УДК 519.8

ВЕРОЯТНОСТНЫЙ ЖАДНЫЙ АЛГОРИТМ ПОИСКА ДЛЯ РЕШЕНИЯ ЗАДАЧ ТЕРРИТОРИАЛЬНОГО ПЛАНИРОВАНИЯ

А. В. Пономарев,
канд. техн. наук
Санкт-Петербургский институт информатики и автоматизации РАН

Описывается опыт создания вероятностного жадного алгоритма поиска и применения его при решении задачи формирования промышленно-сырьевых узлов в ходе осуществления геолого-экономического районирования территорий. Производительность созданного алгоритма сравнивается с производительностью популярного решателя задач линейного и целочисленного программирования IBM ILOG CPLEX.

Ключевые слова *— алгоритм, группировка, задача назначения, целочисленное программирование, жадные алгоритмы, локальный поиск, CPLEX.*

Введение

Геолого-экономическое районирование является методом территориального анализа, позволяющим оценивать структуру и размещение минерально-сырьевых ресурсов и перерабатывающих мощностей исследуемого региона. В ходе районирования происходит выделение кластеров взаимоувязанных сырьевых, промышленных объектов и объектов инфраструктуры, кластеры именуются и наносятся на геолого-экономическую карту.

Одной из целей этого процесса является снижение затрат на разработку месторождений за счет совместного использования инфраструктуры и перерабатывающих мощностей. Это позволяет, в частности, производить добычу на тех месторождениях, которые самостоятельно разрабатывать неэффективно.

Районирование — это многоуровневая задача, исходными данными для которой являются месторождения, объекты перерабатывающей отрасли и инфраструктуры. Первый уровень районирования заключается в формировании так называемых промышленно-сырьевых узлов (ПСУ).

Под *промышленно-сырьевым узлом* понимается группа сближенных месторождений, обладающих одинаковым набором полезных ископаемых и единой технологией получения первого товарного продукта. Упрощенно, ПСУ — это одно или несколько месторождений и центр обогащения руды.

Одним из наиболее употребляемых критериев при формировании ПСУ является обеспечение максимального интегрального дохода от освоения ресурсов недр [1, 2]. Интегральный доход от создания одного узла представляет собой экономический эффект от переработки руды, добываемой на месторождениях, входящих в состав узла, в узловом обогатительном предприятии (центре узла) с учетом затрат на транспортировку руды. Таким образом, формирование ПСУ сводится к выделению на заданной территории такого набора узлов, чтобы суммарный интегральный доход по всем этим узлам был максимален.

Постановка задачи

Пусть S — множество месторождений, а C — множество возможных точек размещения узловых обогатительных предприятий (в том числе и уже действующих). Количество месторождений обозначим n, а количество возможных точек обработки — m. В предлагаемой модели каждое месторождение $s_i \in S$ характеризуется следующими параметрами: потенциальной стоимостью всех минеральных ресурсов месторождения (v_i), запасами месторождения в натуральных единицах (w_i), суммарными затратами на добычу всех минеральных ресурсов ($a_i^{(e)}$) и эксплуатационными затратами на переработку всех минеральных ресурсов месторождения ($o_i^{(p)}$).

Капитальные затраты на переработку, т. е. вложения в строительство или модернизацию обогатительного предприятия, будем считать пропорциональными планируемой нагрузке на это предприятие, а именно суммарным ресурсам всех ме-

сторождений, входящих в узел, центром которого предприятие является.

В том случае, если в точке c_j еще нет предприятия, капитальные затраты на его создание вычисляются следующим образом:

$$c_j^{(p)} = k_j \sum_{i|G(i,j)} w_i,$$

где k_j — коэффициент пропорциональности, а предикат $G(i, j)$ выражает факт вхождения месторождения s_i в узел, центр которого находится в точке c_j.

