УДК. 681.3.015

©2000 г. Д.И. Батищев, Н.В. Старостин

K-РАЗБИЕНИЕ ГРАФОВ

Рассматривается задача k-разбиения простого графа, под которой понимается разрезание графа на k подграфов заданных размеров, вес сечения между которыми имеет наименьшее значение. Для решения данной задачи предлагается использовать генетический алгоритм. Приводится описание символьной модели и операторов размножения.

1. Постановка задачи. В ряде областей технической кибернетики, например, при распределении элементов на электронных платах, декомпозиции различных систем, распределении программ между машинами вычислительной системы и т.п., возникает задача разбиения графа на минимально связные подграфы, которую можно сформулировать следующим образом.

Пусть задан неориентированный взвешенный граф $G(V, E, w)$ порядка n, где $V = \{v_1, v_2, \ldots, v_n\}$ - множество вершин; $E \subseteq V \times V$ — множество ребер; $w : E \to R^+$ — отображение, определяющее вес каждого ребра.

Разбиение $(V_1, V_2, \ldots, V_k)$ множества вершин V, порождающее разбиение исходного графа на k подграфов, назовем *k-разбиением*. Множества вершин $V_1, \ldots, V_k$ должны удовлетворять следующим условиям:

$$V_i \cap V_j = \emptyset, \quad i, j = 1, 2, \ldots, k, \quad i \neq j; \tag{1}$$

$$\bigcup_{i=1}^{k} V_i = V; \tag{2}$$

$$|V_i| = n_i, \quad i = 1, 2, \ldots, k, \tag{3}$$

где k фиксировано и $n_1, \ldots, n_k$ — целые положительные числа, которые задаются как параметры перед решением задачи разбиения. Система требований (1) — (3), предъявляемых к разбиению $(V_1, V_2, \ldots, V_k)$, определяет область поиска D.

Совокупность ребер, соединяющих вершины из разных подграфов, будем называть *сечением разбиения*. Сечение разбиения $(V_1, V_2, \ldots, V_k)$ будем обозначать через $C(V_1, V_2, \ldots, V_k)$.

В качестве *критерия оптимальности* Q, определяющего эффективность k-разбиения $(V_1, V_2, \ldots, V_k)$, будем рассматривать вес сечения, равный сумме весов всех ребер сечения:

$$Q(V_1, V_2, \ldots, V_k) = \sum_{(v_i, v_j) \in C(V_1, V_2, \ldots, V_k)} w(v_i, v_j). \tag{4}$$

Тогда *оптимальным k-разбиением* назовем разбиение $(V_1^*, V_2^*, \ldots, V_k^*) \in D$ такое, что

$$Q(V_1^*, V_2^*, ..., V_k^*) = \min_{(V_1, V_2, ..., V_k) \in D} Q(V_1, V_2, \ldots, V_k).$$

Задача разбиения графа на минимально связные подграфы является NP-полной. В [1] и [2] предложены методы типа ветвей и границ для нахождения оптимального решения. В [3] рассмотрена связь задачи разбиения графа с задачей поиска раскраски графа и приводится эвристический метод ее решения. Использование детерминированных алгоритмов на графах с большой размерностью приводит к значительным затратам машинного времени даже для приближенных эвристических алгоритмов, не говоря уже о таких алгоритмах, как "ветви и границы". Как показывают экспериментальные исследования, введение элементов рандомизации позволяет сократить эти затраты и, соответственно, увеличить размерность обрабатываемых графов.

В настоящей работе для решения задачи поиска оптимального k-разбиения предлагается генетический алгоритм. Схема работы генетического алгоритма для задачи дихотомического разбиения графа подробно описана в [4] и [5].

