ДИСКРЕТНЫЙ АНАЛИЗ И ИССЛЕДОВАНИЕ ОПЕРАЦИЙ
Май—июнь 2012. Том 19, № 3. С. 13–26

УДК 519.718

О СЛОЖНОСТИ ОПТИМАЛЬНОЙ РЕКОМБИНАЦИИ ДЛЯ ОДНОЙ ЗАДАЧИ СОСТАВЛЕНИЯ РАСПИСАНИЙ С ПЕРЕНАЛАДКАМИ *)

А. В. Еремеев, Ю. В. Коваленко

Аннотация. Рассматривается вычислительная сложность оптимальной рекомбинации для одной задачи составления расписаний с переналадками. Доказана NP-трудность в сильном смысле этой задачи и предложен точный алгоритм её решения. Показано, что трудоёмкость данного алгоритма полиномиальна для «почти всех» индивидуальных задач оптимальной рекомбинации.

Ключевые слова: расписание, переналадка, генетический алгоритм, оптимальная рекомбинация.

Введение

Рассматривается следующая задача теории расписаний. Задано множество работ $V = \{v_1, \ldots, v_k\}$, которые должны быть выполнены на единственном имеющемся устройстве. Каждая работа $v \in V$ характеризуется длительностью $p_v \in \mathbb{R}^+$ ($\mathbb{R}^+$ — множество положительных вещественных чисел). Прерывание выполнения работ не допускается. В каждый момент времени устройство не может быть задействовано более чем в одной работе. При этом если устройство переключается с одной работы на другую, то необходимо выполнять переналадку. Пусть $s_{vu} \in \mathbb{R}_+$ — длительность переналадки с работы v на работу u для всех $v, u \in V$, где $v \neq u$ ($\mathbb{R}_+$ — множество неотрицательных вещественных чисел). Требуется составить расписание, минимизирующее время завершения всех работ.

Обозначим через $\pi = (\pi_1, \ldots, \pi_k)$ перестановку, определяющую порядок выполнения работ, а именно, π_i — работа, выполняемая i-й по счёту. Пусть $s(\pi) = \sum\limits_{i=1}^{k-1} s_{\pi_i, \pi_{i+1}}$. Тогда задача эквивалентна поиску такой

*) Исследование выполнено при финансовой поддержке проекта РФФИ (проект 12–01–00122), интеграционного проекта СО РАН (проект № 7Б) и Президиума РАН (проект № 15.8).

© 2012 Еремеев А. В., Коваленко Ю. В.

перестановки π^*, при которой минимизируется суммарная длительность переналадки $s(\pi^*)$.

В соответствии с [15] данная задача обозначается через $1|s_{vu}|C_{\max}$ и является NP-трудной в сильном смысле, так как к ней полиномиально сводится NP-полная в сильном смысле задача Гамильтонов путь [3].

В настоящей работе исследуется вычислительная сложность задачи поиска для произвольных заданных *родительских* решений π^1 и π^2 такой перестановки π', при которой

(i) $\pi'_i = \pi^1_i$ или $\pi'_i = \pi^2_i$ для всех $i = 1, \ldots, k$;

(ii) π' имеет минимальное значение целевой функции $s(\pi')$ среди всех перестановок, удовлетворяющих (i).

Нахождение такой перестановки есть задача оптимальной рекомбинации и возникает при поиске наилучшего возможного результата оператора кроссинговера (рекомбинации) в генетических алгоритмах (см., например, [14, 18]) для задачи $1|s_{vu}|C_{\max}$.

В [12] изучался вопрос о применимости оптимальной рекомбинации исследуемого типа к задачам комбинаторной оптимизации при различных способах представления решений в генетическом алгоритме. В частности, экспериментальным путём показано преимущество использования такой оптимальной рекомбинации по сравнению с другими известными операторами кроссинговера, в составе генетических алгоритмов для задач 3-Выполнимость и построения конвейерного расписания (flow shop). В [5] аналогичные результаты получены для одной задачи составления расписаний многопродуктового производства. Теоретический и экспериментальный анализ иных формулировок задачи оптимальной рекомбинации в случаях, когда множество допустимых решений составляют перестановки, представлен в [4, 11, 19].

Статья построена следующим образом. В разд. 1 с использованием результатов А. И. Сердюкова [9] показана NP-трудность в сильном смысле представленной задачи оптимальной рекомбинации. В разд. 2 предложен алгоритм решения исследуемой задачи, основанный на переборе совершенных паросочетаний в специальном двудольном графе [9]. Доказано, что для «почти всех» пар родительских решений трудоёмкость данного алгоритма равна $O(k \cdot \ln(k))$. В разд. 3 содержатся заключительные замечания.

1. NP-трудность задачи оптимальной рекомбинации

Рассмотрим задачу о кратчайшем гамильтоновом пути с предписаниями вершин следующего вида. Задан полный ориентированный граф $G = (X, U)$, где $X = \{x_1, \ldots, x_n\}$ — множество вершин, $U = \{(x, y) \mid x, y \in X,\ x \neq y\}$ — множество дуг, которым приписаны веса $\rho(x, y) \in \mathbb{R}_+$, $(x, y) \in U$. Также задана система подмножеств (предписаний) $X^i \subseteq X$, $i = 1, \ldots, n$, удовлетворяющая условиям:

($\mathcal{C}1$) $|X^i| \leqslant 2$ для всех $i = 1, \ldots, n$;

($\mathcal{C}2$) $1 \leqslant |\{i \mid x \in X^i,\ i = 1, \ldots, n\}| \leqslant 2$ для всех $x \in X$;

($\mathcal{C}3$) если $x \in X^i$ и $x \in X^j$, где $i \neq j$, то $|X^i| = |X^j| = 2$, а если $x \in X^i$ только при одном значении i, то $|X^i| = 1$.

