

УДК 519.854.2

## О ГЕНЕРАТОРЕ ИСХОДНЫХ ДАННЫХ ДЛЯ ЗАДАЧИ МИНИМИЗАЦИИ СУММАРНОГО ВЗВЕШЕННОГО ЗАПАЗДЫВАНИЯ

Р.Г. Сабиров, В.Р. Фазылов

### Аннотация

В статье обсуждаются особенности популярного генератора псевдослучайных данных для задачи минимизации суммарного взвешенного запаздывания для одной машины и предлагается модификация генератора, улучшающая его свойства.

**Ключевые слова:** теория расписаний, генератор задач.

### Введение

Задача минимизации суммарного взвешенного запаздывания для одного исполнителя определяется следующим образом (см., например, [1]). Необходимо выполнить набор из  $n$  операций (пронумерованных числами  $1, 2, \dots, n$ ) на одной машине, которая не может одновременно выполнять более одной операции. Операция с номером  $i$  имеет длительность выполнения  $p_i$ , директивный срок окончания выполнения  $d_i$ , штрафной коэффициент за запаздывание  $w_i$  и готова к выполнению в момент времени  $r_i$ . В настоящей статье рассматривается случай  $r_i = 0, i = 1, \dots, n$ . Требуется построить расписание выполнения операций, минимизирующее  $\sum_{i=1}^n w_i T_i$ , где  $T_i = \max\{0, C_i - d_i\}$ ,  $C_i$  – момент завершения выполнения  $i$ -й операции.

Оценка качества алгоритмов решения этой задачи обычно производится экспериментально на множестве псевдослучайных задач, данные которых генерируются специальной программой – *генератором исходных данных*. В настоящей работе исследуются свойства одного из широко применяемых генераторов и предлагается способ его улучшения.

### 1. Генератор исходных данных

В ряде работ (см., например, [1–7]), посвященных исследованию алгоритмов решения задачи минимизации суммарного взвешенного запаздывания для одной машины, применен генератор, основанный на двух параметрах задачи:  $TF$  (*tightness factor*) и  $RDD$  (*range of due dates*), вычисляемых по формулам:

$$TF = 1 - \frac{\sum_{i=1}^n d_i}{n \sum_{i=1}^n p_i}, \quad RDD = \frac{d_{\max} - d_{\min}}{\sum_{i=1}^n p_i},$$

где  $d_{\max} = \max_{1 \leq i \leq n} d_i$ ,  $d_{\min} = \min_{1 \leq i \leq n} d_i$ .

### Алгоритм генерации исходных данных задачи

0. Выбираются количество операций  $n$  и значения  $TF$ ,  $RDD$  из интервала  $[0,1]$ .

1. Из целых чисел интервала  $[1,100]$  случайно (по равномерному закону) выбираются значения  $\{p_i\}_{i=1}^n$ , вычисляется  $P = \sum_{i=1}^n p_i$ .

2. Из целых чисел интервала  $[P(1 - TF - RDD/2), P(1 - TF + RDD/2)]$  выбираются случайно (по равномерному закону) значения  $\{d_i\}_{i=1}^n$ , отрицательные директивные сроки обнуляются.

3. Из целых чисел интервала  $[1,10]$  случайно (по равномерному закону) выбираются значения  $\{w_i\}_{i=1}^n$ .

Как видно из описания алгоритма (п. 2), предварительный выбор  $\{d_i\}_{i=1}^n$  осуществляется из интервала

$$[P(1 - TF - RDD/2), P(1 - TF + RDD/2)],$$

а при  $TF > 0.5$  и  $RDD > 2 - 2 \cdot TF$  левая граница интервала предварительных значений директивных сроков будет отрицательной, значит, в этом случае могут получаться задачи с отрицательными директивными сроками операций.

Очевидно, что при условии  $r_i = 0$ ,  $i = 1, \dots, n$ , задача с отрицательными директивными сроками операций будет эквивалентна задаче, в которой все отрицательные директивные сроки заменены нулями, так как их целевые функции отличаются лишь на константу, равную  $\sum_{i: d_i < 0} w_i |d_i|$ . Но заметим, что параметры  $TF$ ,  $RDD$  у этих задач будут различными.

