Лекция 7. Метаэвристики

Екатерина Вячеславовна Алексеева

Новосибирский Государственный Университет
Факультет Информационных Технологий
http://math.nsc.ru/~alekseeva/

5 мая, 2012 г.

Содержание лекции

Обозначения

$$\min_{s \in S} f(s)$$

f — целевая функция
S — множество допустимых решений
$s \in S$ — допустимое решение
$\mathcal{N} \subseteq S \times S$ — окрестность на множестве S
$\mathcal{N}(s) = \{s' \in S | \mathcal{N}(s, s')\} \subseteq S$ — окрестность решения s
$f^* = \min\{f(s') | s' \in \mathcal{N}(s)\}$
$\mathcal{N}^*(s) = \{s' \in \mathcal{N}(s) | f(s') \leq f(s)\}$

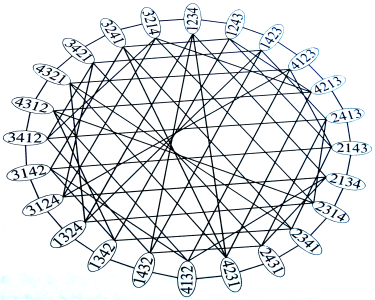
$s \in S$ (строгий) локальный минимум

если $(f(s) < f(s'))$ $f(s) \leq f(s')$ $\forall s' \in \mathcal{N}(s)$

Разновидности окрестностей

окрестность k-замена

решения отличаются в k компонентах $\mathcal{N}_k(s) = \{s' \in S | d(s, s') = k\}$, $1 \leq k \leq n$ где d — расстояние Хэмминга, n — число компонент в решении.



Множество всех решений заданных перестановкой и их соседи по окрестности 2-замена, $|\mathcal{N}_2(s)| = \frac{n(n-1)}{2}$

Пример. Соседи на перестановках

- перестановка местами двух разных компонент

Пример. Соседи на перестановках

- перестановка местами двух разных компонент

перемещение

1 2 3 4 5 6 9 7 8 ⇒ 1 2 3 7 5 6 9 4 8

- $|\mathcal{N}_{tran}(s)| = \frac{n(n-1)}{2}$

Пример. Соседи на перестановках

- перестановка местами двух разных компонент

  перемещение

  1 2 3 4 5 6 9 7 8 ⟹ **1 2 3 7 5 6 9 4 8**

- $|\mathcal{N}_{tran}(s)| = \frac{n(n-1)}{2}$
- перестановка двух рядом стоящих компонент

  инверсия

  1 2 3 4 5 6 9 7 8 ⟹ **1 2 3 5 4 6 9 7 8**

Пример. Соседи на перестановках

- перестановка местами двух разных компонент

перемещение

1 2 3 4 5 6 9 7 8 ⟹ 1 2 3 7 5 6 9 4 8

- $|\mathcal{N}_{tran}(s)| = \frac{n(n-1)}{2}$
- перестановка двух рядом стоящих компонент

инверсия

1 2 3 4 5 6 9 7 8 ⟹ 1 2 3 5 4 6 9 7 8

- $|\mathcal{N}_{inv}(s)| = n - 1$

Пример. Соседи на перестановках

- перестановка местами двух разных компонент

$$1\ 2\ 3\ \underbrace{4\ 5\ 6\ 9\ 7}\ 8 \xrightarrow{\textit{перемещение}} 1\ 2\ 3\ 7\ 5\ 6\ 9\ 4\ 8$$

- $|\mathcal{N}_{tran}(s)| = \frac{n(n-1)}{2}$
- перестановка двух рядом стоящих компонент

$$1\ 2\ 3\ \underbrace{4\ 5}\ 6\ 9\ 7\ 8 \xrightarrow{\textit{инверсия}} 1\ 2\ 3\ 5\ 4\ 6\ 9\ 7\ 8$$

- $|\mathcal{N}_{inv}(s)| = n - 1$
- перестановка компоненты на любое другое место в перестановке

$$1\ 2\ 3\ \vee\ \underbrace{4\ 5\ 6\ 9\ 7}\ 8 \xrightarrow{\textit{вставка}} 1\ 2\ 3\ 7\ 4\ 5\ 6\ 9\ 8$$

Пример. Соседи на перестановках

- перестановка местами двух разных компонент

$$1\,2\,3\,\underbrace{4\,5\,6\,9\,7}\,8 \xrightarrow{\textit{перемещение}} 1\,2\,3\,7\,5\,6\,9\,4\,8$$