Обозначим через F множество взаимно однозначных отображений из $X_n = \{1, \ldots, n\}$ в X таких, что $f(i) \in X^i$, $i = 1, \ldots, n$, для любого $f \in F$. Требуется найти такое отображение $f^* \in F$, что $\rho(f^*) = \min\limits_{f \in F} \rho(f)$, где $\rho(f) = \sum\limits_{i=1}^{n-1} \rho(f(i), f(i+1))$ для всех $f \in F$. Обозначим эту задачу через I.

Для задачи I всегда существует допустимое решение f^1. Действительно, существование допустимого решения задачи I равносильно существованию совершенного паросочетания W в двудольном графе $\overline{G} = (X_n, X, \overline{U})$ с равными по мощности долями вершин X_n, X и множеством рёбер $\overline{U} = \{(i, x) \mid i \in X_n,\ x \in X^i\}$. Заметим, что если степень вершины $i \in X_n$ в графе $\overline{G}$ равна d $(1 \leqslant d \leqslant 2)$, то ввиду условий ($\mathcal{C}2$) и ($\mathcal{C}3$) степень всех смежных с ней вершин также равна d. Тогда существование W следует из теоремы Кёнига — Холла [1], так как

$$|Y| \leqslant |\{x \in X \mid x \in X^i,\ i \in Y\}|$$

для любого $Y \subseteq X_n$.

Кроме того, совершенное паросочетание

$$W = \{(1, x^1), (2, x^2), \ldots, (n, x^n)\} \subseteq \overline{U}$$

может быть найдено за полиномиальное время, например, с помощью алгоритма Кёнига — Холла [1]. Полагая $f^1(i) = x^i$, $i = 1, \ldots, n$, получаем допустимое решение задачи I.

Ясно, что при $|X^i| = 1$, $i = 1, \ldots, n$, задача I тривиальна в том смысле, что искомое отображение определяется однозначно. Поэтому далее будем предполагать, что для задачи I найдётся $i \in X_n$ такое, что $|X^i| = 2$. Тогда для этой задачи существует ещё по крайней мере одно допустимое решение f^2, где $f^2(i) = X^i \backslash \{f^1(i)\}$, если $|X^i| = 2$, и $f^2(i) = f^1(i)$ в противном случае, $i = 1, \ldots, n$.

Перейдём к анализу вычислительной сложности задачи оптимальной рекомбинации (i)–(ii) для $1|s_{vu}|C_{\max}$. Покажем, что к ней сводится задача I. Вершине $x_i \in X$ графа G поставим в соответствие работу v_i, $i = 1, \ldots, n$. Пусть число работ k равно n, а длительность переналадки s_{xy} равна $\rho(x, y)$ для всех $x, y \in X$, где $x \neq y$. Полагая $\pi^1 = f^1$ и $\pi^2 = f^2$, получаем полиномиальную сводимость задачи I к исследуемой задаче. Ввиду свойств данной сводимости из доказанной ниже NP-трудности в сильном смысле задачи I следует NP-трудность в сильном смысле задачи оптимальной рекомбинации (i)–(ii).

А. И. Сердюковым [9] показана NP-трудность в сильном смысле задачи коммивояжёра с предписаниями вершин, в которой система предписаний удовлетворяет только условию ($\mathcal{C}1$) и требуется найти такое отображение $\widetilde{f}^*$, что $\widetilde{\rho}(\widetilde{f}^*) = \min\limits_{f \in F} \widetilde{\rho}(f)$, где

$$\widetilde{\rho}(f) = \sum_{i=1}^{n-1} \rho(f(i), f(i+1)) + \rho(f(n), f(1))$$

для всех $f \in F$. Обозначим эту задачу через $\widetilde{I}$. Докажем NP-трудность в сильном смысле задачи I с помощью сводимости по Тьюрингу к ней задачи $\widetilde{I}$.

Утверждение 1. *Задача I NP-трудна в сильном смысле.*

Доказательство. Покажем, что за полиномиальное время для системы предписаний X^i, $i = 1, \ldots, n$, задачи $\widetilde{I}$ можно построить эквивалентную ей систему предписаний, удовлетворяющую условиям ($\mathcal{C}1$)–($\mathcal{C}3$), или установить, что задача $\widetilde{I}$ не имеет допустимых решений. Для этого последовательно выполним следующие преобразования.

(a) Если найдётся вершина $x \in X$ такая, что $|\{i \in X_n \mid x \in X^i\}| = 0$, то задача $\widetilde{I}$ неразрешима и преобразования окончены.

(b) Выполняем следующие действия до тех пор, пока не останутся только двухэлементные предписания. Находим подмножество $X^i = \{x\}$, т. е. $|X^i| = 1$, и удаляем вершину x из других подмножеств, её содержащих. Если после этого найдётся j такое, что $|X^j| = 0$, то задача $\widetilde{I}$ неразрешима и преобразования окончены. В противном случае удаляем вершину x и подмножество X^i из дальнейшего рассмотрения в процессе преобразования.

