

О сложности оптимальной рекомбинации для одной задачи составления расписаний с переналадками*

А.В. Еремеев, Ю.В. Коваленко

19 октября 2012 г.

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 ее решения. Показано, что трудоемкость 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 $V = \{v_1, \dots, v_k\}$, которые должны быть выполнены на единственном имеющемся устройстве. Каждая работа $v \in V$ характеризуется длительностью $p_v \in \mathbb{R}^+$ (здесь \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 $v \neq u$ (здесь и далее \mathbb{R}_+ – множество неотрицательных вещественных чисел). Требуется составить расписание, минимизирующее время завершения всех работ.

Обозначим через $\pi = (\pi_1, \dots, \pi_k)$ перестановку, определяющую порядок выполнения работ, а именно, π_i – работа, выполняемая i -той по счету. Пусть $s(\pi) = \sum_{i=1}^{k-1} s_{\pi_i, \pi_{i+1}}$. Тогда задача эквивалентна поиску такой перестановки π^* , при которой минимизируется суммарная длительность переналадки $s(\pi^*)$.

В соответствии с известными обозначениями [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].

*Это препринт Статьи, принятой для опубликования в журнале «Дискретный анализ и исследование операций» (© Сибирское отделение РАН, 2012 (год), © Институт математики им. С.Л. Соболева СО РАН, 2012 (год)).

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

- 1) $\pi'_i = \pi_i^1$ или $\pi'_i = \pi_i^2$ для всех $i = 1, \dots, k$;
- 2) π' имеет минимальное значение целевой функции $s(\pi')$ среди всех перестановок, удовлетворяющих условию 1).

Поиск такой перестановки представляет собой задачу *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 работе С. Cotta, Е. Alba и J.M. Troya [12] изучался вопрос о применимости оптимальной рекомбинации исследуемого типа к задачам комбинаторной оптимизации при различных способах представления решений в генетическом алгоритме. В частности, экспериментальным путем было показано преимущество использования такой оптимальной рекомбинации, по сравнению с другими известными операторами кроссинговера, в составе генетических алгоритмов для задачи 3-ВЫПОЛНИМОСТЬ и задачи 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 работах [4, 11, 19].

Статья построена следующим образом. В § 1 с использованием результатов А.И. Сердюкова [9] показана NP-трудность в сильном смысле представленной задачи оптимальной рекомбинации. В § 2 предложен алгоритм решения исследуемой задачи, основанный на переборе совершенных паросочетаний в специальном двудольном графе [9]. Доказано, что для «почти всех» пар родительских решений трудоемкость данного алгоритма равна $O(k \cdot \ln(k))$. В последнем параграфе 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 $X = \{x_1, \dots, x_n\}$ – множество вершин, $U = \{(x, y) : x, y \in X, x \neq y\}$ – множество дуг, которым приписаны веса $\rho(x, y) \in \mathbb{R}_+$, $(x, y) \in U$. Также задана система подмножеств (предписаний) $X^i \subseteq X$, $i = 1, \dots, n$, удовлетворяющая условиям:

C1: $|X^i| \leq 2$ для всех $i = 1, \dots, n$;

C2: $1 \leq |\{i : x \in X^i, i = 1, \dots, n\}| \leq 2$ для всех $x \in X$;

C3: 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 $X_n = \{1, \dots, n\}$ в X таких, что $f(i) \in X^i$, $i = 1, \dots, n$, для любого $f \in F$. Требуется найти такое отображение $f^* \in F$, что $\rho(f^*) = \min_{f \in F} \rho(f)$, где $\rho(f) = \sum_{i=1}^{n-1} \rho(f(i), f(i+1))$ 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 $\bar{G} = (X_n, X, \bar{U})$ с равными по мощности долями

вершин X_n , X и множеством ребер $\bar{U} = \{(i, x) : i \in X_n, x \in X^i\}$. Заметим, что если степень вершины $i \in X_n$ в графе \bar{G} равна d ($1 \leq d \leq 2$), 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 для любого $Y \subseteq X_n$ имеет место неравенство $|Y| \leq |\{x \in X : x \in X^i, i \in Y\}|$.

