Конкурентные модели размещения производства

Ю. А. Кочетов, А. В. Кононов, А.В.Плясунов *
(630090, Новосибирск, пр-т Ак. Коптюга 4, Ин-т матем. СО РАН
e-mail: {jkochet, alvenko, apljas}@math.nsc.ru)

Аннотация

Рассматриваются два класса конкурентных моделей размещения производства, в которых несколько лиц (игроков) последовательно или одновременно принимают решения об открытии предприятий для обслуживания клиентов. Первый класс представляется в виде дискретных моделей двухуровневого программирования. Вторым классом являются игровые модели с несколькими независимыми игроками, преследующими эгоистические интересы. Для первого класса показана его связь с псевдобулевыми функциями и предложен оригинальный способ построения семейства верхних и нижних оценок оптимума. Для второго класса установлена плотная PLS полнота задачи нахождения равновесий по Нэшу.
Ключевые слова: задачи размещения, локальные оптимумы, равновесия по Нэшу, PLS-полные задачи.

Введение

Задачи размещения составляют широкий пласт математических моделей исследования операций, интересный как с практической точки зрения, так и с точки зрения теории комбинаторной оптимизации. Своими корнями это направление уходит к П. Ферма (1601-1665) и Е. Торричелли (1608-1647) [15], но как самостоятельное направление оно сформировалось в 70-80 годы прошлого столетия. На сегодняшний день имеется ряд монографий в этой области [4, 12, 11, 16, 19]. Ежегодно проводятся специализированные конференции, организуемые европейской рабочей группой EWGLA (http://www.vub.ac.be/EWGLA/) и американской рабочей группой SOLA (http://www.ent.ohiou.edu/~thale/sola/sola.html).

В СССР пионерами этого направления были В. Черенин, В. Хачатуров, В. Трубин, С. Лебедев, а в Сибирском отделении Академии наук В. Береснев, Э. Гимади, В. Дементьев. Столь большой интерес к данной проблематике связан в первую очередь с приложениями, которые возникают не только при размещении предприятий, складов и магазинов. Аналогичные модели появляются, в частности, при решении задач унификации и стандартизации, когда выбирается состав системы технических средств, предназначенных для выполнения заданного списка работ [5]. В качестве целевой функции используются либо величина суммарных затрат на создание и функционирование системы технических средств, либо суммарная эффективность системы, т.е. объем выполняемых работ.

В данной работе рассматриваются два новых класса конкурентных задач размещения производства, в которых несколько лиц (игроков) последовательно или одновременно

*Исследование выполнено при финансовой поддержке Российского фонда фундаментальных исследований (проект 06-01-00075).

принимают решения об открытии предприятий для обслуживания клиентов. Если игроки последовательно принимают решения, то получается игра Штаккельберга и соответствующая математическая модель может быть представлена в виде дискретной модели двухуровневого программирования. Если решения принимаются одновременно, то получается игра нескольких лиц, которые стремятся максимизировать свою прибыль. Предполагается, что они не образуют коалиций, действуют независимо друг от друга и преследуют исключительно эгоистические интересы. Для первой модели в случае двух игроков предложен способ построения семейства верхних и нижних оценок оптимального значения целевой функции и показана связь с задачами о псевдобулевых функциях. Для второго класса игровых моделей вводится понятие равновесий по Нэшу. Показана их связь с локальными оптимумами.

Статья организована следующим образом. В разделе 1 рассматриваются классические модели размещения, в которых имеется одно лицо, принимающее решения (ЛПР). Обсуждается связь этих задач с псевдобулевыми функциями. Показано, как с помощью таких функций можно преобразовать исходные данные, чтобы максимальным образом сократить размерность. Во втором разделе рассматривается задача размещения с предпочтениями клиентов. В отличие от предшествующих моделей здесь два уровня принятия решений. Сначала фирма открывает предприятия. Затем клиенты выбирают поставщиков. Известно, что эта задача может быть сведена к задаче целочисленного линейного программирования (ЦЛП) и эквивалентным образом сформулирована в терминах псевдобулевых функций. По аналогии с первым разделом показана возможность сокращения размерности задачи. В третьем разделе формулируется двухуровневая модель размещения производства, в которой два ЛПР последовательно принимают решения об открытии предприятий. Приводится математическая формулировка задачи в терминах псевдобулевых функций и способ построения семейства верхних и нижних оценок. В последнем разделе рассматривается игровая модель для нескольких ЛПР, одновременно открывающих предприятия для обслуживания клиентов. Вводится понятие равновесных решений и устанавливается плотная PLS–полнота задачи нахождения таких решений.

1 Простейшая задача размещения

В подавляющем большинстве моделей размещения производства авторы исходят из предположения, что имеется одно ЛПР. Можно считать, что это руководитель фирмы, размещающий свои предприятия. Для заданного множества клиентов $J = \{1, \dots, n\}$ он знает затраты $c_{ij} \geq 0$, связанные с производством и доставкой продукции j-му клиенту из i-го пункта производства, если оно будет там открыто. Список возможных пунктов размещения производства $I = \{1, \dots, m\}$ предполагается конечным. Для каждого $i \in I$ известна стоимость $f_i \geq 0$ открытия предприятия в этом пункте. Задача состоит в выборе подмножества $S \subseteq I$, которое позволяет обслужить всех клиентов с минимальными суммарными затратами, т. е. найти

$$\min_{S \subseteq I} \Big\{ \sum_{i \in S} f_i + \sum_{j \in J} \min_{i \in S} c_{ij} \Big\}.$$

Первое слагаемое в целевой функции задает затраты на открытие предприятий. Второе слагаемое определяет производственно–транспортные расходы. Сформулированная задача известна в литературе как простейшая задача размещения производства. Она является NP–трудной в сильном смысле даже в случае, когда матрица (c_{ij}) удовлетворяет неравенству треугольника [18].

Существует тесная связь между этой задачей и задачей минимизации псевдобулевых функций. Впервые это было отмечено в работах П. Хаммера и С. Рудеани. Позже в работах В. Береснева было предложено новое оригинальное сведение простейшей задачи размещения к задаче минимизации псевдобудевых функций с положительными коэффициентами при нелинейных членах. Более того, установлена эквивалентность этих задач.

Пусть для некоторого вектора $g_i, i \in I$ известна такая перестановка $i_1, \dots, i_m$, что $g_{i_1} \le g_{i_2} \le \dots \le g_{i_m}$. Положим

$$\Delta g_0 = g_{i_1};$$

$$\Delta g_l = g_{i_{l+1}} - g_{i_l}, \quad 1 \le l < m;$$

$$\Delta g_m = g_{i_m}.$$

Лемма 1. [5] Для любого вектора $z_i \in \{0,1\}, i \in I, z \neq \{1, \dots, 1\}$ справедливы следующие равенства:

$$\min_{i|z_i=0} g_i = \Delta g_0 + \sum_{l=1}^{m-1} \Delta g_l z_{i_1} \dots z_{i_l},$$

$$\max_{i|z_i=0} g_i = \Delta g_m - \sum_{l=1}^{m-1} \Delta g_{m-l} z_{i_{m-l+1}} \dots z_{i_m}.$$

Для j-го столбца матрицы (c_{ij}) введем такую перестановку $i_1^j, \dots, i_m^j$, что $c_{i_1^j j} \le c_{i_2^j j} \le \dots \le c_{i_m^j j}$. Пользуясь Леммой 1, представим целевую функцию задачи размещения в виде псевдобулевой функции:

$$b(z) = \sum_{i \in I} f_i (1 - z_i) + \sum_{j \in J} (\Delta c_{0j} + \sum_{l=1}^{m-1} \Delta c_{lj} z_{i_1^j} \dots z_{i_l^j}).$$

Теорема 1. [5] Задача минимизации псевдобулевой функции $b(z), z \neq (1, \dots, 1)$ и простейшая задача размещения производства эквивалентны. Для оптимальных решений z^*, S^* этих задач справедливы соотношения $z_i^* = 0 \Leftrightarrow i \in S^*, i \in I$ и значения целевых функций на этих решениях совпадают.