(c) Выполняем следующие действия до тех пор, пока не останутся только вершины, содержащиеся в двух предписаниях, каждое из которых будет двухэлементным. Находим вершину x, содержащуюся только в одном подмножестве $X^i = \{x, y\}$. Если вершина y также содержится только в X^i, то задача $\widetilde{I}$ неразрешима и преобразования окончены.

В противном случае полагаем $X^i = \{x\}$ и удаляем вершину x и подмножество X^i из дальнейшего рассмотрения в процессе преобразования.

Ясно, что оставшиеся двухэлементные подмножества и удалённые из рассмотрения одноэлементные подмножества образуют систему предписаний, эквивалентную исходной и удовлетворяющую условиям $(\mathcal{C}1)$–$(\mathcal{C}3)$. Далее будем предполагать, что система предписаний задачи $\widetilde{I}$ приведена к такому виду.

Теперь сведём по Тьюрингу задачу $\widetilde{I}$ к задаче I. Пусть существует гипотетическая подпрограмма $\mathcal{S}$ решения задачи I с системой предписаний $\overline{X}^i$, $i = 1, \ldots, n$. Построим алгоритм $\mathcal{A}$, который будет решать задачу $\widetilde{I}$ с системой предписаний X^i, $i = 1, \ldots, n$, используя не более четырёх раз подпрограмму S для задач вида I, образуемых фиксацией в предписаниях X^1 и X^n одного из их элементов. Заметим, что такая фиксация может привести к нарушению условия $(\mathcal{C}3)$, тогда в алгоритме $\mathcal{A}$ полученная система предписаний преобразуется в эквивалентную ей систему предписаний, удовлетворяющую условиям $(\mathcal{C}1)$–$(\mathcal{C}3)$. Опишем предлагаемый алгоритм по шагам.

Алгоритм $\mathcal{A}$

Шаг 1. Обозначим через $\widetilde{f}'$ лучшее найденное решение задачи $\widetilde{I}$, а через $\widetilde{\rho}'$ — значение целевой функции, ему соответствующее. Полагаем $\widetilde{\rho}' = +\infty$.

Шаг 2. Для каждой вершины $x \in X^1$ выполняем шаги 2.1–2.2.

Шаг 2.1. Полагаем $\widetilde{X}^1 = \{x\}$, $\widetilde{X}^i = X^i$, $i = 2, \ldots, n$, и если $|X^1| = 2$, то преобразуем систему предписаний $\widetilde{X}^i$, $i = 1, \ldots, n$, таким образом, чтобы для неё выполнялось условие $(\mathcal{C}3)$. Для этого находим $j \neq 1$ такое, что $\widetilde{X}^j = \{x, z\}$, и полагаем $\widetilde{X}^j = \{z\}$. Далее выполняем аналогичные действия для вершины z и т. д.

Шаг 2.2. Для каждой вершины $y \in \widetilde{X}^n$ выполняем шаги 2.2.1–2.2.2.

Шаг 2.2.1. Полагаем $\overline{X}^n = \{y\}$, $\overline{X}^i = \widetilde{X}^i$, $i = 1, \ldots, n-1$, и если $|\widetilde{X}^n| = 2$, то преобразуем систему предписаний $\overline{X}^i$, $i = 1, \ldots, n$, таким образом, чтобы для неё выполнялось условие $\mathcal{C}3$ (см. шаг 2.1).

Шаг 2.2.2. Решаем полученную задачу I с помощью алгоритма $\mathcal{S}$. Пусть f^* — решение этой задачи. Если $\rho(f^*) + \rho(\overline{X}^n, \overline{X}^1) < \widetilde{\rho}'$, то полагаем $\widetilde{\rho}' = \rho(f^*) + \rho(\overline{X}^n, \overline{X}^1)$ и $\widetilde{f}' = f^*$.

Ясно, что решение $\widetilde{f}'$, полученное алгоритмом $\mathcal{A}$, оптимальное для задачи $\widetilde{I}$. Так как $|X^1| \leqslant 2$ и $|X^n| \leqslant 2$, а трудоёмкость преобразования системы предписаний равна $O(n^2)$, сводимость полиномиальна. Из свойств данной сводимости следует NP-трудность в сильном смысле задачи I. Утверждение 1 доказано.

Теорема 1. *Задача оптимальной рекомбинации* (i)–(ii) *для* $1|s_{vu}|C_{\max}$ NP-*трудна в сильном смысле.*

Отметим, что ориентированный случай задачи I (граф G ориентированный) легко сводится к неориентированному заменой каждой вершины тремя вершинами (см., например, [6]) и заданием соответствующей системы предписаний X^i, $i = 1, \ldots, n$. Поэтому задача I в неориентированном случае NP-трудна в сильном смысле, и имеет место

Теорема 2. *Задача оптимальной рекомбинации* (i)–(ii) *в задаче расписания* $1|s_{vu} = s_{uv}|C_{\max}$ NP-*трудна в сильном смысле.*

2. Решение задачи оптимальной рекомбинации

Если работе $v_i \in V$ поставить в соответствие вершину $x_i \in X$ графа G, $i = 1, \ldots, k$, положить число вершин n равным k, $\rho(x, y) = s_{xy}$ для всех $x, y \in X$, где $x \neq y$, $X^i = \left\{\pi_i^1, \pi_i^2\right\}$, $i = 1, \ldots, k$, то множество допустимых решений задачи I будет взаимно однозначно отображаться на множество допустимых решений задачи оптимальной рекомбинации (i)–(ii), причём оптимальному решению будет соответствовать оптимальное.