- $|\mathcal{N}_{tran}(s)| = \frac{n(n-1)}{2}$
- перестановка двух рядом стоящих компонент

$$1\,2\,3\,\underbrace{4\,5}\,6\,9\,7\,8 \xrightarrow{\textit{инверсия}} 1\,2\,3\,5\,4\,6\,9\,7\,8$$

- $|\mathcal{N}_{inv}(s)| = n - 1$
- перестановка компоненты на любое другое место в перестановке

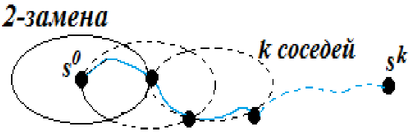
$$1\,2\,3 \vee \underbrace{4\,5\,6\,9\,7}\,8 \xrightarrow{\textit{вставка}} 1\,2\,3\,7\,4\,5\,6\,9\,8$$

- $|\mathcal{N}_{ins}(s)| = n(n-2) + 1$

Разновидности окрестностей

окрестность Лина-Кернигана $\mathcal{N}(k,s)$

$s^0 := s$, $i := 1$, $Tabu := \emptyset$;
пока $i \le k$
выбрать $s^i | f(s^i) = \min_{s' \in \mathcal{N}_2(s^{i-1}) \setminus Tabu} f(s')$,
обновить $Tabu$, *добавив компоненты, которые участвовали в перестроении решения* s^{i-1} *в* s^i;
$i := i + 1$;



Важно учитывать при выборе окрестности:

- мощность
- сложность оценивания соседнего решения
- окрестность называется *точной*, если любой локальный оптимум является глобальным

Пример точной и полиномиальной окрестности

Важно учитывать при выборе окрестности:

- мощность
- сложность оценивания соседнего решения
- окрестность называется *точной*, если любой локальный оптимум является глобальным

Пример точной и полиномиальной окрестности

- *Линейное программирование. Переход от одного базисного решения к другому в симплекс-методе.*

Важно учитывать при выборе окрестности:

- мощность
- сложность оценивания соседнего решения
- окрестность называется *точной*, если любой локальный оптимум является глобальным

Пример точной и полиномиальной окрестности

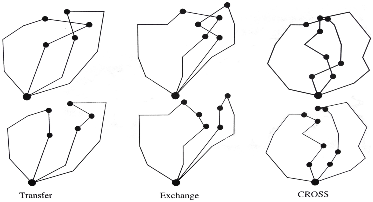
- *Линейное программирование. Переход от одного базисного решения к другому в симплекс-методе.*
- *Минимальное остовное дерево. Операция локальной перестройки остовного дерева.*

Задача о маршрутизации транспортных средств

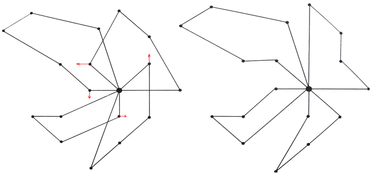
- Дано: бесконечное множество грузовиков, вместимостью V кг
  m грузов, весом v_i кг, $i = 1, ..., m$
  m клиентов
  d_{ij} — расстояние между клиентами i и j
  грузовики выезжают из депо и возвращаются обратно
- Найти: такие маршруты развозки грузов клиентам, чтобы суммарное пройденное расстояние было минимальным.

Построение окрестности для задачи о маршрутизации
Выталкивающие пути

Наиболее часто используемые окрестности

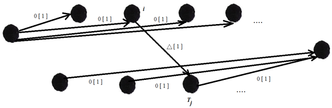


Новая окрестность на основе выталкивающих путей



Построение окрестности для задачи о маршрутизации Выталкивающие пути. Задача о потоке минимальной стоимости в двухслойной сети

Вершины сети: источник, сток, клиенты и обходы.
Дуги сети:
единичной пропускной способности и нулевой стоимости между источником и клиентами, между вершинами, соответствующими обходам и стоку
между вершинами–клиентами и вершинами-обходами, если клиента i можно переместить в обход T_j (согласно $TabuList$), стоимость дуги определяется выгодой от перемещения клиента в новый обход.



Решается задача о поиске потоков минимальной стоимости.

Общая схема алгоритма локального спуска (LS)

Шаг 1. Выбрать начальное решение $s \in S$, вычислить $f(s)$;
Шаг 2. *Выбрать* соседнее решение s' из окрестности $\mathcal{N}(s)$;
Шаг 3. Если $f(s') < f(s)$, то $s := s'$ и вернуться на шаг 2
иначе STOP.

Правила замещения

- *Спуск в направлении наилучшего элемента*. На каждом шаге локального спуска в множестве $\mathcal{N}^*$ выбирается допустимое решение с наименьшим значением целевой функции.
  $\ominus$ требует просмотра всей окрестности.

