ГЕНЕТИЧЕСКИЙ АЛГОРИТМ ДЛЯ КОНКУРЕНТНОЙ ЗАДАЧИ О P-МЕДИАНЕ[1]

Е.В. Алексеева [2], А.В. Орлов [3]

[2] *Институт математики им. С.Л. Соболева СО РАН, Новосибирский Государственный Университет, Новосибирск*
[3] *Институт динамики систем и теории управления СО РАН, Иркутск*
e-mail:ekaterina2@math.nsc.ru, anor@icc.ru

Аннотация. Рассматривается двухуровневая задача о p-медиане, в которой две фирмы, Лидер и Конкурент, последовательно размещают предприятия для обслуживания клиентов. Предлагается генетический алгоритм для нахождения приближенного решения задачи. Приводятся результаты численных экспериментов.

Ключевые слова: двухуровневое программирование, дискретные задачи размещения, генетические алгоритмы.

1. Введение

Теория дискретных задач размещения является одной из интенсивно развивающихся областей в исследовании операций. Возникновение этих задач и первые попытки их решения приписывают французским и итальянским математикам 17 века. По-настоящему бурное развитие модели размещения получили с рождением вычислительной техники. Линейные модели, модели частично–целочисленного программирования, статистические и динамические модели размещения, а также модели с нелинейными целевыми функциями рождались из приложений по размещению нефтяных и газовых станций, размещению предприятий, станций метро, милицейских и пожарных участков и др. Одной из таких задач является задача о p-медиане. В ней задано два конечных множества $I = \{1, \dots, m\}$ и $J = \{1, \dots, n\}$. Первое множество можно рассматривать как набор пунктов размещения предприятий для производства некоторого однородного продукта, второе — множество клиентов, использующих этот продукт. Известны величины $c_{ij}, \quad i \in I, \quad j \in J$, задающие транспортные затраты на обслуживание j-го клиента из i-го пункта. Требуется найти такое подмножество $S \subset I$ мощности не более p, чтобы суммарные затраты на обслуживание всех клиентов были минимальными. Комбинаторная формулировка задачи имеет следующий вид:

$$\min_{S \subset I} \{ \sum_{j \in J} \min_{i \in S} c_{ij} : |S| \leq p \}.$$

В этой модели предполагается, что решение принимает одно лицо, стоящее на высшем уровне иерархии. Отчасти это объясняется тем, что многие задачи размещения возникли в период централизованного управления. В последнее время в условиях рыночной экономики более актуальными становятся ситуации, когда несколько лиц на разных уровнях иерархии участвуют в процессе принятия решений. В таких моделях участники процесса имеют собственные цели и предпочтения, которые могут быть противоположными. В результате возникают конфликтные ситуации. Процесс принятия решений в таких системах выглядит следующим образом. Первым принимает решение верхний уровень иерархии. Это решение передаётся нижестоящим и уже не меняется. Каждый уровень иерархии, получив решение вышестоящих уровней, принимает

[1]Работа выполнена при поддержке РФФИ (№ грантов 07-06-13500-офи_ц, 08-07-00037), фонда содействия отечественной науке и гранта президента РФ МК-1497.2008.1

своё решение. Он преследует свои цели и использует имеющиеся у него возможности и ресурсы. Результат деятельности иерархической системы в целом зависит от работы всех уровней иерархии. Задача состоит в том, чтобы найти такое решение верхнего уровня, которое приводит систему к достижению глобальной цели.

Задачи, в которых рассматриваются только два уровня иерархии, называются задачами двухуровневого программирования. Обзор современного состояния в области двухуровневого программирования можно найти, например, в [3].

