Градиентный метод формирования динамических расписаний обработки данных в конвейерной системе при различных моментах времени их поступления и разных приоритетах

К. В. Кротов*, Т. Ю. Кротова

Севастопольский государственный университет, Россия

*Контактный e-mail: krotov_k1@mail.ru

Реализация запланированного хода вычислительного процесса обработки данных в конвейерной системе может подвергаться возмущающим воздействиям. Одним из видов возмущающих воздействий является поступление в ходе реализации вычислительного процесса данных с высоким приоритетом. Влияние возмущающих воздействий на ход вычислительного процесса может быть уменьшено путем построения динамических расписаний. В работе обосновываются модель составления расписаний обработки данных и метод построения динамических расписаний обработки данных при различных моментах времени их поступления в систему и разных приоритетах.

Ключевые слова: конвейерная система, обработка данных, расписание выполнения программ, жадный алгоритм, динамическое расписание, приоритет.

Введение

Конвейеризация программ обработки данных предполагает для выполнения закрепление за каждым сегментом конвейера определенных их частей [1]. Введем обозначения: i — идентификатор типа данных, обрабатываемых в системе; n — количество типов данных ($i = \overline{1,n}$); n^i — количество данных i-го типа, которые должны быть обработаны. Данные i-го типа обрабатываются соответствующей им программой, тогда i — идентификатор программы, выполняемой в составе конвейера, обрабатывающей данные i-го типа. Фрагменты программ, выполняющих обработку данных, находятся в оперативной памяти каждого из сегментов конвейера. Однократное выполнение конвейеризированной программы i-го типа обеспечивает обработку одного элемента данных i-го типа. Постановка задачи управления вычислительным процессом предполагает, что $n^i = 1$ ($i = \overline{1,n}$), реализуется обработка единичных данных. Обозначим через l индекс сегмента конвейера, осуществляющего выполнение l-й части программы ($l = \overline{1,L}$). Все сегменты конвейерной системы имеют равную и неизменную во времени производительность работы. Выполнение на каждом l-м сегменте назначенной ему части i-го типа программы

характеризуется параметром длительности обработки данных, однозначно соответствующей объему выполняемых вычислений. Дисциплина обслуживания данных в системе предполагает последовательное прохождение ими всех сегментов конвейера.

Через d_i обозначим момент времени поступления в систему данных i-го типа $(\overline{1,n})$, а через R_i — приоритеты данных, для которых $d_i = 0$. Данные с более высоким приоритетом, чем R_i, поступают на обработку в моменты времени $d_i > 0$, они прерывают обработку данных с $d_i = 0$. Для данных с $d_i > 0$ введем обозначение их типа как $i_{\text{пр}}$, а приоритет $i_{\text{пр}}$-го типа данных обозначим как $R_{i_{\text{пр}}}$. Запланированный для данных i-х типов (с $d_i = 0$) ход вычислительного процесса подвергается возмущающему воздействию, соответствующему поступлению в систему в моменты времени $d_{i_{\text{пр}}} > 0$ данных $i_{\text{пр}}$ с приоритетом $R_{i_{\text{пр}}} > R_i$.

1. Анализ существующих методов построения динамических расписаний

В работе [1] выполнен анализ основных подходов к решению задач формирования расписаний обработки данных. Для решения задач теории расписаний развиваются точные методы (ветвей и границ, ветвей и отсечений), приближенные методы и методы локальной оптимизации (метод отжига, генетические алгоритмы и т. д.). В частности, в работе [2] рассматриваются методы управления процессом обработки данных в динамических условиях, предполагающие распределение нагрузки при масштабировании вычислительных ресурсов с учетом времени, отводимого на выполнение задач.

В работах [3–10] развиваются методы формирования динамических расписаний, среди которых рассматриваются как методы на основе эвристических правил [3–8], так и методы, позволяющие получать локально оптимальные решения [9, 10]. В работе [3] приводятся различные эвристические правила, позволяющие формировать статические расписания обработки данных, они могут быть использованы также при формировании динамических расписаний. Работа [4] имеет аналогичный характер, в ней рассматривается использование функций приоритета, позволяющих вычислять значения приоритета (веса) обрабатываемых данных, на основе которых эти данных упорядочиваются в очереди на обработку. В соответствии с этим реализуется определение эффективных (с точки зрения введенных критериев) эвристических правил и функций приоритетов при решении задачи построения динамических расписаний для данных, поступающих на обработку в различные моменты времени. Аналогичная задача решается в [5], где на основе эвристических правил реализуется упорядочивание данных в очередях на обработку и приводится анализ эффективности правил при решении задачи построения расписаний с динамической нагрузкой (поступление данных в различные моменты времени).