Правила замещения

- *Спуск в направлении наилучшего элемента*. На каждом шаге локального спуска в множестве $\mathcal{N}^*$ выбирается допустимое решение с наименьшим значением целевой функции.
  $\ominus$ требует просмотра всей окрестности.
- *Спуск в направлении наихудшего элемента*. В $\mathcal{N}^*$ выбирается элемент с наибольшим значением целевой функции.
  $\oplus$ пологий спуск к локальному минимуму
  $\ominus$ может потребоваться больше шагов, чем в предыдущем случае

Правила замещения

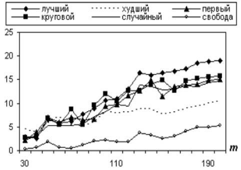
- *Спуск в направлении наилучшего элемента*. На каждом шаге локального спуска в множестве $\mathcal{N}^*$ выбирается допустимое решение с наименьшим значением целевой функции.
  $\ominus$ требует просмотра всей окрестности.
- *Спуск в направлении наихудшего элемента*. В $\mathcal{N}^*$ выбирается элемент с наибольшим значением целевой функции.
  $\oplus$ пологий спуск к локальному минимуму
  $\ominus$ может потребоваться больше шагов, чем в предыдущем случае
- *Спуск в направлении случайного элемента* в $\mathcal{N}^*$ выбирается элемент случайным образом, например, с равномерным распределением.

Правила замещения

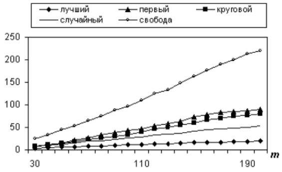
- *Спуск в направлении наилучшего элемента*. На каждом шаге локального спуска в множестве $\mathcal{N}^*$ выбирается допустимое решение с наименьшим значением целевой функции.
  $\ominus$ требует просмотра всей окрестности.
- *Спуск в направлении наихудшего элемента*. В $\mathcal{N}^*$ выбирается элемент с наибольшим значением целевой функции.
  $\oplus$ пологий спуск к локальному минимуму
  $\ominus$ может потребоваться больше шагов, чем в предыдущем случае
- *Спуск в направлении случайного элемента* в $\mathcal{N}^*$ выбирается элемент случайным образом, например, с равномерным распределением.
- *Спуск в направлении первый подходящий*. Поиск соседнего решения завершается, как только обнаружен первый элемент из множества $\mathcal{N}^*$.
  $\oplus$ не требуется просмотра всей окрестности.
  Если просмотр окрестности начинается со случайной точки, то близко к правилу *случайного элемента*. Чаще используется один и тот же порядок, например, лексикографический, и просмотр начинают с наименьшего элемента.

Сравнение правил замещения

p-медиана с предпочтениями

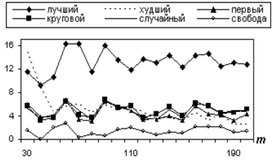


Средняя относительная погрешность (%), $p = m/10$

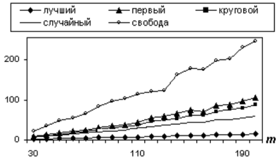


Среднее число шагов, $p = m/10$

p-медиана



Средняя относительная погрешность(%), $p = m/10$



Среднее число шагов, $p = m/10$

Алгоритм имитации отжига (SA: S. Kirkpatrick, C. Gelatt, M. P. Vecchi, 1982)

Выбрать начальное решение $s \in S$, вычислить $f(s)$
Задать начальную температуру T
Пока не выполнен критерий остановки выполнить:
    выбрать решение $s' \in \mathcal{N}(s)$ случайным образом;
    если $f(s') < f(s)$, то $s := s'$
    иначе с вероятностью $e^{\frac{f(s)-f(s')}{T}}$ положить $s := s'$;
    понизить температуру $T := T \cdot r$;
Выдать в качестве ответа s — локальный оптимум.

Поведение алгоритма имитации отжига

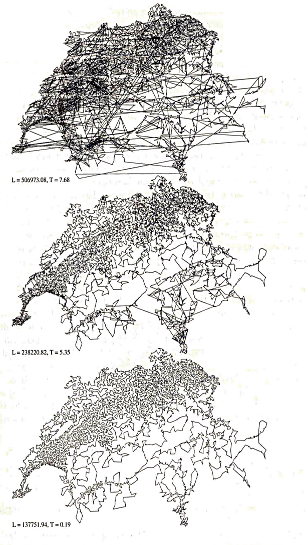


Fig. 1.2. The traveling salesman problem (13206 nodes of the Swiss road network):

Алгоритм поиска с запретами (TS: F. Glover, 1986)

Выбрать начальное решение $s \in S$, вычислить $f(s)$
Пока не выполнен критерий остановки выполнить:
  выбрать наилучшее решение $s' \in \mathcal{N}(s) \setminus TabuList$ с $f(s') < f(s)$
  $s := s'$
  по решению s' обновить $TabuList$
Выдать в качестве ответа s — локальный оптимум.

