Нижегородский государственный университет им. Н.И. Лобачевского

Факультет вычислительной математики и кибернетики

Кафедра информатики и автоматизации
научных исследований

Методические указания

«Улучшающий генетический алгоритм»

по курсу «Генетические алгоритмы:
теория и приложения решения оптимизационных задач»
специальность «Прикладная информатика»

Н.Новгород, 2008

УДК. 681.3.015

Методические указания «Улучшающий генетический алгоритм» по курсу «Генетические алгоритмы: теория и приложения решения оптимизационных задач» для студентов факультета ВМК специальности «Прикладная информатика».

/Нижегородский государственный университет, 2008, 10 с.

В методических указаниях излагается один из способов построения улучшающих алгоритмов с применением методов эволюционно-генетического поиска. В качестве примера приводится процедура построения алгоритма, осуществляющего оптимизацию k-разбиения графа.

Методическое пособие подготовлено соискателем А.В.Филимоновым.

Рецензент д.т.н., проф. Д.И.Батищев. (М.Х. Прилуцкий?)

Содержание

Введение

Класс задач, для которых не существует точных решающих алгоритмов с полиномиальной оценкой вычислительной сложности, очень обширен. Для решения таких задач, как правило, вводится понятие приемлемости решения и строятся алгоритмы, находящие решения, не являющиеся точными, но приемлемые с точки зрения исследователя. Часто такого рода методы носят характер улучшающих алгоритмов и сценарий получения решения следующий: строится некоторое начальное решение, затем это решение улучшается до тех пор, пока исследователь не сочтет решение приемлемым, или алгоритм не исчерпает свой ресурс по улучшению решения. Основная проблема такого рода эвристик, особенно основанных на жадном принципе, - трудности преодоления локальных экстремумов. Предлагается для преодоления такого рода трудностей и расширения области пространства поиска, которая потенциально может быть исследована алгоритмом, интегрировать в улучшающий алгоритм элементы эволюционно-генетического поиска.

Классический генетический алгоритм (ГА) подразумевает манипулирование так называемыми кодировками (генотипами). Каждая кодировка представляет собой некоторое представление решения задачи в форме, удобной для применения различных операторов ГА. Каждая кодировка имеет оценку, называемую приспособленностью, которая характеризует качество решения, соответствующего данной кодировке. В результате процесса, эмулирующего эволюцию набора кодировок, определяется «кодировка-победитель», обладающая наилучшей приспособленностью, которой соответствует решение задачи с наивысшим качеством.

Поскольку классический ГА – это «решающий» алгоритм, а цель – построить улучшающий, то, очевидно, необходимо изменить состав информации закодированный в генотипе. Предлагается кодировать не само решение, а некий сценарий получения нового решения из существующего.

Принцип построения улучшающего ГА

Рассмотрим самый слабый с точки зрения способности преодолевать локальные экстремумы класс алгоритмов – жадные улучшающие алгоритмы. Жадный улучшающий алгоритм на каждом шаге выбирает то направление улучшения решения, которое подразумевает максимальное его улучшение, т.е. выбирает одно из нескольких возможных направлений движения по пространству поиска (ПП), оставляя остальные направления неисследованными. Предлагается внедрить генетический алгоритм в качестве управляющей схемы, принимающей решение в каком направлении двигаться по ПП на каждом шаге, что позволит существенно расширить исследуемую область ПП.

Для реализации предложенного подхода модифицируем жадный алгоритм таким образом, чтобы решение о том, какой шаг сделать в ПП принимал не он, а ГА в соответствии с маршрутом ПП, закодированном в генотипе. Фактически, модифицированный алгоритм формирует набор направлений, в которых можно двигаться по ПП с целью улучшения решения, а ГА принимает решение в каком *именно* направлении сделать шаг.

На рис.1 схематично изображен процесс движения по ОП; точки обозначают решения, через которые проходит маршрут, стрелки – возможные направления движения на каждом шаге; на каждом шаге выбирается одно из направлений определяемое соответствующим геном.