В работе [6] на основе имитационной модели определяются оптимальные значения параметров эвристического правила планирования загрузки многопроцессорной системы — значения уровня мультипрограммирования (количества задач, распределяемых по процессорам с учетом динамической нагрузки) и кванта времени процессора, выделяемого выполняемым задачам. То есть рассматривается эвристическое правило, реализующее планирование при поступлении данных в различные моменты времени, и способ определения его параметров. В [7] также исследуется применение эвристического правила, предполагающего сдвиг начала обработки данных как можно ближе к директивному

сроку окончания их обработки, для определения расписания обработки периодических и апериодических задач (т. е. задач, поступающих на обработку в различные моменты времени). Полученные с применением данного правила расписания содержат интервалы времени простоев обрабатывающих приборов, которые используются для обработки поступающих апериодических задач.

В [8] обосновывается применение эвристического правила для формирования расписаний обработки задач, взаимодействие которых представляется в виде параметризованного графа выполнения задач. При этом граф может быть дополнен вершинами, соответствующими динамически поступающим задачам, которые с использованием этого же эвристического правила распределяются по очередям обработки процессорных элементов.

В [9] реализуется применение одновременно эвристических правил для формирования начальных решений и аппарата генетических алгоритмов (ГА) с целью определения локально оптимальных решений при построении динамических расписаний. Генетические алгоритмы также используются для построения динамических расписаний в [10]. При этом учитываются динамические свойства расписаний: исключаются данные, обработанные к моменту поступления новых данных, и операции, завершившиеся к этому моменту; исключаются обрабатывающие приборы, занятые обработкой данных. Этот подход может быть использован при построении динамических расписаний обработки данных с одинаковыми приоритетами, поступающими в систему в различные моменты времени.

Таким образом, первая группа исследований связана с применением эвристических правил к построению динамических расписаний и направлена на определение эффективных правил (и их параметров) с точки зрения введенных критериев. Вторая группа исследований связана с применением ГА при решении задачи определения динамических расписаний. Этот подход не является универсальным, так как требует применения различных операторов ГА (мутации, скрещивания, селекции) при разных исходных данных для определения локально оптимальных решений. Применение одновременно всех возможных операторов ГА значительно усложняет процесс поиска локально оптимального решения. Так, эвристические правила, предложенные в [3, 4], могут быть адаптированы для построения расписаний обработки данных с разными приоритетами, поступающих в различные моменты времени. Все остальные методы реализуют формирование динамических расписаний обработки данных с равными приоритетами, поступающими в различные моменты времени. Поэтому решение задачи построения динамических расписаний при учете возмущающих воздействий, изменяющих ход вычислительного процесса (поступлении на обработку данных с высокими приоритетами в моменты времени $d_{i_{\text{пр}}}$), является актуальным.

Цель настоящей работы состоит в совершенствовании методов построения расписаний обработки данных в конвейерных системах. Совершенствование методов связано с разработкой подходов к формированию динамических расписаний, учитывающих влияние возмущающих воздействий (поступление данных с $R_{i_{\text{пр}}}$, $i = \overline{1,n}$, в моменты времени $d_{i_{\text{пр}}}$) на ход вычислительного процесса, реализуемый в соответствии со статическим расписанием. Это позволит осуществить построение динамических расписаний в конвейерных системах и снизить влияние возмущений. Для достижения этой цели необходимо выполнить решение задач: формирование модели оптимизации расписаний обработки данных и обоснование метода локальной оптимизации для построения динамических расписаний.