Если в точке c_j уже находится предприятие, то капитальные затраты определяются разницей между объемом ресурсов, который планируется включить в узел, и текущим объемом ресурсов, назначенных этому предприятию. Пусть l_j — текущий объем ресурсов, назначенных предприятию. Тогда для существующего предприятия в точке c_j

$$c_j^{(p)} = k_j \left(\sum_{i|G(i,j\}} w_i - l_j \right).$$

Недостатком такого способа оценки затрат является то, что в случае, когда в некотором оцениваемом варианте решения существующему предприятию предлагается назначить меньше ресурса, чем ему назначено фактически, затраты, вычисленные по формуле, становятся отрицательными. Другими словами, к целевой функции задачи оптимизации прибавляется некоторая величина, эквивалентная возврату уже сделанных вложений в это предприятие. В решаемой задаче такой эффект нежелателен, поэтому окончательное выражение для затрат будет иметь следующий вид:

$$c_j^{(p)} = \max\left(0, k_j \left(\sum_{i|G(i,j\}} w_i - l_j \right)\right).$$

Таким образом, каждая возможная точка создания узла характеризуется k_j — коэффициентом, выражающим зависимость капитальных вложений в предприятие от суммарного ресурса, который должен быть предприятию назначен, и l_j — текущим объемом ресурсов, назначенных предприятию (для точек, в которых нет предприятий, очевидно, $l_j = 0$).

Предполагается также, что задана матрица транспортных затрат, элемент которой t_{ij} соответствует затратам на перевозку всех ресурсов месторождения s_i в точку c_j.

Введем два вида независимых переменных: y_j — бинарную переменную, соответствующую факту создания узла с центром в c_j; x_{ij} — бинарную переменную, соответствующую факту вхождения месторождения s_i в узел с центром в c_j.

Тогда задачу поиска такого множества узлов, что интегральный доход по всем узлам максимален, можно записать следующим образом:

— максимизировать

$$z = \sum_{i=1}^{n}\sum_{j=1}^{m} x_{ij}\left(v_i - t_{ij} - a_i^{(e)} - o_i^{(p)}\right) - \\ - \sum_{j=1}^{m} y_j \left(\max\left(0, k_j \left(\sum_{i=1}^{n} (x_{ij} w_i) - l_j \right)\right)\right) \qquad (1)$$

при ограничениях

$$\sum_{i=1}^{m} x_{ij} \le 1, \ i = \overline{1,n}; \qquad (2а)$$

$$x_{ij} \le y_j, \ i = \overline{1,n}, \ j = \overline{1,m};$$

$$x_{ij}, \ y_j \in \{0,1\}, \ i = \overline{1,n}, \ j = \overline{1,m}. \qquad (2б)$$

Ограничение (2а) выражает требование того, что месторождение может относиться не более чем к одному ПСУ, а ограничение (2б) обеспечивает целостность набора независимых переменных задачи — месторождение может входить в узел только с таким возможным центром c_j, который является действительным центром в текущем решении.

Упростим целевую функцию. Заметим, что выражение $v_i - t_{ij} - a_i^{(e)} - o_i^{(p)}$ состоит только из параметров исходных объектов; обозначим его через d_{ij}. Внесем переменную y_j внутрь выражения max() (это допустимо, учитывая ограничения на значения y_j). Саму функцию max() представим в виде набора переменных g_j, на значения которых наложены соответствующие ограничения (и приняв во внимание, что в практической интерпретации задачи k_j, l_j, w_i неотрицательны). Наконец, заметим, что с учетом (2б) $x_{ij} y_j = x_{ij}$. Тогда задачу можно переписать следующим образом:

— максимизировать

$$z = \sum_{i=1}^{n}\sum_{j=1}^{m} x_{ij} d_{ij} + \sum_{j=1}^{m} q_j$$

при ограничениях

$$q_j \le 0, \qquad j = \overline{1,m};$$

$$q_j \le y_j k_j l_j - k_j \sum_{i=1}^{n} x_{ij} w_i, \quad j = \overline{1,m};$$

$$\sum_{j=1}^{m} x_{ij} \le 1, \qquad i = \overline{1,n};$$

$$x_{ij} \le y_j, \qquad i = \overline{1,n}, \quad j = \overline{1,m};$$

$$x_{ij}, \ y_j \in \{0,1\}, \qquad i = \overline{1,n}, \quad j = \overline{1,m}.$$

Данная задача оптимизации представляет собой задачу целочисленного (даже булева) линейного программирования и на практике решается

обычно с применением промышленных пакетов оптимизации (IBM ILOG CPLEX, GUROBI Optimizer, MOSEK и т. п.).