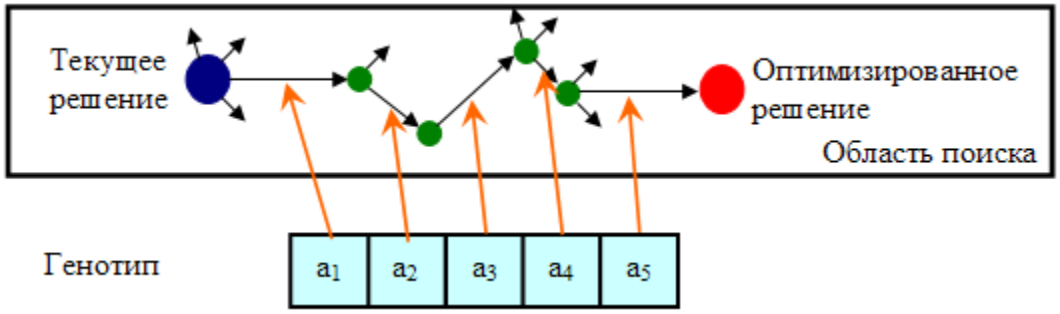

Рис 1. Маршрут поиска закодирован в генотипе

После того, как проделаны все шаги, закодированные в генотипе, для дальнейшего улучшения к получившемуся решению может быть применен немодифицированный жадный алгоритм. Т.е. маршрут в ПП можно поделить на две части – управляемая, т.е. прокладываемая под действием генотипа, и неуправляемая, проходимая жадным алгоритмом (Рис.2).

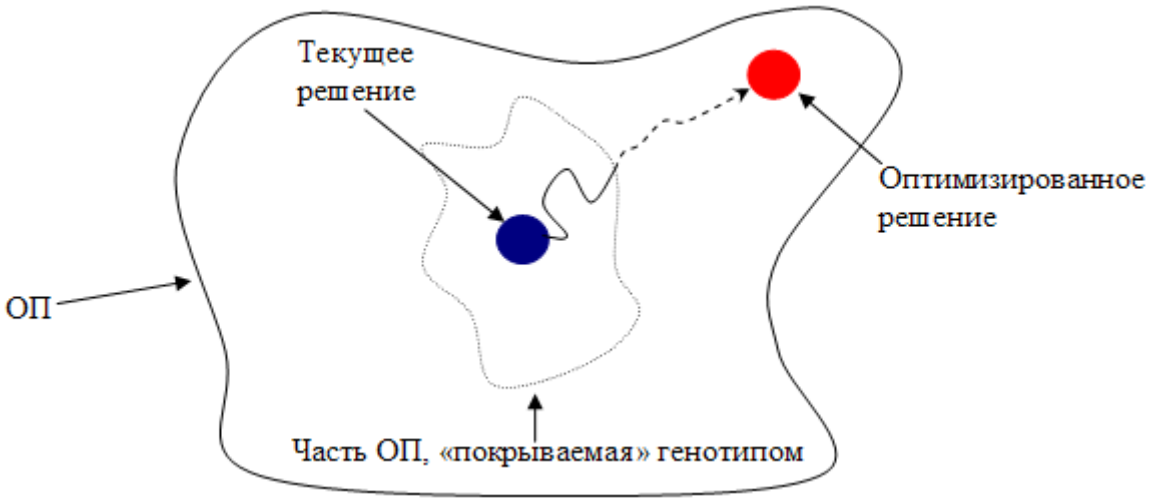

Рис 2. Маршрут в ОП, прокладываемый декодером

По-сути, механизмом улучшения решения по прежнему является жадный алгоритм, управляемая часть маршрута предназначена лишь для того, чтобы вывести этот жадный алгоритм из области локального экстремума. Такой синтез ГА и жадных техник позволяет построить достаточно быстрый улучшающий алгоритм, лишенный недостатка жадных алгоритмов быстро скатываться в локальный экстремум.