2. Обоснование модели оптимизации расписаний обработки данных в конвейерной системе

Особенностью решаемой задачи является задание двух групп данных, обрабатываемых в системе. Первая группа содержит данные n типов, для которых выполняются условия: 1) время поступления данных на обработку $d_i = 0$ $(i = \overline{1,n})$; 2) приоритеты данных являются одинаковыми, $R_i = R_{i'}$. Вторая группа — данные $i_{\text{пр}}$, поступающие на обработку в моменты времени $d_{i_{\text{пр}}} > 0$, значения приоритетов этих данных $R_{i_{\text{пр}}} > R_i$ $(i = \overline{1,n})$. При этом как значения приоритетов R_i данных i-х типов $(i = \overline{1,n})$, так и приоритетов $R_{i_{\text{пр}}}$ данных $i_{\text{пр}}$ задаются входными параметрами алгоритма построения расписаний, т. е. исходными параметрами решаемой задачи построения расписаний. Эти значения не изменяются в ходе реализации алгоритма (являются статическими). Значения приоритетов $R_{i_{\text{пр}}}$ данных $i_{\text{пр}}$, поступающих в систему в моменты $d_{i_{\text{пр}}}$, одинаковые. Тогда прерывание обработки данных $i_{\text{пр}}$ с приоритетом $R_{i_{\text{пр}}}$ поступившими данными $i'_{\text{пр}}$ с приоритетом $R_{i'_{\text{пр}}}$ невозможно (при $d_{i'_{\text{пр}}} > d_{i_{\text{пр}}}$ в силу $R_{i'_{\text{пр}}} = R_{i_{\text{пр}}}$). Фактически различаются две градации приоритета: низкий — для данных i-х типов $(i = \overline{1,n})$ с $d_i = 0$, высокий — для данных $i_{\text{пр}}$ с $d_{i_{\text{пр}}} > 0$.

Управление вычислительным процессом состоит в определении порядка обработки данных на каждом из сегментов. Порядки обработки данных на сегментах определяют расписание обработки. Каждому из сегментов соответствует последовательность π^l обработки данных, расписание обработки данных определяется как совокупность последовательностей π^l $(l = \overline{1,L})$. Последовательность π^l $(l = \overline{1,L})$ для данных, поступивших на обработку в систему в момент $d_i = 0$ (n типов данных с приоритетом R_i), имеет вид $\pi = \left(i_1^l, i_2^l, \ldots, i_{n-1}^l, i_n^l\right)$. Построение расписания предполагает определение эффективных с точки зрения введенного критерия видов последовательностей π^l $(l = \overline{1,L})$. Данные $i_{\text{пр}}$ с приоритетом $R_{i_{\text{пр}}}$ прерывают обработку данных, входящих в π^l на каждом из сегментов конвейера, либо обрабатываются в течение интервала времени простоя сегментов. Тогда для данных $i_{\text{пр}}$ позиция в последовательностях π^l не определяется. Для формализации видов последовательностей π^l расписания π в рассмотрение введены матрицы P^l $(l = \overline{1,L})$ порядков обработки данных в системе. Элемент $p_{ij}^l = 1$, если данные i-го типа занимают в последовательности π^l j-ю позицию, $p_{ij}^l = 0$ в том случае, если данные i-го типа в π^l j-ю позицию не занимают. Матрицы P^l $(l = \overline{1,L})$ имеют размерность $n \times n$.

Для формализации вида модели вычислительного процесса обработки данных в конвейерной системе введены обозначения: t_i^l — длительность обработки данных i-го типа на l-м сегменте $(l = \overline{1,L},\ i = \overline{1,n})$; $t_{i_{\text{пр}}}^l$ — длительность обработки данных $i_{\text{пр}}$ на l-м сегменте конвейера; $\left(t_{ji}^{0l}\right)$ — матрица моментов времени начала обработки данных i-х типов, занимающих в π^l j-е позиции, через $\overline{t_{ji}^l}$ обозначен момент времени окончания обработки данных i-го типа в j-й позиции в π^l. Значения элементов матриц $\left(t_{ji}^{0l}\right)$ $(l = \overline{1,L})$ определяются в соответствии с видом матриц P^l следующим образом: $t_{ji}^{0l} = 0$ для элемента матрицы $\left(t_{ji}^{0l}\right)$, который соответствует $p_{ij}^l = 1$. В случае $p_{ij}^l = 0$ соответствующий ему элемент t_{ji}^{0l} матрицы $\left(t_{ji}^{0l}\right)$ равен 0. Для первого сегмента конвейера элементы матрицы $\left(t_{ji}^{0l}\right)$ определяются с учетом матрицы P^l: $t_{1i}^{01} = 0$, $t_{2i}^{01} = \sum\limits_{h=1}^{n} t_h^1 p_{h1}^1$, $t_{3i}^{01} = \sum\limits_{j=1}^{2}\sum\limits_{h=1}^{n} t_h^1 p_{h,j}^1$, $t_{4i}^{01} = \sum\limits_{j=1}^{3}\sum\limits_{h=1}^{n} t_h^1 p_{h,j}^1$ и т. д. В общем виде выражение для определения t_{ji}^{01} следующее:

