Санкт-Петербургский государственный университет
Прикладная математика и информатика
Исследование операций и принятие решений в задачах оптимизации, управления и экономики

Пьянков Роман Вадимович

Задачи составления расписаний с временами поступления и директивными сроками

Выпускная квалификационная работа

Научный руководитель:
к. ф.-м. н., доцент Н. С. Григорьева

Рецензент:
Старший преподаватель
А. С. Зациорский

Санкт-Петербург
2017

Saint Petersburg State University
Applied Mathematics and Computer Science
Operation Research and Decision Making in Optimisation, Control and Economics Problems

Pyankov Roman Vadimovich

Scheduling problem with receipt time and due dates

Graduation Project

Scientific Supervisor:
PhD, Associate professor
N. S. Grigoryeva

Reviewer:
Senior Lecturer A. S. Zaciorsky

Saint Petersburg
2017

Оглавление

Введение

В данной работе рассматривается задача составления расписания на одном процессоре, где каждая работа имеет время выпуска, время обработки и время доставки. Прерывания во время обработки запрещены. Каждая работа должна начинать свое выполнение не раньше момента наступления. Процесс доставки начинается сразу после завершения выполнения.

Целью является минимизация максимального времени завершения всех работ.

Данная задача является NP–трудной, в связи с чем актуальным становится применение приближенных алгоритмов.

В главе 1 приводится формальная постановка задачи.

В главе 2 рассматриваются аппроксимационные алгоритмы с гарантированной оценкой точности, а также вводятся основные понятия, необходимые для их понимания.

В главе 3 представлены результаты вычислительных экспериментов.

Основной целью работы является реализация приближенных алгоритмов с гарантированной оценкой точности, проведение численных экспериментов, а также анализ полученных результатов.

Глава 1

Постановка задачи

Рассмотрим следующую задачу планирования. На одной машине должны быть выполнены n работ. Каждая i-я работа имеет:

- r_i — время поступления, до которого работа не может быть поставлена на выполнение;
- p_i — время выполнения на машине;
- q_i — время доставки. Процесс доставки начинается сразу после того, как работа выполнена.

Через π будем обозначать перестановку из n элементов, задающую последовательность работ на машине.

Рассматриваемой задачей является минимизация значения целевой функции

$$C_{max}(\pi) = \max_{1 \leqslant i \leqslant j \leqslant n} \left(r_{\pi(i)} + \sum_{k=i}^{j} p_{\pi(k)} + q_{\pi(j)} \right).$$

Определение 1.0.1. *Перестановка π^*, минимизирующая $C_{\max}(\pi)$ среди всех перестановок π из n элементов, называется* ***оптимальной.***

Глава 2

Аппроксимационные алгоритмы с гарантированной оценкой точности

2.1. Основные определения

В связи с тем, что построение точных и эффективных алгоритмов проблематично для рассматриваемой задачи, актуальным становится построение и анализ приближенных алгоритмов с гарантированной оценкой точности.

Определение 2.1.1. ***Оценкой точности*** *некоторого алгоритма A будем называть отношение:*

$$\frac{C_{\max}(\pi^A)}{C_{\max}(\pi^*)}.$$

Определение 2.1.2. *Говорят, что алгоритм имеет* ***гарантированную оценку точности*** *const, если*

$$\frac{C_{\max}(\pi^A)}{C_{\max}(\pi^*)} \leqslant const.$$

Определение 2.1.3. ***Путем*** *(i, j) в перестановке π называется последовательность работ $\{\pi(i), \pi(i+1), \ldots, \pi(j)\}$, где $1 \leqslant i \leqslant j \leqslant n$.*

Определение 2.1.4. *Путь (a, b), для которого*

$$a = \min \left\{ i : C_{\max}(\pi) = r_{\pi(i)} + \sum_{k=i}^{b} p_{\pi(k)} + q_{\pi(b)} \right\},$$

называется ***критическим путем*** *в перестановке π.*

Рассмотрим пример задачи, в которой количество работ $n = 7$.