Кроме того, совершенное паросочетание $W = \{(1, x^1), (2, x^2), \dots, (n, x^n)\} \subseteq \bar{U}$ может быть найдено за полиномиальное время, например, с помощью алгоритма Кёнига-Холла [1]. Полагая $f^1(i) = x^i$, $i = 1, \dots, n$, получаем допустимое решение задачи I .

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

Перейдем к анализу вычислительной сложности задачи оптимальной рекомбинации 1) – 2) 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 , $i = 1, \dots, n$. Пусть число работ $k = n$, длительность переналадки $s_{xy} = \rho(x, y)$ 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 NP-трудности в сильном смысле задачи I будет следовать NP-трудность в сильном смысле задачи оптимальной рекомбинации 1) – 2).

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

$\sum_{i=1}^{n-1} \rho(f(i), f(i+1)) + \rho(f(n), f(1))$ для всех $f \in F$. Обозначим эту задачу через \tilde{I} . Докажем NP-трудность в сильном смысле задачи I с помощью сводимости по Тьюрингу к ней задачи \tilde{I} .

Утверждение 1. *Задача I является NP-трудной в сильном смысле.*

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

1. Если найдется вершина $x \in X$ такая, что $|\{i \in X_n : x \in X^i\}| = 0$, то задача \tilde{I} неразрешима и преобразования окончены.

2. 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, ее содержащих. Если после этого найдется j такое, что $|X^j| = 0$, то задача \tilde{I} неразрешима и преобразования окончены. В противном случае удаляем вершину x и подмножество X^i из дальнейшего рассмотрения в процессе преобразования.

3. 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 \tilde{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 удаленные из рассмотрения одноэлементные подмножества образуют систему предписаний, эквивалентную исходной и удовлетворяющую условиям $C1 - C3$. Далее будем предполагать, что система предписаний задачи \tilde{I} приведена к такому виду.

Теперь сведем по Тьюрингу задачу \tilde{I} к задаче I . Пусть существует гипотетическая подпрограмма \mathcal{S} решения задачи I с системой предписаний \bar{X}^i , $i = 1, \dots, n$. Построим алгоритм \mathcal{A} , который будет решать задачу \tilde{I} с системой предписаний X^i , $i = 1, \dots, n$, используя не более четырех раз подпрограмму \mathcal{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 $C3$, тогда в алгоритме \mathcal{A} полученная система предписаний преобразуется в эквивалентную ей систему предписаний, но удовлетворяющую условиям $C1 - C3$. Опишем предлагаемый алгоритм по шагам.

Алгоритм \mathcal{A}

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

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

2.1. Полагаем $\tilde{X}^1 = \{x\}$, $\tilde{X}^i = X^i$, $i = 2, \dots, n$, и если $|X^1| = 2$, то преобразуем систему предписаний \tilde{X}^i , $i = 1, \dots, n$, таким образом, чтобы для нее выполнялось условие $C3$. Для этого находим $j \neq 1$ такое, что $\tilde{X}^j = \{x, z\}$, и полагаем $\tilde{X}^j = \{z\}$. Далее выполняем аналогичные действия для вершины z и т. д.

2.2. Для каждой вершины $y \in \tilde{X}^n$ выполняем шаги 2.2.1-2.2.2:

2.2.1. Полагаем $\tilde{X}^n = \{y\}$, $\tilde{X}^i = \tilde{X}^i$, $i = 1, \dots, n - 1$, и если $|\tilde{X}^n| = 2$, то преобразуем систему предписаний \tilde{X}^i , $i = 1, \dots, n$, таким образом, чтобы для нее выполнялось условие $C3$, аналогично шагу 2.1.

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

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

Таким образом, справедлива

Теорема 1. *Задача оптимальной рекомбинации 1) – 2) для $1|s_{vu}|C_{\max}$ NP-трудна в сильном смысле.*

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

Теорема 2. *Задача оптимальной рекомбинации 1) – 2) для $1|s_{vu} = s_{uv}|C_{\max}$ 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 , $i = 1, \dots, k$; положить число вершин $n = k$; веса дуг $\rho(x, y) = s_{xy}$ для всех $x, y \in X$, где $x \neq y$; предписания $X^i = \{\pi_i^1, \pi_i^2\}$, $i = 1, \dots, k$, то множество допустимых решений задачи I будет взаимно однозначно отображаться на множество допустимых решений задачи оптимальной рекомбинации 1) – 2), причем оптимальному решению будет соответствовать оптимальное.

