Эволюционный алгоритм для задачи 2-Correlation-Clustering

Моршинин Александр
Институт математики им. С.Л. Соболева СО РАН
Исследование поддержано грантом РНФ № 22-71-10015

Постановка задачи

В задаче кластеризации необходимо разбить заданное множество объектов на несколько подмножеств (*кластеров*) на основе сходства объектов друг с другом.

В задаче k-Correlation-Clustering (k-CC) необходимо «нарезать» граф на не более чем k компонент связности (*кластеров*), каждая из которых является *полным графом*, удаляя и добавляя ребра. При этом нужно минимизировать количество операций добавления и удаления.

Известно, что данная задача NP-трудна.

Приближенные алгоритмы для 2-CC

- 3-приближенный алгоритм **Bansal-Blum-Chawla (BBC)**. Для каждой вершины первый кластер – вершина и ее *окрестность*, второй кластер – все остальное. Выбирает наилучшее. В n-вершинном графе строит n решений.
- 2-приближенный алгоритм **Coleman-Saunderson-Wirth (CSW)**. Для каждого решения алгоритма **BBC** применяется локальный поиск. Выбирает наилучший локальный оптимум. В n-вершинном графе строит n решений.

Эволюционный алгоритм

Идея эволюционного алгоритма – известные приближенные алгоритмы строят начальную популяцию, затем применяются методы эволюционных вычислений.

Функция приспособленности = целевая функция.

Кодирование генотипов

Способ кодирования особей для задачи 2-СС это бинарный вектор длины n. 0 означает, что вершина попадает в первый кластер, 1 – вершина попадает во второй кластер. Можно считать, что вершина с номером 1 всегда лежит в первом кластере. Тогда достаточно бинарного вектора длины n - 1.

Селекция, мутация и скрещивание

- Выбрана турнирная селекция с размером турнира равным 2
- При скрещивании создается копия каждой из родительских особей – потомки. В каждой позиции генотипа с вероятностью 0.5 гены остаются на месте, с вероятностью 0.5 обмениваются
- При мутации каждый ген с вероятностью 0.001 мутирует (поточечная мутация)

Итерация алгоритма

На каждой итерации формируется *n* новых особей. Т.е. пока в выборке нет *n* особей, делаем следующее:

0. Добавляем в популяцию элитную особь (лучшую на предыдущей итерации алгоритма).

1. Выбираем 2 особи с помощью оператора селекции

2. Скрещиваем и мутируем эти особи

3. Добавляем особи в популяцию только если их в ней еще нет

Повторяем шаги 0-3 заданное количество раз (1000).

Комментарии к итерации алгоритма

Без элитной особи алгоритм расходится. Т.е. наилучшее решение на последней итерации сильно хуже наилучшего решения в начальной популяции.

Если не отсеивать дубли, то алгоритм стремится заполнить всю популяцию одной наиболее приспособленной особью, что не позволяет алгоритму находить новые решения.

Разведывательный анализ для задачи 2-CC

Алгоритм был реализован на языке программирования C++.

Было взято 100 графов по 200 вершин в каждом. Плотность графа – 0.33.

Применение алгоритма к популяции, построенной алгоритмом **BBC**, позволяет улучшить значение целевой функции в среднем на 11.5%.

Применение алгоритма к популяции, построенной алгоритмом **CSW**, позволяет улучшить значение целевой функции в среднем всего лишь на 0.0094

Промежуточные выводы

- Построенный алгоритм не может эффективно выскочить из множества локальных оптимумов задачи 2-CC. Однако он достаточно неплохо улучшает начальное решение, если начальная популяция не состоит из локальных оптимумов
- Можно сначала строить приближенные решения алгоритмом **BBC**, применять к ним GA, а потом применять к ним локальный поиск
- Попробовать алгоритм для других значений k