Оптимальное отображение $f^* \in F$ в задаче I можно найти за время $O(2^k)$ путём перебора решений (среди которых будут как допустимые, так и недопустимые), формируемых выбором в каждом подмножестве системы предписаний одного из элементов. Такую же трудоёмкость имеет очевидная модификация алгоритма из [17] для задачи коммивояжёра. Однако можно построить более эффективный алгоритм для решения задачи I, используя подход А. И. Сердюкова [9], разработанный для оценки мощности множества допустимых решений в задаче $\widetilde{I}$.

Рассмотрим двудольный граф $\overline{G} = (X_k, X, \overline{U})$ (см. разд. 1). Заметим, что между множествами допустимых решений F задачи I и совершенных паросочетаний $\mathcal{W}$ в $\overline{G}$ существует взаимно однозначное соответствие.

Ребро $(i, x) \in \overline{U}$ назовём *особым*, если (i, x) принадлежит любому совершенному паросочетанию в графе $\overline{G}$. *Помеченными* будем называть вершины $\overline{G}$, инцидентные особым рёбрам. Под *блоком* в $\overline{G}$ будем понимать максимальный двусвязный подграф [7], содержащий не менее двух рёбер. Заметим, что в каждом блоке j графа $\overline{G}$ степень любой вершины равна двум, $j = 1, \ldots, q(\overline{G})$, где $q(\overline{G})$ — число блоков в $\overline{G}$. Тогда рёбра $(i, x) \in \overline{U}$ такие, что $|X^i| = 1$, особые и блокам не принадлежат, а рёбра $(i,x) \in \overline{U}$ такие, что $|X^i| = 2$, принадлежат. Кроме того, каждый блок $j = 1, \dots, q(\overline{G})$ графа $\overline{G}$ имеет ровно два максимальных (совершенных) паросочетания, наборы рёбер которых различны, поэтому он не содержит особых рёбер. Следовательно, ребро $(i,x) \in \overline{U}$ является особым тогда и только тогда, когда $|X^i| = 1$, и любое совершенное паросочетание в графе $\overline{G}$ взаимно однозначно определяется набором максимальных паросочетаний (по одному из каждого блока) и совокупностью особых рёбер.

Для примера рассмотрим задачу I, в которой $n = k = 7$ и система предписаний имеет вид: $X^1 = \{x_3, x_7\}$, $X^2 = \{x_3, x_7\}$, $X^3 = \{x_2\}$, $X^4 = \{x_5\}$, $X^5 = \{x_1, x_4\}$, $X^6 = \{x_4, x_6\}$, $X^7 = \{x_1, x_6\}$. Соответствующий задаче I двудольный граф $\overline{G} = (X_7, X, \overline{U})$, его особые рёбра и блоки представлены на рис. 1. Здесь рёбра каждого блока, выделенные жирным, образуют одно максимальное паросочетание блока, а остальные рёбра блока — второе.

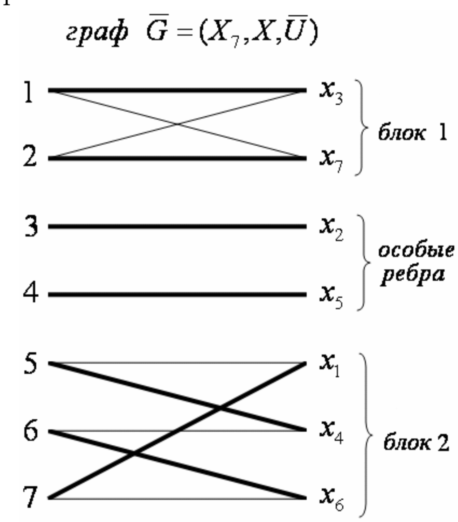

Рис. 1. Пример графа $\overline{G} = (X_7, X, \overline{U})$, содержащего два особых ребра и два блока

Блоки в графе $\overline{G}$ могут быть вычислены за время $O(k)$, например, с помощью алгоритма «поиск в глубину» [7]. Особые рёбра и максимальные паросочетания в блоках находятся очевидным образом за время $O(k)$.

Таким образом, для решения задачи I можно предложить следующий алгоритм. Строим двудольный граф $\overline{G}$, определяем в нём набор особых рёбер и блоков, а также находим максимальные паросочетания в блоках.

Перебираем все совершенные паросочетания $W \in \mathcal{W}$ в $\overline{G}$ (формируя их из максимальных паросочетаний в блоках и особых рёбер). Каждому $W \in \mathcal{W}$ ставим в соответствие $f \in F$ и вычисляем $\rho(f)$. В результате находим $f^* \in F$ такое, что $\rho(f^*) = \min\limits_{f \in F} \rho(f)$.

Поскольку $|F| = |\mathcal{W}| = 2^{q(\overline{G})}$, вычислительная сложность представленного алгоритма равна $O(k \cdot 2^{q(\overline{G})})$, причём $q(\overline{G}) \leqslant \lfloor \frac{k}{2} \rfloor$, и последняя оценка достижима. Рассмотрим модификацию данного алгоритма, трудоёмкость которой равна $O(q(\overline{G}) \cdot 2^{q(\overline{G})})$.