Оптимальное отображение $f^* \in F$ в задаче I можно найти за время $O(2^k)$ путем перебора решений (среди которых будут как допустимые, так и недопустимые), формируемых выбором в каждом подмножестве системы предписаний одного из его элементов. Такую же трудоемкость имеет очевидная модификация алгоритма М. Held и R.M. Кагр [17], известного для задачи коммивояжера. 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 \tilde{I} .

Рассмотрим двудольный граф $\bar{G} = (X_k, X, \bar{U})$, определенный в § 1. Заметим, что между множеством допустимых решений F задачи I и множеством совершенных паросочетаний \mathcal{W} в графе \bar{G} существует взаимно однозначное соответствие.

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

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 данной задаче двудольный граф $\bar{G} = (X_7, X, \bar{U})$, его особые ребра и блоки представлены на рис. 1. Здесь ребра каждого блока, выделенные жирным, образуют одно максимальное паросочетание блока, а остальные ребра блока – второе.

Блоки в графе \bar{G} могут быть вычислены за время $O(k)$, например, с помощью алгоритма «поиск в глубину» [7]. Особые ребра 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 \bar{G} , определяем в нем набор особых ребер и блоков, а также находим максимальные паросочетания в блоках. Перебираем все совершенные паросочетания $W \in \mathcal{W}$ в графе \bar{G} (формируя их из максимальных паросочетаний в блоках и особых ребер). Каждому $W \in \mathcal{W}$ ставим в соответствие $f \in F$ и вычисляем $\rho(f)$. В

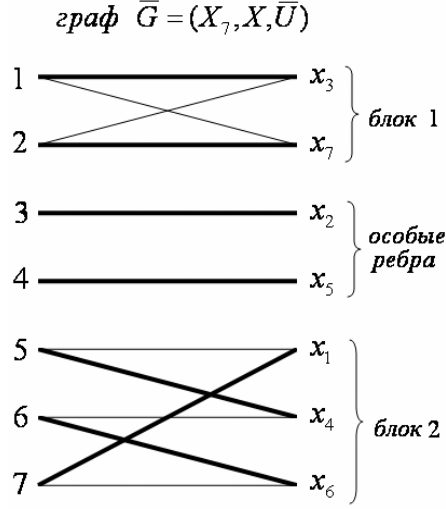


Рис. 1. Пример графа $\bar{G} = (X_7, X, \bar{U})$, содержащего два особых ребра и два блока.

результате находим $f^* \in F$ такое, что $\rho(f^*) = \min_{f \in F} \rho(f)$.

Поскольку $|F| = |\mathcal{W}| = 2^{q(\bar{G})}$, то вычислительная сложность представленного алгоритма равна $O(k \cdot 2^{q(\bar{G})})$, причем $q(\bar{G}) \leq \lfloor \frac{k}{2} \rfloor$ и данная оценка достижима. Рассмотрим модификацию данного алгоритма, трудоемкость которой равна $O(q(\bar{G}) \cdot 2^{q(\bar{G})})$.

Прежде, чем приступить к перебору всех возможных комбинаций максимальных паросочетаний в блоках, проведем предварительные подсчеты, 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 \bar{G} , причем одна из этих двух вершин принадлежит блоку j , а другая – блоку j' (или особому ребру). Контактom внутри блока будем называть пару вершин левой доли блока, если их порядковые номера отличаются на единицу. Для каждого блока j , $j = 1, \dots, q(\bar{G})$, проверим наличие контактов внутри блока j , между блоком j и всеми особыми ребрами, а также между блоком j и каждым другим блоком. Трудоемкость проверки всех вершин в левой доле одного блока на наличие контактов есть $O(k)$.

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

Вышеуказанные величины вычисляются для каждого блока, поэтому общая трудо-

емкость предварительной процедуры есть $O(k \cdot q(\bar{G}))$.