Пример 2.1.1. *Задача с количеством работ, равным 7:*

	1	2	3	4	5	6	7
r_i	10	24	0	7	31	0	0
p_i	9	13	5	16	10	18	9
q_i	15	46	17	8	7	15	18

Рассмотрим некоторую перестановку π. *Путь* $(1, 5)$ *будет являться критическим.*

π	7	3	1	6	2	4	5	$C_{\max}(\pi) = 100$

Определение 2.1.5. *Пусть* (a, b) *— критический путь перестановки* π. *Работа* c, *такая что*

$$c = \pi(k'),\ a \leqslant k' \leqslant b,$$

$$q_{\pi(k')} < q_{\pi(b)}, \quad q_{\pi(k)} \geqslant q_{\pi(b)},$$

где $k = k' + 1, \ldots, b$, *называется* ***интерференционной работой***.

Пример 2.1.2. *Интерференционной работой для примера 2.1.1 будет являться работа* 6.

π	7	3	1	6	2	4	5	$C_{\max}(\pi) = 100$

2.2. Расширенное правило Джексона

Базовым подходом к рассматриваемой задаче является алгоритм Schrage (см. [5]), называемый *расширенным правилом Джексона.*

Расширенное правило Джексона: Всякий раз, когда машина свободна, и одна или более работ доступны для выполнения, выбирается работа с наибольшем временем доставки.

Для введенной ранее задачи (см. 2.1.1) рассмотрим перестановку π^S, полученную с помощью описанного правила.

Пример 2.2.1. *Перестановка π^S для примера 2.1.1:*

π^S	7	3	1	6	2	4	5	$C_{\max}(\pi) = 100$

Теорема 2.2.1 (Kise, см. [2]) *Пусть π^S — перестановка, полученная по расширенному правилу Джексона. Тогда оценка точности значения целевой функции:*

$$\frac{C_{\max}(\pi^S)}{C_{\max}(\pi^*)} < 2.$$

2.3. Алгоритм H

Для рассматриваемой задачи E.Nowicki и C.Smutnicki представили более эффективный алгоритм [1], который в отличии от алгоритма Schrage имеет гарантированную оценку точности $3/2$ и вычислительную сложность порядка $O(n \log n)$. Данный алгоритм, называемый *алгоритмом H*, использует в качестве базовой перестановку, полученную с использованием расширенного правила Джексона. Вычислениия состоят из трех основных шагов.

Шаг 1: Используя расширенное правило Джексона, находим начальную перестановку π^S и интерференционную работу $c \in \pi^S$. Если такая работа c не найдена, то заканчиваем вычисления и полагаем

$$\pi^H := \pi^S.$$

Шаг 2: Находим

- $A = \{i \in N \setminus \{c\} : r_i \leqslant q_i\}$,
- $B = \{i \in N \setminus \{c\} : r_i > q_i\}$,
- перестановку π_A такую, что r_i в π_A не убывают,
- перестановку π_B такую, что q_i в π_B не возрастают.

Положим

$$\pi^{AB} := \pi_A c \pi_B.$$

Шаг 3: Находим $\pi^H \in \{\pi^S, \pi^{AB}\}$ такую, что

- $A = \{i \in N \setminus \{c\} : r_i \leqslant q_i\}$,
- $B = \{i \in N \setminus \{c\} : r_i > q_i\}$,
- перестановку π_A такую, что r_i в π_A не убывают,
- перестановку π_B такую, что q_i в π_B не возрастают.

Положим

$$\pi^{AB} := \pi_A c \pi_B.$$

Теорема 2.3.1 (Lenstra, см. [4]) *Пусть π^H - перестановка, полученная алгоритмом H. Тогда значение целевой функции оценивается неравеством:*

$$\frac{C_{\max}(\pi^H)}{C_{\max}(\pi^*)} \leqslant 3/2.$$

Рассмотрим работу алгоритма на примере 2.1.1.

Пример 2.3.1. *На первом шаге воспользуемся расширенным правилом Джексона и получим перестановку* π^S*:*

π^S	7	3	1	6	2	4	5	$C_{\max}(\pi) = 100$

Путь $(1, 5)$ *будет являться критическим, а работа* 6 *— интерференционной. Далее найдем множества* A *и* B*, а также соответствующие им перестановки:*

$$A = \{1, 2, 3, 4, 7\},\ B = \{5\}.$$

π_A	3	7	4	1	2

π_B	5

Положим $\pi^{AB} := \pi_A c \pi_B$*:*

π^{AB}	3	7	4	1	2	6	5	$C_{\max}(\pi) = 98$

Таким образом, в качестве ответа алгоритма будет выбрана перестановка π^{AB}*.*

Глава 3

Вычислительные эксперименты и результаты

3.1. Генерация тестовой выборки

Для тестирования работы рассматриваемых алгоритмов был написан генератор на языке программирования Python 3.6. Код программы находится в приложении А. Каждый тестовый пример записывается в отдельный файл.