Запрет повторяющихся решений

$$hash(x) = \sum_{i=1}^{m} x_i 3^{m-i}, \text{ где } x_i \in \{0, 1, 2\}$$

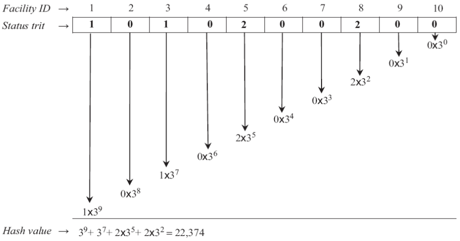


Fig. 1. The unique ternary string representation and hash value of a sample BFCLP solution σ.

Формирование Tabu List

Пусть решение задано перестановкой чисел от 1 до n.
$s = (\ldots, \underline{i, j,} k, p, \ldots)$

Формирование Tabu List

Пусть решение задано перестановкой чисел от 1 до n.
$s = (..., \underline{i, j,} k, p, ...)$

- $s^1 = (..., j, i, \underline{k, p}, ...)$, $TabuList := TabuList \cup (j, i)$

Формирование Tabu List

Пусть решение задано перестановкой чисел от 1 до n.
$s = (..., \underline{i, j,} k, p, ...)$

- $s^1 = (..., j, i, \underline{k, p}, ...)$, $TabuList := TabuList \cup (j, i)$
- $s^2 = (..., j, \underline{i, p}, k, ...)$, $TabuList := TabuList \cup (p, k)$

Формирование Tabu List

Пусть решение задано перестановкой чисел от 1 до n.
$s = (..., \underline{i, j,} k, p, ...)$

- $s^1 = (..., j, i, \underline{k, p}, ...)$, $TabuList := TabuList \cup (j, i)$
- $s^2 = (..., j, \underline{i, p}, k, ...)$, $TabuList := TabuList \cup (p, k)$
- $s^3 = (..., \underline{j}, p, i, \underline{k}, ...)$, $TabuList := TabuList \cup (p, i)$

Формирование Tabu List

Пусть решение задано перестановкой чисел от 1 до n.
$s = (..., \underline{i, j,} k, p, ...)$

- $s^1 = (..., j, i, \underline{k, p}, ...)$, $TabuList := TabuList \cup (j, i)$
- $s^2 = (..., j, \underline{i, p}, k, ...)$, $TabuList := TabuList \cup (p, k)$
- $s^3 = (..., \underline{j}, p, i, \underline{k}, ...)$, $TabuList := TabuList \cup (p, i)$
- $s^4 = (..., \underline{k}, p, \underline{i}, j, ...)$, $TabuList := TabuList \cup (k, j)$

Формирование Tabu List

Пусть решение задано перестановкой чисел от 1 до n.
$s = (..., \underline{i, j,} k, p, ...)$

- $s^1 = (..., j, i, \underline{k, p}, ...)$, $TabuList := TabuList \cup (j, i)$
- $s^2 = (..., j, \underline{i, p}, k, ...)$, $TabuList := TabuList \cup (p, k)$
- $s^3 = (..., \underline{j}, p, i, \underline{k}, ...)$, $TabuList := TabuList \cup (p, i)$
- $s^4 = (..., \underline{k}, p, \underline{i}, j, ...)$, $TabuList := TabuList \cup (k, j)$
- $s^5 = (..., i, \underline{p}, k, \underline{j}, ...)$, $TabuList := TabuList \cup (i, k)$

Формирование Tabu List

Пусть решение задано перестановкой чисел от 1 до n.
$s = (..., \underline{i, j}, k, p, ...)$

- $s^1 = (..., j, i, \underline{k, p}, ...)$, $TabuList := TabuList \cup (j, i)$
- $s^2 = (..., j, \underline{i, p}, k, ...)$, $TabuList := TabuList \cup (p, k)$
- $s^3 = (..., \underline{j}, p, i, \underline{k}, ...)$, $TabuList := TabuList \cup (p, i)$
- $s^4 = (..., \underline{k}, p, \underline{i}, j, ...)$, $TabuList := TabuList \cup (k, j)$
- $s^5 = (..., i, \underline{p}, k, \underline{j}, ...)$, $TabuList := TabuList \cup (i, k)$
- $s^6 = (..., i, j, k, p, ...)$, $TabuList := TabuList \cup (j, p)$
  Как нужно было формировать запреты, чтобы не было цикла?