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 $\delta = (\delta_1, \dots, \delta_{q(\bar{G})})$ 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 $\bar{\delta}$ к вектору δ , при котором изменяется паросочетание в блоке j , то значение целевой функции $\rho(f_\delta)$ вычисляется через значение целевой функции $\rho(f_{\bar{\delta}})$ по формуле $\rho(f_\delta) = \rho(f_{\bar{\delta}}) - P_j^{\bar{\delta}_j} + P_j^{\delta_j} - \sum_{j' \in A(j)} P_{jj'}^{(\bar{\delta}_j, \bar{\delta}_{j'})} + \sum_{j' \in A(j)} P_{jj'}^{(\delta_j, \delta_{j'})}$, где $A(j)$ – множество блоков, контактирующих с блоком j . Поскольку $|A(j)| \leq q(\bar{G})$, то пересчет целевой функции требует времени $O(q(\bar{G}))$, и общая трудоемкость представленной модификации алгоритма решения задачи I равна $O(q(\bar{G}) \cdot 2^{q(\bar{G})})$.

Таким образом, задача оптимальной рекомбинации 1) – 2), как и задача I , может быть решена за время $O(q(\bar{G}) \cdot 2^{q(\bar{G})})$. Однако, как будет показано ниже, для «почти всех» пар родительских решений выполняется неравенство $q(\bar{G}) \leq 1.1 \cdot \ln(k)$, т. е. «почти все» индивидуальные задачи оптимальной рекомбинации 1) – 2) могут быть решены за время $O(k \cdot \ln(k))$, и, кроме того, имеют не более чем k допустимых решений.

Определение 1. [9] Граф $\bar{G} = (X_k, X, \bar{U})$ назовем «хорошим», если $q(\bar{G}) \leq 1.1 \cdot \ln(k)$, и «плохим» – в противном случае.

Определение 2. Пару родительских решений π^1 и π^2 в задаче оптимальной рекомбинации 1) – 2) назовем «хорошей», если соответствующий ей граф $\bar{G} = (X_k, X, \bar{U})$ является «хорошим», и «плохой» – в противном случае.

Заметим, что вместо константы 1.1 в определении 1 можно было бы выбрать любую другую константу, равную $1 + \varepsilon$, где $\varepsilon \in (0, \log_2(e) - 1]$. При любом таком выборе константы ε обеспечивается разрешимость задачи оптимальной рекомбинации за время $O(k \cdot \ln(k))$.

Обозначим через $\bar{\mathfrak{R}}_k$ ($\bar{\mathfrak{R}}_k$) множество «хороших» графов (множество «хороших» пар родительских решений), $\tilde{\mathfrak{R}}_k$ ($\tilde{\mathfrak{R}}_k$) – множество «плохих» графов (множество «плохих» пар родительских решений). Положим $\mathfrak{S}_k = \bar{\mathfrak{S}}_k \cup \tilde{\mathfrak{S}}_k$, $\mathfrak{R}_k = \bar{\mathfrak{R}}_k \cup \tilde{\mathfrak{R}}_k$.

Также введем следующие вспомогательные обозначения:

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

\bar{S}_l – множество подстановок из S_l с числом циклов, не превосходящим $1.1 \cdot \ln(l)$.

Положим $\tilde{S}_l = S_l \setminus \bar{S}_l$. Из результатов [9] следует

Утверждение 2. $|\tilde{S}_l|/|S_l| \rightarrow 0$ при $l \rightarrow \infty$.

Используя данное утверждение, докажем следующую теорему.

Теорема 3. $|\bar{\mathfrak{R}}_k|/|\mathfrak{R}_k| \rightarrow 1$ при $k \rightarrow \infty$.