Параметры генератора:

- $n\_samples$ — количество тестовых примеров;
- n_jobs — количество работ в каждом примере;
- $range\_r$ — диапазон генерации r_i;
- $range\_p$ — диапазон генерации p_i;
- $range\_q$ — диапазон генерации q_i.

Параметры задачи r_i, p_i, q_i генерируются в соответствии с нормальным распределением и заданными диапазонами.

3.2. Вычисление оптимального решения

Для нахождения оптимальных решений задач, в которых количество работ n достаточно мало, был реализован наивный алгоритм, перебирающий все возможные перестановки π.

3.3. Результаты вычислений

Результаты каждого тестового примера были получены с помощью алгоритмов, описанных ранее. В колонках таблиц, представленных ниже, отражены доли тестовых примеров, удовлетворяющие заданным оценкам точности.

Эксперимент 1: Рассмотрим следующие параметры задачи:

- $n\_samples = 10000$;
- $n\_jobs = 3$;
- $range\_r = 10$;
- $range\_p = 30$;
- $range\_q = 30$.

Оценка точности	Джексон	Алгоритм H
$C_* = C_A$	68.9%	73%
$C_A/C_* > 1.6$	0.02%	0%
$C_A/C_* > 1.4$	0.03%	0.001%
$C_A/C_* > 1.2$	5%	4.3%
$C_A/C_* > 1.1$	14%	12.2%
$C_A/C_* > 1.05$	22.2%	18.7%

Эксперимент 2: Рассмотрим следующие параметры задачи:

- $n\_samples = 10000$;
- $n\_jobs = 3$;
- $range\_r = 10$;
- $range\_p = 30$;
- $range\_q = 50$.

Оценка точности	Джексон	Алгоритм Н
$C_* = C_A$	57.2%	65.6%
$C_A/C_* > 1.6$	0.01%	0%
$C_A/C_* > 1.4$	0.05%	0%
$C_A/C_* > 1.2$	9.5%	7%
$C_A/C_* > 1.1$	22.8%	17.8%
$C_A/C_* > 1.05$	32.2%	23.3%

Эксперимент 3: Рассмотрим следующие параметры задачи:

- $n\_samples = 10000$;
- $n\_jobs = 3$;
- $range\_r = 10$;
- $range\_p = 30$;
- $range\_q = 70$.

Оценка точности	Джексон	Алгоритм H
$C_* = C_A$	52.1%	59.3%
$C_A/C_* > 1.6$	0.01%	0%
$C_A/C_* > 1.4$	0.2%	0%
$C_A/C_* > 1.2$	10.4%	5.7%
$C_A/C_* > 1.1$	25.9%	18%
$C_A/C_* > 1.05$	36.5%	21%

Эксперимент 4: Рассмотрим следующие параметры задачи:

- $n\_samples = 10000$;
- $n\_jobs = 5$;
- $range\_r = 30$;
- $range\_p = 30$;
- $range\_q = 30$.

Оценка точности	Джексон	Алгоритм Н
$C_* = C_A$	81.5%	87%
$C_A/C_* > 1.6$	0%	0%
$C_A/C_* > 1.4$	0.02%	0%
$C_A/C_* > 1.2$	0.5%	0.02%
$C_A/C_* > 1.1$	3.6%	1.3%
$C_A/C_* > 1.05$	9%	7%

Эксперимент 5: Рассмотрим следующие параметры задачи:

- $n\_samples = 10000$;
- $n\_jobs = 5$;
- $range\_r = 30$;
- $range\_p = 30$;
- $range\_q = 50$.

Оценка точности	Джексон	Алгоритм Н
$C_* = C_A$	68.2%	72.7%
$C_A/C_* > 1.6$	0%	0%
$C_A/C_* > 1.4$	0.01%	0%
$C_A/C_* > 1.2$	1.4%	0.1%
$C_A/C_* > 1.1$	9.3%	8%
$C_A/C_* > 1.05$	19.1%	17.2%

Эксперимент 6: Рассмотрим следующие параметры задачи:

- $n\_samples = 10000$;
- $n\_jobs = 5$;
- $range\_r = 30$;
- $range\_p = 30$;
- $range\_q = 70$.