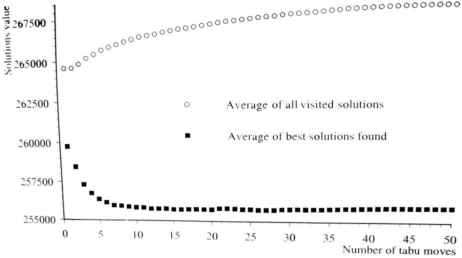
Формирование Tabu List

Пусть решение задано перестановкой чисел от 1 до n.
$s = (..., \underline{i, j}, k, p, ...)$

- $s^1 = (..., j, i, \underline{k, p}, ...)$, $TabuList := TabuList \cup (j, i)$
- $s^2 = (..., j, \underline{i, p}, k, ...)$, $TabuList := TabuList \cup (p, k)$
- $s^3 = (..., \underline{j}, p, i, \underline{k}, ...)$, $TabuList := TabuList \cup (p, i)$
- $s^4 = (..., \underline{k}, p, \underline{i}, j, ...)$, $TabuList := TabuList \cup (k, j)$
- $s^5 = (..., i, \underline{p}, k, \underline{j}, ...)$, $TabuList := TabuList \cup (i, k)$
- $s^6 = (..., i, j, k, p, ...)$, $TabuList := TabuList \cup (j, p)$
  Как нужно было формировать запреты, чтобы не было цикла?
- Например, пусть p_i — место элемента i в перестановке. Тогда $TabuList := TabuList \cup (i, p_i), (j, p_j)$

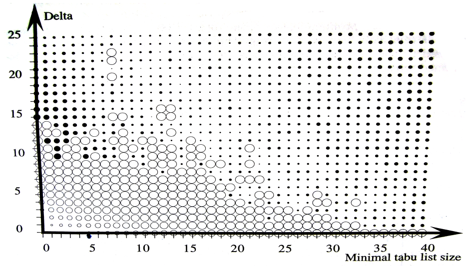
Длина Tabu List.
Результаты экспериментов для квадратичной задачи о назначениях

$n = 12$, 3000 примеров, 50 итераций TS.



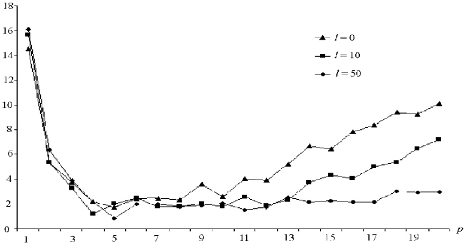
Варьирование длины Tabu List.
Результаты экспериментов для квадратичной задачи о назначениях

$n = 15$, 500 примеров
длина $TabuList$ варьируется от минимального до (минимального $+\, Delta$).
Размер круга — среднее число шагов TS до получения оптимума.
Окружность означает зацикливание.
Размер окружности — пропорционален количеству решенных задач.



Погрешность получаемых решений при разной длине $TabuList$ и степени рандомизации
Результаты экспериментов для многостадийной задачи размещения

l — длина $TabuList$, $l = 0, 10, 50$
$\varepsilon(T)$ — относительная погрешность после T шагов алгоритма
p — рандомизация окрестности



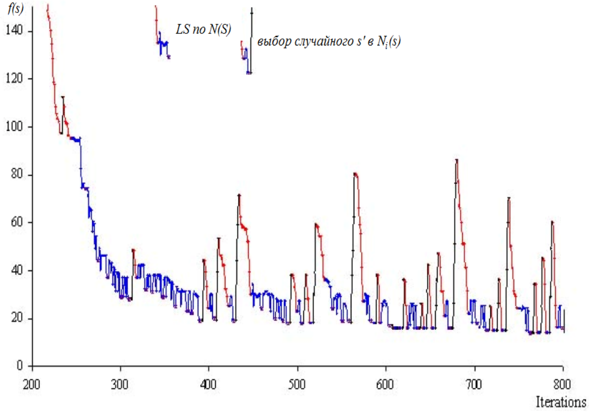
Изменение $\varepsilon(p)$ *при* $T = 20000$

Поиск с чередующимися окрестностями (VNS: N. Mladenović, P. Hansen, 1997)

Задана система разнотипных окрестностей $\mathcal{N}_1, \ldots, \mathcal{N}_k$.

