Тема 3. ИСЧЕРПЫВАЮЩИЙ ПОИСК

3.5. Эвристические алгоритмы

Методы исследований, основанные на неформальных, интуитивных соображениях, на общем опыте решения родственных задач, называются *эвристическими*. Такие методы позволяют сократить количество исследуемых вариантов при поиске решения задачи и обычно не гарантируют наилучшего решения.

Если экспоненциальный алгоритм поиска оптимального решения, основанный на методе ветвей и границ, не применим для решения реальной задачи из-за недопустимо большого времени вычислений, делается попытка найти не оптимальное решение, а некоторое приближение к нему. Для этого определяется некоторая *эвристика*, ослабляющая требования к критерию оптимальности, что может дать полиномиальный алгоритм, результат которого будет разумно близким к оптимальному. Поэтому такие алгоритмы называются *эвристическими* (*приближенными*).

Типичным эвристическим методом в задачах оптимизации является поиск решений, которые оптимальны локально, а не глобально. Простейшую эвристику можно показать на примере решения задачи коммивояжера, в которой коммивояжер должен посетить n городов точно по одному разу и возвратиться в исходный пункт. Известны стоимости перемещений между всеми парами городов (стоимость перемещения из города i в город j равна C_{ij}). Эвристика заключается в следующем. В качестве начальной точки произвольно выбирается один из городов. Среди всех еще не посещавшихся городов в качестве следующего выбирается ближайший к последнему выбранному городу. Если все города уже посещались, возвращаются в начальный город. Таким образом, условие локальной оптимизации заключается в выборе на каждом шаге города, ближайшего к последнему пройденному. Этот алгоритм, назовем его алгоритмом *ближайшего соседа*, легко реализуется за время $O(n^2)$.

Для иллюстрации работы алгоритма воспользуемся матрицей стоимостей (см. рис. 3.3, *а*).

$i \backslash j$	1	2	3	4	5
1	∞	25	30	5	22
2	12	∞	8	16	19
3	17	24	∞	21	14
4	23	11	20	∞	9
5	27	13	7	26	∞

В качестве начальной точки выберем город 1. Ближайшим к нему является город 4, стоимость перемещения $C_{1,4} = 5$. Затем выбирается город 5 как ближайший к городу 4 ($C_{4,5} = 9$). Следующим будет город 3 ($C_{5,3} = 7$). Добавляем последний не посещавшийся город 2 ($C_{3,2} = 24$) и возвращаемся в исходный город 1 ($C_{2,1} = 12$). В результате получен маршрут 1 – 4 – 5 – 3 – 2 – 1 со стоимостью 57, который не является оптимальным (стоимость оптимального маршрута равна 52), но достаточно близок по стоимости к оптимальному. Очевидно, что выбор другой исходной точки может привести к совершенно другому результату. Например, если в качестве исходного взять город 3, то алгоритм сформирует маршрут 3 – 5 – 2 – 1 – 4 – 3 со стоимостью 64.

Алгоритм ближайшего соседа относится к *жадным* алгоритмам. Такие алгоритмы на каждом шаге выбирают локально оптимальный вариант и являются основой многих эвристических алгоритмов. Во многих случаях они не позволяют получить оптимальное решение (небольшое отклонение от локальной оптимальности на некотором начальном шаге может дать существенную экономию позже). Однако для ряда задач жадные алгоритмы в действительности находят оптимальные решения.

Пусть N_n – порожденный алгоритмом ближайшего соседа маршрут со стоимостью $cost(N_n)$ и O_n – оптимальный маршрут со стоимостью $cost(O_n)$. Если матрица стоимостей симметрична ($C_{ij} = C_{ji}$ для любых i и j) и удовлетворяет неравенству треугольника ($C_{ij} \le C_{ik} + C_{kj}$ для любых i, j и k), то

$$\frac{cost(N_n)}{cost(O_n)} \le \frac{1}{2}\left(\lceil \log n \rceil + 1\right).$$

Ясно, что это не лучшая эвристика для задачи коммивояжера; существуют более сложные и тонкие эвристики, которые позволяют получать результат, более близкий к оптимальному (в среднем и худшем случаях).

$i \backslash j$	1	2	3	4	5
1	∞	25	30	5	22
2	12	∞	8	16	19
3	17	24	∞	21	14
4	23	11	20	∞	9
5	27	13	7	26	∞

$i \backslash j$	1	2	3	4	5
1	∞	25	30	5	22
2	12	∞	8	16	19
3	17	24	∞	21	14
4	∞	11	20	∞	9
5	27	13	7	26	∞

$i \backslash j$	1	2	3	4	5
1	∞	25	30	5	22
2	12	∞	8	16	19
3	17	24	∞	21	14
4	∞	11	20	∞	9
5	∞	13	7	∞	∞

$i \backslash j$	1	2	3	4	5
1	∞	25	30	5	22
2	12	∞	8	16	19
3	∞	24	∞	∞	∞
4	∞	11	20	∞	9
5	∞	13	7	∞	∞

$i \backslash j$	1	2	3	4	5
1	∞	25	30	5	22
2	12	∞	∞	∞	∞
3	∞	24	∞	∞	∞
4	∞	11	20	∞	9
5	∞	13	7	∞	∞

1 – 4 – 5 – 3 – 2 – 1

5 + 9 + 7 + 24 + 12 = 57