2. Эволюционно-генетический подход. В 70-х годах в рамках случайного поиска Растригиным Л.А. был предложен ряд алгоритмов, использующих идеи бионического поведения особей [6]. Развитие этих идей нашло отражение в работах Бутаковой И.Л. по эволюционному моделированию [7]. Большую роль в развитии методов генетического поиска сыграли Holland J.H. [8], Goldberg D.E. [9], Devis L. [10], которые заложили и развили основы генетических алгоритмов, моделирующих механизмы - аналоги биологических процессов популяционной генетики.

Генетические алгоритмы работают с совокупностью "*особей*" (*популяцией*), каждая из которых представляет возможное решение исследуемой проблемы. Так, в задаче оптимального k-разбиения в качестве особи выступает конкретное решение $(V_1, V_2, \ldots, V_k)$, удовлетворяющее условиям (1) - (3), что позволяет интерпретировать сам процесс решения экстремальной задачи как эволюционный процесс, связанный с перераспределением вершин $v_i \in V$ графа $G(V, E, w)$ по k подграфам соответствующих порядков с целью отыскания глобального минимума критерия оптимальности (4). В этом и заключается в данном случае цель эволюционного развития особей.

Решения исследуемой задачи определенным образом кодируются. Код решения задачи представляет из себя цепочку символов, которую можно интерпретировать как *хромосому (генотип)*, содержащую сцепленные между собой *гены*, расположенные в линейной последовательности "слева — направо". Местоположение определенного гена в хромосоме называется *локусом*, а альтернативные формы одного и того же гена, расположенные в одинаковых локусах хромосомы, называются *аллелями*.

Каждая "особь" оценивается мерой ее "приспособленности" согласно тому, насколько "хорошо" соответствующее ей решение задачи. Наиболее приспособленные особи получают возможность участвовать в процессе "воспроизводства" потомства. Это приводит к появлению новых особей, которые сочетают в себе некоторые особенности, наследуемые ими от родителей. Наименее приспособленные особи с меньшей вероятностью смогут воспроизвести потомков, так что те свойства, которыми они обладали, будут постепенно исчезать из популяции в процессе эволюции.

Так и воспроизводится вся новая популяция возможных решений, выбирая лучших представителей предыдущего поколения, скрещивая их и получая множество

новых особей. Это новое поколение содержит более высокое соотношение характеристик, чем в предыдущем поколении. Таким образом, из поколения в поколение хорошие характеристики распространяются по всей популяции. В конечном итоге, популяция будет сходиться к оптимальному решению задачи.

В генетическом алгоритме можно выделить шесть основных операторов:

1. Оператор формирования первоначальной популяции.

2. Оператор скрещивания. Осуществляет подбор брачных пар особей. Примером такого подбора может служить панмиксия: "родители" случайным равновероятным образом выбираются из популяции для образования брачной пары.

3. Оператор размножения (кроссовер). Из выбранных родительских особей производит новых особей (потомков).

4. Оператор мутации. С определенной вероятностью производит случайные нарушения в генетическом коде родительских особей с целью повышения вероятности локализации точки глобального оптимума.

5. Оператор оценивания особей. Для всех полученных новых особей рассчитывает их меры приспособленности.

6. Оператор отбора. Формирует популяцию следующего поколения, являясь механизмом "выживания" наиболее приспособленных особей. Примером такого отбора может служить "жесткая" схема естественного отбора, согласно которой чем больше приспособленность особи, тем выше вероятность ее включения в следующее поколение.

Процедуры с 2 по 6 последовательно выполняются в цикле до тех пор, пока не наступят условия останова генетического алгоритма. Таким условием может быть, например, условие сходимости генетического алгорима к некоторому решению, когда вся популяция представлена одним решением исследуемой задачи.

В последние годы реализовано много различных схем генетических алгоритмов. Исследователи экспериментировали с различными типами представлений, операторов кроссовера и мутации, специальных операторов и различных подходов к воспроизводству и отбору.

