5. Jan ˇSochman, Ji˘r´ıMatas, «AdaBoost», Center for Machine Perception, Czech Technical University, Prague, 2010

М.Р. Рахмонова, Д.Т. Мухамедиева

РЕШЕНИЕ ЗАДАЧИ ВЫБОРА МАРШРУТА С ИСПОЛЬЗОВАНИЕМ ГЕНЕТИЧЕСКОГО АЛГОРИТМА

(Ташкентский университет информационных технологий)

Задача о выборе маршрута относится к классу NP-трудных задач. Все эффективные (сокращающие полный перебор) методы решения задачи выборе маршрута — методы эвристические («жадные алгоритмы»). В большинстве эвристических методов находится не самый эффективный маршрут, а приближённое решение. Зачастую востребованы так называемые any-time алгоритмы, то есть постепенно улучшающие некоторое текущее приближенное решение.

Задача выборе маршрута может быть сформулирована как целочисленная введением булевых переменных $x_{ij} = 1$, если маршрут включает переезд из города i непосредственно в город j и $x_{ij} = 0$ в противном случае. Тогда можно задать математическую модель задачи, то есть записать целевую функцию и систему ограничений

$$Z = \sum_{i=1}^{n} \sum_{j=1}^{n} x_{ij} c_{ij} \to \min \quad (1)$$

$$\sum_{i=1}^{n} x_{ij} = 1 \qquad j = \overline{1,n} \quad (2)$$

$$\sum_{j=1}^{n} x_{ij} = 1 \qquad i = \overline{1,n} \quad (3)$$

$$x_{ij} \geq 0 \qquad i,j = \overline{1,n} \quad (4)$$

Условия (2)–(4) в совокупности обеспечивают, что каждая переменная x_{ij} равна или нулю, или единице. При этом ограничения (2), (3) выражают условия, что коммивояжер побывает в каждом городе один раз в него прибыв (ограничение (2)), и один раз из него выехав (ограничение (3)).

Однако этих ограничений не достаточно для постановки задачи коммивояжера, так как они не исключают решения, где вместо простого цикла, проходящего через n вершин, отыскиваются 2 и более отдельных цикла (подцикла), проходящего через меньшее число вершин. Поэтому задача, описанная уравнениями (2)–(4) должна быть дополнена ограничениями, обеспечивающими связность искомого цикла.

Для того, чтобы исключить при постановке задачи все возможные подциклы в систему ограничений задачи включают следующее ограничение:

$U_i - U_j + n \cdot X_{ij} \leq n-1$, где $U_i \equiv \sum_{j=1}^{n} x_{ij}, U_j = \sum_{i=1}^{n} x_{ij}, i = \overline{2,n}, j = \overline{2,n}$ и $i \neq j$.

Методы решения задачи о выборе маршрута различны.

К эвристическим методам решения задачи выборе маршрута следует отнести «жадный» алгоритм, на каждом шаге выбирающий ребро наименьшей стоимости из множества рёбер, не нарушающих корректности решения. Эти методы имеют большую погрешность. Хорошо исследована область генетических алгоритмов, показавших свою эффективность для данной задачи, но они довольно громоздки. Метод перебора прост, но только лишь при небольшом количестве итераций.

Решить задачу выборе маршрута можно с помощью алгоритма Крускала. Также нам могут помочь алгоритмы Борувки и Прима, так называемые «жадные алгоритмы» Эти методы ускоряют разработку алгоритма по сравнению с методом полного перебора, однако не всегда дают оптимальное решение. Дадим немного понятия о них.

Итак, имеется n городов, которые нужно обойти. Для этого достаточно проложить $(n-1)$ путей между городами. Как соединить города так, чтобы суммарная стоимость путешествия была минимальна?

В общем случае, задачу можно сформулировать так. Пусть дан связный, неориентированный граф с весами на ребрах $G(V,E)$, в котором V-множество вершин (городов), а E-множество их возможных попарных соединений (дороги). Пусть для каждого ребра (u,v) однозначно определено некоторое вещественное число $w(u,v)$-его вес (длина или стоимость пути). $w(u,v)$ называется *весовой функцией*. Задача состоит в нахождении такого связного ациклического подграфа $T \subset G$, содержащего все вершины, что суммарный вес его ребер будет минимален.

В алгоритме Борувки на первом шаге A состоит из всех вершин G и пустого множества ребер. В начале очередной фазы алгоритма Борувки, для каждой компоненты связности промежуточного остовного леса выбирается *лидер* или *корень* – вершина, сопоставляемая каждой компоненте. Сделать это можно в простейшем случае с помощью обхода A в глубину: вершина, с которой начинается обход очередной компоненты, и будет ее лидером.

После того, как лидеры выбраны, для каждой компоненты связности находится безопасное для нее ребро, по существу методом грубой силы. Как только все такие ребра отобраны, они добавляются к A. Процесс продолжается до тех пор, пока в A присутствует больше одной компоненты связности.