$$t_{ji}^{01} = \sum_{q=1}^{j-1} \sum_{h=1}^{n} t_h^1 p_{h,q}^1 \quad \text{при } j > 1. \tag{1}$$

Для l-го сегмента (при $l \neq 1$) элементы матрицы $\left(t_{ji}^{0l}\right)$ определяются выражениями вида:

а) первая строка (первая позиция для данных i-го типа, $j = 1$)

$$t_{1,i}^{0l} = \sum_{h=1}^{n} p_{ih}^{l-1} \left(t_{h,i}^{0l-1} + t_i^{l-1}\right), \tag{2}$$

где тип данных i определяется по матрице P^l $(l = \overline{1, L})$ как занимающих первую позицию в последовательности π^l (i-й тип данных в позиции $j = 1$ на l-м сегменте);

б) j-я строка $(j \neq 1)$ в матрице $\left(t_{ji}^{0l}\right)$ $(l = \overline{2, L})$ для данных i-го типа

$$t_{ji}^{0l} = \max \left\{ \sum_{h=1}^{n} p_{i,h}^{l-1} \left(t_{h,i}^{0l-1} + t_i^{l-1}\right); \ \sum_{h=1}^{n} \left(t_{j-1,h}^{0l} + t_h^l\right) p_{h,j-1}^l \right\}. \tag{3}$$

Выражения (1)–(3) являются моделью вычислительного процесса обработки данных в конвейерной системе. Сформированное решение по порядку обработки данных имеет вид $[P^l, \left(t_{ji}^{0l}\right) \mid l = \overline{1, L}]$.

Особенностями алгоритма определения расписания обработки данных, реализующего "жадный" подход [1], являются:

1) добавление на s-м шаге данных i-го типа в конец последовательностей $\pi^l(s-1)$ $(l = \overline{1, L})$, сформированных на предыдущем $(s-1)$-м шаге;
2) определение эффективных позиций рассматриваемых данных i-го типа в каждой из последовательностей $\pi^l(s)$ на текущем s-м шаге алгоритма;
3) если изменение положения данных i-го типа в последовательностях π^l $(l = \overline{1, L})$ не приводит к уменьшению значения критерия, то реализуется изменение положения данных i-го типа одновременно в двух последовательностях π^l $(l = \overline{1, L})$, в трех последовательностях и т. д. Тогда для текущего решения реализуется переход к более эффективному решению в рамках его окрестностей с различными метриками.

В соответствии с особенностями алгоритма формирования расписаний обработки данных [1] на некотором s-м шаге выполняется решение отдельной подзадачи добавления и размещения в последовательностях π^L $(l = \overline{1, L})$ данных одного i-го типа, при этом на последующих шагах алгоритма предполагается решение подзадачи размещения в последовательностях π^l $(l = \overline{1, L})$ данных оставшихся $(n-i)$-х типов. Тогда на s-м шаге алгоритма выполняется "жадный" выбор по определению локально оптимального решения (эффективных j-х позиций в π^l $(l = \overline{1, L})$ данных i-го типа). На основе локально оптимального решения для данных i-го типа формируется решение по определению позиций данных следующих $(n-i)$-х типов. Каждый последующий жадный выбор связан с добавлением в π^l данных одного типа, определением их позиций в последовательностях π^l $(l = \overline{1, L})$. Решение по размещению в π^l $(l = \overline{1, L})$ данных i-го типа формируется на основе локально оптимального решения, сформированного для данных предшествующих $(i-1)$-х типов.

Данные не всех n типов одновременно находятся в последовательностях $\pi^l(s)$ $(l = \overline{1, L})$, а только их текущее количество. Тогда для обоснования критерия эффективности решения приведем следующие рассуждения.

