ТЕМА 15. КОМБИНАТОРНАЯ ОПТИМИЗАЦИЯ.

Задача заключается в том, что коммивояжер должен посетить каждый город на некоторой территории ровно один раз, а затем вернуться к исходной точке. Учитывая стоимость (расстояния, время поездки) проезда между всеми городами, необходимо так спланировать свой маршрут, чтобы стоимость (длина пути всего тура, время поездки) была минимальна. Поиск маршрута состоит в наборе перестановок из n городов. Оптимальным решением является перестановка, которая дает минимальную стоимость тура. Размерность пространства поиска равна (n-1)! для асимметричной задачи и (n-1)!/2 – для симметричной задачи. Другими словами, размерность тура N определяется множеством точек $v=\{v_1,v_2,...,v_n\}$, где v_i – i-й город, заданный координатами x_i, y_i. Длина пути одного из маршрутов (одной из перестановок) определяется формулой

$$f=\sum_{i=1}^{n-1}\sqrt{(x_i-x_{i+1})^2+(y_i-y_{i+1})^2}+\sqrt{(x_n-x_1)^2+(y_n-y_1)^2}\,. \qquad (3.14)$$

Решением задачи является способ планирования $T=(T[1],T[2],...,T[n],T[1])$, где $T[i]$ является перестановкой на множестве {1, 2,..., N }.

Введем матрицу расстояний между городами v_i и v_j

$$d(i,j)=\sqrt{(x_i-x_j)^2+(y_i-y_j)^2}\,, \qquad (3.15)$$

тогда формула (3.14) примет вид

$$f(T)=\sum_{i=1}^{n-1}d(T[i],T[i+1])+d(T[n],T[1])\,. \qquad (3.16)$$

Задача коммивояжера заключается в минимизации функции $f(T)$ по аргументу T, где $T=(T[1],T[2],...,T[n],T[1])$.

Детерминированные алгоритмы

Задача коммивояжера может быть редуцирована к задаче целочисленного линейного программирования. Пусть города пронумерованы числами от 1 до n. Для каждой пары городов зададим расстояние (или стоимость переезда) c_{ij}, $i,j=1,...,n$. Введем бинарные переменные x_{ij}

$$x_{ij} = \begin{cases} 1, \\ 0, \end{cases}$$

Тогда имеем следующую оптимизационную задачу. Найти переменные x_{ij}, $i, j = 1,...n$, обращающие в минимум функцию

$$f(x) = \sum_{i=1}^{n} \sum_{j=1}^{i-1} c_{ij} x_{ij} \tag{3.18}$$

и удовлетворяющие ограничениям

$$\begin{aligned} &\forall i \in N = \{1,2,...,n)\} \quad \sum_{j \neq i}^{n-1} x_{ij} = 2, \\ &for \ \ each \ \ S \subset N \quad \sum_{i \in S} \sum_{j \notin S} x_{ij} \geq 2. \end{aligned} \tag{3.19}$$

Первое ограничение означает, что каждая вершина должна сообщаться через пару ребер с остальными вершинам, то есть, через входное и выходное ребро. Второе ограничение связано с устранением подмаршрутов, т.е. каждое множество вершин $S \subset N$ является либо пустым, либо содержит все вершины, сочетающееся с остальными вершинам через минимум два ребра.

Для решения задачи можно использовать точные методы, такие как комбинаторный метод ветвей и границ и метод отсекающих плоскостей (метод Гомори). Однако с ростом размерности задачи экспоненциально возрастает требуемый объем памяти и времени счета. Поэтому для задач большой размерности точные методы не используются.

В течение последних десятилетий появилось несколько приближенных алгоритмов для аппроксимации оптимального решения: метод ближайшего соседа, жадный алгоритм, метод включения ближайшего города, метод имитации отжига и др.

Задача коммивояжера относится к категории NP-полных задач, т.е. задач решаемых методом полного перебора всех вариантов.

Формализация задачи

Ген – это число, характеризующее номер посещаемого города.

Генотип – строка из чисел длиной N, описывающая порядок посещения городов.

Особь – конкретная строка из чисел (допустимый вариант решения задачи).

Представление тура

Существуют различные представления тура: бинарное представление, представление путей и представление смежности. В данной работе будет рассмотрено представление путей. Представление путей представляет собой упорядоченный список городов, которые необходимо посетить, причем если город i является j-ым элементом списка, то город i необходимо посетить j-м по счету. Например, тур 3-2-4-1-7-5-8-6 представляется просто как (3 2 4 1 7 5 8 6)

Генерация начальной популяции

Начальная популяция обусловливает скорость и сходимость алгоритма. Для этого можно применить несколько методов для генерации начальной совокупности:

- Случайная генерация начальной популяции.
- Генерация первой особи случайным образом, затем она будет мутирована N-1 раз с помощью оператора мутации.
- Генерация первой особи с использованием эвристического механизма. Преемник первого города расположен на расстоянии, меньшем по сравнению с другими. Затем мы используем оператор мутации на маршруте, полученном для того, чтобы сгенерировать (N-2) других индивидуумов, которые будут составлять начальную популяцию.