В данной работе рассматривается конкурентная модель о p-медиане, в которой два лица, Лидер и Конкурент, последовательно принимают решения. Первым принимает решение Лидер. Он открывает свое множество предприятий S. Затем, зная это решение, второе лицо – Конкурент открывает собственные предприятия, множество S_1. Каждый клиент из множества открытых предприятий $S \cup S_1$ выбирает одно предприятие, согласно предпочтениям. Обслуживание каждого клиента приносит доход. В зависимости от размещения предприятий множество клиентов делится между Лидером и Конкурентом. Каждое лицо стремится максимизировать число обслуженных клиентов. Задача состоит в том, чтобы найти множество S, позволяющее максимизировать доход Лидера. Эта задача содержит в себе классическую задачу о p-медиане, а следовательно даже нахождение допустимого решения конкурентной задачи о p-медиане является NP-трудной проблемой. Поэтому для её решения используются различные эвристические алгоритмы. На сегодняшний день эффективные методы решения данной задачи неизвестны. Первые шаги в этом направлении сделаны в [10], где излагается точная схема частичного перебора. В работе [1] предложен способ построения верхних оценок. В настоящее время достаточно успешными при решении практических задач являются итерационные методы локального поиска. К ним можно отнести поиск с запретами [4], поиск с чередующимися окрестностями [8], генетические алгоритмы [5], алгоритмы имитации отжига [2] и другие метаэвристики [7, 9]. Генетические алгоритмы доказали свою конкурентоспособность при решении многих NP-трудных задач и особенно в практических приложениях, где математические модели имеют сложную структуру, а применение стандартных методов типа ветвей и границ, динамического или линейного программирования крайне затруднено.

1. Математическая постановка задачи

Введем следующие обозначения:
$J = \{1, \ldots, n\}$ — множество клиентов;
$I = \{1, \ldots, m\}$ — множество потенциальных предприятий;
$p^0 \geq 1$, — число предприятий, открываемых Лидером;
$p^1 \geq 1$, — число предприятий, открываемых Конкурентом;
$r_j \geq 0,\ j \in J$ — доход от обслуживания j-го клиента;
$g_{ij} \geq 0,\ i \in I, j \in J$ — матрица предпочтений клиентов, если $g_{i_1 j} < g_{i_2 j}$, то j-й клиент из открытых предприятий i_1, i_2 выберет предприятие i_1.
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

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

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

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

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

Целевая функция задачи определяет 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.

2. Генетический алгоритм

Алгоритм решения задач оптимизации, основанный на идеях наследственности в биологических популяциях, впервые был предложен Дж.Холландом в 1975г. Дальнейшее развитие эти идеи, как и название – генетические алгоритмы, получили в работах Гольдберга и Де Йонга [6]. К решению оптимизационных задач впервые были применены в середине 70-х годов прошлого столетия. Цель генетического алгоритма при решении задачи оптимизации состоит в том, чтобы найти приближённое решение, близкое, но не гарантированно оптимальное решение. Стандартный генетический алгоритм начинает свою работу с формирования начальной *популяции*– конечного набора допустимых решений задачи. Эти решения могут быть выбраны случайным образом, получены с помощью вероятностных жадных алгоритмов или другими методами. Выбор начальной популяции не имеет значения для сходимости процесса в асимптотике, однако формирование "хорошей" начальной популяции (например из множества локальных оптимумов) может заметно сократить время достижения глобального оптимума.

Для оценки допустимых решений кроме целевой функции часто используется функция пригодности. Она выбирается с учетом специфики конкретной оптимизационной задачи и должна быть чувствительна к небольшим изменениям аргумента. Целевая функция не всегда удовлетворяет этому свойству как, например, в задачах упаковки в контейнеры, задачах теории расписаний на параллельных машинах или при календарном планировании в условиях

ограниченных ресурсов. Удачный выбор функции пригодности может существенно повысить эффективность генетических алгоритмов, хотя в простейшем случае в качестве нее можно выбирать целевую функцию задачи.

На каждом шаге генетического алгоритма с помощью вероятностного оператора *селекции* выбираются из популяции два решения x^1, x^2 в качестве *родителей*. Оператор *скрещивания* по этим решениям строит новое решение x'. Оно подвергается небольшим случайным модификациям, которые принято называть *мутациями*. Затем решение добавляется в популяцию, а наихудшее решение удаляется из нее. Так выглядит общая канва генетического алгоритма, при этом каждый его шаг может быть адаптирован к рассматриваемой задаче. Общая схема предлагаемого в работе генетического алгоритма выглядит следующим образом.