1. При $\sum\limits_{i=1}^{n} t_{ji}^{0l} p_{ij}^{l} > \sum\limits_{i=1}^{n} (t_{j-1,i}^{0l} + t_i^l) p_{i,j-1}^{l}$ l-й сегмент конвейера ожидает готовности для обработки данных в j-й позиции после их обработки в (j–1)-й позиции. Длительность простоев l-го сегмента в ожидании данных, занимающих j-ю позицию в последовательности π^l, определяется следующим образом:

$$\sum_{i=1}^{n} t_{j,i}^{0l} p_{i,j}^{l} - \sum_{i=1}^{n} (t_{j-1,i}^{0l} + t_i^l) p_{i,j-1}^{l},$$

тогда длительность простоев l-го сегмента при выполнении текущего количества программ обработки данных, находящихся в π^l, определяется выражением

$$\sum_{j=2}^{n} \left(\sum_{i=1}^{n} t_{j,i}^{0l} p_{i,j}^{l} - \sum_{i=1}^{n} (t_{j-1,i}^{0l} + t_i^l) p_{i,j-1}^{l} \right).$$

2. Для последовательности π^l простои первого сегмента в ожидании готовности данных отсутствуют, тогда суммарное время простоев $(L-1)$-го сегмента конвейера будет определено выражением вида

$$\sum_{l=2}^{L} \sum_{j=2}^{n} \left(\sum_{i=1}^{n} t_{j,i}^{0l} p_{i,j}^{l} - \sum_{i=1}^{n} (t_{j-1,i}^{0l} + t_i^l) p_{i,j-1}^{l} \right).$$

3. Если данные занимают первую позицию в последовательностях π^l ($l = \overline{2, L}$), то для l-го сегмента ($l \neq 1$) время ожидания готовности данных для обработки в первой позиции определяется выражением $\sum\limits_{i=1}^{n} t_{1,i}^{0l} p_{i1}^{l}$. Так как $t_{1i}^{01} = 0$, простои всех l-х сегментов конвейера ($l = \overline{2, L}$) будут определены выражением $\sum\limits_{l=2}^{L} \sum\limits_{i=1}^{n} t_{1,i}^{0l} p_{i1}^{l}$.

В соответствии с приведенными рассуждениями вид критерия эффективности формируемых решений по порядку обработки данных в π^l ($l = \overline{1, L}$) следующий:

$$f = \sum_{l=2}^{L} \sum_{i=1}^{n} t_{1,i}^{0l} p_{i,1}^{l} + \sum_{l=2}^{L} \sum_{j=2}^{n} \left(\sum_{i=1}^{n} t_{ji}^{0l} p_{ij}^{l} - \sum_{i=1}^{n} \left(t_{j-1,i}^{0l} + t_i^l \right) p_{i,j-1}^{l} \right). \qquad (4)$$

Введем обозначение окрестности текущего решения с метрикой k ($k = \overline{1, k_{\max}}$), в которой выполняется поиск локально оптимального решения O_k. Значение метрики k определяется как

$$k = \sum_{i=1}^{n} \sum_{j=1}^{n} |p_{i,j}(s+g) - p_{i,j}(s)|/2,$$

где g — индекс промежуточного шага алгоритма, на котором выполняется формирование нового решения $\pi(s+g)$ на базе текущего локально оптимального решения $\pi(s)$.

3. Обоснование метода построения динамических расписаний обработки данных

Предложенный в [1] метод построения статических расписаний обработки данных в конвейерных системах является основой для обоснования метода построения динамических расписаний (с учетом поступления данных $i_{\text{пр}}$ в моменты времени $d_{i_{\text{пр}}} > 0$). Использование одинаковых структур данных при построении статических и динамических расписаний позволяет унифицировать подход к решению рассматриваемых задач.