В известном наборе тестовых задач OR-Library, расположенном по адресу <http://people.brunel.ac.uk/~mastjjb/jeb/info.html>, по-видимому, также используется генератор с положительной срезкой директивных сроков, хотя в описании приводится генератор без срезки.

## 2. Анализ генератора

Как было отмечено выше, описанный генератор используется многими исследователями задачи минимизации суммарного взвешенного запаздывания для одной машины. Обычно для проведения численных экспериментов генерируются серии задач заданной размерности с различными значениями параметров  $TF$ ,  $RDD$ , причем значения  $TF$ ,  $RDD$  табулируются независимо от 0.2 до 1.0 с шагом  $h$ , равным 0.2.

Нами был проведен эксперимент по выявлению соответствия параметров  $TF$ ,  $RDD$  как параметров генератора параметрам  $TF$ ,  $RDD$  как параметрам сгенерированных задач. Для того чтобы отличать  $TF$ ,  $RDD$  генератора и задачи, параметры генератора будем обозначать как  $TF_{\text{ген}}$ ,  $RDD_{\text{ген}}$ , а параметры задачи —  $TF_{\text{пр}}$ ,  $RDD_{\text{пр}}$ .

Будем считать, что задача, сгенерированная при параметрах  $TF_{\text{ген}}$ ,  $RDD_{\text{ген}}$ , соответствует своему классу, если ее параметры  $TF_{\text{пр}}$  и  $RDD_{\text{пр}}$  принадлежат интервалам  $[TF_{\text{ген}} - h/2, TF_{\text{ген}} + h/2]$  и  $[RDD_{\text{ген}} - h/2, RDD_{\text{ген}} + h/2]$  соответственно, где  $h = 0.2$ .

В первом эксперименте мы проверили соответствие параметров генератора параметрам сгенерированных задач в рамках традиционной схемы генерации задач.

Схема эксперимента такова:

1)  $TF_{\text{ген}}$ ,  $RDD_{\text{ген}}$  табулировались с шагом 0.2, начиная от 0.2 до 1.0;

Табл. 1

$TF_{pr} \setminus RDD_{pr}$	0 – 0.1	0.1 – 0.3	0.3 – 0.5
0.0 – 0.1	0 / 0 / 0	0 / 0 / 0	0 / 0 / 0
0.1 – 0.3	0 / 0 / 0	100 / 0 / 0	100 / 0 / 0
0.3 – 0.5	0 / 0 / 0	100 / 0 / 0	100 / 0 / 0
0.5 – 0.7	0 / 0 / 0	100 / 0 / 0	100 / 0 / 0
0.7 – 0.9	0 / 0 / 0	100 / 0 / 0	343 / 0 / 243
0.9 – 1.1	100 / 0 / 100	200 / 100 / 200	57 / 100 / 57

$TF_{pr} \setminus RDD_{pr}$	0.5 – 0.7	0.7 – 0.9	0.9 – 1.1
0.0 – 0.1	0 / 0 / 0	0 / 0 / 0	0 / 0 / 0
0.1 – 0.3	100 / 0 / 0	100 / 0 / 0	100 / 0 / 0
0.3 – 0.5	100 / 0 / 0	101 / 0 / 1	99 / 1 / 0
0.5 – 0.7	101 / 0 / 1	200 / 0 / 100	0 / 100 / 0
0.7 – 0.9	199 / 100 / 199	0 / 100 / 0	0 / 100 / 0
0.9 – 1.1	0 / 100 / 0	0 / 100 / 0	0 / 100 / 0

2) для каждой пары  $TF_{gen}, RDD_{gen}$  было сгенерировано по 100 задач размерности 100 операций;

3) для каждой задачи были вычислены  $TF_{pr}, RDD_{pr}$ .