Генетический алгоритм

1. Выбрать начальную популяцию.
2. Пока не выполнен критерий остановки делать следующее.
   2.1. Выбрать родителей x^1, x^2 из популяции.
   2.2. Построить x' по решениям x^1, x^2.
   2.3. Применить к решению x' оператор мутации.
   2.3. Выполнить процедуру локального подъема.
   2.4. Если полученный локальный оптимум лучше наихудшего элемента в популяции, то обновить популяцию.
3. Выбрать наилучший элемент популяции в качестве ответа.

Генетический алгоритм начинает работу с формирования начальной популяции – набора допустимых решений Лидера. Для оценки допустимых решений используется целевая функция Лидера. Для вычисления её значения необходимо решить задачу Конкурента. Оптимальное решение задачи Конкурента находим с помощью метода ветвей и отсечений, заложенного в коммерческом пакете CPLEX для решения задач смешанного целочисленного программирования. Для улучшения качества начальной популяции предлагается следующая стратегия выбора начальных решений. Пусть x^* оптимальное решение задачи Лидера, минимизирующее суммарные расстояния до клиентов, в случае когда на рынке отсутствует Конкурент. Модифицируем решение x^*. Обозначим через S множество номеров открытых Лидером предприятий $S = \{i | x_i^* = 1, i \in I\}$. Выберем случайным образом пару элементов (i_1, i_0), где $i_1 \in S, i_0 \notin S$ и поменяем их местами. К полученному таким образом решению применим процедуру локального улучшения. Полученный локальный оптимум добавим в популяцию. Повторим выбор пар и процедуру локального улучшения пока не сформируем популяцию нужного объема.

Реализация стандартной процедуры локального подъема требует просмотра всей окрестности, то есть перебора всех возможных пар (i_1, i_0) и вычисления целевой функции Лидера. Это является трудоемким процессом из-за решения внутренней задачи Конкурента. Поэтому в работе предлагается просматривать не всю окрестность, а только её часть. Для каждого элемента из множества S определим l ближайших предприятий из множества $\overline{S}$. В качестве i_1 выберем случайным образом элемент из множества S, а в качестве i_0 один из l ближайших к i_1 предприятиям. Решим задачу Конкурента на новом множестве открытых предприятий S'. Если значение целевой функции Лидера окажется лучше на этом множестве, то алгоритм продолжает работу с новым решением. Иначе, алгоритм ищет другую пару (i_1, i_0) среди не участвовавших ранее, это повторяется t раз. Таким образом получается алгоритм локального подъема по окрестности 1-замена из $(t * l)$ решений. Параметры t и l выбираются на основе эмпирических исследований.

На каждом шаге 2.1 генетического алгоритма с помощью турнирной селекции [5] из популяции выбирается пара решений. По этим решениям однородным оператором скрещивания

строится новое решение x'. Оно наследует общие координаты родителей. Остальные координаты выбираются с вероятностью от одного из них. Для нахождения значения целевой функции Лидера решается задача Конкурента. К полученному решению применяется оператор мутации, который меняет одну из ненулевых координат вектора x' на противоположную. Далее к модифицированному решению применяется описанная выше процедура локального подъема. Её использование позволяет генетическому алгоритму сосредоточиться только на локальных оптимумах. Если в результате этой процедуры найденный локальный оптимум оказывается лучше наихудшего элемента в популяции, то худший элемент удаляется из нее, а новый добавляется.

3 . Численные эксперименты

Предложенный алгоритм тестировался на примерах размерности $n = m = 100$. Матрица g_{ij} задавалась следующим образом. На квадрате со стороной 7000 случайным образом с равномерным распределением выбирались n точек. Величина g_{ij} полагалась равной евклидовому расстоянию между точками i, j. Другими словами, каждый клиент среди открытых предприятий выбирает ближайшее в евклидовой метрике. Величины r_j полагались равными 1.

В таблице 1 приведены результаты расчетов для $p^0 = p^1 = 10$.