Для дальнейших рассуждений по обоснованию метода построения динамических расписаний выполнен переход к обозначениям данных и их позиций в последовательности π^l $(l=\overline{1,L})$ в виде i^l_j, где j — позиция данных i-го типа в последовательности π^l. Поступление в систему данных $i_{\text{пр}}$ приводит к прерыванию обработки данных i^l_j на соответствующих сегментах. В момент времени $d_{i_{\text{пр}}} > 0$ $(l=1)$-й сегмент конвейера немедленно приступает к обработке данных $i_{\text{пр}}$, прерывая обработку данных i^1_j. В момент $t^{\overline{l-1}}_{i_{\text{пр}}}$ $(l=\overline{2,L})$ оканчивается обработка данных $i_{\text{пр}}$ на $(l-1)$-м сегменте, на l-м сегменте прерывается обработка данных i^l_j. Прерывание обработки данных i^l_j на l-х сегментах конвейера (в моменты времени $t^{\overline{l-1}}_{i_{\text{пр}}}$) продолжается до полной обработки данных $i_{\text{пр}}$ в системе. В силу того, что значение $d_{i_{\text{пр}}} > 0$ определяется в момент поступления данных $i_{\text{пр}}$ на обработку и значения длительностей $t^l_{i_{\text{пр}}}$ $(l=\overline{1,L})$ известны, могут быть определены моменты $t^{0l}_{i_{\text{пр}}}$ начала обработки данных $i_{\text{пр}}$ на l-х сегментах $(l=\overline{1,L})$:

$$t^{0l}_{i_{\text{пр}}} = t^{0l-1}_{i_{\text{пр}}} + t^{l-1}_{i_{\text{пр}}}, \quad t^{\overline{l-1}}_{i_{\text{пр}}} = t^{0l}_{i_{\text{пр}}}, \quad t^{0l}_{i_{\text{пр}}} = d_{i_{\text{пр}}}. \tag{5}$$

Моменты времени $t^{0l}_{i_{\text{пр}}}$ начала обработки данных $i_{\text{пр}}$ определяются по (5), тогда моменты времени прерывания обработки данных i^l_j также вычисляются по (5). При прерывании данными $i_{\text{пр}}$ обработки данных i^l_j выполняется условие $t^{0l}_{i_{\text{пр}}} \in \left[t^{0l}_{i_j}, \overline{t^l_{i_j}}\right]$, где $t^{0l}_{i_j}$, $\overline{t^l_{i_j}}$ — моменты времени начала и окончания обработки данных i^l_j. В случае выполнения этого условия при построении динамического расписания длительность $t^l_{i_j}$ обработки данных i^l_j, занимающих в π^l j-ю позицию, и момент времени окончания обработки модифицируются следующим образом:

$$\left(t^l_{i_j}\right)' = t^l_{i_j} + t^l_{i_{\text{пр}}}, \quad \left(\overline{t^l_{i_j}}\right)' = \overline{t^l_{i_j}} + t^l_{i_{\text{пр}}}, \tag{6}$$

где $t^l_{i_j}$ — длительность обработки данных i-го типа на l-м сегменте; $\left(t^l_{i_j}\right)'$ — длительность обработки данных i-го типа, используемая при построении динамического расписания.

Так как момент времени $d_{i_{\text{пр}}} > 0$ поступления данных $i_{\text{пр}}$ на обработку задан и моменты времени начала обработки данных $i_{\text{пр}}$ на l-х сегментах вычисляются с использованием (5), при построении динамического расписания длительности $t^l_{i_j}$ обработки данных i^l_j и моменты времени окончания их обработки $\overline{t^l_{i_j}}$ вычисляются по (6). В результате позиции данных $i_{\text{пр}}$ в последовательностях π^l $(l=\overline{1,L})$ не определяются, а при построении динамического расписания модифицируются значения $t^l_{i_j}$ и $\overline{t^l_{i_j}}$.

Если для $t^{0l}_{i_{\text{пр}}}$ реализуются условия $t^{0l}_{i_{\text{пр}}} \notin \left[t^{0l}_{i_j}, \overline{t^l_{i_j}}\right]$ и $t^{0l}_{i_{\text{пр}}} \in \left[\overline{t^l_{i_j-1}}, t^{0l}_{i_j}\right]$, то момент начала обработки данных $i_{\text{пр}}$ на l-м сегменте принадлежит интервалу времени простоя этого сегмента в ожидании начала обработки данных i-го типа в j-й позиции в π^l после окончания обработки данных i'-го типа в $(j-1)$-й позиции. Поскольку данные i^l_j являются следующими за $i_{\text{пр}}$ в последовательности π^l, значения $t^{0l}_{i_j}$ изменяются следующим образом:

$$\left(t^{0l}_{i_j}\right)' = \overline{t^l_{i_{\text{пр}}}}, \quad \text{если} \quad t^{0l}_{i_j} \le \overline{t^l_{i_{\text{пр}}}}; \quad \left(t^{0l}_{i_j}\right)' = t^{0l}_{i_j}, \quad \text{если} \quad t^{0l}_{i_j} > \overline{t^l_{i_{\text{пр}}}}. \tag{7}$$