Рассмотрим следующий пример: $I = J = \{1, 2, 3\}$, $f_i = \begin{pmatrix} 10 \\ 10 \\ 10 \end{pmatrix}$ $c_{ij} = \begin{pmatrix} 0 & 3 & 10 \\ 5 & 0 & 0 \\ 10 & 20 & 7 \end{pmatrix}$. Соответствующая псевдобулева функция имеет вид $b(z) = 10(1-z_1) + 10(1-z_2) + 10(1-z_3) + (5z_1 + 5z_1 z_2) + (3z_2 + 17 z_1 z_2) + (7z_2 + 3z_2 z_3) = 15 + 5(1 - z_1) + 0(1 - z_2) + 10(1 - z_3) + 22 z_1 z_2 + 3 z_2 z_3$. Восстановим по этой функции задачу размещения. Получаем $I' = I, J' = \{1, 2\}$ $f_i' = \begin{pmatrix} 5 \\ 0 \\ 10 \end{pmatrix}$ $c_{ij}' = \begin{pmatrix} 0 & 3 \\ 0 & 0 \\ 22 & 0 \end{pmatrix}$. Новая задача имеет меньше клиентов, $|J'| < |J|$. Более того, $f_2' = 0$. Следовательно, в оптимальном решении второе предприятие можно считать открытым. Другими словами, новая задача имеет меньшую размерность и эквивалентна исходной.

Итак, по простейшей задаче размещения можно построить псевдобулеву функцию с положительными коэффициентами при нелинейных членах. Разные примеры могут давать одну и ту же функцию. Значит, прежде чем решать задачу, можно сначала найти эквивалентную ей, но с меньшей размерностью. Поиск такой задачи может быть осуществлен за полиномиальное время.

Теорема 2. Для заданной псевдобулевой функции с положительными коэффициентами при нелинейных членах эквивалентная ей простейшая задача размещения с минимальным числом клиентов может быть найдена с полиномиальной трудоемкостью от n и m.

Доказательство. Рассмотрим произвольную псевдобулеву функцию $b(z)$ с положительными коэффициентами при нелинейных членах

$$b(z) = \sum_{i\in I} \alpha_i z_i + \sum_{l\in L} \beta_l \prod_{i\in I_l} z_i, \quad \beta_l > 0, I_l \subset I, l \in L.$$

Семейство подмножеств $\{I_l\}_{l\in L}$ множества I с отношением порядка $I_{l'} < I_{l''} \Leftrightarrow I_{l'} \subset I_{l''}$ образует частично упорядоченное множество. Любую последовательность подмножеств $I_{l_1} < I_{l_2} < \dots < I_{l_K}$ называют *цепью.* Произвольное разбиение семейства $\{I_l\}_{l\in L}$ на непересекающиеся цепи порождает матрицу транспортных затрат (c_{ij}) простейшей задачи размещения. Каждый элемент такого разбиения соответствует одному клиенту. Требование найти матрицу с минимальным числом клиентов означает найти разбиение частично упорядоченного множества на минимальное число непересекающихся цепей. Эта известная задача решается с полиномиальной трудоемкостью от n и m с помощью конструктивного доказательства теоремы Дилворта [24]. Теорема доказана.

Задача минимизации функции $b(z)$ эквивалентна простейшей задаче размещения, но обладает новыми свойствами. Рассмотрим задачу минимизации этой функции для непрерывных переменных $z_i \in [0, 1], i \in I$. Для простейшей задачи размещения такой переход связан с появлением разрыва целочисленности, который может оказаться сколь угодно близким к единице [17]. Для функции $b(z)$ этот разрыв равен нулю! Легко проверить, что среди оптимальных решений задачи минимизации функции $b(z)$ с непрерывными переменными всегда существует целочисленное оптимальное решение.

2 Задача размещения с предпочтениями клиентов

До сих пор предполагалось, что имеется одно лицо, принимающее решение, которое стремится минимизировать суммарные затраты на организацию производства и обслуживание клиентов. Однако клиент часто имеет возможность сам выбирать поставщиков продукции, исходя из собственных предпочтений [1]. Он не обязан минимизировать производственно–транспортные затраты фирмы.

Пусть матрица (g_{ij}) задает предпочтения клиентов на множестве I. Если $g_{i_1j} < g_{i_2j}$, то j-й клиент предпочитает предприятие i_1. Для упрощения модели будем предполагать, что в каждом столбце матрицы g_{ij} все элементы различные. В противном случае придется рассматривать оптимистические и пессимистические стратегии и вводить дополнительные определения, что следует называть оптимальным решением задачи. Итак, цель ЛПР по прежнему состоит в выборе подмножества $S \subseteq I$, которое позволяет обслужить всех клиентов с минимальными суммарными затратами, но теперь приходится учитывать предпочтения клиентов по выбору поставщиков. Введем переменные задачи:

$$x_i = \begin{cases} 1, & \text{если в пункте } i \text{ открывается предприятие,} \\ 0 & \text{в противном случае,} \end{cases}$$

$$x_{ij} = \begin{cases} 1, & \text{если клиент } j \text{ обслуживается из предприятия } i, \\ 0 & \text{в противном случае.} \end{cases}$$

Математическая модель может быть представлена в виде дискретной задачи двухуровневого программирования [7]: найти

$$\min_{x_i} \Big\{ \sum_{i\in I} f_i x_i + \sum_{j\in J} \sum_{i\in I} c_{ij} x^*_{ij}(x_i) \Big\}$$

при условиях

$$x_i \in \{0,1\}, \quad i \in I,$$

где $x^*_{ij}(x_i)$ — оптимальное решение задачи клиентов:

$$\min_{x_{ij}} \sum_{j\in J} \sum_{i\in I} g_{ij} x_{ij}$$

при условиях

$$\sum_{i\in I} x_{ij} = 1, \qquad j \in J,$$

$$x_{ij} \le x_i, \qquad i \in I, j \in J,$$

$$x_{ij} \in \{0,1\}, \quad i \in I, j \in J.$$

Целевая функция ЛПР по прежнему задает суммарные затраты на открытие предприятий и обслуживание клиентов. Отличие состоит в том, что теперь множество допустимых решений задается как требованием целочисленности переменных $x_i, i \in I$ (что соответствует условию $S \subseteq I$), так и вспомогательной оптимизационной задачей. Первое ограничение в ней требует назначить поставщика каждому клиенту. Второе ограничение позволяет выбирать поставщиков только из открытых предприятий. Вектор x_i при этом считается заданным.

Известно несколько способов сведения данной двухуровневой задачи к ЦЛП [1, 6]. Заметим, что для каждого $j \in J$ важно только упорядочение элементов g_{ij}, а не их числовые значения. Упорядочим элементы j-го столбца $g_{i_1 j} < g_{i_2 j} < \ldots < g_{i_m j}$ и положим $S_{ij} = \{l \in I |\ g_{lj} < g_{ij}\},\ i \in I$. Оптимальное решение $x^*_{ij}(x_i)$ задачи клиентов имеет следующее свойство: $x^*_{ij} = 1 \Rightarrow x_l = 0, \quad l \in S_{ij}$. Используя это соотношение, исходную двухуровневую задачу можно представить в следующем виде [1, 6]: найти

$$\min \sum_{i\in I} f_i x_i + \sum_{j\in J} \sum_{i\in I} c_{ij} x_{ij}$$

при условиях

$$x_{ij} + x_l \le 1, \quad l \in S_{ij}, i \in I, j \in J,$$

$$\sum_{i\in I} x_{ij} = 1, \quad j \in J,$$

$$0 \le x_{ij} \le x_i, \qquad i \in I, j \in J,$$

$$x_i, x_{ij} \in \{0,1\}, \quad i \in I, j \in J.$$

Первое неравенство гарантирует оптимальность x_{ij} в задаче клиентов. Если матрицы (c_{ij}) и (g_{ij}) совпадают, то получаем простейшую задачу размещения.

Первое неравенство можно записать в другой эквивалентной форме:

$$\sum_{l\in S_{ij}} x_l \le |S_{ij}|(1 - x_{ij}), \quad i \in I, j \in J,$$

или

$$x_i \le x_{ij} + \sum_{l \in S_{ij}} x_l, \quad i \in I, j \in J,$$

или

$$x_i \le x_{ij} + \sum_{l \in S_{ij}} x_{lj}, \quad i \in I, j \in J.$$

Можно показать, что последнее неравенство приводит к лучшей линейной релаксации, чем три других [1].