Код примера	Верхняя оценка	Генетический алгоритм	Нижняя оценка
111	74	43	41
211	70	45	41
311	73	47	46
411	72	46	41
511	70	48	48
611	67	45	42
711	73	50	49
811	74	45	42
911	68	49	49
1011	70	47	46

Таблица 1. Результаты генетического алгоритма в сравнении с верхними и нижними оценками на значение целевой функции Лидера

В таблице 1 каждая строка соответствует одному случайно сгенерированному примеру. Код примера приводится в первой колонке таблицы. Во второй колонке приводится значение верхней оценки оптимума. В третьей колонке содержится приближённое решение, полученное разработанным генетическим алгоритмом. Параметры t, l были подобраны так, что просматривалась одна пятая часть от окрестности 1-замена. Четвёртая колонка содержит нижнюю оценку, полученную Лидером по следующей стратегии. Предположим, что Лидер не подозревает о существовании Конкурента. Он открывает свои предприятия стремясь минимизировать суммарные расстояния от клиентов до открываемых предприятий. После открытия предприятий Лидером на рынок выходит Конкурент. Он открывает свои предприятия, стремясь переманить у Лидера максимальное число клиентов. Оставшееся число клиентов (доход Лидера) при такой стратегии приводится в четвёртой колонке таблицы.

Представленные результаты демонстрируют большое расхождение между верхней оценкой и результатами генетического алгоритма. В то же время решения генетического алгоритма, мало отличаются от нижней оценки. С одной стороны это может свидетельствовать о том, что нижняя оценка является хорошей и может быть использована для получения начального

приближенного решения в эвристиках. С другой стороны большой разрыв между верхней оценкой и решением генетического алгоритма оставляет поле деятельности для улучшения и доработки этого алгоритма, что представляет несомненный интерес.

Авторы выражают искреннюю благодарность А.В. Еремееву за предоставленную возможность проведения расчетов в алгебраической моделирующей системе GAMS.

Список литературы

[1] Кочетов Ю.А. Верхние оценки для одной двухуровневой задачи о p-медиане // Российская конференция «Дискретная оптимизация и исследование операций». Материалы конф. 2007. С. 129.

[2] Aarts E. H. L, Korst J.H.M., van Laarhoven P. J. M. Simulated annealing. Local Search in Combinatorial Optimization. Chichester: Wiley. 1997. P. 91–120.

[3] Dempe S.J. Foundations of bilevel programming. Dordrecht: Kluwer Academic Publishers, 2002.

[4] Glover F., Laguna M. Tabu search. Boston: Kluwer Acad. Publ., 1997

[5] Goldberg D. E. Genetic algorithm in search, optimization and machine learning. Reading, MA: Addison-Wesley. 1989.

[6] Goldberg D. E. Simple genetic algorithms and the minimal deceptive problem. Genetic Algorithms and Simulated Annealing. Chapter 6. Los Altos, CA, Morgan Kauffman. 1987. P. 74–88.

[7] Kochetov Y., Alekseeva E., Levanova T., Loresh M. Large neighborhood local search for the p-median problem // Yugoslav Journal of Oper. Res. 2005. V. 15, № 1. P. 53–63.

[8] Hansen P., Mladenović N. Developments of variable neighborhood search. Essays and Surveys of Metaheuristics. Boston: Kluwer Acad. Publ., 2002. P. 415–440.

[9] P*é*trowski D.,Taillard S. Metaheuristics for Hard Optimization. Methods and Case Studies. Springer. 2006.

[10] Rodriguez C. M. C., Perez J. A. M. Multiple voting location problems // European J. Oper. Res. – 2008. (to appear).

MEMETIC ALGORITHM FOR THE COMPETITIVE P-MEDIAN PROBLEM

E. Alekseeva, A. Orlov

Sobolev Institute of Mathematics, Novosibirsk
Institute of System Dynamics and Control Theory, Irkutsk
e-mail: ekaterina2@math.nsc.ru, anor@icc.ru

Abstract. The bilevel p-median problem where two competitive firms, Leader and Follower, open facilities consecutively for delivering the clients is considered. A memetic algorithm produced an approximate solution is suggested. Computational results are discussed.

Key words: bilevel programming, discrete location competitive problems, memetic algorithms.