



Московский государственный университет имени М.В. Ломоносова  
Факультет вычислительной математики и кибернетики  
Кафедра автоматизации систем вычислительных комплексов

Савицкий Илья Павлович

# Построение многопроцессорного расписания с использованием жадных стратегий и ограниченного перебора

Курсовая работа

**Научный руководитель:**  
Доцент, к.т.н  
Костенко Валерий Алексеевич

Москва, 2022

## **Аннотация**

Для построения расписаний можно использовать много всякой херни. вот одна из них.

# Содержание

<b>1</b>	<b>Введение</b>	<b>4</b>
<b>2</b>	<b>Постановка задачи</b>	<b>5</b>
<b>3</b>	<b>Обзор предметной области</b>	<b>7</b>
3.1	Жадные алгоритмы . . . . .	7
3.2	Жадные алгоритмы с процедурой ограниченного перебора . . . . .	7
<b>4</b>	<b>Алгоритм решения</b>	<b>8</b>
4.1	Дополнительные обозначения . . . . .	8
4.2	Словесное описание алгоритма . . . . .	8
4.3	Блок-схема алгоритма . . . . .	10
<b>5</b>	<b>Программная реализация алгоритма</b>	<b>11</b>
<b>6</b>	<b>Экспериментальное исследование алгоритма</b>	<b>12</b>
<b>7</b>	<b>Заключение</b>	<b>13</b>

# 1 Введение

## 2 Постановка задачи

### Дано

1. Ориентированный граф работ  $G$  без циклов, в котором дуги - зависимости по данным, а вершины - задания. Вершин  $n$ , дуг  $m$
2. Вычислительная система, состоящая из  $p$  различных процессоров
3. Матрица  $C_{ij}$  длительности выполнения работ на процессорах,  $i = 1 \dots n, j = 1 \dots p$
4. Матрица  $D_{kl}$  передач данных между процессорами,  $k = 1 \dots p, l = 1 \dots p, D_{kk} = 0$

### Определение расписания

Расписание программы определено, если

1. Множества процессор и работ
2. Привязка - всюду определенная на множестве работ функция, которая задает распределение работ по процессорам
3. Порядок - заданные ограничения на последовательность выполнения работ и является отношением частичного порядка, удовлетворяющим условиям ацикличности и транзитивности. Отношение порядка на множестве работ, распределенных на один процессор, является отношением полного порядка.

### Требуется

1. Построить расписание  $HP$ , то есть для  $i$ -й работы определить время начала ее выполнения  $s_i$  и процессор  $p_i$  на котором она будет выполняться
2. В расписании требуется минимизировать время выполнения набора работ, данных в графе  $G$
3. В задаче так же присутствуют дополнительные ограничения, котрым расписание обязано удовлетворять.

### Различные постановки задачи:

1. Задача с однородными процессорами (длительность выполнения работы не зависит от того, на каком процессоре она выполняется) и дополнительными ограничениями на количество передач:
  - $CR = \frac{m_{ip}}{m} < 0.4$ , где  $m_{ip}$  - количество передач данных между работами на каждый процессор
  - $CR2 = \frac{m_{2edg}}{m} < 0.05$ , где  $m_{2edg}$  - количество дуг, начальный и конечный узлы которых назначены на процессоры, не соединенных напрямую
2. Задача с однородными процессорами и дополнительным ограничением сбалансированности распределения работ:
  - $BF = \lceil 100 \cdot (\frac{a_{max} \cdot p}{n} - 1) \rceil < 10$ , где  $a_{max}$  - наибольшее, по всем процессорам, количество работ на процессоре

3. Задача с неоднородными процессорами, но без дополнительных ограничений на расписание

## **3 Обзор предметной области**

### **3.1 Жадные алгоритмы**

Жадные алгоритмы подразумевают декомпозицию задачи на ряд более простых подзадач.

### **3.2 Жадные алгоритмы с процедурой ограниченного перебора**

## 4 Алгоритм решения

### 4.1 Дополнительные обозначения

1.  $D = (d_1, d_2, \dots, d_l)$ , где  $l$  - количество вершин, доступных для добавления (т.е. у которых нет предшественников в исходном графе) - множество вершин, доступных для добавления в расписание.
2.  $k$  - вектор длин критических путей от "головной" вершины до каждой вершины графа.
3.  $(s_i, p_i)$  - достаточное количество информации для размещения работы в расписании.

