УДК 519.176
ББК 22.176

АЛГОРИТМ ВЕТВЕЙ И ГРАНИЦ В ЗАДАЧЕ ОБ ОПТИМАЛЬНОЙ СВЯЗЫВАЮЩЕЙ СЕТИ[1]

Гинз В.Н.[2], **Губко М.В.**[3]
(*Учреждение Российской академии наук Институт проблем управления им. В.А. Трапезникова РАН, Москва*)

Рассматривается модель оптимизации связывающей сети в условиях т.н. «аддитивной» функции затрат, когда стоимость вершины, добавляемой для маршрутизации потоков между фиксированными основными вершинами, зависит от количества связей этой вершины и от суммарного объема протекающего через нее потока. Для фиксированной древовидной топологии вершин-коммутаторов предлагается алгоритм ветвей и границ для поиска оптимального распределения основных вершин по коммутирующим вершинам. Используемая алгоритмом нижняя оценка затрат сети основана на непрерывной релаксации и линеаризации задачи, а также на результатах алгебраической теории графов.

Ключевые слова: оптимизация структуры, связывающая сеть, нижние оценки, алгоритм ветвей и границ.

Введение

Многие природные и искусственные системы имеют сетевую структуру. Нас окружают сети – сети связи, транспортные сети, социальные сети и многие другие. Эффективность функционирования сети в большой степени определяется её строением.

[1] *Работа выполнена при финансовой поддержке РФФИ, грант №13-07-00491а*

[2] *Василий Николаевич Гинз, инженер* (*vasikos@yandex.ru*)

[3] *Михаил Владимирович Губко, кандидат технических наук,* (*mgoubko@mail.ru*)

В связи с этим возникает задача оптимизации внутренней структуры сети.

В [1, 2] была предложена модель оптимизации связывающей сети. Задача состоит в построении связывающей сети в соответствии с потребностями в связи. В рамках этой модели были предложены нижние оценки затрат сети для частного случая функции затрат: так называемой аддитивной функции. Эти оценки основаны на результатах алгебраической теории графов. Особенностью оценок является то, что они подходят только для замкнутой сети, то есть сети, не допускающей потоков, выходящих за ее пределы.

В настоящем докладе статье описываются модификации этих оценок, позволяющие использовать их в алгоритме ветвей и границ. Также предлагается схема алгоритма ветвей и границ для нахождения оптимального распределения основных вершин по дополнительным в сети с заданной топологией.

1. Постановка задачи

Рассматриваемая в настоящей работе постановка задачи является частным случаем предложенной в [1] *модели связывающей сети*.

Пусть задано *множество основных вершин* $W = \{1, ..., n\}$, которые необходимо соединить *каналами связи*. Через $x_{ww'} \in [0, +\infty)$ обозначим объем потока (информационного или материального), который должен протекать между вершинами $w \in W$ и $w' \in W$, а через $R = (x_{ww'})_{w,w'=1}^{n}$ – матрицу объемов. Если $x(w, w') > 0$, будем говорить, что вершины w и w' *испытывают потребность в связи*.

Непосредственно соединять все пары вершин, испытывающие потребность в связи, может оказаться накладым, и близкие потоки группируются в одном канале связи.

Пусть G – простой неориентированный граф G с множеством вершин $V(G)$. Для произвольной вершины $v \in V(G)$ обозначим через $d_G(v)$ *степень* вершины v в графе G, то есть

количество инцидентных ей ребер. Если $d_G(v) = 1$, вершина $v \in V(G)$ называется *висячей*.

Определение 1. *Простой неориентированный граф H с множеством вершин V_H и множеством дуг E_H будем называть связывающей сетью над множеством основных вершин W, если граф H связен, $W \subseteq V_H$ и основные вершины являются висячими, то есть $d_H(w) = 1$ для всех $w \in W$.*

Вершины множества $M_H = V_H \backslash W$ будем называть *дополнительными*. Дополнительные вершины занимаются маршрутизацией потоков. Будем считать, что основные вершины не обладают способностью к маршрутизации, именно поэтому в связывающей сети они являются висячими вершинами.