Для оценки применимости промышленных пакетов оптимизации и, в общем, оценки трудоемкости данного вида задач для классических методов решения задач целочисленного линейного программирования был проведен вычислительный эксперимент. Оценивалось время решения задачи для сгенерированных случайным образом наборов исходных данных двух размерностей: $m = n = 100$ и $m = n = 200$. В каждой размерности было оценено время для 400 наборов данных. Эксперимент выявил существенную разницу во времени нахождения решения. Для основной массы наборов решение находилось относительно быстро, но для некоторых время поиска решения было в сотни раз больше. Проще всего проиллюстрировать результаты эксперимента с помощью порядковых статистик. В табл. 1 показаны результаты запусков решателя IBM ILOG CPLEX для наборов разного размера с различными требованиями к точности получаемого результата.

Времена работы решателя на исходных наборах были отсортированы по возрастанию, в таблице приведены некоторые процентили этой последовательности. Столбец «Допустимая погрешность» отражает требование к максимальной разнице между оценкой оптимального решения сверху (получаемой обычно через решение ослабленной задачи со снятым ограничением целочисленности) и наилучшим на данный момент решением. То есть при допустимой погрешности 5 % процесс решения останавливается, если разница между оценкой оптимального решения сверху и наилучшим известным решением становится менее 5 % от наилучшего известного решения.

Из таблицы видно, что для практических задач, типичные размерности которых не превышают 200, применение CPLEX оказывается вполне оправданным. Однако задача формирования ПСУ обладает двумя особенностями, которые вынуждают искать альтернативные пути решения.

■ ***Таблица 1.*** Характеристики времени работы CPLEX

Размер набора	Допустимая погрешность, %	Процентиль по времени работы, с				
		50	60	70	80	90
100	—	2,01	2,65	3,4	4,49	> 60
	1	1,69	2,15	2,94	3,8	6,5
	2	1,53	1,98	2,66	3,41	4,98
	5	1,04	1,46	1,95	2,68	3,68
200	—	11,29	16,34	24,18	> 120	> 120
	1	10,27	15,09	21,84	42,39	> 120
	2	9,6	13,45	19,24	30,99	> 120
	5	6,21	8,42	12,5	18,29	29,92

Особенностями этими являются размерность (типовая — около 300 объектов) и повышенные требования к скорости получения решения. Последняя особенность связана с тем, что формирование ПСУ представляет собой интерактивный процесс, в ходе которого приведенная задача математического программирования решается с различными исходными данными, а результаты сопоставляются. Вместе с тем, учитывая такой характер решения задачи, находимое решение не обязательно должно быть оптимальным. В совокупности указанные особенности допускают применение для решения задачи формирования ПСУ эвристических алгоритмов.

Вероятностные жадные алгоритмы поиска

Вероятностные жадные алгоритмы поиска (Greedy Randomized Adaptive Search Procedures — GRASP) — эвристика, нашедшая применение при решении задач комбинаторной оптимизации [3–8]. Применение GRASP представляет собой итеративный процесс, каждый шаг которого состоит из двух фаз: фазы конструирования, где происходит формирование допустимого решения, и фазы локального поиска, когда исходное решение улучшается до локально-оптимального. Наилучшее решение сохраняется и является результатом работы алгоритма в целом.

На фазе конструирования допустимое решение формируется поэлементно, причем семантика «элемента» тесно связана с моделью решаемой задачи. В задачах поиска пути таким элементом может быть переход из вершины в вершину, в задачах назначения — одно назначение и т. п. На каждом шаге фазы конструирования один элемент добавляется в создаваемое решение. Выбор осуществляется на основе упорядочения всех элементов, которые могут быть добавлены (допустимых элементов) по значению некоторой функции оценки g.

Эта функция показывает привлекательность выбора соответствующего элемента (с точки зрения получения наилучшего исходного решения задачи). Эвристика является адаптивной в том смысле, что на каждом шаге фазы конструирования кандидаты заново упорядочиваются по значению функции g. Вероятностный компонент связан с тем, что выбирается не всегда наилучший (в смысле значения функции g) кандидат, но один из ограниченного списка наилучших кандидатов. Этот список в исходном описании эвристики получил название список кандидатов (Restricted Candidate List — RCL). Факт выбора случайного элемента на каждом шаге позволяет получать в конце фазы конструирования различные решения, увеличивая охват исследуемой части пространства решений.