Известно [6], что задача размещения с предпочтениями клиентов может быть сведена к задаче минимизации псевдобулевой функции. Пусть перестановка $i_1^j, i_2^j, \dots, i_m^j$ задает упорядочение элементов j-го столбца матрицы (g_{ij}). Для $j \in J$ положим

$$\nabla c_{i_1 j} = c_{i_1 j}$$

$$\nabla c_{i_l j} = c_{i_l j} - c_{i_{l-1} j}, \quad 1 < l \le m,$$

и определим псевдобулеву функцию

$$B(z) = \sum_{i \in I} f_i (1 - z_i) + \sum_{j \in J} \sum_{i \in I} \nabla c_{ij} \prod_{l \in S_{ij}} z_l.$$

Задача размещения с предпочтениями клиентов эквивалентна задаче минимизации псевдобулевой функции $B(z)$, $z \ne (1, \dots, 1)$. Оптимальные решения z^*, x^* этих задач связаны соотношением $z_i^* = 1 - x_i^*$, $i \in I$ и значения целевых функций на этих решениях совпадают.

Заметим, что коэффициенты ∇c_{ij} могут иметь произвольный знак. Другими словами, для любой псевдобулевой функции можно построить пример задачи размещения с предпочтениями клиентов и наоборот. Более того, по аналогии с Теоремой 2 для любой задачи размещения с предпочтениями клиентов можно за полиномиальное время от n и m перестроить исходные данные так, чтобы минимизировать числом клиентов.

Теорема 3. Для произвольной псевдобулевой функции эквивалентная ей задача размещения с предпочтениями клиентов может быть построена за полиномиальное время от n и m так, чтобы число клиентов было минимальным.

3 Антагонистические размещения

Рассмотрим теперь ситуацию, когда две фирмы последовательно принимают решения о размещении предприятий. Сначала на рынок выходит первая фирма — Лидер и открывает свое подмножество предприятий $S_0 \subset I$. Затем, зная это решение, конкурирующая фирма открывает собственные предприятия, подмножество $S_1 \subset I, S_0 \cap S_1 = \emptyset$. Каждый клиент выбирает из множества открытых предприятий $S_0 \cup S_1$ одно предприятие, согласно собственным предпочтениям. Если обслуживание j-го клиента приносит доход $r_j > 0$, Лидер открывает p^0 предприятий, а Конкурент — p^1 предприятий, то в зависимости от размещения этих предприятий рынок (множество клиентов) будет как-то разделен между двумя фирмами. Каждая фирма будет стремиться максимизировать свою долю рынка. Получаем игру двух лиц с противоположными интересами. Игроки неравноправны. Сначала делает ход Лидер. Он может размещать предприятия в любом месте. Затем делает ход Конкурент. Он знает ход Лидера. Получаем игру Штакельберга, в которой требуется максимизировать долю рынка (суммарный доход) первого игрока. Введем переменные

задачи:
Лидер:

$$x_i = \begin{cases} 1, & \text{если Лидер в пункте } i \text{ открывает предприятие,} \\ 0 & \text{в противном случае,} \end{cases}$$

Конкурент:

$$y_i = \begin{cases} 1, & \text{если Конкурент в пункте } i \text{ открывает предприятие,} \\ 0 & \text{в противном случае,} \end{cases}$$

Клиенты:

$$u_j = \begin{cases} 1, & \text{если клиент } j \text{ обслуживается из предприятия Лидера,} \\ 0, & \text{если клиент } j \text{ обслуживается из предприятия Конкурента.} \end{cases}$$

При заданном векторе $x_i \in \{0,1\}, i \in I$ определим множество

$$I_j(x) = \{i \in I |\, g_{ij} < \min_{l \in I}(g_{lj} |\, x_l = 1)\}, \quad j \in J.$$

Это множество задает пункты размещения предприятий, позволяющие Конкуренту *захватить* j-го клиента. С использованием введенных переменных соответствующая задача двухуровневого программирования записывается следующим образом: найти

$$\max_{x_i} \sum_{j \in J} r_j u_j^*(x_i)$$

при условиях

$$\sum_{i \in I} x_i = p^0,$$

$$x_i \in \{0,1\}, \quad i \in I,$$

где $u_j^*(x_i)$ — оптимальное решение задачи Конкурента:

$$\max_{y_i, u_j} \sum_{j \in J} r_j (1 - u_j)$$

при условиях

$$\sum_{i \in I} y_i = p^1,$$

$$1 - u_j \leq \sum_{i \in I_j(x)} y_i, \quad j \in J,$$

$$y_i + x_i \leq 1, \quad i \in I,$$

$$y_i, u_j \in \{0,1\}, \quad i \in I, j \in J.$$

Целевая функция задачи задает суммарный доход Лидера. Множество допустимых решений описывается с помощью задачи Конкурента. Вектор $x_i,\ i \in I$ и множества $I_j(x),\ j \in J$ в задаче Конкурента считаются известными. На сегодняшний день эффективные методы решения данной задачи неизвестны. Первые шаги в этом направлении сделаны в [21], где предлагается точная схема частичного перебора. В [20] исследуется частный случай задачи, когда Конкурент размещает только одно предприятие.

В задаче Конкурента предполагалось, что нельзя открывать предприятия в тех местах, где уже есть предприятия Лидера, т.е. $S_0 \cap S_1 = \emptyset$. От этого условия можно отказаться. Удаление ограничения $y_i + x_i \leq 1$, $i \in I$ из задачи Конкурента не меняет его оптимального решения. Конкуренту невыгодно открывать предприятия в том месте, где уже есть предприятие Лидера. Это не дает ему дополнительных клиентов. Однако ситуация может измениться, если положить $I_j(x) = \{i \in I | g_{ij} \le \min_{l \in I}(g_{lj} | x_l = 1)\}, \quad j \in J$. В этом случае Лидер будет терять клиента даже в том случае, когда Конкурент открывает столь же предпочтительное предприятие, что и Лидер. Таким образом получаем модель с *любознательными* клиентами, которые при прочих равных условиях тянутся к новому предприятию. Обзор различных стратегий поведения клиентов в конкурентных моделях размещения можно найти в [14].

Покажем связь предложенной задачи двухуровневого программирования с псевдобулевыми функциями. Заметим, что $u_j^* = \prod\limits_{i \in I_j(x)} (1 - y_i^*)$. Тогда задачу можно представить следующим образом: найти

$$\max \sum_{j \in J} r_j \prod_{i \in I_j(x)} (1 - y_i^*(x_i))$$

при условиях

$$\sum_{i \in I} x_i = p^0, \quad x_i \in \{0, 1\},\ i \in I,$$

где $y_i^*(x_i)$ — оптимальное решение задачи Конкурента

$$\max \sum_{j \in J} r_j \Big(1 - \prod_{i \in I_j(x)} (1 - y_i)\Big)$$

при условиях

$$\sum_{i \in I} y_i = p^1, \quad y_i \in \{0, 1\},\ i \in I.$$

Удаляя константу из целевой функции Конкурента, получаем задачу двухуровневого программирования для псевдобулевой функции $\sum_{j \in J} r_j \prod_{i \in I_j(x)} (1 - y_i)$.

Покажем, как с помощью ЦЛП можно получить семейство верхних оценок на оптимальное значение целевой функции Лидера и, соответственно, нижних оценок на оптимальное значение целевой функции Конкурента. Общая идея построения таких оценок выглядит следующим образом. Добавим в систему ограничений задачи Конкурента дополнительное ограничение. Оптимум в задаче Конкурента от этого может только уменьшиться, а оптимум в задаче Лидера может только возрасти. Если введение дополнительного ограничения позволяет свести исходную задачу двухуровневого программирования к ЦЛП, то получаем нужную оценку. Проблема состоит в том, чтобы найти подходящее ограничение, обладающее требуемым свойством.