Добавление каждой дополнительной вершины $m \in M_H$ в связывающую сеть H влечет некоторые затраты $c(m, H)$, и *затраты* связывающей сети $(H) = \sum^{m \in M_H} c(m, H)$ складываются из затрат дополнительных вершин.

Определение 2. *Связывающим деревом называется связывающая сеть без циклов. Множество связывающих деревьев над множеством основных вершин W обозначим через $\mathcal{T}(W)$.*

В произвольном связывающем дереве $T \in \mathcal{T}(W)$ существует единственный путь между каждой парой основных вершин, поэтому суммарный объем потока $x_T(m)$ через любую дополнительную вершину $m \in M_T$ связывающего дерева T полностью определяется структурой дерева. Для краткости обозначим $d := d_T(m)$. Если из T удалить вершину m и инцидентные ей ребра, получим d компонент связности. Если обозначить через $s_1, ..., s_d \subseteq W$ множества основных вершин в каждой из этих d компонент, то легко видеть, что суммарный поток через вершину m в связывающем дереве T записывается как

$$x_T(m) = \sum_{w,w \in W} x_{ww'} - \sum_{i=1}^{d} \sum_{w,w' \in s_i} x_{ww'}, \tag{1}$$

то есть через вершину m текут все потоки кроме тех, что текут внутри компонент связности, на которые удаление вершины m разбивает дерево T.

Определение 3. *Функцию затрат вершины будем называть аддитивной, если для произвольной вершины m связывающего дерева T она записывается как $c(m,T) = c_1(d_T(m)) + c_2(x_T(m))$, где $c_1(\cdot)$ и $c_2(\cdot)$ – некоторые неотрицательные функции, а $x_T(m)$ определяется выражением* (1).

Ниже будем решать задачу об оптимальном связывающем дереве для аддитивной функции затрат, которая состоит в поиске связывающего дерева с минимальными затратами, то есть дерева

$$(2)\quad T^* \in \mathrm{Argmin}_{T\in\mathcal{T}(W)} C(T) = \mathrm{Argmin}_{T\in\mathcal{T}(W)} \left[c_1(d_T(m)) + c_2(x_T(m))\right].$$

Очевидно, что достаточно рассматривать такие деревья $T \in \mathcal{T}(W)$, в которых произвольная дополнительная вершина $m \in M_T$ имеет степень $d_T(m) \geqslant 3$. Действительно, если вершина $m \in M_T$ висячая, то ее можно удалить вместе с инцидентным ребром, а в случае $d_H(m) = 2$ ее инцидентные вершины можно соединить напрямую, также удалив m вместе с ее инцидентными ребрами без нарушения связности дерева.

В [1, 2] описывается более общая модель оптимизации связывающей сети. В недревовидной сети между парой основных вершин может иметься несколько путей, и для однозначного определения потоков через вершины сети необходимо дополнительно вводить функции маршрутизации, определяющие путь, по которому протекает поток.

Однако даже более простая модель связывающего дерева позволяет описывать многие интересные прикладные задачи, такие как оптимизация индекса Винера для графа со взвешенными вершинами (см. подробнее в [5]) или описываемая ниже задача об оптимальной декомпозиции бизнес-процессов.

2. Алгоритм ветвей и границ оптимизации распределения основных вершин по дополнительным

Для фиксированного множества основных вершин W и матрицы связи R нахождение оптимального связывающего дерева T

сводится к определению:

1) Количества дополнительных вершин q в дереве T,

2) Топологии дополнительных вершин, то есть некоторого дерева T_q с множеством вершин $M = \{1, ..., q\}$,

3) Вектора $k = (k_1, ..., k_q)$, где k_m равно количеству основных вершин, инцидентных дополнительной вершине $m \in M$,