Операторы скрещивания

Одними из удачных операторов кроссинговера для решения ЗК признаны [3, 4] Partially-Mapped Crossover (PMX) и Order Crossover (OX), или, как их ещё называют, частично соответствующий кроссинговер и упорядоченный кроссинговер соответственно. Операторы PMX и OX представляет собой обмен частей туров двух родительских особей. Для начала случайно выбираются две секущие точки для выделения частей туров из родительских особей (так называемый двухточечный кроссинговер). При таком обмене часто могут возникать частичные туры, например, тур $T = (5\ 7\ 1\ 2\ 5\ 4\ 3\ 6)$ содержит частичный тур: коммивояжер, выйдя из города 5, вернется в город 5, не посетив города 4, 3 и 6. Для того, чтобы не возникало частичных туров, в этих операторах существуют механизмы создания правильных туров. Алгоритм создания потомков с помощью оператора PMX представлен ниже и проиллюстрирован на рисунках 3.1 и 3.2.

Шаг 1. Случайно выбрать две секущие точки для выделения части тура из родителя 1.

Шаг 2. Скопировать все города из родителя 2 в потомка.

Шаг 3. Для каждого города C из передаваемой части тура родителя 1:

Шаг 3.1. Найти город C в потомке. На его место поставить тот город, который заменяется городом C.

Шаг 3.2. Копировать город C из родителя 1 в потомка на ту же позицию.

Шаг 4. Для создания второго потомка поменять местами родителей и перейти к шагу 1.

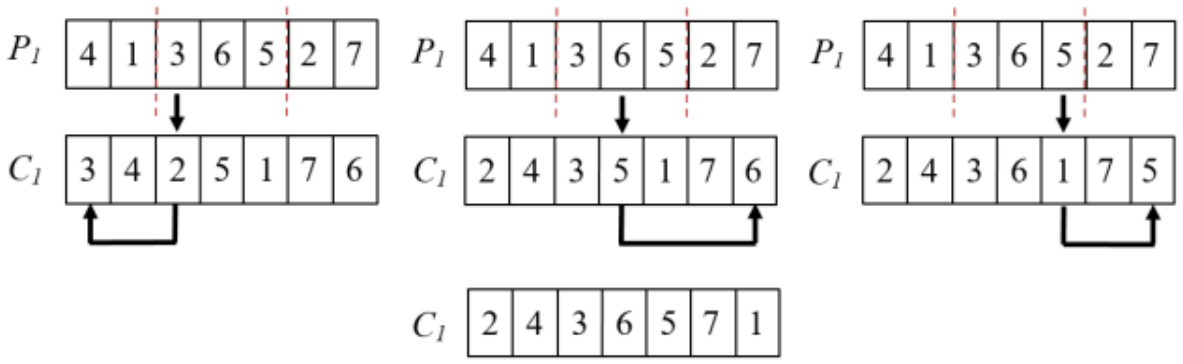

Рис. 3.1. Формирование первого потомка с помощью PMX

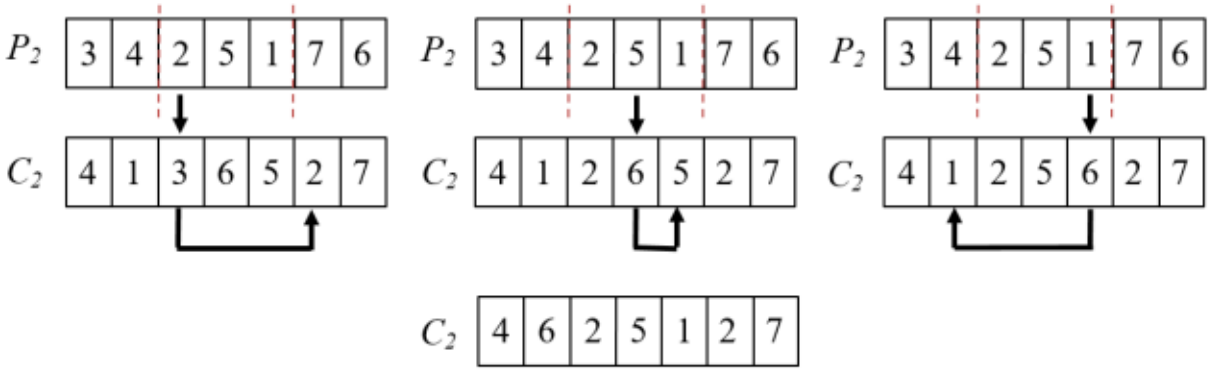

Рис. 3.2. Формирование второго потомка с помощью PMX

Особенность OX состоит в том, что он сохраняет порядок следования городов, а не их расположение в туре. Он строит потомка с использованием части тура одного из родителей и сохраняет порядок следования городов, представленный в другом родителе. Алгоритм создания потомков с помощью OX представлен ниже и проиллюстрирован на рисунке 3.3.