Предположим, что Конкурент при размещении предприятий действует по следующему правилу. Он упорядочивает возможные места размещения предприятий согласно некоторому критерию. Это упорядочение производится до решения задачи и доводится до сведения Лидера. Как только Лидер объявляет свое решение, Конкурент размещает свои предприятия согласно выбранному порядку в тех местах, которые не занял Лидер. Такая стратегия не гарантирует нахождения оптимального решения Конкурента и, следовательно, приводит к нижней оценке оптимума для Конкурента. Эта нижняя оценка зависит от выбранного упорядочения. Меняя упорядочение, получаем разные нижние оценки. Таким образом, получаем семейство из $m!$ нижних оценок для Конкурента или верхних оценок на оптимум Лидера.

Представим сведение к ЦЛП задачи с указанным ограничением на поведение Конкурента. Без ограничения общности будем считать, что возможные места размещения предприятий уже упорядочены в соответствии с предпочтениями Конкурента. Первое предприятие $i = 1$ является наиболее желательным для него, последнее предприятие $i = m$ —

наименее желательным. Тогда при выбранных значениях x_i, $i \in I$ поведение Конкурента однозначно определяется следующей системой ограничений

$$x_i + y_i \leq 1, \quad i \in I,$$

$$\sum_{i \in I} y_i = p^1,$$

$$x_i + y_i \geq y_k, \quad i, k \in I, k > i.$$

Последнее неравенство запрещает Конкуренту открывать предприятие в пункте k, если хотя бы одно из предприятий с меньшим номером не было открыто Лидером или Конкурентом. Введем дополнительные переменные:

$$z_{jk} = \begin{cases} 1, & \text{если клиент } j \text{ обслуживается из предприятия } k \\ 0 & \text{в противном случае.} \end{cases}$$

Переменные z_{jk} однозначно определяются по значениям x_i и y_i из следующей системы ограничений:

$$\sum_{k \in I} z_{jk} = 1, \quad j \in J,$$

$$z_{jk} \leq x_k + y_k, \quad j \in J, k \in I,$$

$$z_{jk} + \sum_{l \in S_{kj}} z_{jl} \geq x_k + y_k, \quad k \in I, j \in J.$$

Первое равенство требует обслужить каждого клиента из предприятий Лидера или Конкурента. Второе ограничение разрешает обслуживание только из открытых предприятий. Последнее неравенство устанавливает приоритет в выборе предприятий при обслуживании клиентов. Согласно этому ограничению наиболее предпочтительное из открытых предприятий Лидера или Конкурента будет выбрано для обслуживания данного клиента. Кроме того, это ограничение доминирует первое ограничение в предыдущей системе, так как $z_{jk} + \sum_{l \in S_{kj}} z_{jl}$ всегда не больше единицы.

При заданных значениях z_{jk}, x_i, y_i переменные u_j однозначно определяются следующей системой ограничений:

$$u_j \geq z_{jk} - y_k, \quad j \in J, k \in I,$$

$$1 - u_j \geq z_{jk} - x_k, \quad j \in J, k \in I.$$

Первое неравенство требует включить клиента j в область обслуживания Лидера, если $z_{jk} = 1$ и $y_k = 0$. Если же $z_{jk} = 1$ и $y_k = 1$, то второе неравенство требует включить этого клиента в область обслуживания Конкурента.

Таким образом, выбрав значения x_i, $i \in I$, можно однозначно определить значения y_i, $i \in I$, z_{jk}, $j \in J, k \in I$ и u_j, $j \in J$. Оптимальные значения всех переменных определяются следующей задачей: найти

$$\max \sum_{j \in J} r_j u_j$$

при условиях

$$\sum_{i \in I} x_i = p^0,$$

$$\sum_{i \in I} y_i = p^1,$$

$$x_i + y_i \geq y_k, \quad i, k \in I, k > i,$$

$$\sum_{k\in I} z_{jk} = 1, \quad j \in J,$$

$$z_{jk} \le x_k + y_k, \quad j \in J, k \in I,$$

$$z_{jk} + \sum_{l\in S_{kj}} z_{jl} \ge x_k + y_k, \quad k \in I, j \in J,$$

$$u_j \ge z_{jk} - y_k, \quad j \in J, k \in I,$$

$$1 - u_j \ge z_{jk} - x_k, \quad j \in J, k \in I,$$

$$x_i, y_i, z_{jk}, u_j \in \{0,1\}, \quad k, i \in I, j \in J.$$

Пусть $(x_i^*, y_i^*, z_{jk}^*, u_j^*)$ — оптимальное решение сформулированной задачи. Оно дает верхнюю оценку на оптимум Лидера. Линейная релаксация данной задачи, полная или частичная, также дает верхнюю оценку. Для получения нижней оценки достаточно решить задачу Конкурента при заданных значениях x_i^*. Другие подходы к построению нижних оценок и их сравнение можно найти в [2, 3]. Таким образом, получаем приближенное решение исходной задачи двухуровневого программирования с апостериорной оценкой точности.

4 Игровая модель размещения

Рассмотрим теперь ситуацию, когда p фирм одновременно открывают свои предприятия для обслуживания клиентов. Цель каждой фирмы (игрока) состоит в максимизации прибыли. Будем предполагать, что игроки принимают решения независимо друг от друга, не образуют коалиций и преследуют исключительно эгоистические интересы. Для простоты изложения будем считать, что каждый игрок открывает не более одного предприятия, хотя все дальнейшие рассуждения можно обобщить и на случай произвольного числа предприятий.

Пусть величина r_j задает максимальную цену, которую клиент j согласен заплатить за предлагаемую продукцию. Если бы игроки объединились, стремясь получить максимальную прибыль, то их оптимальную стратегию можно найти из следующей оптимизационной задачи:

$$\max \sum_{j\in J} (r_j - \sum_{i\in I} c_{ij} x_{ij})$$

при условиях

$$\sum_{i\in I} x_{ij} \le 1, \quad j \in J,$$

$$x_i \ge x_{ij}, \quad i \in I, j \in J,$$

$$\sum_{i\in I} x_i = p,$$

$$x_i, x_{ij} \in \{0,1\}, \quad i \in I, j \in J.$$

Если первое ограничение выходит на равенство для всех $j \in J$, то каждый клиент заплатит максимальную приемлемую для него цену, и весь доход достанется игрокам.

Рассмотрим теперь ситуацию, когда игроки действуют независимо друг от друга. Каждый из них может открывать собственное предприятие в любом из пунктов множества I. Производственно-транспортные затраты k-го игрока зададим матрицей $c_{ij}^k, j \in J$. Без ограничения общности можно считать, что $c_{ij}^k \le r_j, j \in J$. В противном случае положим $c_{ij}^k = r_j$ и игрок не будет иметь прибыль от обслуживания этого клиента.

Пусть игроки открыли предприятия $i_1, i_2, \ldots, i_p$. Какую цену q_j заплатит j-й клиент за их продукцию? Обозначим через c_j минимальные производственно-транспортные затраты по обслуживанию этого клиента из открытых предприятий:

$$c_j = \min\{c^1_{i_1 j}, c^2_{i_2 j}, \ldots, c^p_{i_p j}\}, \quad j \in J.$$

Цена q_j не может быть меньше c_j, так как игрокам невыгодно поставлять продукцию по столь низкой цене. Предположим, что игроки выставили цену $q_j = r_j$. Тогда j-му клиенту все равно, какой из поставщиков будет его обслуживать. Так как каждый из игроков хочет оказаться поставщиком, то они начнут снижать цену. Пусть $i(j) = \arg\min\{c^1_{i_1 j}, c^2_{i_2 j}, \ldots, c^p_{i_p j}\}$ и c'_j — второй по величине минимальный элемент среди $c^1_{i_1 j}, c^2_{i_2 j}, \ldots, c^p_{i_p j}$. Процесс падения цены остановится на c'_j, когда игрок $i(j)$ останется единственным поставщиком, кому все еще выгодно обслуживание j-го клиента. Ниже опускать цену нецелесообразно. Выше поднимать цену нельзя, так как появляется конкурент, которому также становится выгодным обслуживание этого клиента. Получаем $q_j = c'_j$, $j \in J$. Если $c_j = c'_j$ для некоторого $j \in J$, то $q_j = c_j$, и обслуживание этого клиента не приносит прибыли ни одному из игроков.