Прежде, чем приступить к перебору всех возможных комбинаций максимальных паросочетаний в блоках, проведём предварительные подсчёты, которые позволят ускорить вычисления значений целевой функции в процессе перебора. Условимся называть *контактом между блоком j и блоком $j' \neq j$* (или *особым ребром*) пару вершин $(i, i+1)$ левой доли графа $\overline{G}$, причём одна из этих двух вершин принадлежит блоку j, а другая — блоку j' (или особому ребру). *Контактом внутри блока* будем называть пару вершин левой доли блока, если их порядковые номера отличаются на единицу. Для каждого блока $j = 1, \ldots, q(\overline{G})$ проверим наличие контактов внутри блока j, между j и всеми особыми рёбрами, а также между j и каждым другим блоком. Трудоёмкость проверки всех вершин в левой доле одного блока на наличие контактов равна $O(k)$.

Каждое из двух максимальных паросочетаний $w^{0,j}$ и $w^{1,j}$ блока j определяет некоторую дугу графа G для любых контакта $(i, i+1)$ внутри данного блока и контакта блока j с особыми рёбрами. Просуммируем веса дуг графа G по контактам внутри этого блока и по всем контактам блока j с особыми рёбрами и обозначим полученные величины для паросочетаний $w^{0,j}$ и $w^{1,j}$ через P_j^0 и P_j^1 соответственно. Если блок j контактирует с $j' \neq j$, то каждая комбинация максимальных паросочетаний данных блоков определяет некоторую дугу графа G для любого контакта $(i, i+1)$ между этими блоками. Просуммируем веса дуг графа G по всем контактам между блоками j и j' и обозначим четыре полученные величины через $P_{jj'}^{(0,0)}$, $P_{jj'}^{(0,1)}$, $P_{jj'}^{(1,0)}$ и $P_{jj'}^{(1,1)}$ соответственно.

Вышеуказанные величины вычисляются для каждого блока, поэтому общая трудоёмкость предварительной процедуры равна $O(k \cdot q(\overline{G}))$.

После этого перебор всех возможных комбинаций максимальных паросочетаний в блоках осуществляется с помощью кода Грея (см., например, [8]) таким образом, что каждая следующая комбинация отличается от предыдущей заменой максимального паросочетания только в одном из блоков. Пусть двоичный вектор $\delta = (\delta_1, \ldots, \delta_{q(\overline{G})})$ определяет назначение максимальных паросочетаний в блоках, а именно, $\delta_j = 0$, если в блоке j

выбрано паросочетание $w^{0,j}$, и $\delta_j = 1$, если в блоке j выбрано паросочетание $w^{1,j}$. Таким образом, каждому вектору δ взаимно однозначно сопоставляется допустимое решение f_δ задачи I.

Если осуществляется переход от вектора $\overline{\delta}$ к вектору δ, при котором изменяется паросочетание в блоке j, то значение целевой функции $\rho(f_\delta)$ вычисляется через значение целевой функции $\rho(f_{\overline{\delta}})$ по формуле

$$\rho(f_\delta) = \rho(f_{\overline{\delta}}) - P_j^{\overline{\delta}_j} + P_j^{\delta_j} - \sum_{j' \in A(j)} P_{jj'}^{(\overline{\delta}_j, \overline{\delta}_{j'})} + \sum_{j' \in A(j)} P_{jj'}^{(\delta_j, \delta_{j'})},$$

где $A(j)$ — множество блоков, контактирующих с j. Поскольку $|A(j)| \leqslant q(\overline{G})$, пересчёт целевой функции требует времени $O(q(\overline{G}))$ и общая трудоёмкость представленной модификации алгоритма решения задачи I равна $O(q(\overline{G}) \cdot 2^{q(\overline{G})})$.

Таким образом, задача оптимальной рекомбинации (i)–(ii), как и задача I, может быть решена за время $O(q(\overline{G}) \cdot 2^{q(\overline{G})})$. Однако, как показано ниже, для «почти всех» пар родительских решений выполняется неравенство $q(\overline{G}) \leqslant 1{,}1 \cdot \ln(k)$, т. е. «почти все» индивидуальные задачи оптимальной рекомбинации (i)–(ii) могут быть решены за время $O(k \cdot \ln(k))$ и, кроме того, имеют не более k допустимых решений.

Определение 1 [9]. Граф $\overline{G} = (X_k, X, \overline{U})$ назовём *хорошим*, если $q(\overline{G}) \leqslant 1{,}1 \cdot \ln(k)$, и *плохим* — в противном случае.

Определение 2. Пару родительских решений π^1 и π^2 в задаче оптимальной рекомбинации (i)–(ii) назовём *хорошей*, если соответствующий ей граф $\overline{G} = (X_k, X, \overline{U})$ хороший, и *плохой* — в противном случае.

Заметим, что вместо константы 1,1 в определении 1 можно было бы выбрать любую другую, равную $1 + \varepsilon$, где $\varepsilon \in (0, \log_2(e) - 1]$. При таком выборе константы ε обеспечивается разрешимость задачи оптимальной рекомбинации за время $O(k \cdot \ln(k))$.