Выбрать начальное решение $s \in S$, вычислить $f(s)$
Пока не выполнен критерий остановки повторять:
  $i := 1$;
  Пока $i \leq k$ выполнить:
    выбрать решение $s' \in \mathcal{N}_i(s)$ случайным образом;
    применить процедуру локального улучшения с окрестностью $\mathcal{N}$ к s',
    s''—полученное решение
    если $f(s'') < f(s)$, то $s := s''$
    иначе $i := i + 1$;
Выдать в качестве ответа s — локальный оптимум.

Поведение VNS



Вероятностный жадный алгоритм (GRASP: Feo, Resende, 1989)

Пока не выполнен критерий остановки выполнить:

жадной стратегией покомпонентно построить решение s

запустить процедуру локального поиска со стартовым решением s

Выдать в качестве ответа локальный оптимум.

GRASP для задачи MAX-SAT

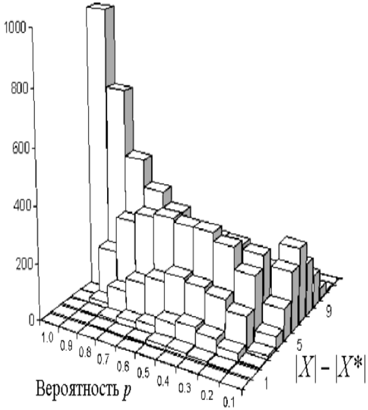
Пусть L множество переменных,
$h(i, v)$ — количество выполненных дизъюнкций при $x_i = v$, $v \in \{0, 1\}$.
i выбирается из списка, состоящего из k-лучших компонент, т.е. с наибольшем значением $h(i, v)$.
Построение списка:
определить $h_{min} = \min\{h(i), i \in L\}$, $h_{max} = \max\{h(i), i \in L\}$
i-ая компонента попадает в список, тогда и только тогда, когда
$h(i) \leq h_{min} + \alpha(h_{max} - h_{min})$

Алгоритм муравьиных колоний (ACO: M. Dorigo, V. Maniezzo, 1991)

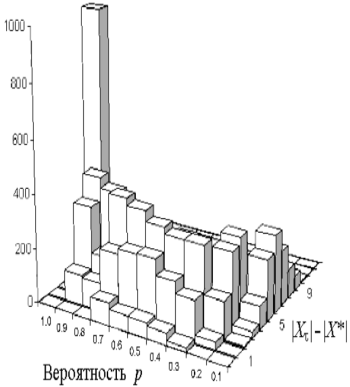
1. Задать исходную статистическую информацию.
2. Пока не выполнен критерий остановки делать следующее:
    2.1. Построить популяцию решений рандомизированным алгоритмом
    2.2. Применить процедуру локального поиска к решениям из популяции
    2.3. Выбрать часть наилучших решений из популяции
    2.4. По выбранным решениям обновить статистическую информацию

В качестве ответа предъявить наилучшее найденное решение.

Влияние статистической информации на качество получаемых решений



Влияние статистической информации



Эволюционные алгоритмы

Гибридная схема генетического алгоритма

1. Выбрать начальную популяцию из k решений. Запомнить рекорд $f^* = \min_{i=1,\dots,k} f(s_i)$.
2. Пока не выполнен критерий остановки делать следующее:
   2.1. Выбрать "родителей" s_{i_1} s_{i_2} из популяции.
   2.2. Применить к s_{i_1} s_{i_2} оператор скрещивания и получить новое решение s'.
   2.3. Применить к s' оператор мутации и получить новое решение s''.
   2.4. Применить к s'' оператор локального улучшения и получить новое решение s'''.
   2.5. Если $f(s''') < f^*$, то сменить рекорд $f^* := f(s''')$.
   2.6. Добавить s''' к популяции и удалить из нее наихудшее решение.

Генетический алгоритм (GA, Holland, 1975, Goldberg, 1989)

Имитационный алгоритм без вспомогательной процедуры локального улучшения на шаге 2.4.

Разновидности операторов кроссовера

- *Равномерный*
  новое допустимое решение y': если $y_i^1 = y_i^2$ для некоторого $i \in I$, то $y_i' = y_i^1 = y_i^2$. Для остальных $i \in I$ сначала полагается $y_i' = 0$, а затем выбирается одна из координат, для которой $y_i' = 0$ и полагается $y_i' = 1$. Выбор координаты проводится случайным образом с равномерным распределением.