Жадные критерии

1.  $GR1$  - критерий, используемый в выборе работы на постановку
2.  $GR2$  - критерий, используемый в выборе места постановки работы

Процедуры ограниченного перебора

1.  $H1$  - процедура перебора для создания места для постановки работы
2.  $H2$  - процедура перебора для приближения времени старта работы к длине критического пути до нее

### 4.2 Словесное описание алгоритма

1. Сформировать множество вершин, у которых нет предшественников. Множество  $D = (d_1, d_2 \dots d_i)$  где  $d_i$  – номер работы, доступной для добавления в расписание (т.е. у которой нет предшественников в исходном графе)
2. В случае, если в множестве  $D$  одна вершина – обозначим ее за  $d$ , в противном случае – создадим фиктивную вершину с нулевой длительностью, у которой все потомки будут из множества  $D$ , и обозначим ее за  $d$
3. Зададим вектор  $k$  – вектор длин критических путей до вершин от  $d$ . При помощи алгоритма Дейкстры этот вектор заполняется значениями  $k_i$ , где  $i$  – номер вершины. Поскольку алгоритм Дейкстры работает со взвешенными графами, каждое ребро получает вес минимального времени работы на вычислительной системе вершины, из которой исходит
4. По жадному критерию выбора работы выбирается работа из множества  $D$  для размещения в расписании. Пусть выбранная работа –  $d_i$
5. Производится пробное размещение работы  $d$  в расписании с учетом жадного критерия выбора места работы и дополнительных ограничений. В случае, если не удалось найти подходящее место для работы – запускается процедура ограниченного перебора с проверяемым критерием возможности добавления работы в расписание. Становится известно  $s$  – время старта работы и  $p$  – процессор, на котором работа выполняется. Выбор места выполнения работы происходит по системе допусков. Изначально места ранжируются по жадному критерию и берутся верхние  $n\%$  ( $n$  – параметр алгоритма) списка. Оставшиеся места ранжируются по первому дополнительному ограничению, после чего снова берутся верхние  $n\%$ , и так далее по всем ограничениям. После прохода по всем ограничениям, из оставшегося списка берется место в соответствии с жадным критерием



6. Если  $s_i$  больше длины критического пути (с точностью до  $\Delta$ , где  $\Delta$  – параметр алгоритма), то вызывается процедура ограниченного перебора с проверяемым критерием  $S = \sum s_k$ , где  $s_k$  – времена начал всех перебираемых работ, в результате которой работа размещается в расписании. Если работу разместить не удалось – завершить алгоритм. Если  $s_i$  не превосходит длину критического пути (с точностью  $\Delta$ ), то работа размещается в расписании.
7.  $d_i$  удаляется из списка размещенных работ и в графе  $G$  удаляется соответствующая вершина и все дуги, исходящие из нее.
8. Обновляется множество  $D$ . Если  $D$  не пустое, то алгоритм переходит на пункт 4.