Обозначим через $\overline{\Im}_k$ ($\overline{\Re}_k$) множество хороших графов (множество хороших пар родительских решений), $\widetilde{\Im}_k$ ($\widetilde{\Re}_k$) — множество плохих графов (множество плохих пар родительских решений). Положим $\Im_k = \overline{\Im}_k \cup \widetilde{\Im}_k$, $\Re_k = \overline{\Re}_k \cup \widetilde{\Re}_k$.

Введём следующие вспомогательные обозначения:

S_l — множество подстановок множества $\{1, \ldots, l\}$, не содержащих циклов единичной длины;

$\overline{S}_l$ — множество подстановок из S_l с числом циклов, не превосходящим $1{,}1 \cdot \ln(l)$.

Положим $\widetilde{S}_l = S_l \backslash \overline{S}_l$. Из результатов [9] следует

Утверждение 2. $|\widetilde{S}_l|/|\overline{S}_l| \longrightarrow 0$ *при* $l \to \infty$.

Теорема 3. $|\overline{\Re}_k|/|\Re_k| \longrightarrow 1$ *при* $k \to \infty$.

Доказательство. 1. Аналогично [9] оценим величины $|\overline{\Im}_k|$ и $|\widetilde{\Im}_k|$. Для этого поставим в соответствие произвольной подстановке $\sigma \in S_l$, $l \leqslant k$, множество двудольных графов $\Im_k(\sigma) \subset \Im_k$ следующим образом. Выделим произвольные $k-l$ рёбер в качестве особых. Непомеченые вершины $\{i_1, i_2, \ldots, i_l\} \subset X_k$ левой доли, где $i_j < i_{j+1}$, $j = 1, \ldots, l-1$, разобьём на $\xi(\sigma)$ блоков, где $\xi(\sigma)$ — число циклов в подстановке σ, таким образом, чтобы вершины с номерами $\{i_{t_1}, i_{t_2}, \ldots, i_{t_r}\}$ принадлежали одному блоку в том и только том случае, когда элементы $(t_1, t_2, \ldots, t_r)$ составляют цикл в подстановке σ. В этом случае потребуем, чтобы для любой пары вершин i_{t_j}, $i_{t_{j+1}}$ существовала смежная с ними вершина из правой правой доли X графа, $j = 1, \ldots, r$, $i_{t_{r+1}} = i_{t_1}$.

Рассмотрим подстановку $\sigma = \binom{1\ 2\ 3\ 4\ 5}{2\ 3\ 1\ 5\ 4} \in S_5$, содержащую циклы $c_1 = (1, 2, 3)$ и $c_2 = (4, 5)$. Два примера графов из класса $\Im_7(\sigma)$ приведены на рис. 2. Здесь блок j соответствует циклу c_j, $j = 1, 2$.

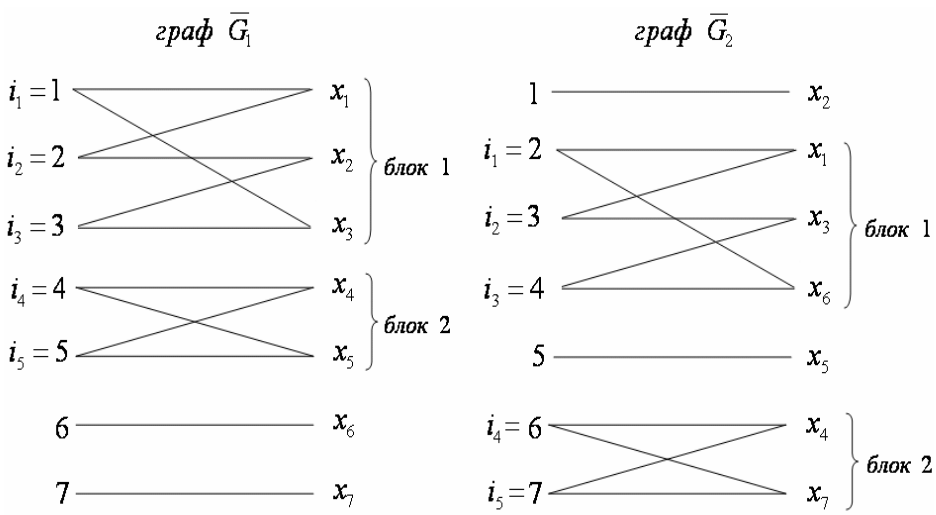

Рис. 2. Примеры графов из класса $\Im_7(\sigma)$, где $\sigma = \binom{1\ 2\ 3\ 4\ 5}{2\ 3\ 1\ 5\ 4} \in S_5$

Существует $k!$ способов выбора соответствия между вершинами левой и правой долей, поэтому количество различных графов из класса $\Im_k(\sigma)$, $\sigma \in S_l$, $l \leqslant k$, равно $|\Im_k(\sigma)| = C_k^l \frac{k!}{2^{\xi_1(\sigma)}}$, где $\xi_1(\sigma)$ — число циклов длины два в подстановке σ. Деление на $2^{\xi_1(\sigma)}$ обусловлено тем, что для любого блока, соответствующего циклу длины два в подстановке σ, существует два эквивалентных способа нумерации вершин его правой доли.