Результаты экспериментов в табл. 1 приведены в формате  $A/B/C$ . Для каждого класса, соответствующего параметрам  $TF_{gen}, RDD_{gen}$ ,  $A$  означает общее количество сгенерированных задач, оказавшихся в этом классе,  $B$  – количество задач, сгенерированных при соответствующих классу значениях  $TF_{gen}, RDD_{gen}$ , но не попавших в свой класс (ушедшие задачи),  $C$  – количество задач, попавших в класс, но сгенерированных при несоответствующих классу параметрах  $TF_{gen}, RDD_{gen}$  (пришедшие задачи).

Заметим, что классы с  $TF_{pr} \in [0.0, 0.1]$  или с  $RDD_{pr} \in [0.0, 0.1]$  не должны были содержать ни одной задачи, так как для них в эксперименте не было соответствующих параметров  $TF_{gen}, RDD_{gen}$ , но в классе с  $TF_{pr} \in [0.9, 1.1]$  и  $RDD_{pr} \in [0.0, 0.1]$  оказалось 100 задач.

Далее, из табл. 1 видно, что в некоторые классы, имеющие соответствующие параметры  $TF_{gen}, RDD_{gen}$ , не попало ни одной задачи (см., например, класс с  $TF_{gen} = 1.0, RDD_{gen} = 1.0$ , который должен бы соответствовать классу  $TF_{pr} \in [0.9, 1.1], RDD_{pr} \in [0.9, 1.1]$ ). В то же время в некоторых классах наблюдается много задач из «чужих» классов (см., например, класс с  $TF_{gen} = 0.8, RDD_{gen} = 0.4$ , который должен бы соответствовать классу  $TF_{pr} \in [0.7, 0.9], RDD_{pr} \in [0.3, 0.5]$ ). Отметим также классы с  $TF_{gen} = 0.8, RDD_{gen} = 0.6$ , с  $TF_{gen} = 1.0, RDD_{gen} = 0.2$ , с  $TF_{gen} = 1.0, RDD_{gen} = 0.4$ , которые потеряли все «свои» задачи, но приобрели «чужие».

Таким образом, видно, что  $TF_{gen}, RDD_{gen}$  и  $TF_{pr}, RDD_{pr}$  могут не соответствовать друг другу, и поэтому выводы об эффективности методов на этих классах задач, сделанные на основании численных экспериментов, будут некорректными.

Во втором эксперименте проверялось соответствие параметров  $TF_{gen}, RDD_{gen}$  параметрам  $TF_{pr}, RDD_{pr}$  для задач из библиотеки OR-Library.

Схема эксперимента такова:

1) оценивались задачи из набора OR-Library при  $n = 100$ , в которых  $TF_{gen}, RDD_{gen}$  изменяются от 0.2 до 1.0 с шагом 0.2;

Табл. 2

$TF_{pr} \setminus RDD_{pr}$	0.0 – 0.1	0.1 – 0.3	0.3 – 0.5	0.5 – 0.7	0.7 – 0.9	0.9 – 1.1
0.0 – 0.1	0 / 0 / 0	0 / 0 / 0	0 / 0 / 0	0 / 0 / 0	0 / 0 / 0	0 / 0 / 0
0.1 – 0.3	0 / 0 / 0	5 / 0 / 0	5 / 0 / 0	5 / 0 / 0	5 / 0 / 0	5 / 0 / 0
0.3 – 0.5	0 / 0 / 0	5 / 0 / 0	5 / 0 / 0	5 / 0 / 0	5 / 0 / 0	5 / 0 / 0
0.5 – 0.7	0 / 0 / 0	5 / 0 / 0	5 / 0 / 0	5 / 0 / 0	10 / 0 / 5	0 / 5 / 0
0.7 – 0.9	0 / 0 / 0	5 / 0 / 0	16 / 0 / 11	10 / 5 / 10	0 / 5 / 0	0 / 5 / 0
0.9 – 1.0	4 / 0 / 4	11 / 4 / 10	4 / 5 / 4	0 / 5 / 0	0 / 5 / 0	0 / 5 / 0