Рассмотрим один из популярных методов выбора кандидата. Пусть задан параметр $\alpha \in [0, 1]$, пусть C — множество элементов, которые могут быть присоединены к решению на очередном шаге: $s^M = \max\{g(t) \mid t \in C\}$, $s^m = \min\{g(t) \mid t \in C\}$, и решается задача поиска максимума. Тогда RCL будет формироваться как $\{c \mid g(c) \geq s^M - \alpha(s^M - s^m)\}$, а добавляемый элемент — выбираться из RCL случайным образом. Параметр α, таким образом, управляет степенью «жадности» и случайности алгоритма.

При значении $\alpha = 0$ алгоритм становится жадным (детерминированным), а при $\alpha = 1$ — полностью случайным.

Другим распространенным способом формирования RCL является задание максимального размера RCL — ρ. В этом случае в RCL всегда отбираются ρ лучших кандидатов вне зависимости от конкретных значений.

Полученное таким образом решение, скорее всего, не будет локально оптимальным по отношению к какому-либо естественному для задачи определению окрестности, поэтому применение фазы локального поиска практически всегда способно это решение улучшить.

GRASP для задачи формирования ПСУ

Для применения GRASP к задаче следует определить представление ее решения, алгоритм фазы конструирования решения и понятие окрестности, с использованием которого будет производиться локальный поиск.

Представление задачи

Пронумеруем все месторождения натуральными числами от 1 до n, а все точки возможного размещения предприятий — от 1 до m. Будем кодировать текущее назначение месторождений к центрам в виде массива A из n элементов, причем значение элемента с индексом i соответствует индексу центра того узла, в который входит месторождение i, или 0, если месторождение i не включено ни в один узел. Кроме того, в массиве P, состоящем из m элементов, будем для каждой точки возможного размещения предприятия j запоминать значение суммы $\sum_{i|A_i=j} w_i$. С учетом информации, находящейся в этих двух массивах, операции включения месторождения в узел и исключения месторождения из узла будут выполнимы за O(1).

Алгоритм фазы конструирования

Под элементами, которые могут быть добавлены в решение на очередном шаге фазы конструирования, понимаются факты назначения месторождения определенному центру (включения в узел), причем на каждом шаге рассматриваются лишь еще не включенные в узел месторождения. Функция оценки g вычисляется как эффект в целевой функции от такого включения. Выражение для оценки включения месторождения i в узел с центром в j будет записываться так:

$$g(i, j) = \min(0, k_j l_j - k_j(P_j + w_i)) - \\ - \min(0, k_j l_j - k_j P_j) + d_{ij}.$$

Для ускорения фазы конструирования применяется расширенная очередь с приоритетами PQ, в которой находятся все элементы, активные на данном шаге. Очередь упорядочена по значению функции g для элементов таким образом, что первым элементом очереди является элемент с наибольшим значением g, т. е. самый предпочтительный кандидат на включение в конструируемое решение. Расширение же интерфейса очереди заключается в обеспечении ею операции удаления произвольного элемента (не обязательно с наибольшим значением g).

В алгоритме используется RCL фиксированного размера (ρ), для его формирования выбирается ρ элементов из PQ с наибольшими значениями g.

На каждой итерации фазы конструирования после выбора элемента, добавляемого в решение ((i_a, j_a) с оценкой g_a), осуществляется согласованная корректировка A, P и PQ. А именно, из очереди исключаются те элементы, которые в качестве распределяемого месторождения содержат i_a. Изменяются приоритеты в PQ для элементов, представляющих собой присоединение к j_a еще не входящих в узлы месторождений, Ai_a устанавливается равным j_a, а Pj_a увеличивается на g_a.

Сложность одного шага алгоритма оценивается как $O((\rho + m + n)\log(mn))$.

Алгоритм фазы локального поиска

Пусть A — массив назначений месторождений решения Q, а A' — массив назначений месторождений для решения Q'. Решение Q' входит в окрестность, через которую определяется процедура локального поиска, если выполняется одно из условий:

— A' и A различаются значением одного компонента:

$$|\{i \mid A_i \neq A'_i\}| = 1;$$

— в Q' для какой-либо одной пары месторождений, включенных в Q в узлы, номера узлов поменяны местами:

$$|\{(i, j) \mid i \neq j \vee A_i \neq A_j \vee A'_i = A_j \vee A'_j = A_i\}| = 1.$$

На каждой итерации поиска происходит переход к тому решению из окрестности, для которого целевая функция принимает наибольшее значение.