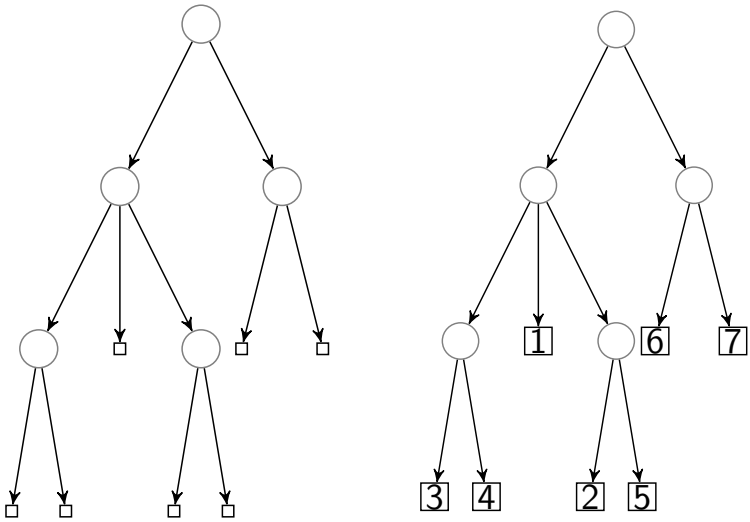
4) Распределения основных вершин по дополнительным, то есть такого разбиения множества W на непересекающиеся подмножества W_1, ..., W_q, что $|W_m| = k_m$, $m \in M$.

Искомое дерево T получается добавлением к дереву T_q мно жества вершин W и множества ребер mw, где $m \in M$, $w \in W_m$.

Полностью задача об оптимальном связывающем дереве пока не решена. Можно показать, что она является обобщением NP-трудной задачи о минимальном k-разрезе графа [3]. В настоящем докладе мы предлагаем алгоритм решения задачи этапа 4 – определения оптимального распределения основных вершин по дополнительным для фиксированного количества дополнительных вершин q, заданной их топологии T_q и вектора «емкостей» дополнительных вершин k. Даже эта задача является сложной, поскольку к ней сводится NP-полная в сильном смысле задача [6] об остове ограниченной пропускной способности.

Для поиска оптимального распределения мы предлагаем использовать алгоритм ветвей и границ.

Алгоритм ветвей и границ имеет структуру дерева над множеством возможных решений. Поиск решения начинается с корня дерева, и в каждой вершине множество разбивается на несколько частей. В отличие от полного перебора, этот алгоритм на основе сравнения вычисляемой нижней оценки с текущим *рекордом* – лучшим найденным решением – отсекает ветку (а вместе с ней и часть дерева). Таким образом уменьшается количество проверяемых решений.



а) б)

Рис. 1. Топология дополнительных вершин до (а) и после (б) распределения основных вершин.

Ветвление в предлагаемом нами алгоритме осуществляется выбором дополнительный вершины, на которую назначается основная вершина. То есть на каждом шаге алгоритма для очередной основной вершины перебираются дополнительные вершины, которым она еще может быть назначена исходя из уже назначенных вершин и вектора k.

Таким образом, путь от корня дерева ветвлений до листа описывает распределение всех основных вершин по дополнительным. Первый полностью построенный такой путь выбирается в качестве *рекорда*, а соответствующие затраты – в качестве рекордного значения затрат. Для сокращения числа ветвлений (для «отсечения») используется нижняя оценка затрат сети, построенная с учетом того, что распределение части основных вершин по дополнительным уже фиксировано. Если нижняя оценка превышает рекорд затрат, то эта ветвь “обрезается”, и дальнейшее

ветвление не производится. Иначе продолжается перебор дополнительных вершин для следующей основной вершины из еще не зафиксированных.

Если стоимость очередного полученного решения (напомним, что решению однозначно соответствует путь от корня до листа дерева ветвлений) меньше чем рекорд затрат, то рекорд обновляется новым решением.

В результате такого обхода дерева ветвлений рекорд будет соответствовать оптимальному распределению основных вершин по дополнительным.

3. Нижние оценки затрат связывающей сети

Для работы описанного алгоритма ветвей и границ требуется строить нижнюю оценку затрат связывающего дерева. В [2] были предложены нижние оценки, основанные на применении аппарата алгебраической теории графов. Однако непосредственно их нельзя использовать в алгоритме, поскольку при их расчете критично, что оцениваемая сеть замкнута, то есть все потоки протекают только между основными вершинами, принадлежащими сети, и являющихся предметом оптимизации.