Решение задачи построения динамических расписаний обеспечивается рассмотрением последовательностей π^l полученного статического расписания, оптимизированного для данных n типов, в виде двух последовательностей — π^l_1 и π^l_2 $(\pi^l = \pi^l_1 \cup \pi^l_2$ $(l=\overline{1,L}))$. Последовательность π^l_1 образуют данные, обработанные и обрабатываемые к моменту

поступления данных $i_{\text{пр}}$ в систему и обработка которых на l-м сегменте будет закончена до поступления данных $i_{\text{пр}}$ на него (последовательность π_1^l образуют данные, порядок обработки которых не может быть изменен).

Для определения данных i_j^l, включаемых в π_1^l ($l = \overline{1, L}$), сформулированы условия следующего вида:

а) данные $i_j^l \in \pi_1^l$, обработанные к моменту $d_{i_{\text{пр}}} > 0$:

$$t_{i_j}^{0l} < d_{i_{\text{пр}}}, \quad \overline{t_{i_j}^l} < d_{i_{\text{пр}}}; \tag{8}$$

б) данные $i_j^l \in \pi_1^l$ обрабатываются к моменту $d_{i_{\text{пр}}} > 0$, но их обработка завершится к моменту $t_{i_{\text{пр}}}^{0l}$ (до поступления данных $i_{\text{пр}}$ на обработку на l-й сегмент конвейера):

$$t_{i_j}^{0l} < d_{i_{\text{пр}}}, \quad \overline{t_{i_j}^l} > d_{i_{\text{пр}}}, \quad \overline{t_{i_j}^l} < t_{i_{\text{пр}}}^{0l}; \tag{9}$$

в) данные i_j^l, обрабатываемые к моменту $d_{i_{\text{пр}}} > 0$, но их обработка к моменту $t_{i_{\text{пр}}}^{0l}$, т. е. к поступлению данных $i_{\text{пр}}$ на обработку на l-й сегмент конвейера, завершена не будет:

$$t_{i_j}^{0l} < d_{i_{\text{пр}}}, \quad t_{i_{\text{пр}}}^{0l} \le \overline{t_{i_j}^l}. \tag{10}$$

Значения $t_{i_j}^{0l}$ и $\overline{t_{i_j}^l}$ ($l = \overline{1, L}$) соответствуют статическому расписанию, значения $t_{i_{\text{пр}}}^{0l}$ ($l = \overline{1, L}$) вычисляются по формуле (5).

При выполнении условий $t_{i_j}^{0l} < d_{i_{\text{пр}}}$, $t_{i_{\text{пр}}}^{0l} \le \overline{t_{i_j}^l}$ данные i_j^l являются "граничными" для последовательности π_1^l и их длительность (момент времени окончания их обработки $\overline{t_{i_j}^l}$) модифицируется в соответствии с (6). Обозначив через i_q^l данные, которые являются "границей" последовательности π_1^l, эту последовательность можно представить в одном из следующих видов:

$$\pi_1^l = \left\{i_1^l, i_2^l, \ldots, i_q^l\right\}, \quad \pi_1^l = \left\{i_1^l, i_2^l, \ldots, i_q^l, i_{\text{пр}}^l\right\}, \quad \pi_1^l = \left\{i_1^l, i_2^l, \ldots, \left(i_q^l\right)_1, i_{\text{пр}}^l, \left(i_q^l\right)_1\right\}, \tag{11}$$

где i_j^l ($j = \overline{1, q}$) — данные, порядок которых в π^l не может быть изменен; данные i_q^l — "граница" между последовательностями π_1^l и π_2^l; $\left(i_q^l\right)_1$ и $\left(i_q^l\right)_2$ — первая и вторая стадии обработки данных i_q^l при прерывании их обработки данными $i_{\text{пр}}$ на l-м сегменте.