Экспериментальное исследование алгоритма

Эксперименты с GRASP производились на ПК с процессором Pentium Dual-Core CPU @2.2ГГц. Аналогичная конфигурация использовалась и в экспериментах с CPLEX. Алгоритм был реализован на языке C++, в качестве очереди приоритетов использовался класс relaxed_heap из библиотеки Boost. Машинный код получен с помощью Microsoft Visual Studio 2005 Express Edition, режим оптимизации «/O2».

Наборы для экспериментального исследования и сравнения алгоритмов были сгенерированы случайным образом. Для части наборов с помощью решателя IBM ILOG CPLEX были получены точные решения, однако для некоторых наборов данных получение точного решения на стендовом ПК было затруднено из-за исчерпания виртуальной памяти. В отличие от CPLEX, существенной разницы во времени решения с помощью GRASP для наборов одинаковой размерности не выявлено (при одинаковых параметрах алгоритма).

Пример результата одного из экспериментов приведен в табл. 2. Размер исходного набора данных: $m = n = 200$; точное решение, полученное с помощью CPLEX за 27 с, 5335,11. Было произведено 10 запусков алгоритма GRASP с различными начальными значениями генератора случайных чисел. Для каждого запуска было установлено ограничение в 1000 итераций. Размер RCL был установлен равным 32. В таблице показано, в скольких запусках, на какой итерации алгоритма и за какое время было достигнуто решение, превышающее значение, указанное в первой колонке.

■ *Таблица 2.* Экспериментальное исследование GRASP

Значение	Количество	Итерации			Время, с		
		мин.	макс.	средн.	мин.	макс.	средн.
5329	10	10	117	36,00	0,62	7,44	2,35
5330	10	11	285	76,00	0,74	18,40	4,95
5331	10	12	318	191,00	0,77	20,88	12,45
5332	6	29	989	462,00	1,86	63,83	30,13
5333	2	318	989	653,00	20,72	63,83	42,28
5334	0	–	–	–	–	–	–
5335	0	–	–	–	–	–	–

Видно, что хотя оптимальное решение для этого набора данных получено не было, алгоритму удавалось находить решения, отличающиеся от оптимального на 0,1 %.

Заключение

В статье описан вероятностный алгоритм поиска (GRASP) для решения задачи дискретной оптимизации, возникающей при оптимальном формировании ПСУ на заданной территории. Построенный алгоритм, протестированный на наборах данных, сгенерированных случайным образом, показал способность к быстрому нахождению решений, близких к оптимальным.

Литература

1. **Куклина Е. А.** Природно-промышленные комплексы как основа устойчивого развития природно-ресурсных регионов России // Образование, экономика, общество. 2007. № 4. С. 44–51.
2. **Куклина Е. А.** Методологический подход к локализации промышленно-сырьевых узлов как центров формирования горно-промышленных комплексов // Современные проблемы экономики и организации промышленных предприятий. — СПб.: СПГИЭУ, 2007. С. 112–119.
3. **Resende M. G. C., Werneck R. F.** A hybrid multistart heuristic for the uncapacitated facility location problem // European J. of Operational Research. 2006. Vol. 174. P. 54–68.
4. **Resende M. G. C., Marti R., Gallego M., Duarte A.** GRASP and path relinking for the max-min diversity problem // Computers and Operations Research. 2010. Vol. 37. P. 498–508.
5. **Oliveira C. A., Pardalos P. M., Resende M. G. C.** GRASP with path relinking for the quadratic assignment problem // Proc. of III Workshop on Efficient and Experimental Algorithms (WEA2004) / Ed. C. C. Ribeiro and S. L. Martins. 2004. Vol. 3059. P. 356–368.
6. **Moura A., Oliveira J. F.** A GRASP approach to the container-loading problem // IEEE Intelligent Systems. 2005. Vol. 20. P. 50–57.
7. **Festa P., Resende M. G. C.** An annotated bibliography of GRASP. Part I: Algorithms // International Transactions in Operational Research. 2009. Vol. 16. P. 1–24.
8. **Festa P., Resende M. G. C.** An annotated bibliography of GRASP. Part II: Applications // International Transactions in Operational Research. 2009. Vol. 16. P. 131–172.