Пусть $\sigma = c_1 c_2 \ldots c_{\xi(\sigma)}$ — подстановка из множества S_l, представленная в виде циклов c_i, $i = 1, \ldots, \xi(\sigma)$, и c_j — произвольный цикл подстановки σ, содержащий не менее трёх элементов, $1 \leqslant j \leqslant \xi(\sigma)$. Преобразуем подстановку σ в σ^1 путём замены цикла c_j обратным:

$$\sigma^1 = c_1 c_2 \ldots c_{j-1} c_j^{-1} c_{j+1} \ldots c_{\xi(\sigma)}. \tag{1}$$

Ясно, что σ^1 порождает то же множество графов из класса $\Im_k$, что и σ. Таким образом, для любых двух подстановок σ^1 и σ^2 из множества S_l, $l \leqslant k$, порождаемые ими множества графов из класса $\Im_k$ совпадают, если одну из них можно получить из другой путём применения нескольких преобразований вида (1), и не пересекаются — в противном случае. Кроме того, $\Im_k(\sigma^1) \cap \Im_k(\sigma^2) = \varnothing$, если $\sigma^1 \in S_{l_1}$, $\sigma^2 \in S_{l_2}$, $l_1 \neq l_2$.

Если $\sigma \in \overline{S}_l$, $l \leqslant k$, то $\Im_k(\sigma) \subseteq \overline{\Im}_k$, а если $\sigma \in \widetilde{S}_l$, то при $l < k$ не исключается возможность того, что $\Im_k(\sigma) \subseteq \overline{\Im}_k$. С учётом вышесказанного имеем

$$|\overline{\Im}_k| \geqslant \sum_{l=2}^{k} \sum_{\sigma \in \overline{S}_l} C_k^l \frac{k!}{2^{\xi_1(\sigma)} 2^{\xi(\sigma)-\xi_1(\sigma)}} = \sum_{l=2}^{k} \sum_{\sigma \in \widetilde{S}_l} C_k^l \frac{k!}{2^{\xi(\sigma)}}, \tag{2}$$

$$|\widetilde{\Im}_k| \leqslant \sum_{l=\lfloor 1{,}1 \cdot \ln(k) \rfloor}^{k} \sum_{\sigma \in \widetilde{S}_l} C_k^l \frac{k!}{2^{\xi_1(\sigma)} 2^{\xi(\sigma)-\xi_1(\sigma)}} = \sum_{l=\lfloor 1{,}1 \cdot \ln(k) \rfloor}^{k} \sum_{\sigma \in \widetilde{S}_l} C_k^l \frac{k!}{2^{\xi(\sigma)}}. \tag{3}$$

2. Оценим величины $|\overline{\Re}_k|$, $|\widetilde{\Re}_k|$ и докажем справедливость утверждения теоремы. Напомним, что каждый граф $\overline{G} \in \Im_k(\sigma)$, $\sigma \in S_l$, $l \leqslant k$, имеет $\xi(\sigma)$ блоков. Обозначим через $w^j = \{(i_1, x^{i_1}), (i_2, x^{i_2}), \ldots, (i_{m_j}, x^{i_{m_j}})\}$, $\overline{w}^j = \{(i_1, \overline{x}^{i_1}), (i_2, \overline{x}^{i_2}), \ldots, (i_{m_j}, \overline{x}^{i_{m_j}})\}$ максимальные паросочетания, на которые разбиваются рёбра блока j графа $\overline{G}$, $j = 1, \ldots, \xi(\sigma)$. Тогда в любой задаче оптимальной рекомбинации (i)–(ii), порождающей граф $\overline{G}$, либо $\pi^1_{i_m} = x^{i_m}$, $\pi^2_{i_m} = \overline{x}^{i_m}$, $m = 1, \ldots, m_j$, либо $\pi^1_{i_m} = \overline{x}^{i_m}$, $\pi^2_{i_m} = x^{i_m}$, $m = 1, \ldots, m_j$, для всех $j = 1, \ldots, \xi(\sigma)$. Следовательно, каждый двудольный граф из класса $\Im_k(\sigma)$ соответствует $2^{\xi(\sigma)}$ парам родительских решений в задаче оптимальной рекомбинации (i)–(ii) (пары родителей $\pi^1 = a$, $\pi^2 = b$ и $\pi^1 = b$, $\pi^2 = a$ считаются различными). Учитывая (2) и (3), получаем

$$|\overline{\Re}_k| \geqslant \sum_{l=2}^{k} \sum_{\sigma \in \overline{S}_l} C_k^l \frac{k!}{2^{\xi(\sigma)}} 2^{\xi(\sigma)} \geqslant \sum_{l=\lfloor 1{,}1 \cdot \ln(k) \rfloor}^{k} |\overline{S}_l| C_k^l k!, \tag{4}$$

$$|\widetilde{\Re}_k| \leqslant \sum_{l=\lfloor 1,1\cdot\ln(k)\rfloor}^{k} \sum_{\sigma\in\widetilde{S}_l} C_k^l \frac{k!}{2^{\xi(\sigma)}} 2^{\xi(\sigma)} = \sum_{l=\lfloor 1,1\cdot\ln(k)\rfloor}^{k} |\widetilde{S}_l| C_k^l k!. \tag{5}$$

Полагая $\psi(k) = \max\limits_{l=\lfloor 1,1\cdot\ln(k)\rfloor,\ldots,k} |\widetilde{S}_l|/|\overline{S}_l|$, в силу (4), (5) и утверждения 2 имеем

$$|\widetilde{\Re}_k|/|\overline{\Re}_k| \leqslant \psi(k) \to 0 \quad \text{при } k\to\infty. \tag{6}$$

Справедливость утверждения теоремы следует из (6). Теорема 3 доказана.