Использование ГА подразумевает выбор способа кодирования маршрута (символьной модели). Эта проблема специфична для каждой задачи и для каждого модифицируемого алгоритма, но в некоторых случаях маршрут ПП можно закодировать k-ичной строкой длины p. В этом случае p – это длина управляемой части маршрута. Эта константа регулирует часть ПП, которая исследуется ГА и оказывает непосредственное влияние на скорость работы улучшающего алгоритма. Иными словами, длина генотипа – это средство контроля баланса между качеством находимого решения и скоростью работы

алгоритма. Константа k может быть определена и интерпретирована несколькими способами. Если проблема и решающий алгоритм позволяют оценить максимально возможное количество направлений улучшения решения на каждом шаге, то выбор k очевиден. В других случаях может быть полезен такой способ: константа k выбирается случайным образом. Если на очередном шаге количество возможных направлений поиска больше k, тогда единственное значение гена соответствует нескольким направлениям, и выбор между направлениями, определенными значением гена, осуществляется случайным образом. Если же направлений поиска меньше k, тогда каждое направление характеризуется несколькими значениями гена и конкретное направление выбирается однозначно. При использованиии данного приема, следует понимать, что значение k должно быть достаточно большим, в противном случае элементы случайного поиска будут играть слишком большую роль в работе ГА и снижать эффективность улучшающего алгоритма.

Стоит отметить, что описанный выше способ кодирования делает возможным применение классических операторов ГА для работы с k-ичными строками, что позволяет сделать структуру ГА простой и прозрачной. Еще одно преимущество этих операторов – скорость работы; она достаточно высока, поскольку над генотипом не производится никаких сложных вычислений. Единственная ресурсоемкая операция в составе предложенного ГА – оценка решения, кодируемого генотипом, поскольку для ее вычисления требуется процедура декодирования, которая подразумевает прохождения всего маршрута улучшения решения.

Выше описана весьма общая концепция построения улучшающего ГА, которая была опробована при проектировании улучшающего ГА для задачи k-разбиения графа. Модификации подвергался жадный алгоритм перемещения вершин, принадлежащих ребрам, между подграфами разбиения. Суть алгоритма в том, что он пытается перенести вершины, принадлежащие ребру, в один подграф разбиения, не нарушая ограничений задачи, так, чтобы максимально уменьшить вес сечения. Для построения ГА была выбрана символьная модель, описанная выше. Выбор констант k и p для этого случая очевиден: k – количество подграфов разбиения, p – количество ребер; каждый ген кодирует перемещение вершин какого-либо ребра в тот или иной подграф разбиения.

Алгоритм оптимизации k-разбиения графа

Постановка задачи разбиения

Проблема k-разбиения взвешенного графа G(V,E) состоит в распределении множества вершин V по k подмножествам $V_1,\dots V_k$, таким образом, чтобы удовлетворялся набор ограничений, называемых декомпозиционными ограничениями, а оптимизируемая функция (функция цели, критерий оптимальности, функционал), определенная над разбиением, принимала экстремальное значение. Распределение предполагает, что множества $V_1,\dots V_k$ попарно не пересекаются и при объединении составляют исходное множество V:

$$V_i \cap V_j = \varnothing, \quad i, j = \overline{1,k}, \quad i \neq j,$$
$$\bigcup_{i=1}^{k} V_i = V. \tag{1}$$

Подмножества разбиения определяют подграфы разбиения, т.е. каждый подграф разбиения $G_i(V_i, E_i)$ определяется множеством вершин V_i и множеством ребер $E_i = \{e = \{v_l, v_q\} \mid v_l \in V_i, v_q \in V_i\}$. Значение k фиксировано и задается как параметр задачи.

Определим декомпозиционные ограничения так, чтобы вес каждого подграфа разбиения $w(V_i) = \sum_{v_q \in V_i} w(v_q)$, где $w(v_q)$ – вес вершины v_q, лежал в следующих пределах:

$$L_i \le w(V_i) \le U_i, i = \overline{1,k}, \quad (2)$$

L_i и U_i – минимальный и максимальный вершинные веса подграфов разбиения.

В ограничениях (2) константы L_i и U_i определяют пределы, в которых могут варьироваться веса подграфов разбиения.