ДОКАЗАТЕЛЬСТВО. 1. Аналогично [9] оценим величины $|\bar{\mathfrak{S}}_k|$ и $|\tilde{\mathfrak{S}}_k|$. Для этого поставим в соответствие произвольной подстановке $\sigma \in S_l$, $l \leq k$, множество двудольных графов $\mathfrak{S}_k(\sigma) \subset \mathfrak{S}_k$ следующим образом. Выделим произвольные $k-l$ ребер в качестве особых. Непомеченные вершины $\{i_1, i_2, \dots, i_l\} \subset X_k$ левой доли, где $i_j < i_{j+1}$, $j = 1, \dots, l-1$, разобьем на $\xi(\sigma)$ блоков, где $\xi(\sigma)$ – число циклов в подстановке σ , таким образом, чтобы вершины с номерами $\{i_{t_1}, i_{t_2}, \dots, i_{t_r}\}$ принадлежали одному блоку в том и только том случае, когда элементы (t_1, t_2, \dots, 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, $j = 1, \dots, r$, $i_{t_{r+1}} = i_{t_1}$.

Рассмотрим подстановку $\sigma = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 2 & 3 & 1 & 5 & 4 \end{pmatrix} \in S_5$, содержащую циклы $c_1 = (1, 2, 3)$ и $c_2 = (4, 5)$. Два примера графов из класса $\mathfrak{S}_7(\sigma)$ приведены на рис. 2. Здесь блок j соответствует циклу c_j , $j = 1, 2$.

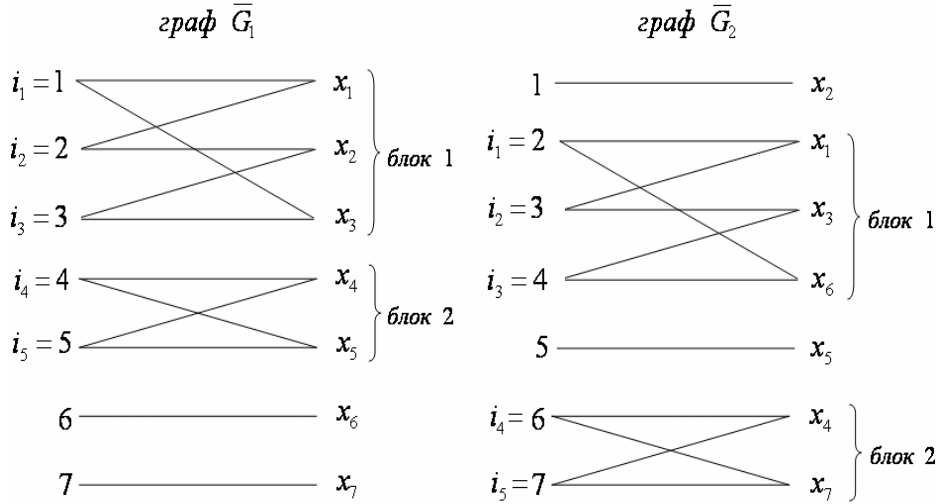


Рис. 2. Примеры графов из класса $\mathfrak{S}_7(\sigma)$, где $\sigma = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 2 & 3 & 1 & 5 & 4 \end{pmatrix} \in S_5$.

Существует $k!$ способов выбора соответствия между вершинами левой и правой долей, поэтому количество различных графов из класса $\mathfrak{S}_k(\sigma)$, $\sigma \in S_l$, $l \leq k$, равно $|\mathfrak{S}_k(\sigma)| = C_k^l \frac{k!}{2^{\xi_1(\sigma)}}$, 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 $\sigma = c_1 c_2 \dots c_{\xi(\sigma)}$ – подстановка из множества S_l , представленная в виде циклов c_i , $i = 1, \dots, \xi(\sigma)$, и c_j – произвольный цикл подстановки σ , содержащий не менее трех элементов, $1 \leq j \leq \xi(\sigma)$. Преобразуем подстановку σ в σ^1 :

$$\sigma^1 = c_1 c_2 \dots c_{j-1} c_j^{-1} c_{j+1} \dots c_{\xi(\sigma)} \quad (1)$$

путем замены цикла c_j на обратный. Ясно, что подстановка σ^1 порождает то же множество графов из класса \mathfrak{S}_k , что и подстановка σ . Таким образом, для любых двух

подстановок σ^1 и σ^2 из множества S_l , $l \leq k$, порождаемые ими множества графов из класса \mathfrak{S}_k совпадают, если одну из них можно получить из другой путем применения нескольких преобразований вида (1), и не пересекаются – в противном случае. Кроме того, $\mathfrak{S}_k(\sigma^1) \cap \mathfrak{S}_k(\sigma^2) = \emptyset$, если $\sigma^1 \in S_{l_1}$, $\sigma^2 \in S_{l_2}$, $l_1 \neq l_2$.