### **Жадный критерий выбора работы**

- Максимальное количество потомков у работы

### **Жадный критерий выбора места работы в расписании**

- Скорейшее завершение частично построенного расписания

### **Ограниченный перебор**

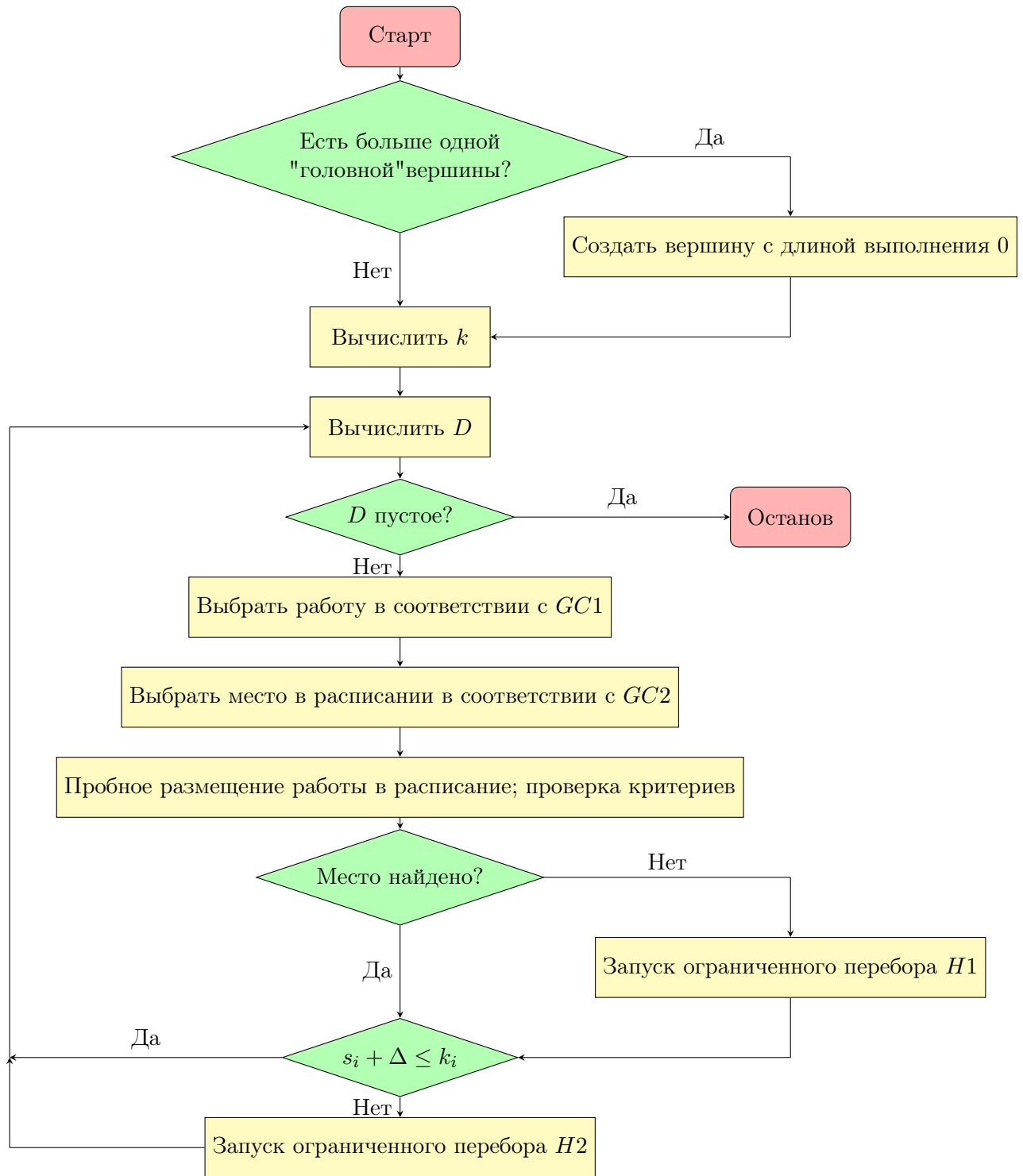
После неудачной пробной постановки работы в расписание алгоритм создает набор  $K = (k_1, k_2, \dots, k_t)$ , состоящий из  $t$  последних добавленных работ ( $t$  – параметр алгоритма). Далее, процедурой полного перебора пробуются различные расписания до тех пор, пока не получится расписание, удовлетворяющее критерию критичности пути до последней поставленной работы и удовлетворяющее дополнительным ограничениям.

### **Расчет времени начала работы**

Для того, чтобы рассчитать время начала для конкретной работы на процессоре  $p$  требуется:

1. Вычислить вектор  $PJ_{k=1}^L$ , где  $L$  – количество предшественников у работы. Элементами этого вектора будут являться суммы вида  $s_k + C_{kr} + D_{rj}$ , где  $r$  – номер процессора, на котором размещен предшественник.
2. Максимумом этого вектора и будет являться первое доступное начало выполнения работы на данном процессоре.  $n_j = \max PJ$

### 4.3 Блок-схема алгоритма



## 5 Программная реализация алгоритма

## 6 Экспериментальное исследование алгоритма

## 7 Заключение

## Список литературы