiza setej / D. Filips, A. Garsia-Dias – M.: Izdatel'stvo "Mir", 1984. – 496 s.