Алгоритм Крускала объединяет вершины графа в несколько связных компонент, каждая из которых является деревом. На каждом шаге из всех ребер, соединяющих вершины из различных компонент связности, выбирается ребро с наименьшим весом. Необходимо проверить, что оно является безопасным. Безопасность ребра гарантируется ранее показанной теоремой о безопасных ребрах. Так как ребро является самым легким из всех ребер, выходящих из данной компоненты, оно будет безопасным.

Остается понять, как реализовать выбор безопасного ребра на каждом шаге. Для этого в алгоритме Крускала все ребра графа G перебираются по возрастанию веса. Для очередного ребра проверяется, не лежат ли концы ребра в разных компонентах связности, и если это так, ребро добавляется, и компоненты объединяются.

Удобно использовать для хранения компонент структуры данных для непересекающихся множеств, как, например, списки или, что лучше, лес непересекающихся множеств со сжатием путей и объединением по рангу (один из самых быстрых известных методов). Элементами множеств являются вершины графа, каждое множество содержит вершины одной связной компоненты.

Указанные алгоритмы легко выполняются при малом количестве вершин в графе. При увеличении их количества задача поиска кратчайшего пути усложняется. Здесь на помощь приходит генетический алгоритм.

Генетический алгоритм — это алгоритм, который позволяет найти удовлетворительное решение к аналитически неразрешимым проблемам через последовательный подбор и комбинирование искомых параметров с использованием механизмов, напоминающих биологическую эволюцию. Является разновидностью эволюционных вычислений. Отличительной особенностью генетического алгоритма является акцент на использование оператора «кроссовера», который производит операцию, роль которой аналогична роли скрещивания в живой природе.

Задача кодируется таким образом, чтобы её решение могло быть представлено в виде вектора («хромосома»). Случайным образом создаётся некоторое количество начальных векторов («начальная популяция»). Они оцениваются с использованием «функции приспособленности», в результате чего каждому вектору присваивается определённое значение («приспособленность»), которое определяет вероятность выживания организма, представленного данным вектором. После этого с использованием полученных значений приспособленности выбираются вектора (селекция), допущенные к «скрещиванию». К этим векторам применяются «генетические операторы» (в большинстве случаев «скрещивание» - crossover и «мутация» - mutation), создавая таким образом следующее «поколение». Хромосомы следующего поколения также оцениваются, затем производится селекция, применяются генетические операторы и т.д. Так моделируется «эволюционный процесс», продолжающийся несколько жизненных циклов (поколений), пока не будет выполнен критерий останова алгоритма. Таким критерием может быть: нахождение глобального, либо субоптимального решения; исчерпание числа поколений, отпущенных на эволюцию; исчерпание времени, отпущенного на эволюцию. Генетические алгоритмы служат, главным образом, для поиска решений в очень больших, сложных пространствах поиска.

Таким образом, можно выделить следующие этапы генетического алгоритма:

- создание начальной популяции;
- вычисление функций полезности для особей популяции (оценивание);

- выбор индивидов из текущей популяции (селекция);
- скрещивание и/или мутация;
- вычисление функций полезности для всех особей;
- формирование нового поколения;
- если условия совпали, то решение найдено (конец цикла), если нет, то цикл повторяется.

При разработке генетических процедур основное влияние уделялось разработке с учетом знаний о предметной области методов кодирования решений, модификации генетических операторов и организации эволюционного процесса.

Универсальность генетического алгоритма заключается в том, что от конкретной задачи зависят только такие параметры, как функция приспособленности и кодирование решений. Остальные шаги для всех задач производятся одинаково.

Генетические алгоритмы оперируют совокупностью особей (популяцией), которые представляют собой строки, кодирующие одно из решений задачи. Этим генетический алгоритм отличается от большинства других алгоритмов оптимизации, которые оперируют лишь с одним решением, улучшая его.

Литература

1. Олифер В., Олифер Н. Искусство оптимизации трафика. "Журнал сетевых решений LAN". 2001. №12.
2. Барсегян А.А., Куприянов М.С., Степаненко В.В., Холод И.И. Технологии анализа данных: Data Mining, Visual Mining, Text Mining, OLAP. Санкт-Петербург, Изд-во БХВ-Петербург, 2007

А.С. Семёнова

МОДУЛЬ ОЦЕНКИ РЕЦЕНЗЕНТОМ РАБОТ ШКОЛЬНИКОВ

(Самарский государственный техничекий университет)

В настоящее время становится актуальным проблема поддержки одаренной и творческой молодежи. В Самарской области был создан Координационный совет по работе с одаренной молодежью в сфере науки и техники при Администрации Губернатора Самарской области[1]. В рамках работы Координационного совета была сформирована единая Самарская областная система мер по выявлению и развитию творчески одаренной молодежи в сфере науки, техники и технологий и инновационному развитию Самарской области - система «ВЗЛЕТ».

В рамках этой системы, школьники разных классов из разных уголков Самарской области выполняют индивидуальные проекты научной направленности. Эти проекты школьник выполняет совместно с учителем из школы. Для