В алгоритме же ветвей и границ распределение части основных вершин уже зафиксировано, и потоки между ними и еще не зафиксированными вершинами можно интерпретировать как внешние потоки, втекающие или вытекающие

Таким образом, необходима нижняя оценка затрат связывающего дерева, учитывающая то, что распределение части основных вершин по дополнительным уже фиксировано. Поскольку при фиксированной топологии роутеров и векторе k степени всех дополнительных вершин фиксированы, можно считать, что подлежащая оптимизации функция затрат $C(T) = C_2(T) = \sum^{m \in M_T} c_2(x_T(m))$.

3.1. ЛИНЕЙНАЯ ФУНКЦИЯ ЗАТРАТ

Введем индикаторную матрицу Y размера $n \times q$, элемент y_{wm} которой равен единице если вершина $w \in W$ инцидентна

$m \in M$, иначе равен нулю. На эту матрицу наложены следующие ограничения:

1) $Y^T \cdot 1_n = k$,

2) $Y \cdot 1_q = 1_n$,

3) $y_{wm} \in \{0, 1\}, w = 1, ..., n, m = 1, ..., q$.

Здесь 1_n - это вектор-столбец, состоящий из n единиц. Множество матриц, удовлетворяющих этим условиям, обозначим через Ξ.

Поскольку на каждом шаге алгоритма ветвей и границ могут быть основные вершины как с фиксированным, так и с не фиксированным распределением на дополнительные вершины, то представим матрицу Y как блочную $Y = \begin{bmatrix} Y_0 \\ Y_1 \end{bmatrix}$. Здесь $n_0 \times q$-матрица Y_0 описывает фиксированное распределение n_0 зафиксированных вершин, а $n_1 \times q$-матрица Y_1 – еще не зафиксированное распределение оставшихся $n_1 := n - n_0$ вершин.

Рассмотрим случай, когда функция $c_2(\cdot)$ линейна по потоку. Тогда, аналогично [2], затраты произвольного связывающего дерева T можно оценить снизу как

$$C_2(T) \geqslant \min_{Y_1 \in \Xi_1} [q 1_n^T R 1_n - tr(Y_0^T R_{00} Y_0 + 2 Y_0^T R_{01} Y_1 + Y_1^T R_{11} Y_1) P], \tag{3}$$

где $\Xi_1 = \{Y_1 : [Y_0 Y_1]^T \in \Xi\}$, $R = \begin{bmatrix} R_{00} & R_{01} \\ R_{10} & R_{11} \end{bmatrix}$, а матрица P – это положительно определенная $q \times q$-матрица, являющаяся дополнением матрицы D расстояний графа T_q (то есть $P = (q - 1)1_q 1_q^T - D$).

Поскольку минимум суммы не превышает суммы минимумов, то нижнюю оценку для C_2^* можно представить как

$$C_2(T) \geqslant q 1_n^T R 1_n - tr Y_0^T R_{00} Y_0 P - \max_{Y_1 \in \Xi_1} tr 2 Y_0^T R_{01} Y_1 P - \max_{Y_1 \in \Xi_1} tr Y_1^T R_{11} Y_1 P. \tag{4}$$

Таким образом, задача нахождения нижней оценки сводится к максимизации двух слагаемых из формулы (4).

Рассмотрим первое максимизируемое слагаемое:

$$(5)\quad \max_{Y_1\in\Xi_1} tr2Y_0^T R_{01}Y_1P \;\leqslant\; LB_1(Y_0) \;=\; \max_{Y_1\in\widetilde{\Xi}_1} tr2Y_0^T R_{01}Y_1P,$$

где $\widetilde{\Xi}_1$ – непрерывная релаксация множества Ξ_1, то есть $n_1 \times q$-матрица $Y_1 \in \widetilde{\Xi}_1$ тогда и только тогда, когда

1) $Y_1^T \cdot 1_{n_1} = k' := k - Y_0^T \cdot 1_{n_0}$,

2) $Y_1 \cdot 1_q = 1_{n_1}$,

3) $(Y_1)_{wm} \in [0,1]$.