Хотя модель эволюционного развития, применяемая в генетических алгоритмах, сильно упрощена по сравнению со своим природным аналогом, тем не менее данный класс алгоритмов является достаточно мощным средством и может с успехом применяться для широкого класса прикладных задач, включая те, которые трудно, а иногда и вовсе невозможно, решить другими методами. Примером тому может быть задача оптимального k-разбиения графа. Однако, генетические алгоритмы не гарантируют обнаружения глобального оптимума за полиномиальное время. Они не гарантируют и того, что глобальный оптимум будет найден, однако они пригодны для поиска "достаточно хорошего" решения задачи "достаточно быстро".

3. Символьная модель. Для того, чтобы в рамках эволюционно-генетического подхода реализовать эффективные генетические алгоритмы, необходимо для исследуемой задачи глобальной оптимизации построить символьную модель.

Представим k-разбиение $(V_1, \ldots, V_k)$ графа $G(V, E, w)$ порядка n в виде строки длины n: для $i = 1, \ldots, n$ в i-ой позиции стоит число j_i такое, что вершина v_i принадлежит подмножеству V_{j_i}. При этом каждая строка должна удовлетворять условиям, связанным с требованиями k-разбиения: число j, $j = 1, \ldots, k$, в такой строке должно встречаться ровно n_j раз, где n_j — мощность множества вершин V_j. Такую строку можно интерпретировать как хромосому, тогда k-ичные числа выступают в качестве генов, которые определяют, к какой части разбиения $V_1, V_2, \ldots, V_k$ принадлежат соответствующие им вершины графа G.

На рис. 1 изображен граф порядка 12. Веса всех ребер равны 1. Рассмотрим, для примера, разбиение данного графа на 3 подграфа: $V_1 = \{v_1, v_4, v_5, v_{12}\}$, $V_2 = \{v_2, v_3, v_9, v_{10}\}$, $V_3 = \{v_6, v_7, v_8, v_{11}\}$.

Вес сечения данного разбиения будет равен 14. Хромосома будет выглядеть следующим образом:

v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9	v_{10}	v_{11}	v_{12}
1	2	2	1	1	3	3	3	2	2	3	1

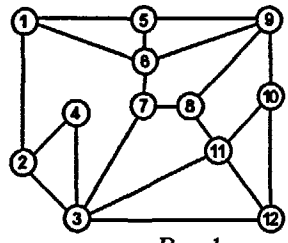

Рис.1

Тем самым мы свели задачу k-разбиения к задаче поиска такой строки (кодировки) фиксированной длины n, которой соответствует разбиение с наименьшим весом сечения. Будем решать данную задачу, используя генетический алгоритм. Поиск оптимального решения в нем осуществляется путем прямого манипулирования с совокупностью из нескольких допустимых решений, закодированных в k-ичном коде.

4. Классические схемы размножения. В качестве операторов воспроизводства потомков в генетическом алгоритме обычно используются классические схемы одноточечного, двухточечного и равномерного кроссоверов [4], [5], [8-10]. Работа данных операторов заключается в разрыве родительских хромосом и сцеплении соответствующих их участков в хромосомы потомков. При этом точки разрыва выбираются случайно без использования какой бы то ни было информации об особенностях исследуемой задачи.

На рис. 2 показана схема получения генотипов потомков одноточечным кроссовером из двух родительских хромосом.

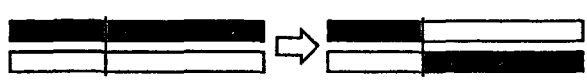
Рис.2

Применение данных схем размножения для задачи k-разбиения графа может привести к получению хромосом, которые не будут удовлетворять требованиям, накладываемым на генотипы. Поэтому такие некорректные генотипы необходимо корректировать. Предлагается алгоритм случайной коррекции, который заключается в следующем: случайным образом в некорректной хромосоме потомка заменяются гены, содержащие избыточные аллели.