3. Заключение

Полученные результаты показывают, что задача оптимальной рекомбинации для $1|s_{vu}|C_{\max}$ NP-трудна в сильном смысле, однако для получения оптимального решения-потомка существует более быстрый алгоритм, чем модификация известного метода динамического программирования решения задачи коммивояжёра [17]. Кроме того, трудоёмкость данного алгоритма полиномиальна для «почти всех» пар родительских решений.

Отметим универсальность представленного алгоритма в том смысле, что он может использоваться не только при целевой функции, минимизирующей общий момент завершения выполнения работ, но и при других критериях (см. примеры в [10, 16, 19]).

Что же касается NP-трудности исследуемой оптимальной рекомбинации, то из неё следует NP-трудность оптимальной рекомбинации подобного типа для более общих задач составления расписаний, когда имеется несколько устройств и каждая работа может быть выполнена различными способами с использованием одного или нескольких устройств [2, 5, 13]. Для дальнейшего анализа представляет интерес экспериментальное исследование предложенного оператора оптимальной рекомбинации и его обобщений в составе генетических алгоритмов для задач составления расписаний с переналадками.

Авторы благодарны рецензенту и А. В. Кононову за полезные замечания.

ЛИТЕРАТУРА

1. **Берж К.** Теория графов и её применения. — М.: Изд-во иностр. лит-ры, 1962. — 319 с.

2. **Борисовский П. А.** Генетический алгоритм для одной задачи составления производственного расписания с переналадками // Тр. XIV Байк. междунар. шк.-семинара «Методы оптимизации и их приложения». Т. 4. — Иркутск: ИСЭМ СО РАН, 2008. — С. 166–173.
3. **Гэри М., Джонсон Д.** Вычислительные машины и труднорешаемые задачи. — М.: Мир, 1982. — 416 с.
4. **Еремеев А. В.** О сложности оптимальной рекомбинации для задачи коммивояжёра // Дискрет. анализ и исслед. операций. — 2011. — Т. 18, № 1. — С. 27–40.
5. **Еремеев А. В., Коваленко Ю. В.** О задаче составления расписаний с группировкой машин по технологиям // Дискрет. анализ и исслед. операций. — 2011. — Т. 18, № 5. — С. 54–79.
6. **Карп Р. М.** Сводимость комбинаторных проблем // Кибернет. сб. — М.: Мир, 1975. — Вып. 12. — С. 16–38.
7. **Кормен Т., Лейзерсон Ч., Ривест Р.** Алгоритмы: построение и анализ. — М.: МЦИМО, 2001. — 960 с.
8. **Рейнгольд Э., Нивергельт Ю., Део Н.** Комбинаторные алгоритмы, теория и практика. — М.: Мир, 1980. — 476 с.
9. **Сердюков А. И.** О задаче коммивояжёра при наличии запретов // Управляемые системы. — 1978. — Вып. 17. — С. 80–86.
10. **Танаев В. С., Ковалев М. Я., Шафранский Я. М.** Теория расписаний. Групповые технологии. — Минск: Ин-т техн. кибернетики НАН Беларуси, 1998. — 290 с.
11. **Cook W., Seymour P.** Tour merging via branch-decomposition // INFORMS J. Comput. — 2003. — Vol. 15, N 2. — P. 233–248.
12. **Cotta C., Alba E., Troya J. M.** Utilizing dynastically optimal forma recombination in hybrid genetic algorithms // Proc. 5th Int. Conf. Parallel Problem Solving from Nature. — Berlin: Springer-Verl., 1998. — P. 305–314. (Lect. Notes Comput. Sci.; Vol. 1498.)
13. **Dolgui A., Eremeev A. V., Kovalyov M. Y.** Multi-product lot-sizing and scheduling on unrelated parallel machines // Res. Rep. N 2007–500–011. – St. Etienne: Ecole des Mines de St. Etienne, 2007. — 15 p.
14. **Eremeev A. V.** On complexity of optimal recombination for binary representations of solutions // Evolutionary Comput. — 2008. — Vol. 16, N 1. — P. 127–147.
15. **Graham R. L., Lawler E. L., Lenstra J. K., Rinnooy Kan A. H. G.** Optimization and approximation in deterministic sequencing and scheduling: a survey // Ann. Discrete Math. — 1979. — Vol. 5. — P. 287–326.
16. **Hazir Ö., Günalay Y., Erel E.** Customer order scheduling problem: a comparative metaheuristics study // Int. J. Adv. Manuf. Technology. — 2008. — Vol. 37. — P. 589–598.
17. **Held M., Karp R. M.** A dynamic programming approach to sequencing problems // SIAM J. Appl. Math. — 1962. — Vol. 10. — P. 196–210.

18. **Reeves C. R.** Genetic algorithms for the operations researcher // INFORMS J. Comput. — 1997. — Vol. 9, N 3. — P. 231–250.
19. **Yagiura M., Ibaraki T.** The use of dynamic programming in genetic algorithms for permutation problems // Eur. J. Oper. Res. — 1996. — Vol. 92. — P. 387–401.

Еремеев Антон Валентинович,
e-mail: eremeev@ofim.oscsbras.ru

Коваленко Юлия Викторовна,
e-mail: juliakoval86@mail.ru

Статья поступила
19 июля 2011 г.

Переработанный вариант —
17 ноября 2011 г.