Когда игроки действовали сообща, они держали цену q_j равной r_j и присваивали весь доход. Теперь $q_j = c'_j$ и прибыль $r_j - c_j$ делится между поставщиком $i(j)$, который получает $q_j - c_j$, и клиентом, который экономит величину $r_j - q_j$. Обозначим через Γ_k множество клиентов, обслуживаемых k-м игроком при заданном выборе $i_1, \ldots, i_p$. Тогда прибыль k-го игрока равна

$$w_k = \sum_{j \in \Gamma_k} (q_j - c_j),$$

суммарная экономия клиентов определяется величиной

$$\nu(i_1, \ldots, i_p) = \sum_{j \in J} (r_j - q_j),$$

суммарная прибыль игроков и экономия клиентов составляет величину

$$\mu(i_1, \ldots, i_p) = \sum_{j \in J} (r_j - c_j) = \sum_{k=1}^{p} w_k + \nu(i_1, \ldots, i_p).$$

Решение $(i_1, \ldots, i_p)$ называют равновесием по Нэшу или равновесным решением, если ни один из игроков не может увеличить свою прибыль при условии, что другие игроки не меняют свой выбор. Оптимальным решением называют такое решение, при котором величина $\mu(i_1, \ldots, i_p)$ достигает максимального значения. Такое решение действительно является наилучшим для общества, так как приносит наибольший «эффект» от размещения производства и выпуска продукции. Ниже будет показано, что оптимальное решение является равновесным, но не каждое равновесное решение является оптимальным. Для таких игр в [22] вводится понятие цены анархии. Эта величина равна отношению оптимального решения к наихудшему среди равновесий по Нэшу. Известно [25], что для данной задачи цена анархии не превосходит 2. Однако вопрос о трудоемкости нахождения равновесных решений до сих пор остается открытым. Неясно, можно ли найти хоть одно равновесное решение за полиномиальное время от длины записи исходных данных.

Рассмотрим следующую оптимизационную задачу:

$$\min \sum_{j \in J} \sum_{k=1}^{p} \sum_{i \in I} c^k_{ij} x^k_{ij}$$

при условиях

$$\sum_{k=1}^{p}\sum_{i\in I} x_{ij}^k = 1, \quad j \in J,$$

$$x_i^k \geq x_{ij}^k, \quad i \in I, j \in J, k = 1,\dots,p,$$

$$\sum_{i\in I} x_i^k = 1, \quad k = 1,\dots,p,$$

где переменные x_i^k и x_{ij}^k имеют следующий смысл:

$$x_i^k = \begin{cases} 1, & \text{если игрок } k \text{ открывает предприятие в пункте } i, \\ 0 & \text{в противном случае,} \end{cases}$$

$$x_{ij}^k = \begin{cases} 1, & \text{если клиент } j \text{ обслуживается игроком } k \text{ из предприятия } i, \\ 0 & \text{в противном случае.} \end{cases}$$

Целевая функция задачи задает производственно–транспортные затраты игроков. Первое ограничение требует обслужить всех клиентов. Второе ограничение позволяет обслуживать клиентов только из открытых предприятий. Последнее ограничение позволяет каждому игроку открывать только одно предприятие.

Заметим, что в этой задаче несколько игроков могут открывать предприятия в одном месте. Поэтому такая постановка близка по смыслу к задаче о p-медиане [10], но не идентична ей. Будем обозначать ее через *p-median game* или PMG. Пусть (x_i^k, x_{ij}^k) — допустимое решение этой задачи и при данных значениях x_i^k, величины x_{ij}^k задают оптимальное распределение клиентов между открытыми предприятиями. Другими словами, будем предполагать, что x_{ij}^k определяются по переменным x_i^k и решение можно задавать только этими переменными. Тогда задачу можно переписать следующим образом: найти

$$\min \sum_{j\in J} \min_{k=1,\dots,p} \min_{i\in I} c_{ij}^k x_i^k$$

при условиях

$$\sum_{i\in I} x_i^k = 1, \quad k = 1,\dots,p,$$

$$x_i^k \in \{0,1\}, \quad k = 1,\dots,p, i \in I.$$

Пусть (x_i^k) — допустимое решение задачи. Под окрестностью $Swap(x)$ будем понимать множество допустимых решений, которые могут быть получены из решения (x_i^k) выбором некоторого k и заменой предприятия этого игрока на любое другое. Локальным минимумом для данной окрестности называется такое решение, для которого любое соседнее решение имеет не меньшие значение целевой функции.

Лемма 2 [9, 23] Между локальными минимумами задачи PMG и равновесными решениями существует взаимно–однозначное соответствие.

Доказательство. Пусть $(i_1,\dots,i_p)$ — равновесное решение. Поставим ему в соответствие решение (x_i^k), определяемое следующим образом: $x_i^k = 1$, если $i = i_k$ и $x_i^k = 0$ в противном случае. Проверим, что это решение является локальным минимумом тогда и только тогда, когда $(i_1,\dots,i_p)$ — равновесное решение. Для этого достаточно показать, что при смене предприятия k-м игроком изменение его прибыли равно изменению целевой функции задачи PMG при переходе от решения (x_i^k) к соответствующему соседнему решению.

Пусть k-й игрок сменил предприятие i_k на l и новое множество его клиентов $\widetilde{\Gamma}_k$ не пересекается с множеством Γ_k. Для $j \in \widetilde{\Gamma}_k$ прибыль игрока от обслуживания этого клиента составит величину $c_j - c^k_{lj} \geq 0$. Ровно на столько же изменится j-е слагаемое целевой функции PMG при переходе к соседнему решению:

$$\widehat{x}^t_i = \begin{cases} 1, \text{ если } & i = l, \\ 0, \text{ если } & i \neq l, \end{cases} \quad t = k.$$

При $j \in \Gamma_k$ игрок имел прибыль $q_j - c_j \geq 0$ и теперь теряет ее. Ровно на столько же возрастает j-е слагаемое целевой функции задачи PMG. Таким образом, смена предприятия k-м игроком ведет к изменению его прибыли с величины $\sum_{j\in\Gamma_k}(q_j - c_j)$ на величину $\sum_{j\in\widetilde{\Gamma}_k}(c_j - c^k_{lj})$ и в точности таким же образом меняется значение целевой функции в задаче PMG. Лемма доказана.

Таким образом, вопрос о нахождении равновесных решений тесно связан с поиском локальных минимумов. Рассмотрим эту задачу более подробно. Напомним основные определения [8, 27].

Определение 1. *Оптимизационная задача* OP определяется следующим набором объектов $< \mathcal{I}, Sol, F, goal >$, где

1. $\mathcal{I}$ – множество входов задачи OP;

2. Sol – функция, которая каждому входу $x \in \mathcal{I}$ сопоставляет множество допустимых решений $Sol(x)$;

3. F – функция, которая задаёт вес $F(s, x)$ допустимого решения s на входе x;

4. величина $goal \in \{\min, \max\}$ уточняет является ли задача OP задачей на максимум или минимум.

В оптимизационной задаче необходимо найти оптимальное решение для заданного входа x.

Определение 2. *Задача локального поиска* определяется парой $\Pi = (OP, N)$, где OP — оптимизационная задача, а N — функция окрестности, которая каждому допустимому решению s на входе x ставит в соответствие множество $N(s, x) \subseteq Sol(x)$ соседних решений. Задача локального поиска заключается в отыскании локального минимума для заданного входа x.

Определение 3. Задача локального поиска Π принадлежит классу PLS, если существует три полиномиальных алгоритма A, B, C и полином q такие, что

1. Алгоритм A определяет является ли любое слово x входом задачи. Если $x \in \mathcal{I}$, то алгоритм находит допустимое решение задачи OP;

2. Для любого входа задачи $x \in \mathcal{I}$ и любого слова s алгоритм B определяет является ли s допустимым решением. Если $s \in Sol(x)$, то за полиномиальное время алгоритм находит значение целевой функции $F(s, x)$;

3. Для любого входа $x \in \mathcal{I}$ и любого решения $s \in Sol(x)$ алгоритм C определяет является ли s локальным оптимумом. Если нет, то алгоритм находит соседа $s' \in N(s, x)$ с меньшим значением целевой функции;

4) Для любого входа $x \in \mathcal{I}$ длина любого допустимого решения $s \in Sol(x)$ полиномиально ограничена длиной входа задачи, т.е. $|s| \leq q(|x|)$.