Классические схемы размножения работают исключительно с генотипами, но никак не с разбиениями. Поэтому их использование оказывается неэффективным. Предлагается совершенно иной принцип построения генотипов потомков, основанный на информации о структуре разрезаемого графа.

5. Структурный оператор размножения. *Структурный оператор размножения* работает не с хромосомами, но с разбиениями, и из двух родительских

разбиений получает одно новое — разбиение потомка. Формирование нового разбиения происходит в два этапа. На первом этапе осуществляется начальное формирование всех подграфов разбиения. Подграфы формируются из компонент связности исходного графа без ребер сечения первого родительского разбиения и из компонент связности исходного графа без ребер сечения второго родительского разбиения. На втором этапе подграфы полностью достраиваются путем последовательного включения тех вершин, которые не вошли ни в один из подграфов.

Алгоритм структурного оператора размножения

1. Процедура начального формирования подграфов разбиения.

   1.1. Положим $i = 1$.

   1.2. Положим $\Psi = \emptyset$ (Ψ — множество вершин, распределенных по подграфам).

   1.3. Рассмотрим граф, который получен из исходного графа $G(V, E, w)$ удалением всех ребер сечения первого родительского разбиения и всех вершин из множества Ψ вместе с инцидентными им ребрами. Для построенного графа определяем компоненты связности.

   1.4. Рассмотрим граф, который получен из исходного графа $G(V, E, w)$ удалением всех ребер сечения второго родительского разбиения и всех вершин из множества Ψ вместе с инцидентными им ребрами. Для построенного графа определяем компоненты связности.

   1.5. Для всех компонент связности рассчитываем параметры: размер — число вершин компоненты; вес — суммарный вес ребер компоненты.

   1.6. Из всех компонент, размеры которых не превышают числа вершин i-ого подграфа (в случае, когда отсутствуют компоненты требуемого размера, считаем подграф с номером i инициированным и переходим к п. 1.9), выбирают компоненту связности с наибольшим весом.

   1.7. Все вершины из выбранной компоненты переходят в i-ый подграф (формируют множество V_i).

   1.8. Положим $\Psi = \Psi \cup V_i$.

   1.9. Положим $i = i + 1$.

   1.10. Если все подграфы инициированы ($i > k$), то выход из процедуры, иначе переход к п. 1.3.

2. Процедура достройки подграфов.

   2.1. Если все подграфы полностью сформированы ($|V_1| = n_1, \ldots, |V_k| = n_k$), то выход из оператора, иначе переход к п. 2.2.

   2.2. Случайным образом выбираем вершину v_j из множества вершин, не вошедших ни в один из подграфов.

   2.3. Вершина v_j включается в тот подграф, который еще полностью не сформирован ($|V_i| < n_i$, где $i = 1, \ldots, k$), и имеет наибольший суммарный вес ребер, связывающих вершину v_j с вершинами данного подграфа.

   2.4. Переход к п. 2.1.

6. Локальная адаптация. Оператор локальной адаптации является специальным оператором, который используется для увеличения степени приспособленности особей. В его основе лежит достаточно быстрый эвристический алгоритм улучшения начальных разбиений методом обмена вершин из разных подграфов, описанный в [5], [11].

Оператор локальной адаптации осуществляет поиск некоторого локального оптимума в окрестности относительного обмена - всевозможных разбиений графа $G(V, E, w)$, которые можно получить из текущего разбиения $(V_1, \ldots, V_k)$ с помощью обмена двух вершин из разных подграфов. Оператор может быть применен к заданному разбиению несколько раз путем осуществления нескольких однократных обменов. Число таких однократных обменов назовем *глубиной прижизненной адаптации*. Если разбиение после действия оператора прижизненной адаптации невозможно улучшить с помощью однократных обменов, локальную адаптацию будем называть *полной локальной адаптацией*.

Этот оператор целесообразно использовать для максимизации степени приспособленности потомков и мутантов, получаемых на этапах размножения и мутации.