Шаг 1. Выбрать две секущие точки для выделения части тура из родителя 1.

Шаг 2: Выделить часть тура из родителя 1 и передать потомку. Все города из переданной части тура отметить как использованные в родителе 2.

Шаг 3: Начиная с правой стороны от переданной части тура, помещать неиспользованные города родителя 2 в потомок.

Шаг 4: Для создания второго потомка поменять местами родителей и перейти к шагу 1.

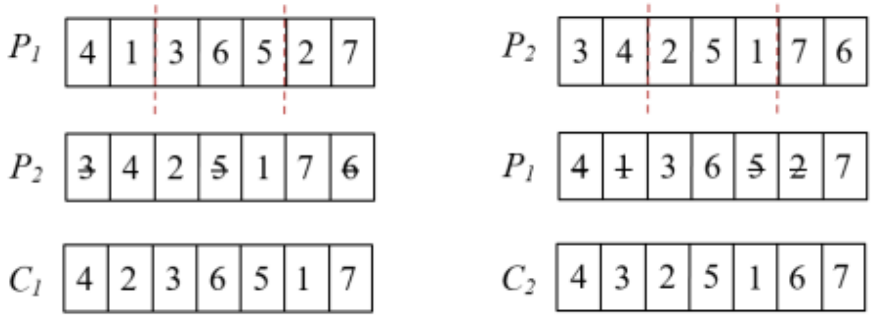

Рис.. 3.3. Формирование потомков с помощью OX

Представленные выше операторы кроссинговера не используют информацию о расстояниях между городами. Эвристический кроссинговер – пример создания потомков с использованием информации о длине пути

между городами. Алгоритм создания потомков с помощью эвристического кроссинговера представлен ниже.

Шаг 1. Выбрать начальным городом тура потомка любой из начальных городов туров родительских особей.

Шаг 2. Взять следующий город из родительских особей, оценить расстояние до них и передать потомку тот город, расстояние до которого будет меньше

Шаг 3. Если выбранный город образует частичный тур, то проверить город другого родителя, а если и он не создает правильный тур, то выбрать случайный город, не образующий цикл.

Шаг 4. Повторять шаги 2 и 3 пока все города не войдут в тур потомка.

Операторы мутации в решении задачи коммивояжера

Простая мутация перестановки (SIM) выбирает случайно два города в списке городов и переставляет их, т.е. меняет местами. Оператор перестановки представлен на рисунке 4.

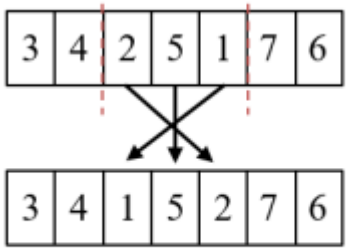

Рис. 3.4. Оператор мутации SIM

При мутации перестановки относительно центра (CIM) хромосома случайным образом разбивается на две части. В каждой части хромосомы гены перестанавливаются в обратном порядке. Оператор представлен на рисунке 3.5.

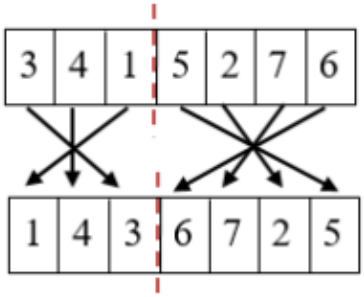

Рис. 3.5. Оператор мутации CIM

Мутация вставки (IM) случайно выбирает ген в хромосоме и переставляет его в другое случайно выбранное место. Оператор представлен на рисунке 3.6.

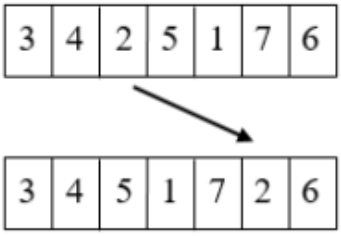

Рис. 3.6. Оператор мутации IM

Мутация перестановки (SM) случайно выбирает два гена и меняет их местами. Оператор представлен на рисунке 3.7.

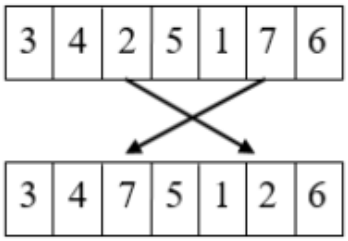

Рис. 3.7. Оператор мутации SM