Легко проверить, что задача поиска локального минимума для задачи PMG с окрестностью $Swap$ принадлежит классу PLS.

Определение 4. Пусть Π_1 и Π_2 — две задачи локального поиска. Задача Π_1 PLS – сводится к задаче Π_2, если существуют такие полиномиально вычислимые функции h и g, что
1) по произвольному входу x задачи Π_1 функция h вычисляет некоторый вход $h(x)$ задачи Π_2;

2) по произвольному решению s входа $h(x)$ функция g находит некоторое решение $g(s, x)$ для входа x;

3) если $x \in \Pi_1$ и s — локальный оптимум для входа $h(x) \in \Pi_2$, то $g(s, x)$ — локальный оптимум для входа x.

Задачу Π из класса PLS называют PLS–полной, если любая задача из этого класса может быть сведена к ней. Примеры PLS-полных задач можно найти в [27]. Ниже будет показано, что поиск локального минимума в задаче PMG с окрестностью *Swap* является PLS-полной задачей, т.е. самой трудной задачей в этом классе. Более того, будет получена экспоненциальная нижняя оценка на число шагов в худшем случае для алгоритмов локального улучшения. Эта оценка не зависит от правила замещения, т.е. от способа выбора лучшего соседа в окрестности, но не закрывает возможности получения локального оптимума (равновесия по Нэшу) за полиномиальное время другими алгоритмами.

Определение 5. *Графом переходов* $TG_{\Pi}(x)$ для входа x задачи Π называется ориентированный граф, вершинами которого являются все допустимые решения задачи. Дуга (s, s') принадлежит графу, если s' —соседнее решение для s и $F(s, x) > F(s', x)$. Высота вершины s есть длина кратчайшего пути в графе $TG_{\Pi}(x)$ из вершины s в сток, т. е. в локальный минимум. Высота графа $TG_{\Pi}(x)$ равна максимальной высоте его вершин.

Высота вершины есть, по сути, оценка снизу на число шагов алгоритма локального улучшения независимо от применяемого правила замещения. Таким образом, алгоритм требует экспоненциального числа шагов тогда и только тогда, когда найдутся исходные данные задачи, для которых высота графа переходов является экспоненциальной функцией от длины записи исходных данных.

Определение 6. Пусть задача Π_1 PLS-сводится к задаче Π_2 и h, g — соответствующие функции. Говорят, что сводимость является *плотной*, если для любого входа x задачи Π_1 можно указать такое подмножество R допустимых решений входа $y = h(x)$ задачи Π_2, что
1) в R содержатся все локальные минимумы для входа y

2) существует полиномиальный алгоритм, который для каждого решения p входа x позволяет находить решение $q \in R$ для входа y такое, что $g(q, x) = p$;

3) пусть граф переходов $TG_{\Pi_2}(y)$ содержит ориентированный путь из вершины $q \in R$ в вершину $q' \in R$ такой, что в нем нет промежуточных вершин из R, и пусть $p = g(q, x)$ $p' = g(q', x)$ — соответствующие решения для входа x. Тогда $p = p'$ или граф переходов $TG_{\Pi_1}(x)$ содержит дугу из вершины p в вершину p'.

Задачу Π из класса PLS называют плотно полной, если все задачи из этого класса плотно к ней сводятся. Известно [26], что следующая задача о раскраске вершин графа в два цвета является плотно полной. Задан неориентированный граф $G = (V, E)$ с весами $w_e, e \in E$ на ребрах. Допустимым решением задачи является любая раскраска вершин в два цвета, т.е. функция вида $c : V \to \{0, 1\}$. В качестве целевой функции используется суммарный вес ребер, вершины которых окрашены в один цвет. Окрестность *Flip* состоит из всех раскрасок, отличающихся от заданной цветом одной вершины. Задача состоит в поиске раскраски, являющейся локальным минимумом для окрестности *Flip*.

Теорема 4 Задача поиска *Flip*-минимальной раскраски плотно сводится к задаче локального поиска (PMG, *Swap*).

Доказательство. По заданному графу построим исходные данные задачи PMG. Положим $I = \{1, \ldots, 2|V|\}$, $J = \{1, ..., |V| + 2|E|\}$, $p = |V|$, $W = \sum\limits_{e \in E} |w_e| + 1$. Сопоставим каждому игроку одну и ту же матрицу c_{ij} следующим образом.

Каждой вершине $v \in V$ поставим в соответствие две строки i_v, i'_v и один столбец j_v. Каждому ребру $e = (j_1, j_2) \in E$ — два столбца $j_1(e), j_2(e) \in J$. При $j_v = 1, \ldots, |V|$ положим

$$c_{ij_v} = \begin{cases} 0, & \text{если } (i = i_v) \vee (i = i'_v), \\ W & \text{в противном случае .} \end{cases}$$

При $e = (j_1, j_2) \in E$ и $w_e \geq 0$ положим

$$c_{ij_1(e)} = \begin{cases} 0, & \text{если } \ (i = j_1(e)) \vee (i = j_2(e)), \\ w_e, & \text{если } \ (i = j'_1(e)) \vee (i = j'_2(e)), \\ W & \text{в противном случае,} \end{cases}$$

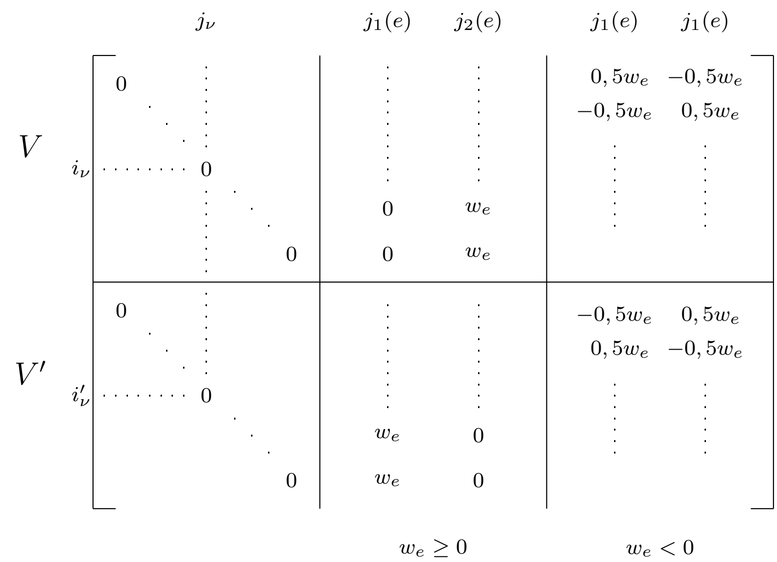
$$c_{ij_2(e)} = \begin{cases} 0, & \text{если } \ (i = j'_1(e)) \vee (i = j'_2(e)), \\ w_e, & \text{если } \ (i = j_1(e)) \vee (i = j_2(e)), \\ W & \text{в противном случае.} \end{cases}$$

Если же $w_e < 0$, то положим

$$c_{ij_1(e)} = \begin{cases} w_e/2, & \text{если } \ (i = j_1(e)) \vee (i = j'_2(e)), \\ -w_e/2, & \text{если } \ (i = j'_1(e)) \vee (i = j_2(e)), \\ W & \text{в противном случае,} \end{cases}$$

$$c_{ij_2(e)} = \begin{cases} w_e/2, & \text{если } \ (i = j'_1(e)) \vee (i = j_2(e)), \\ -w_e/2, & \text{если } \ (i = j_1(e)) \vee (i = j'_2(e)), \\ W & \text{в противном случае.} \end{cases}$$

Структура матрица (c_{ij}) изображена на фиг. 1.



Фиг. 1.