Разновидности операторов кроссовера

- *Равномерный*
  новое допустимое решение y': если $y_i^1 = y_i^2$ для некоторого $i \in I$, то $y_i' = y_i^1 = y_i^2$. Для остальных $i \in I$ сначала полагается $y_i' = 0$, а затем выбирается одна из координат, для которой $y_i' = 0$ и полагается $y_i' = 1$. Выбор координаты проводится случайным образом с равномерным распределением.
- *Жадный*
  Среди претендентов выбирается та координата, для которой уменьшение целевой функции будет наибольшим. Если уменьшение целевой функции таким способом невозможно, то выбирается координата с наименьшим увеличением целевой функции. В остальном жадный оператор совпадает с равномерным. По сути данный оператор предполагает решение исходной задачи с помощью жадной процедуры на множестве координат, где родители отличаются друг от друга.

Разновидности операторов кроссовера

- *Одноточечный*
  выбирается случайным образом одна координата i_0, и полагается $y'_i = y^1_i$ для $i \leq i_0$ и $y'_i = y^2_i$ для $i > i_0$.
  Недостатки:
  единичные компоненты вектора y^2 с большими номерами имеют мало шансов быть унаследованными в векторе y'
  вектор y' может отличаться от родителей по тем координатам, где $y^1_i = y^2_i$.

Разновидности операторов кроссовера

- *Одноточечный*
  выбирается случайным образом одна координата i_0, и полагается $y'_i = y^1_i$ для $i \leq i_0$ и $y'_i = y^2_i$ для $i > i_0$.
  Недостатки:
  единичные компоненты вектора y^2 с большими номерами имеют мало шансов быть унаследованными в векторе y'
  вектор y' может отличаться от родителей по тем координатам, где $y^1_i = y^2_i$.
- *Связывающих путей*
  Из двух решений y^1 и y^2 выбирается решение с меньшим значением целевой функции и строится последовательность допустимых решений $\overline{y}^1, \overline{y}^2, ..., \overline{y}^k$, таких, что $\overline{y}^1 = y^1$, $\overline{y}^k = y^2$ и расстояние Хэмминга между соседними решениями равно двум. Решение y' выбирается среди элементов этой последовательности.

Разновидности операторов кроссовера

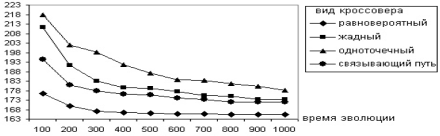
- *Одноточечный*
  выбирается случайным образом одна координата i_0, и полагается $y'_i = y^1_i$ для $i \le i_0$ и $y'_i = y^2_i$ для $i > i_0$.
  Недостатки:
  единичные компоненты вектора y^2 с большими номерами имеют мало шансов быть унаследованными в векторе y'
  вектор y' может отличаться от родителей по тем координатам, где $y^1_i = y^2_i$.
- *Связывающих путей*
  Из двух решений y^1 и y^2 выбирается решение с меньшим значением целевой функции и строится последовательность допустимых решений $\overline{y}^1, \overline{y}^2, \ldots, \overline{y}^k$, таких, что $\overline{y}^1 = y^1$, $\overline{y}^k = y^2$ и расстояние Хэмминга между соседними решениями равно двум. Решение y' выбирается среди элементов этой последовательности.
- Сколько путей для заданных y^1, y^2 можно построить?

Разновидности операторов кроссовера

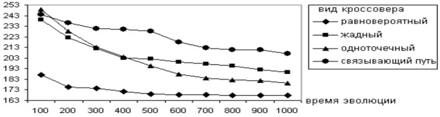
- *Одноточечный*
  выбирается случайным образом одна координата i_0, и полагается $y'_i = y^1_i$ для $i \leq i_0$ и $y'_i = y^2_i$ для $i > i_0$.
  Недостатки:
  единичные компоненты вектора y^2 с большими номерами имеют мало шансов быть унаследованными в векторе y'
  вектор y' может отличаться от родителей по тем координатам, где $y^1_i = y^2_i$.
- *Связывающих путей*
  Из двух решений y^1 и y^2 выбирается решение с меньшим значением целевой функции и строится последовательность допустимых решений $\overline{y}^1, \overline{y}^2, ..., \overline{y}^k$, таких, что $\overline{y}^1 = y^1$, $\overline{y}^k = y^2$ и расстояние Хэмминга между соседними решениями равно двум. Решение y' выбирается среди элементов этой последовательности.
- Сколько путей для заданных y^1, y^2 можно построить?
- Какие пути стоит рассматривать?

Влияние параметров GA на примере задачи о p-медиане с предпочтениями клиентов

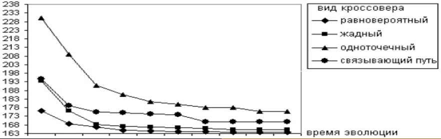
$m = n = 100$, $p = 14$ Среднее значение целевой функции для 10 примеров и 100 испытаний. Размер популяции 20 решений.