Критерии оптимальности, как и декомпозиционные ограничения, определяются спецификой конкретной задачи. Один из наиболее часто используемых критериев – минимизация веса сечения. Вес сечения определяется суммой весов ребер, которые целиком не принадлежат ни одному подграфу разбиения:

$$W_c = \sum_{e_c \in E_c} w(e_c) \to \min, \quad где \quad w(e_c) - вес \quad ребра \quad e_c$$
$$E_c = \left\{ e = \{v_i, v_j\} \in E \mid v_i \in V_a, v_j \in V_b, a \ne b, \quad a, b = \overline{1,k} \right\} \quad (3)$$

Задачу (1)-(3) будем называть задачей k-разбиения графа. Возможны различные расширения проблемы (1)-(3). Например, задача может иметь несколько функций цели (многокритериальный случай), тогда решение состоит в нахождении Парето-области решений.

Жадный улучшающий алгоритм

Пусть имеется какое-либо решение вышеописанной задачи k-декомпозиции графа. Задача предполагает достаточно простой жадный улучшающий алгоритм. Суть алгоритма заключается в вычислении максимального выигрыша, который может принести перенос какой-либо вершины в тот или иной подграф разбиения и осуществлении этого переноса.

Схема работы алгоритма такова:

1) Ребра графа сортируются по неубыванию веса. Пусть $E=(e_1,...e_m)$ – вектор отсортированных ребер.
2) i:=0; f:=0
3) Из k подграфов разбиения выбирается тот, при перемещении в который вершин ребра e_i не нарушаются ограничения задачи декомпозиции, и достигается наибольшее уменьшение веса сечения.
4) Если на шаге 3 перемещение состоялось, то f:=1.
5) i:=i+1
6) Если i ≤m, то переход на шаг 3.

7) Если f=1, то переход на шаг 2.

Нетрудно заметить, что на шаге 3 алгоритма обе вершины ребра e_i оказываются в одном подграфе разбиения, тем самым ребро перестает участвовать в сечении, если до этого участвовало. Назовем этот процесс локализацией ребра. Таким образом, алгоритм локализует сначала самые «тяжелые» ребра, а затем, если это возможно, ребра меньшего веса. Эвристика работает до тех пор, пока возможна локализация хотя бы одного ребра с уменьшением веса сечения[1], т.е. до достижения некоторого локального экстремума.

Генетический улучшающий алгоритм

Для того чтобы значительно расширить исследуемое пространство области поиска задачи предлагается сконструировать гибридный алгоритм улучшения решения, основанный на предложенной жадной методике и эксплуатирующий эволюционно-генетический подход.

Описанный выше жадный алгоритм улучшения решения на каждом шаге принимает решение, руководствуясь наибольшим сокращением веса сечения, т.е. выбирает одно из нескольких возможных направлений движения по ОП, оставляя остальные направления неисследованными. Внедрение генетического алгоритма в качестве управляющей схемы, принимающей решение, в каком направлении двигаться по ОП на каждом шаге, позволяет существенно расширить исследуемую область ОП.

Для реализации предложенного подхода модифицируем описанный выше жадный алгоритм переноса вершин таким образом, чтобы решение о том, какой шаг сделать в ОП принимал не он, а ГА в соответствии с маршрутом ОП, закодированном в генотипе. Фактически, модифицированный жадный алгоритм формирует набор направлений, в которых можно двигаться по ОП с целью улучшения решения, а ГА принимает решение в каком именно направлении сделать шаг. Такой синтез ГА и жадных техник позволяет построить достаточно быстрый улучшающий алгоритм, лишенный недостатка жадных алгоритмов скатываться в локальный экстремум.

Для того чтобы закодировать маршрут в генотипе необходимо выбрать символьную модель (кодировку), способы оценки кодировки и декодирования кодировки в решение задачи.

Символьная модель представляет собой k-ичную строку длины m $A=(a_1,...,a_m)$, где k – количество компонент разбиения, $m=|E|$ - количество ребер графа. Ген a_i говорит о том, что обе вершины, входящие в состав ребра e_i будут перемещены в a_i-ую компоненту разбиения.