Если $\sigma \in \tilde{S}_l$, $l \leq k$, то $\mathfrak{S}_k(\sigma) \subseteq \mathfrak{S}_k$, а если $\sigma \in \tilde{S}_l$, то при $l < k$ не исключается возможность $\mathfrak{S}_k(\sigma) \subseteq \tilde{\mathfrak{S}}_k$. Тогда, учитывая вышесказанное, имеем:

$$|\tilde{\mathfrak{S}}_k| \geq \sum_{l=2}^k \sum_{\sigma \in \tilde{S}_l} C_k^l \frac{k!}{2^{\xi_1(\sigma)} 2^{\xi(\sigma) - \xi_1(\sigma)}} = \sum_{l=2}^k \sum_{\sigma \in \tilde{S}_l} C_k^l \frac{k!}{2^{\xi(\sigma)}}, \quad (2)$$

$$|\tilde{\mathfrak{S}}_k| \leq \sum_{l=\lfloor 1.1 \cdot \ln(k) \rfloor}^k \sum_{\sigma \in \tilde{S}_l} C_k^l \frac{k!}{2^{\xi_1(\sigma)} 2^{\xi(\sigma) - \xi_1(\sigma)}} = \sum_{l=\lfloor 1.1 \cdot \ln(k) \rfloor}^k \sum_{\sigma \in \tilde{S}_l} C_k^l \frac{k!}{2^{\xi(\sigma)}}. \quad (3)$$

2. Оценим величины $|\bar{\mathfrak{R}}_k|$, $|\tilde{\mathfrak{R}}_k|$, и докажем справедливость утверждения теоремы. Напомним, что каждый граф $\bar{G} \in \mathfrak{S}_k(\sigma)$, $\sigma \in S_l$, $l \leq k$, имеет $\xi(\sigma)$ блоков. Обозначим через $w^j = \{(i_1, x^{i_1}), (i_2, x^{i_2}), \dots, (i_{m_j}, x^{i_{m_j}})\}$, $\bar{w}^j = \{(i_1, \bar{x}^{i_1}), (i_2, \bar{x}^{i_2}), \dots, (i_{m_j}, \bar{x}^{i_{m_j}})\}$ максимальные паросочетания, на которые разбиваются ребра блока j графа \bar{G} , $j = 1, \dots, \xi(\sigma)$. Тогда в любой задаче оптимальной рекомбинации 1) – 2), порождающей граф \bar{G} , либо $\pi_{i_m}^1 = x^{i_m}$, $\pi_{i_m}^2 = \bar{x}^{i_m}$, $m = 1, \dots, m_j$, либо $\pi_{i_m}^1 = \bar{x}^{i_m}$, $\pi_{i_m}^2 = x^{i_m}$, $m = 1, \dots, m_j$, для всех $j = 1, \dots, \xi(\sigma)$. Следовательно, каждый двудольный граф из класса $\mathfrak{S}_k(\sigma)$ соответствует $2^{\xi(\sigma)}$ парам родительских решений в задаче оптимальной рекомбинации 1) – 2) (пары родителей $\pi^1 = a$, $\pi^2 = b$ и $\pi^1 = b$, $\pi^2 = a$ считаются различными), тогда, учитывая (2) и (3), получаем:

$$|\bar{\mathfrak{R}}_k| \geq \sum_{l=2}^k \sum_{\sigma \in \tilde{S}_l} C_k^l \frac{k!}{2^{\xi(\sigma)}} 2^{\xi(\sigma)} \geq \sum_{l=\lfloor 1.1 \cdot \ln(k) \rfloor}^k |\tilde{S}_l| C_k^l k!, \quad (4)$$

$$|\tilde{\mathfrak{R}}_k| \leq \sum_{l=\lfloor 1.1 \cdot \ln(k) \rfloor}^k \sum_{\sigma \in \tilde{S}_l} C_k^l \frac{k!}{2^{\xi(\sigma)}} 2^{\xi(\sigma)} = \sum_{l=\lfloor 1.1 \cdot \ln(k) \rfloor}^k |\tilde{S}_l| C_k^l k!. \quad (5)$$