7. Тестовые задачи и эксперименты. В нашем распоряжении было 30 графов различных порядков и со своими характерными особенностями. На данных графах ставились разнообразные задачи k-разбиения графов. На этих задачах проверялась эффективность предложенных подходов. Как показывает практика, в подавляющем большинстве случаев генетический алгоритм со структурным оператором размножения демонстрировал более качественный поиск, нежели классические операторы размножения со случайной коррекцией генотипов. В качестве примера приводятся четыре тестовых задачи и результаты экспериментов с обоими типами операторов размножения.

Задача 1. Имеется граф порядка 132, состоящий из 6 не связанных между собой подграфов порядка 22. Каждый такой подграф представляет собой "склейку" по одной вершине трех полных графов порядка 8. Веса всех ребер равны 1. Граф требуется разбить на 4 подграфа так, что $n_1 = n_2 = 44$, $n_3 = n_4 = 22$. Очевидно, что вес сечения оптимального разбиения равен 0.

Задача 2. Имеется граф порядка 71, состоящий из пяти подграфов порядков 11, 15, 18, 13, 14. Для каждого из подграфов множество ребер формировалось случайным образом так, чтобы средняя степень вершин подграфов приблизительно была равна 3. Подграфы соединены между собой 10 ребрами. Веса всех ребер графа равны 1. Граф требуется разбить на 5 подграфов так, что $n_1 = 11$, $n_2 = 15$, $n_3 = 18$, $n_4 = 13$, $n_5 = 14$. Дополнительные 10 ребер выбраны так, что вес сечения оптимального разбиения равен 10.

Задача 3. Имеется обычный сеточный граф порядка 100, бисекционная ширина которого равна 10. Веса всех ребер равны 1. Граф требуется разбить на 3 подграфа: $n_1 = 50$, $n_2 = 30$, $n_3 = 20$. Можно показать, что вес сечения оптимального разбиения равен 15.

Задача 4. Имеется граф порядка 165, состоящий из пяти подграфов порядков 50, 45, 35, 20, 15. Для каждого из подграфов множество ребер формировалось случайным образом так, чтобы средняя степень вершин подграфов приблизительно была равна 6. Подграфы соединены между собой 12 ребрами. Веса всех ребер графа равны 1.

Граф разбивается на 5 подграфов: $n_1 = 50$, $n_2 = 45$, $n_3 = 35$, $n_4 = 20$, $n_5 = 15$. Дополнительные 12 ребер выбраны так, что вес сечения оптимального разбиения равен 12.

Объектом исследования в вычислительных экспериментах являлся структурный оператор размножения, его влияние на сходимость генетического алгоритма и способность отыскивать оптимальные решения задач. Данный оператор сравнивался с классическим оператором равномерного кроссовера со случайной коррекцией генотипов.

Во всех вычислительных экспериментах использовались следующие параметры генетического алгоритма: случайное формирование начальной популяции; родительские пары формировались случайным образом так, чтобы особи с одинаковыми генотипами не попали в одну пару (панмиксия генотипов); мутация полностью отсутствует; используется полная локальная адаптация потомков; применяется жесткая схема естественного отбора; численность популяции равна 20 (для задачи 1) и 40 (для остальных задач); число брачных пар равно 20 (для задачи 1) и 40 (для остальных задач).

Параметры останова: найдено лучшее решение, или число различных генотипов меньше 3, или число вычислений превысило 2000. Под вычислением понимается общее число исследованных допустимых решений в процессе генетического поиска.

Таблица 1

Задача	Операторы размножения							
	равномерный кроссовер				структурный			
	поколен	вычисл	время	лучшее	поколен	вычисл	время	лучшее
1	17	497	45с	0	5	120	19с	0
2	36	1656	58с	14.25	20	842	46с	10
3	26	1821	73с	15.4	16	861	46с	15
4	27	1692	318с	20.5	6	280	72с	12

8. Результаты экспериментов. Для каждой задачи осуществлялось по 20 экспериментов с различными схемами размножения. Эксперименты проводились на компьютере IBM PC AT со следующей конфигурацией: AMD K5 Pr-100 / 24Mb / Windows 95 OSR. Усредненные результаты экспериментов приведены в таблице 1.