В таком виде (5) – это классическая транспортная задача. Эта задача линейного программирования эффективно решается современными методами для больших размерностей. Через Y_1^* обозначим решение, доставляющее максимум в правой части (5).

Теперь рассмотрим второй максимизируемый член выражения (4):

$$(6)\qquad \max_{Y_1\in\Xi_1} trY_1^T R_{11}Y_1P.$$

Для этой задачи в [2] была предложена следующая оценка, основанная на другой релаксации множества Ξ_1:

$$(7)\quad \max_{Y_1\in\Xi_1} trY_1^T R_{11}Y_1P \leqslant LB_2(Y_0) :=$$

$$:= \max_{Y_1:Y_1^TY_1=diag(k')} trY_1^T R_{11}Y_1P = \sum_{i=1}^{p} \lambda_i(R_{11})\lambda_i(\tilde{P}),$$

где p – количество дополнительных вершин, у которых после фиксированного распределения основных вершин остались «свободные» места, $\lambda_i(A)$ – i-ое по убыванию собственное число матрицы A, а $\tilde{P} := diag(k')^{1/2}Pdiag(k')^{1/2}$.

Таким образом, вычисление данной оценки сводится к нахождению собственных чисел двух матриц. Через Y_1^{**} обозначим матрицу, доставляющую максимум в правой части (7). Согласно [2], $Y_1^{**} = U_1U_2^T diag(k')^{1/2}$, где U_1 – $n_1 \times q$ матрица, столбцы которой – собственные вектора матрицы R, упорядоченные по убыванию соответствующих собственных чисел, а U_2 – $q\times q$-матрица, столбцы которой – собственные вектора матрицы $\tilde{P}$, также упорядоченные по убыванию соответствующих собственных чисел.

Выпишем оценку (4) с использованием описанных выше оценок $LB_1(Y_0)$ и $LB_2(Y_0)$:

(8) $C_2(T) \geqslant$

$$\geqslant q1_n^T R1_n - trY_0^T R_{00} Y_0 P - tr2Y_0^T R_{01} Y_1^* P - trY_1^T R_{11} Y_1 P.$$

Выражение (8) дает нижнюю оценку затрат сети если матрица Y_0 фиксирует распределение первых n_0 основных вершин по дополнительным.

3.2. ВЫПУКЛАЯ ФУНКЦИЯ ЗАТРАТ

Для случая выпуклой функции затрат в [1] был предложен метод линеаризации, при котором функции затрат каждой дополнительной вершины заменяются касательными. В таком случае затраты связывающего дерева T можно оценить снизу как:

$$(9)\quad C_2(T) \geqslant \underline{C}_2^{\cup} = \max_{\alpha_1,\ldots,\alpha_q} \sum_{m=1}^{q} [c_2([c_2']^{-1}(\alpha_m)) - \\ - \alpha_m [c_2']^{-1}(\alpha_m) + \alpha_m 1_n^T R 1_n - \max_{Y \in \Xi} trY^T RYP(\alpha)],$$

где $P(\alpha)$ – положительно определенная $q \times q$-матрица, зависящая от топологии дополнительных вершин T_q и вектора $\alpha = (\alpha_1, ..., \alpha_q)$ наклонов касательных к функциям затрат.

Как и в случае линейной функции затрат, представим матрицу Y в виде двух частей. Тогда зависящая от Y часть нижней оценки затрат связывающего дерева запишется как:

$$(10)\quad trY^T RYP(\alpha) = \\ = tr[(Y_0 R_{00} Y_0 + 2Y_0^T R_{01} Y_1 + Y_1^T R_{11} Y_1) \cdot P(\alpha)].$$

Оценивая это выражение сверху (поскольку оно входит в (9) с отрицательным знаком) получаем аналогично линейному случаю:

$$(11)\quad trY^T RYP(\alpha) \leqslant trY_0 R_{00} Y_0 P(\alpha) + \\ + \max_{Y_1 \in \Xi_1} tr2Y_0^T R_{01} Y_1 P(\alpha) + \max_{Y_1 \in \Xi_1} trY_1^T R_{11} Y_1 P(\alpha).$$

Для слагаемого $\max_{Y_1 \in \Xi_1} 2trY_0^T R_{01} Y_1 P(\alpha)$, как и в линейном случае, получаем транспортную задачу, решение которой будем обозначать Y_1^{nl*}.