Табл. 3

$TF_{pr} \setminus RDD_{pr}$	0.0 – 0.2	0.2 – 0.4	0.4 – 0.6	0.6 – 0.8	0.8 – 1.0
0.0 – 0.2	100 / 0 / 0	100 / 0 / 0	100 / 0 / 0	100 / 0 / 0	100 / 0 / 0
0.2 – 0.4	100 / 0 / 0	100 / 0 / 0	100 / 0 / 0	100 / 0 / 0	100 / 0 / 0
0.4 – 0.6	100 / 0 / 0	100 / 0 / 0	100 / 0 / 0	100 / 0 / 0	100 / 0 / 0
0.6 – 0.8	100 / 0 / 0	100 / 0 / 0	100 / 0 / 0	–	–
0.8 – 1.0	100 / 0 / 0	–	–	–	–

- 2) для каждой пары  $TF_{gen}, RDD_{gen}$  набор содержит по 5 задач;
- 3) для каждой задачи были вычислены  $TF_{pr}, RDD_{pr}$ ;
- 4) считалось, что  $TF_{pr}, RDD_{pr}$  соответствуют  $TF_{gen}, RDD_{gen}$ , если  $TF_{pr} \in [TF_{gen} - 0.1, TF_{gen} + 0.1]$  и  $RDD_{pr} \in [RDD_{gen} - 0.1, RDD_{gen} + 0.1]$ .

Из табл. 2 видно, что второй эксперимент дал результаты, аналогичные результатам первого эксперимента: в наборе OR-Library 11 классов задач из 25 возможных не соответствуют ожиданиям, так как параметры  $TF_{pr}, RDD_{pr}$  для задач из этих классов в основном не соответствуют порождающим их параметрам  $TF_{gen}, RDD_{gen}$ .

### 3. Модификация генератора исходных данных

Для исключения указанных выше несоответствий параметров генератора параметрам сгенерированных задач мы предлагаем использовать вышеописанный генератор со следующим ограничением на параметры:

$$RDD_{gen} < \min\{1; 2 - 2 \cdot TF_{gen}\}, \quad (1)$$

и генерировать задачи, табулируя  $TF_{gen}, RDD_{gen}$  от 0.1 до 0.9 с шагом 0.2.

Заметим, что ограничение (1) обеспечивает неотрицательность нижней границы интервала директивных сроков (см. описание алгоритма) и тем самым снимает необходимость замены отрицательных директивных сроков нулями.

Нами был проведен эксперимент по вышеописанной схеме с предложенным ограничением на параметры  $TF_{gen}, RDD_{gen}$ . В табл. 3 приведены результаты этого эксперимента, прочерки в таблице соответствуют парам  $TF_{gen}, RDD_{gen}$ , не удовлетворяющим условию (1), для этих пар параметров задачи не генерировались.

Как видим, предложенный генератор обеспечивает полное соответствие  $TF_{gen}, RDD_{gen}$  классам сгенерированных задач с  $TF_{pr} \in [TF_{gen} - 0.1, TF_{gen} + 0.1]$  и  $RDD_{pr} \in [RDD_{gen} - 0.1, RDD_{gen} + 0.1]$ .

### 4. Генерация задач с малым числом операций

Нами было проведено численное исследование свойств генератора при  $n < 100$ . Было установлено, что пока  $n$  оставалось не меньше 50, качественного отличия

Табл. 4

 $n = 50$ 

$TF_{pr} \setminus RDD_{pr}$	0.0 – 0.2	0.2 – 0.4	0.4 – 0.6	0.6 – 0.8	0.8 – 1.0
0.0 – 0.2	100 / 0 / 0	100 / 0 / 0	101 / 0 / 1	100 / 1 / 1	100 / 1 / 1
0.2 – 0.4	100 / 0 / 0	100 / 0 / 0	101 / 0 / 1	99 / 1 / 0	99 / 1 / 0
0.4 – 0.6	100 / 0 / 0	100 / 0 / 0	100 / 0 / 0	101 / 0 / 1	99 / 1 / 0
0.6 – 0.8	100 / 0 / 0	100 / 0 / 0	100 / 0 / 0	–	–
0.8 – 1.0	100 / 0 / 0	–	–	–	–