Предложенная конструкция определяет полиномиально вычислимую функцию h из определения 4. Функцию g зададим следующим образом. Пусть (x_i^k) — допустимое решение задачи PMG. Положим

$$c_x(v) = \begin{cases} 0, & \text{если найдётся такое } k, \text{ что } x_{i_v}^k = 1, \\ 1 & \text{в противном случае.} \end{cases}$$

Проверим, что $c_x(v)$ является локальным минимумом относительно окрестности *Flip*, если и только если (x_i^k) — локальный минимум для *Swap*. Заметим, что любой локальный минимум (x_i^k) обладает следующим свойством: для любого $v \in V$ существует единственный номер k такой, что $x_{i_v}^k + x_{i_{v'}}^k = 1$. Покажем сначала, что не существует вершины $v \in V$ такой, что $x_{i_v}^{k_1} = x_{i_v}^{k_2} = 1$, $k_1 \neq k_2$. Действительно допустим, что для некоторой вершины $v \in V$ найдутся такие номера. Так как $p = |V|$, то найдутся вершина $v_0 \in V$ и номер k_0 такие, что $x_{i_{v_0}}^{k_0} = x_{i_{v_0'}}^{k_0} = 0$. Построим новое решение $\overline{x}$, которое будет отличаться от прежнего только компонентами (k_0, i_{v_0}) и (k_1, i_v). Положим $\overline{x}_{i_{v_0}}^{k_0} = 1$ и $\overline{x}_{i_v}^{k_1} = 0$. Это соседнее решение для исходного и оно имеет меньшее значение целевой функции. Действительно, т.к. $x_{i_{v_0}}^{k_0} = x_{i_{v_0'}}^{k_0} = 0$, то слагаемое с номером j_{v_0} в целевой функции задачи PMG равно W. Полагая $\overline{x}_{i_{v_0}}^{k_0} = 1$, получаем уменьшение целевой функции на W. Замена $x_{i_v}^{k_1} = 1$ на $\overline{x}_{i_v}^{k_1} = 0$ может привести к росту целевой функции. Однако этот рост не превосходит суммарного веса ребер, инцидентных вершине v. Так как $W > \sum\limits_{e \in E} |w_e|$, то получаем требуемое. Случаи $x_{i_{v'}}^{k_1} = x_{i_v}^{k_2} = 1$ и $x_{i_{v'}}^{k_1} = x_{i_{v'}}^{k_2} = 1$ рассматриваются аналогично.

Итак, если (x_i^k) — локальный минимум для окрестности *Swap*, то для каждой вершины $v \in V$ найдется единственный номер $k(v)$ такой, что $x_{i_v}^{k(v)} + x_{i_{v'}}^{k(v)} = 1$. Покажем, что $c_x(v)$ является локальным минимумом для окрестности *Flip*. Предположим, что ребро $e \in E$ инцидентно вершинам одного цвета. Рассмотрим два случая.
1) Пусть $w_e \geq 0$. Для ребра $e = (j_1(e), j_2(e))$ найдем строки i_1, i_2 матрицы (c_{ij}^k), соответствующее вершинам $j_1(e), j_2(e)$. Пусть $k_1 = k(j_1(e))$ и $k_2 = k(j_2(e))$. Если $x_{i_1}^{k_1} = x_{i_2}^{k_2} = 1$, то $x_{i_1'}^{k_1} = x_{i_2'}^{k_2} = 0$ и в целевой функции слагаемое для $j = j_1(e)$ равно 0, а для $j = j_2(e)$ равно w_e, т.е. вес w_e учитывается как в задаче о раскраске, так и в задаче PMG. Если же $x_{i_1}^{k_1} = x_{i_2}^{k_2} = 0$, то $x_{i_1'}^{k_1} = x_{i_2'}^{k_2} = 1$ и в целевой функции слагаемое для $j = j_1(e)$ равно w_e, а для $j = j_2(e)$ равно 0. Снова приходим к тому же выводу.

2) Пусть $w_e < 0$. Если $x_{i_1}^{k_1} = x_{i_2}^{k_2} = 1$, то оба слагаемых в целевой функции равны $w_e/2$, что в сумме дает w_e. Если же $x_{i_1'}^{k_1} = x_{i_2'}^{k_2} = 1$, то снова получаем $w_e/2 + w_e/2 = w_e$.

Рассмотрим теперь ребро, соединяющее вершины разного цвета. Пусть $x_{i_1}^{k_1} = 1, x_{i_2}^{k_2} = 0$. Если $w_e \geq 0$, то слагаемые для $j = j_1(e)$ и $j = j_2(e)$ равны 0 и вес ребра не входит в целевую функцию. Если же $w_e < 0$, то слагаемое для $j = j_1(e)$ равно $w_e/2$, а для $j = j_2(e)$ равно $-w_e/2$, что в сумме снова дает ноль. Таким образом, значения целевых функций на решениях (x_i^k) и $c_x(v)$ совпадают.

Предположим, что решение $c_x(v)$ не является локальным минимумом. Тогда найдется вершина $v \in V$, изменение цвета которой приводит к падению целевой функции. В этом случае решение (x_i^k) не является локальным минимумом для окрестности *Swap*, так как изменение цвета вершины v соответствует замене

$$x_{i_v}^k := 1 - x_{i_v}^k, \quad x_{i_{v'}}^k := 1 - x_{i_{v'}}^k \text{ при } k = k(i_v)$$

с тем же изменением целевой функции. Значит $c_x(v)$ — локальный минимум и построенное сведение удовлетворяет требованиям определения 4.

Убедимся, что это сведение плотное. В качестве множества R возьмем все допустимые решения (x_i^k), обладающие следующим свойством: $x_{i_v}^{k(v)} + x_{i_{v'}}^{k(v)} = 1$ для любого $v \in V$. Ранее было показано, что это множество содержит все локальные минимумы задачи PMG, т. е. условие 1 определения 6 выполнено. Проверим условие 2. Пусть $c(v)$ — произвольная раскраска вершин в два цвета. Для k-го игрока найдем вершину v, такую, что $i_v = k$ и положим

$$x_{i_v}^k = \begin{cases} 1, & \text{если вершина } v \text{ имеет цвет } 0, \\ 0 & \text{в противном случае,} \end{cases}$$

$$x_{i_{v'}}^k = \begin{cases} 0, & \text{если вершина } v \text{ имеет цвет } 0, \\ 1 & \text{в противном случае.} \end{cases}$$

При $k \neq i_v$ положим $x_{i_v}^k = x_{i'_v}^k = 0$. Получаем взаимно-однозначное соответствие между раскрасками и элементами множества R.

Проверим последнее условие. Заметим, что в графе переходов задачи PMG нет дуг вида $(q_1, q_2), q_1 \in R, q_2 \notin R$, так как значение целевой функции для q_2 всегда больше чем для q_1. Другими словами, любой путь начинающийся и заканчивающийся в R целиком лежит в R. Путь без промежуточных вершин из R может быть только дугой вида (q_1, q_2), $q_1, q_2 \in R$. Но такой дуге соответствует дуга в графе переходов задачи о раскраске. Теорема доказана.

Рассмотрим теперь вопрос о трудоемкости получения равновесных решений с помощью следующего итерационного алгоритма. Пусть некоторое решение (x_i^k) не является равновесным, т.е. существует игрок, быть может не один, который может увеличить свою прибыль, выбрав другое предприятие. Шаг алгоритма состоит в нахождении такого игрока и выборе для него нового предприятия с увеличением прибыли. Процедура выбора нового предприятия может быть любой. Наша цель будет состоять в оценке числа шагов такого итерационного процесса в худшем случае при любой процедуре выбора.

Следствие 1 Итерационный алгоритм в худшем случае требует экспоненциального числа шагов для нахождения равновесного решения при любой процедуре выбора игрока и лучшего предприятия для него.

Справедливость утверждения следует из плотной PLS полноты задачи локального поиска $(PMG, Swap)$, и существования задачи экспоненциальной высоты в классе PLS [27]. Оценка остаётся экспоненциальной при использовании любого правила выбора игрока и лучшего предприятия для него: детерминированного, вероятностного или любого другого произвольной сложности.

В задаче поиска равновесных решений нужно найти хотя бы одно (любое) равновесное решение. Однако, если решения игроков уже известны, и они готовы их менять только при увеличении прибыли, то возникает другая задача, более важная с практической точки зрения. Найти равновесное решение, достижимое итерационным алгоритмом локального улучшения из заданной начальной позиции. Другими словами, в графе переходов задана стартовая вершина. Требуется найти сток, достижимый из стартовой вершины вдоль некоторого ориентированного пути. Оказывается, что такая задача является значительно более сложной, чем предшествующая.

Следствие 2 Нахождение равновесного решения при заданной начальной позиции игроков является PSPACE-полной задачей.