Задача 1. Оператор равномерного кроссовера и оператор структурного размножения позволяли отыскивать оптимальное решение. Однако, генетический алгоритм с оператором структурного размножения находил его быстрее, затратив, в среднем, в два раза меньше времени.

Задача 2. Оператор равномерного кроссовера обычно приводил генетический алгоритм к преждевременной сходимости. Оператор структурного размножения позволял удерживать генетическое разнообразие популяции на достаточном уровне и всегда находил лучшие решения, затрачивая в среднем меньше времени.

Задача 3. Оператор равномерного кроссовера позволил найти оптимальное решение задачи в 12 случаях из 20 запусков. Оператор структурного размножения во всех 20 экспериментах находил оптимальное решение, затрачивая в среднем почти в два раза меньше времени.

Задача 4. Оператор структурного размножения в 15 случаях из 20 позволял находить лучшие решения, чем оператор равномерного кроссовера, затрачивая на их поиск значительно меньше времени.

Сходные результаты получены и для остальных задач на 30 тестовых графах. На данном основании можно сделать следующий вывод: использование оператора структурного размножения позволяет повысить эффективность генетического алгоритма для задач k-разбиения графов.

Работа выполнена при поддержке гранта РФФИ 00-01-00384.

Список литературы

[1] Гоpинштейн Л.Л. О разбиении графов // Изв. АН СССР. Техническая кибернетика. 1969. N 1. С. 79-85.

[2] Рыжков А.П. Алгоритм разбиения графа на минимально связные подграфы // Изв. АН СССР. Техническая кибернетика. 1975. N 6. С. 122-128.

[3] Орлова Г.И., Дорфман Я.Г. Оптимальное деление графа на несколько подграфов // Изв. АН СССР. Техническая кибернетика. 1972. N 1. С. 118-121.

[4] Батищев Д.И. Генетические алгоритмы решения экстремальных задач / Под ред. Я.Е.Львовича. Учебное пособие. Воронеж. 1995.

[5] Батищев Д.И., Старостин Н.В. Применение генетических алгоритмов к решению задачи дихотомического разбиения графа // Оптимизация и моделирование в автоматизированных системах / Межвузовский сборник научных трудов. Воронеж. 1998. С. 3-10.

[6] Растригин Л.А. Адаптация сложных систем. Методы и приложения. Рига: Зинатне, 1981.

[7] Бутакова И.Л. Эвоинформатика. Теория и практика эволюционного моделирования. М.: Наука, 1991.

[8] Holland J.H. Adaptation in natural and artificial systems. Cambridge, Massachusetts: The MIT Press, 1992.

[9] Goldberg D.E. Genetic algorithms in search, optimization, and machine learning. New York: Addison Wesley, 1989.

[10] Davis L. Handbook of genetic algoithms. New York: Van Nostrand Reinhold, 1991.

[11] Пападимитриу Х., Стайглиц К. Комбинаторная оптимизация. М.: Мир, 1985.

Нижегородский государственный университет им. Н.И. Лобачевского
603600, Нижний Новгород, пр. Гагарина, 23

K-PARTITION OF GRAPHS

D.I.Batishchev, N.V.Starostin
Nizhny Novgorod State University
23 Gagarin Ave., 603600 Nizhny Novgorod, Russia

This paper is dedicated to the graph partitioning problem. The goal is to cut a graph into k subgraphs with given number of nodes. We should minimize total weight of edges which are not belonging to subgraphs. To solve this problem we suggest a kind of genetic algorithm. Also a symbolic model and crossover operations are described.