Табл. 5

 $n = 20$ 

$TF_{pr} \setminus RDD_{pr}$	0.0 – 0.2	0.2 – 0.4	0.4 – 0.6	0.6 – 0.8	0.8 – 1.0
0.0 – 0.2	101 / 0 / 1	107 / 1 / 8	112 / 7 / 19	111 / 21 / 32	70 / 31 / 1
0.2 – 0.4	101 / 0 / 1	103 / 1 / 4	114 / 4 / 18	119 / 20 / 39	67 / 42 / 9
0.4 – 0.6	102 / 0 / 2	106 / 2 / 8	108 / 8 / 16	116 / 16 / 32	60 / 45 / 5
0.6 – 0.8	101 / 0 / 1	110 / 1 / 11	89 / 11 / 0	2 / – / 2	1 / – / 1
0.8 – 1.0	100 / 0 / 0	–	–	–	–

результатов от вышеизложенных не наблюдалось. Как видно из табл. 4, количество ушедших и пришедших задач не превышает 1%.

Однако при уменьшении количества операций наблюдается заметное ухудшение качества генератора. Из табл. 5 видно, что при  $n = 20$  количество ушедших задач достигает 39%, а пришедших – 45%. Особо отметим, в классы, соответствующие парам  $TF_{gen} = 0.7$ ,  $RDD_{gen} = 0.7$  и  $TF_{gen} = 0.7$ ,  $RDD_{gen} = 0.9$  (для этих пар параметров задачи не генерировались), пришли задачи из других классов.

Проведенные эксперименты показывают, что для получения задач с малым числом операций ( $n \leq 50$ ) нужно дополнить генератор фильтром, который отсеивает задачи, не соответствующие параметрам генератора.

Работа выполнена при финансовой поддержке РФФИ (проект № 10-01-00728).

### Summary

*R. G. Sabirov, V. R. Fazylov.* About the Generator of the Data for the Single Machine Total Weighted Tardiness Problem.

In this paper features of popular pseudo-random generator of input data for total weighted tardiness problem for single machine are discussed, and modification of this generator with better properties is offered.

**Key words:** scheduling theory, generator of tasks.

### Литература

1. *Potts C.N., Van Wassenhove L.N.* A branch and bound algorithm for total weighted tardiness problem // Oper. Res. – 1985. – V. 33. – P. 363–377.
2. *Liu N., Abdelrahman M., Ramaswamy S.* A Genetic Algorithm for Single Machine Total Weighted Tardiness Scheduling Problem // Proc. of the 35th Southeastern Symposium on System Theory, Morgantown (West Virginia, USA), 16–18 March 2003 / IEEE. – 2003. – P. 34–38.

3. *Tasgetiren M.F., Sevkli M.* Particle Swarm Optimization Algorithm for Single Machine Total Weighted Tardiness Problem // Congress of Evolutionary Computation, Portland (Oregon, USA), 20–23 June 2004 / IEEE. – 2004. – V. 2. – P. 1412–1419.
4. *Congram R.K., Post C.N., van de Velde S.L.* An Iterated Dynasearch Algorithm for the Single-Machine Total Weighted Tardiness Scheduling Problem // INFORMS J. Computing. – 2002. – V. 14, No 1. – P. 52–67.
5. *Avci S., Akturk S.M., Storer R.H.* A problem space algorithm for single machine weighted tardiness problems // IEEE Transact. – 2004. – V. 35. – P. 479–486.
6. *Akturk S.M., Yildirim B.M.* A new dominance rule for the total weighted tardiness problem // Production Planning & Control. – 1999. – V. 10, No 2. – P. 138–149.
7. *Akturk S.M., Yildirim B.M.* A new lower bounding scheme for the total weighted tardiness problem // Comp. Oper. Res. – 1998. – V. 25, No 4. – P. 265–278.

Поступила в редакцию  
28.04.09

---

**Сабиров Раиль Гарифзянович** – аспирант кафедры экономической кибернетики Казанского государственного университета.

**Фазылов Валерий Рауфович** – доктор физико-математических наук, профессор кафедры экономической кибернетики Казанского государственного университета.

E-mail: *Valery.Fazylov@ksu.ru*