Доказательство. Покажем, что рассматриваемая задача принадлежит классу PSPACE. Из определения класса PLS следует, что длина любого решения ограничена полиномом от длины входа. Так как для работы итерационного алгоритма не требуется хранить все промежуточные позиции игроков и нахождение лучшего соседнего решения осуществляется за полиномиальное время, то требуемый объем памяти также ограничен полиномом от длины входа. Значит, задача принадлежит классу PSPACE.

Известно [27], что в классе PLS есть задачи локального поиска с фиксированной стартовой точкой, которые являются PSPACE-полными. Задача о раскраске вершин графа с окрестностью *Flip* — одна из них. Плотная сводимость, установленная в теореме 4, влечет полиномиальную сводимость соответствующих задач с фиксированными стартовыми решениями [27]. Следовательно, нахождение равновесного решения, достижимого из заданной начальной стратегии игроков является PSPACE-полной задачей. Теорема доказана.

Предположим теперь, что каждый игрок может отрывать несколько предприятий. В этом случае равновесным решением будет называться такое решение, когда ни одному из игроков не удается увеличить свою прибыль за счет смены своего решения на любое другое. Для такого случая легко выписать соответствующую оптимизационную задачу и определить подходящую окрестность. Эта окрестность будет включать в себя окрестность *Swap* и, следовательно, соответствующая задача локального поиска оказывается PLS–полной. Плотная полнота задачи со всеми вытекающими отсюда последствиями установлена в [9].

5 Заключение

В работе рассмотрены два класса конкурентных моделей размещения. Первый класс приводит к дискретным двухуровневым моделям для неравноправных игроков. Построение даже допустимого решения в такой задаче требует точного решения задачи Конкурента, что является известной NP-трудной задачей о покрытии на максимум. В работе предложен оригинальный способ построения семейства верхних и нижних оценок оптимума для Лидера. Это семейство имеет экспоненциальную мощность и дает возможность получения приближенного решения с апостериорной оценкой точности для исходной двухуровневой задачи. Исследование качества таких оценок и трудоемкости их получения требует численных экспериментов. Это направление представляется авторам исключительно важным, так как просмотреть все семейство не представляется возможным, а поиск лучшего элемента может оказаться не проще исходной задачи.

Для второго класса моделей показана связь равновесных решений с локальными оптимумами и установлена плотная PLS-полнота задачи нахождения таких решений. Как следствие получаем, что стандартный алгоритм локального улучшения с любым правилом замещения требует в худшем случае экспоненциального числа шагов. Нахождение равновесного решения при заданной начальной стратегии игроков оказывается PSPACE-полной задачей. Тем не менее, вопрос о трудоемкости нахождения равновесных решений, так же как и локальных оптимумом до сих пор остается актуальным. Если удастся доказать несуществование полиномиальных алгоритмов для нахождения равновесных решений, то отсюда будет следовать, что P$\neq$NP [10].

Список литературы

[1] *Алексеева Е. В., Кочетов Ю. А.* Генетический локальный поиск для задачи о p–медиане с предпочтениями клиентов. // Дискретный анализ и исследование операций, 2007, серия 2, Т. 14, N 1, 3–31.

[2] *Алексеева Е. В., Кочетова Н. А.* Верхние и нижние оценки для конкурентной задачи о p-медиане // Труды 14-й Байкальской международной школы–семинара "Методы оптимизации и их приложения". Т. 1. – Северобайкальск, 2008. С. 563–56.

[3] *Алексеева Е. В., Орлов А. В.* Генетичекий алгоритм для конкурентной задачи о p-медиане. // Труды 14-й Байкальской международной школы–семинара "Методы оптимизации и их приложения". Т. 1. – Северобайкальск, 2008. С. 570–576.

[4] *Береснев В. Л.* Дискретные задачи размещения и полиномы от булевых переменных. – Новосибирск. Изд-во Инст. математики. – 2005. – 408 с.

[5] *Береснев В. Л., Гимади Э. Х., Дементьев В. Т.* Экстремальные задачи стандартизации. – Новосибирск: Наука. – 1978. – 333 с.

[6] *Горбачевская Л. Е.* Полиномиально разрешимые и NP-трудные задачи стандартизации. Канд. дисс. Институт математики им. С. Л. Соболева СО РАН. – Новосибирск. – 1998.

[7] *Горбачевская Л. Е., Дементьев В. Т, Шамардин Ю. В.* Двухуровневая задача стандартизации с условием единственности оптимального потребительского выбора // Дискрет. анализ и исслед. операций. Сер. 2. 1999. Т. 6, № 2. С. 3–11.

[8] *Кочетов Ю. А.* Вычислительные возможности локального поиска в комбинаторной оптимизации // Журнал выч. матем. и матем. физики 2008, Т. 48, № 5. С. 747–764.

[9] *Кочетов Ю. А.* Равновесия по Нэшу в игровых моделях размещения // Труды 14-й Байкальской международной школы–семинара "Методы оптимизации и их приложения". Т. 1. – Северобайкальск, 2008. С. 119–127.

[10] *Кочетов Ю. А., Пащенко М. Г., Плясунов А. В.* О сложности локального поиска в задаче о p-медиане // Дискрет. анализ и исслед. операций. Серия 2. 2005. Т. 12, № 2. С. 44–71.

[11] *Хачатуров В. Р., Веселовский В. Е. и др.* Комбинаторные методы и алгоритмы решения задач дискретной оптимизации большой размерности. — М.: Наука. 2000.— 360 с.

[12] *Chan Y.* Location, transport and land-use. Modelling spatial–temporal information. Berlin. Springer. 2005.

[13] *Drezner Z.* (Ed.) Facility Location. A Survey of Applications and Methods. Springer. – 1995.

[14] *Drezner T., Eiselt H. A.* Consumers in competitive location models. In: Z. Drezner, H. Hamacher (Eds.) Facility Location. Applications and Theory. Springer. – 2004. 151–178.

[15] *Drezner Z., Klamroth K., Schobel A., Wesolowsky G.* The Weber Problem. In: Z. Drezner, H. Hamacher (Eds.) Facility Location. Applications and Theory. Springer. – 2004. 1–36.

[16] *Eiselt H. A., Sandblom C. -L.* Decision Analysis, Location Models, and Scheduling Problems. Springer. – 2004.

[17] *Krarup J., Pruzan P. M.* The simple plant location problem: survey and synthesis. // European J. Oper. Res. 1983. V. 12, P. 36–81.

[18] *Korte B., Vygen J.* Combinatorial Optimization. Theory and Algorithms. Third Edition, Springer. – 2005.

[19] *Mirchandani P. B., Francis R. L.*(Eds.) Discrete Location Theory. Wiley. – 1990.

[20] *Plastria F., Vanhaverbeke L.* Discrete models for competitive location with foresight // Computers and Oper. Res. 2008. V. 35, N. 3. P. 683–700.

[21] *Rodriguez C. M. C., Perez J. A. M.* Multiple voting location problems // European J. Oper. Res. 2008. V. 191, N. 2. P. 437–453.

[22] *Koutsoupias E., Papadimitriou C.* Worst-case equilibria // Proceedings of the 16th Annual Symposium on Theoretical Aspects of Computer Science. Trier, Germany. 1999. 404-413.

[23] *Tardos E., Wexler T.* Network formation games and the potential function method. In: N. Nisan, T. Roughgarden, E. Tardos, V. V. Vazirani (Eds.) Algorithmic Game Theory. Cambridge University Press. 2007. P. 487–516.

[24] *Schrijver A.* Combinatorial Optimization. Polyhedra and Efficiency. Berlin. Springer. 2003.

[25] *Vetta A.* Nash equilibria in competitive societies, with applications to facolity location, traffic routing and auctions // Proceedings of 43rd Annual IEEE symposiun on foundations of computer science, Vancouver, Canada. 2002. P. 416–425.

[26] *Vredeveld T., Lenstra J. K.* On local search for the generalized graph coloring problem // Oper. Res. Lett. 2003. V. 31, N. 4. P. 28–34.

[27] *Yannakakis M.* Computational complexity. In: E. Aarts, J. K. Lenstra (Eds.) Local Search in Combinatorial Optimization. Chichester: Wiley, 1997. P. 19–55.