Теперь, полагая $\psi(k) = \max_{l=\lfloor 1.1 \cdot \ln(k) \rfloor, \dots, k} |\tilde{S}_l| / |\bar{S}_l|$ и учитывая (4), (5) и утверждение 2, имеем

$$|\tilde{\mathfrak{R}}_k| / |\bar{\mathfrak{R}}_k| \leq \psi(k) \rightarrow 0 \text{ при } k \rightarrow \infty. \quad (6)$$

Далее 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 коммивояжера [17]. Кроме того, трудоемкость данного алгоритма полиномиальна для «почти всех» пар родительских решений.

Также 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 нее 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, используя при этом одно или несколько 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 благодарят анонимного рецензента и А.В. Кононова за полезные замечания.

ЛИТЕРАТУРА

- [1] **Берж К.** Теория графов и ее приложение. – М.: Изд-во иност. лит-ры, 1962. – 319 с.
- [2] **Борисовский П.А.** Генетический алгоритм для одной задачи составления производственного расписания с переналадками // Тр. XIV Байкальской международной школы-семинара «Методы оптимизации и их приложения». – Иркутск: ИСЭМ СО РАН, 2008. Т. 4. 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 коммивояжера // Дискретный анализ и исследование 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 // Кибернетический сборник. – 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 коммивояжера при наличии запретов // Управляемые системы, ИМ СО АН СССР. 1978. Вып. 17. С. 80 – 86.
- [10] **Танаев В.С., Ковалев М.Я., Шафранский Я.М.** Теория расписаний. Групповые технологии. – Мн.: Институт технической кибернетики НАН Беларуси, 1998. – 290 с.

- [11] **Cook W., Seymour P.** Tour merging via branch-decomposition // *INFORMS Journal on Computing*, 2003. V. 15, N 2. P. 233 – 248.
- [12] **Cotta C., Alba E., Troya J.M.** Utilizing dynastically optimal forma recombination in hybrid genetic algorithms // *Proceedings of the 5th International Conference on Parallel Problem Solving from Nature. Lecture Notes In Computer Science*, 1998. V. 1498. P. 305 – 314.
- [13] **Dolgui A., Ereemeev A.V., Kovalyov M.Y.** Multi-product lot-sizing and scheduling on unrelated parallel machines // *Research Report No. 2007–500–011.* – Saint-Etienne: Ecole des Mines de Saint-Etienne, 2007. – 15 p.
- [14] **Ereemeev A.V.** On complexity of optimal recombination for binary representations of solutions // *Evolutionary Computation*. 2008. V. 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. V. 5. P. 287 – 326.
- [16] **Hazir Ö., Günalay Y., Erel E.** Customer order scheduling problem: a comparative metaheuristics study // *Int. Journ. of Adv. Manuf. Technol.* 2008. V. 37. P. 589 – 598.
- [17] **Held M., Karp R.M.** A dynamic programming approach to sequencing problems // *SIAM Journal on Applied Mathematics*, 1962. V. 10. P. 196 – 210.
- [18] **Reeves C.R.** Genetic algorithms for the operations researcher // *INFORMS Journ. on Comput.* 1997. V. 9, N 3. P. 231 – 250.
- [19] **Yagiura M., Ibaraki T.** The use of dynamic programming in genetic algorithms for permutation problems // *European Journal of Operational Research*. 1996. V. 92. P. 387 – 401.

On Complexity of Optimal Recombination for one Scheduling Problem with Setup Times

A.V. Ereemeev, J.V. Kovalenko

Abstract. Computational complexity of optimal recombination for one scheduling problem with setup times is considered. Strong NP-hardness of this optimal recombination problem is proven and an algorithm for its solution is proposed. The algorithm is shown to be polynomial for «almost all» instances of the optimal recombination problem.

Keywords: scheduling, setup time, genetic algorithm, optimal recombination.