Оценка точности	Джексон	Алгоритм Н
$C_* = C_A$	66%	75.3%
$C_A/C_* > 1.6$	0%	0%
$C_A/C_* > 1.4$	0%	0%
$C_A/C_* > 1.2$	2.4%	1.2%
$C_A/C_* > 1.1$	14.8%	12.3%
$C_A/C_* > 1.05$	28.6%	24%

Эксперимент 7: Рассмотрим следующие параметры задачи:

- $n\_samples = 10000$;
- $n\_jobs = 7$;
- $range\_r = 50$;
- $range\_p = 30$;
- $range\_q = 30$.

Оценка точности	Джексон	Алгоритм H
$C_* = C_A$	86.7%	93.3%
$C_A/C_* > 1.6$	0%	0%
$C_A/C_* > 1.4$	0%	0%
$C_A/C_* > 1.2$	0.05%	0%
$C_A/C_* > 1.1$	1%	0.002%
$C_A/C_* > 1.05$	3.7%	1.5%

Эксперимент 8: Рассмотрим следующие параметры задачи:

- $n\_samples = 10000$;
- $n\_jobs = 7$;
- $range\_r = 50$;
- $range\_p = 30$;
- $range\_q = 50$.

Оценка точности	Джексон	Алгоритм Н
$C_* = C_A$	71.1%	79.85%
$C_A/C_* > 1.6$	0%	0%
$C_A/C_* > 1.4$	0%	0%
$C_A/C_* > 1.2$	0.2%	0%
$C_A/C_* > 1.1$	3.7%	1%
$C_A/C_* > 1.05$	11.7%	5.9%

Эксперимент 9: Рассмотрим следующие параметры задачи:

- $n\_samples = 10000$;
- $n\_jobs = 7$;
- $range\_r = 50$;
- $range\_p = 30$;
- $range\_q = 30$.

Оценка точности	Джексон	Алгоритм Н
$C_* = C_A$	59.5%	68%
$C_A/C_* > 1.6$	0%	0%
$C_A/C_* > 1.4$	0%	0%
$C_A/C_* > 1.2$	0.48%	0.14%
$C_A/C_* > 1.1$	7.3%	3.4%
$C_A/C_* > 1.05$	19.8%	13%

Заключение

В данной работе были рассмотрены аппроксимационные алгоритмы с гарантированной оценкой точности для задачи с заданными параметрами работ:

– время выпуска, r_i;

– время обработки, p_i;

– время доставки, q_i.

Реализованы программы представленных в работе алгоритмов, а также программа для генерации тестовых примеров.

Вычислительные эксперименты показали, что при увеличении диапазона генерируемых времен доставки, доля совпадающих с оптимальным решением результатов рассматриваемых алгоритмов, уменьшается. Также стоит отметить, что в большей части тестовых примеров аппроксимационные алгоритмы дают решение, близкое к оптимальному.

Все результаты получены с помощью программы, с которой можно ознакомиться по ссылке https://yadi.sk/d/pstEP6aB3JRUdY/2017.

Список литературы

1. Nowicki E., Smutnicki C. An approximation algorithm for a single-machine scheduling problem with release times and delivery times //Discrete Applied Mathematics. North-Holland. 1994. P. 69–79.
2. Kise H., Ibaraki H. Performance analysis of six approximation algorithms for the one-machine maximum lateness scheduling problem with ready times, // J. Oper. Res. Sot. Japan. 1979. V. 22, P. 205–244.
3. Grabowskil J., Nowicki E., Zdralka S. A block approach for single-machine scheduling with release dates and due dates, // European J. Oper. 1986. V. 26, P. 278–285.
4. Lenstra J.K. Sequencing by enumerative methods, // Mathematical Centre Tracts. 1977. V. 69.
5. Schrage L. Obtaining optimal solution to resource constrained network scheduling problem, //Unpublished manuscript. 1971.

Приложение А

Generate samples

```
import numpy as np


n_samples = 10
n_jobs = 10
range_r = 50
range_p = 50
range_q = 50


for i in range(n_samples):
    r = np.random.randint(0, range_r, n_jobs)
    p = np.random.randint(1, range_p, n_jobs)
    q = np.random.randint(1, range_q, n_jobs)
    f_out = open('samples\\sample' + str(i) + '.txt', "w")
    f_out.write(str(n_jobs) + '\n')
    for j in range(n_jobs):
        f_out.write(str(r[j]) + ' ' + str(p[j]) + ' ' +
                    str(q[j]) + '\n')
```