Оценкой кодировки является вес сечения соответствующего решения задачи. Необходимо отметить, что в данном случае считается, что кодировка обладает лучшей приспособленностью, если ей соответствует меньшая оценка.

---

[1] Локализация не каждого ребра способна привести к уменьшению веса сечения, ведь локализация ребра потенциально способна разлокализовать ребра, инцидентные локализуемому.

Алгоритм декодирования для такой кодировки может выглядеть следующим образом:

1) i:=1

2) Если возможно перенести вершины ребра e_i в a_i-ую компоненту разбиения, не нарушая ограничений задачи, то осуществляем перенос.

3) i:=i+1

4) Если i<=m, то переход на шаг 2.

Такая схема декодирования позволит ГА исследовать некоторый район ОП расположенный вокруг исходного решения. Однако, эффективность работы ГА можно повысить, модифицируя декодер. Очевидно, что в общем случае решение, полученное описанным выше декодером можно улучшить, применяя, например, все тот же жадный алгоритм переноса ребер. Таким образом, схема модифицированного декодера такова:

1) i:=1

2) Если возможно перенести вершины ребра e_i в a_i-ую компоненту разбиения, не нарушая ограничений задачи, то осуществляем перенос.

3) i:=i+1

4) Если i<=m, то переход на шаг 2.

5) К полученному решению применяем жадный алгоритм переноса ребер.

Таким образом, декодер представляет собой управляемый улучшающий алгоритм, начальную часть пути в ОП прокладывает в соответствии с кодировкой, а оставшуюся часть – руководствуясь жадным принципом переноса ребер. На рис.2 изображена область поиска задачи и маршрут, проложенный описанным декодером. Часть маршрута, соответствующая управляемой работе декодера, т.е. работе шагов 1-4 алгоритма, показана сплошной линией, а часть маршрута, соответствующая работе жадного алгоритма – пунктирной.

Очевидно, управляемая часть декодера позволяет оценивать лишь решения в некоторой ограниченной подобласти ОП (показана на рис.2 прерывистой линией), в то время как остальная часть потенциально может быть исследована жадным алгоритмом. Однако, этот фактор не критичен, т.к. предложенный ГА не претендует на роль решающего алгоритма, но преследует другую цель: предотвратить схождение жадного алгоритма в локальный экстремум и тем самым увеличить исследуемую область решений. Фактически предложенный ГА можно рассматривать как некий механизм, «выталкивающий» жадный алгоритм из локального оптимума и предоставляющий возможность тому же самому жадному алгоритму отыскать оптимум в другой области ОП.

Описанный выше способ кодирования делает возможным применение классических операторов ГА для работы с k-ичными строками, что позволяет

сделать структуру ГА простой и прозрачной. Еще одно преимущество этих операторов – скорость работы; она достаточно высока, поскольку над генотипом не производится никаких сложных вычислений. Единственная ресурсоемкая операция в составе предложенного ГА – оценка решения, кодируемого генотипом, поскольку для ее вычисления требуется процедура декодирования.

Литература

1. Батищев Д.И., Коган Д.И.. Вычислительная сложность экстремальных задач переборного типа. Н.Новгород, ННГУ, 1994.
2. Батищев Д.И. Генетические алгоритмы решения экстремальных задач / Под ред. Львовича Я.Е.: Учеб. пособие, Воронеж, 1995.
3. Батищев Д.И., Старостин Н.В. Применение генетических алгоритмов к решению задачи дихотомического разбиения графа. Воронеж. Межвузовский сборник науч. трудов «Оптимизация и моделирование в автоматизированных системах», 1998 г., с.3 – 10.
4. Батищев Д.И., Старостин Н.В. Способы повышения эффективности генетического поиска оптимального *k*-разбиения графа. Воронеж. Межвузовский сборник н. трудов “Прикладные задачи моделирования и оптимизации”, 2000 г., Часть 2, стр. 4-17.
5. Батищев Д.И., Старостин Н.В. *k*-разбиение графов. Вестник ННГУ “Математическое моделирование и оптимальное управление”, Н.Новгород, 2000 г., стр. 27-35.