Для второго же слагаемого в [1] дается оценка:

$$\max_{Y_1 \in \Xi_1} trY_1^T R_{11} Y_1 P(\alpha) \leqslant trY_1^{nl**} R_{11} Y_1^{nl**} P(\alpha),$$

где $Y_1^{nl**} = U_1 U_2^T diag(k')^{1/2}$, где U_1 – $n_1 \times q$ матрица, столбцы которой – нормированные на единицу собственные вектора матрицы R, соответствующие q ее наибольшим собственным числам, упорядоченным по убыванию, а U_2 – $q \times q$ матрица, столбцы которой – нормированные на единицу собственные вектора матрицы $\widetilde{P}(\alpha) = K^{1/2} P(\alpha) K^{1/2}$, также упорядоченные по убыванию соответствующих им собственных чисел.

В итоге (9) превращается в:

$$C_2(T) \geqslant \max_{\alpha_1,\ldots,\alpha_q} \sum_{m=1}^{q} [c_2([c_2']^{-1}(\alpha_m)) - \alpha_m [c_2']^{-1}(\alpha_m) + \\ + \alpha_m 1_n^T R 1_n - trY_0 R_{00} Y_0 P(\alpha) - 2trY_0^T R_{01} Y_1^{nl*} P(\alpha) - \\ - trY_1^{nl**} R_{11} Y_1^{nl**} P(\alpha)]. \tag{12}$$

Таким образом, выражение (12) дает нижнюю оценку затрат сети для известной линеаризации (вектора α наклонов касательных).

Для улучшения качества линеаризации используется итерационный процесс, который начинается с $\alpha_m = 1, m \in M$, и вектор линеаризации изменяется по формуле:

$$\alpha_m^{i+1} = \alpha_m^i + \frac{1}{\lg(i)} (c_2'(\underline{x}^i(m)) - \alpha_m^i), \tag{13}$$

где i – номер шага, а $\underline{x}^i(m)$ – это оценка объема потока через вершину m, соответствующая решению, полученному на i-м шаге. Она вычисляется по формуле

$$\underline{x}^i(m) = \alpha_m 1_n^T R 1_n - tr[(Y_0^T R_{00} Y_0 + 2Y_0^T R_{01} Y_1^{nl*} + Y_1^{nl**)^T} R_{11} Y_1^{nl**}) P_m P_m^T],$$

где P_m – $q \times d_{T_q}(m)$-матрица, элемент которой $(P_m)_{m'i}$ равен единице если дополнительная вершина $m' \in M$ достижима через i-ое ребро, инцидентное вершине m.

4. Пример прикладных задач

Одним из примеров применения этого метода является анализ бизнес-процессов. Во время бизнес-анализа происходит декомпозиция функций предприятия, начиная с самого общего описания процессов внутри компании и до элементарных функций. Таким образом получается иерархия, в которой вершина содержит диаграмму процесса со связями и функциями. В дочерних вершинах содержатся диаграммы декомпозированных функций.

Для задач бизнес-анализа важна правильная декомпозиция этих функций, а именно уменьшение количества связей между диаграммами. Это позволяет обеспечить отображение на одной диаграмме наиболее тесно связанных между собой функций. Минимизация общего числа связей между диаграммами в условиях фиксированного шаблона декомпозиции функций (например, по 7 функций на одной диаграмме) сводится к минимизации аддитивной функции для линейной функции затрат $c_2(\cdot)$ и матрицы R, задающей связи между элементарными функциями, соответствующими основным вершинам (см. рис. 2).