а) турнирная селекция, $p_m = 0,02$



б) селекция "лучший и случайный", $p_m = 0,02$



Задача об $(r|p)$-центроиде

Две фирмы, Лидер и Конкурент, последовательно принимают решения о размещении предприятий.
Первым открывает p предприятий Лидер.
Затем, зная это решение, Конкурент открывает r предприятий в свободных местах.

Каждый клиент из $(p+r)$ предприятий выбирает одно, согласно собственным предпочтениям, и приносит доход одной из фирм.
В зависимости от размещения предприятий множество клиентов делится между двумя фирмами.
Каждая фирма стремится максимизировать свою долю рынка.

Необходимо выбрать p предприятий, позволяющие максимизировать суммарный доход Лидера.

Особенности гибридного алгоритма для задачи об $(r|p)$-центроиде

Формирование начальной популяции

- Найти x^L — оптимальное решение задачи Лидера, минимизирующее суммарные расстояния до клиентов, в случае когда на рынке отсутствует Конкурент.

Особенности гибридного алгоритма для задачи об $(r|p)$-центроиде

Формирование начальной популяции

- Найти x^L — оптимальное решение задачи Лидера, минимизирующее суммарные расстояния до клиентов, в случае когда на рынке отсутствует Конкурент.
- Пока популяция не сформирована в нужном объеме повторять:
  выбрать случайным образом пару элементов $(i_1; i_0)$:
  $x^L_{i_1} = 1$, $x^L_{i_0} = 0$,
  построить новое решение $x^{L'}$, в котором $x^{L'}_{i_1} = 0$, $x^{L'}_{i_0} = 1$;
  применить к $x^{L'}$ процедуру локального улучшения;
  добавить полученный локальный оптимум в популяцию.

Особенности гибридного алгоритма для задачи об $(r|p)$-центроиде

Рандомизация окрестности

Пусть $S = \{i | x_i^L = 1\}$, $\overline{S} = \{i | x_i^L = 0\}$

Повторить t раз:

для каждого $i \in S$ найти l ближайших предприятий из множества $\overline{S}$;

выбрать случайным образом $i_1 \in S$,

выбрать случайным образом i_0 из l ближайших к i_1 предприятиям;

решить задачу Конкурента на новом множестве открытых предприятий.

Если значение целевой функции Лидера окажется лучше на новом множестве, то алгоритм продолжает работу с новым решением.

Иначе найти другую пару $(i_1; i_0)$ среди не участвовавших ранее.

Особенности гибридного алгоритма для задачи об $(r|p)$-центроиде

Рандомизация окрестности

Пусть $S = \{i|x_i^L = 1\}$, $\overline{S} = \{i|x_i^L = 0\}$

Повторить t раз:
для каждого $i \in S$ найти l ближайших предприятий из множества $\overline{S}$;
выбрать случайным образом $i_1 \in S$,
выбрать случайным образом i_0 из l ближайших к i_1 предприятиям;
решить задачу Конкурента на новом множестве открытых предприятий.
Если значение целевой функции Лидера окажется лучше на новом множестве, то алгоритм продолжает работу с новым решением.
Иначе найти другую пару $(i_1; i_0)$ среди не участвовавших ранее.

- Какова мощность такой окрестности?

Рекомендации при разработке метаэвристик

- Выбрать подходящую окрестность, чтобы быстро находить соседнее решение

Рекомендации при разработке метаэвристик

- Выбрать подходящую окрестность, чтобы быстро находить соседнее решение
- Изображать графически "посещаемые" решения при фиксированных параметрах алгоритма

Рекомендации при разработке метаэвристик

- Выбрать подходящую окрестность, чтобы быстро находить соседнее решение
- Изображать графически "посещаемые" решения при фиксированных параметрах алгоритма
- Проверять не стал ли метод похож на случайный поиск? Не установлены ли параметры алгоритма на максимум?

Сравнение метаэвристик на примере задачи о съемке кинофильма

Пример	TS	SA	GA
1	2926	2793	2794
2	3191	3191	3308
3	3289	3218	3256
4	3303	3254	3254
5	3299	3184	3223
6	3336	3268	3262
7	3391	3367	3403
8	3223	3213	3294
9	3356	3293	3405
10	3281	3204	3268

Рекомендуемая литература

Е.В. Алексеева, А.В. Орлов Генетический алгоритм для конкурентной задачи о p-медиане // Материалы XIV Байкальской международной школы-семинара "Методы оптимизации и их приложения". Северобайкальск. 2008