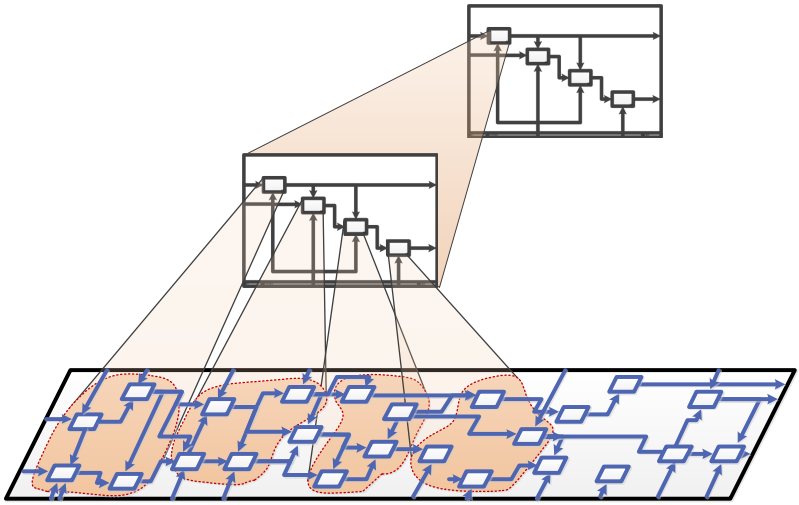


Рис. 2. Декомпозиция бизнес-процессов

Для поиска эффективной декомпозиции функций может быть использован описанный выше алгоритм.

5. Заключение

Таким образом, мы получили нижние оценки затрат связывающего дерева для линейной и выпуклой функции затрат и основанный на них алгоритм ветвей и границ поиска оптимального распределения основных вершин по дополнительным в условиях фиксированной топологии дополнительных вершин сети и вектора k «емкостей» дополнительных вершин.

В докладе мы расскажем о результатах численного моделирования эффективности описанного выше алгоритма ветвей и границ.

Перспективы исследований связаны с повышением эффективности алгоритма ветвей и границ, повышением качества нижних оценок, а также разработкой алгоритма оптимизации топологии дополнительных вершин.

Литература

1. ГУБКО М. В. *Модели и методы оптимизации структуры иерархических систем обработки информации. дисс. на соиск. степени д.ф.-м.-н.* // ИПУ РАН, 2014
2. ГУБКО М. В. *Спектральные нижние оценки затрат связывающей сети*. Труды XII Всероссийского совещания по проблемам управления (ВСПУ-2014), Москва, 16-19 июня 2014 г. С. 1959-1970.
3. Гэри М., Джонсон Д. Вычислительные машины и трудноразрешаемые задачи. – М.: Мир, 1982.
4. Goldschmidt O. and Hochbaum D. S. A Polynomial Algorithm for the k-Cut Problem for Fixed k // Mathematics of Operations Research Vol. 19, No. 1 (Feb., 1994), P. 24-37.
5. Klavzar S. and Gutman I. Wiener number of vertex-weighted graphs and a chemical application // Discrete Appl. Math. V. 80 (1997) P. 73–81.

6. Papadimitriou C.H. The complexity of the capacitated tree problem // Networks V. 8. No 3. 1978. P. 217–230.
7. RENDL F., WOLKOWICZ H. *A projection technique for partitioning the nodes of a graph* // Annals of Operations Research. 1995. V. 58. No 3. P. 155–179.

BRANCH AND BOUND ALGORITHM FOR OPTIMAL COMMUNICATION NETWORK CONSTRUCTION

Vasily Ginz, Moscow Institute of Physics and Technology, Moscow, engineer (vasikos@yandex.ru).
Mikhail Goubko, Institute of Control Sciences of RAS, Moscow, Cand. Sci., (mgoubko@mail.ru)

Abstract: We consider a recently suggested model of communication network optimization. A communication network connects a fixed collection of basic nodes through an arbitrary number of auxiliary nodes, which manage information flows between basic nodes. We contribute to the case of the, so called, additive cost function, when the cost of an auxiliary node in a network depends on the node degree and the total volume of information flowing throw this node. For the fixed tree-shaped topology of auxiliary nodes we suggest a branch-and-bound algorithm to find an optimal attachment of basic nodes to auxiliary ones. The algorithm uses a special lower-bound estimate of network cost, which is based on the continuous relaxation, linearization of cost function, and some results of the algebraic graph theory.

Keywords: structure optimization, communication network, lower-bound estimate, branch-and-bound algorithm.