Для данных i_j^l, включаемых в последовательность π_2^l, выполняются условие $d_{i_{\text{пр}}} < t_{i_j}^{0l}$ (данные i_j^l начинают обрабатываться на l-м сегменте после поступления данных $i_{\text{пр}}$ в систему) и одно из указанных ниже условий.

1. Обработка данных i_j^l завершается раньше поступления данных $i_{\text{пр}}$ на обработку на l-й сегмент конвейера:

$$t_{i_j}^{0l} < t_{i_{\text{пр}}}^{0l}, \quad \overline{t_{i_j}^l} \le t_{i_{\text{пр}}}^{0l} \ \text{ при } d_{i_{\text{пр}}} < t_{i_j}^{0l}. \tag{12}$$

2. Обработка данных i_j^l будет прервана поступлением на обработку данных $i_{\text{пр}}$ ($i = \overline{1, n}$):

$$t_{i_j}^{0l} < t_{i_{\text{пр}}}^{0l}, \quad \overline{t_{i_j}^l} > t_{i_{\text{пр}}}^{0l} \ \text{ при } d_{i_{\text{пр}}} < t_{i_j}^{0l}. \tag{13}$$

3. Обработка данных i_j^l начинается после завершения обработки данных $i_{\text{пр}}$:

$$\overline{t_{i_{\text{пр}}}^l} < t_{i_j}^{0l}. \tag{14}$$

В результате для данных $i_j^l \in \pi_2^l$ выполняются условия:

а) $d_{i_{\text{пр}}} < t_{i_j}^{0l}$ и условие (12), если в статическом расписании обработка данных i_j^l предшествует поступлению данных $i_{\text{пр}}$ на l-й сегмент конвейера;

б) $d_{i_{\text{пр}}} < t_{i_j}^{0l}$ и условие (13), если обработка данных i_j^l на l-м сегменте прерывается поступлением данных $i_{\text{пр}}$;

в) $d_{i_{\text{пр}}} < t_{i_j}^{0l}$ и условие (14), если обработка данных i_j^l выполняется после завершения обработки данных $i_{\text{пр}}$.

Выражения (12)–(14) при $d_{i_{\text{пр}}} < t_{i_j}^{0l}$ определяют условия включения данных i_j^l в π_2^l. Их порядок в π_2^l ($l = \overline{1, L}$) может быть изменен при построении динамического расписания.

Последовательности π_2^l ($l = \overline{1, L}$) в динамическом расписании при выполнении условия $d_{i_{\text{пр}}} < t_{i_j}^{0l}$ и условий, определяемых выражениями (12)–(14), имеют следующий вид:

а) данные $i_{\text{пр}}$ не прерывают обработку данных i_j^l в последовательности π_2^l, а строго следуют за ними (в соответствии с условием (12)):

$$\pi_2^l = \left\{ i_{q+1}^l, i_{\text{пр}}^l, i_{q+2}^l, i_{q+3}^l, \ldots, i_n^l \right\};$$

б) данные $i_{\text{пр}}$ прерывают обработку данных i_j^l в последовательности π_2^l (в соответствии с условием (13)):

$$\pi_2^l = \left\{ i_{q+1}^l, i_{q+2}^l, \ldots, \left(i_j^l\right)_1, i_{\text{пр}}^l, \left(i_j^l\right)_2, i_{j+1}^l, i_{j+2}^l, \ldots, i_n^l \right\},$$

где $\left(i_j^l\right)_1$ и $\left(i_j^l\right)_2$ — первая и вторая стадии обработки данных i_j^l при прерывании этой обработки данными $i_{\text{пр}}$;

в) данные $i_{\text{пр}}$ не прерывают обработку данных i_j^l в последовательности π_2^l (в соответствии с условием (14)):

$$\pi_2^l = \left\{ i_{q+1}^l, i_{q+2}^l, \ldots, i_j^l, i_{j+1}^l, \ldots, i_n^l \right\}.$$

Данные, входящие в последовательность π_2^l ($l = \overline{1, L}$), могут занимать в ней позиции от $(q+1)$-й до n-й (j-я позиция данных i-го типа, входящих в последовательность π_2^l, определяется с точки зрения минимизации критерия (4) при условии $q + 1 \le j \le n$). Позиции для данных $i_{\text{пр}}$ в последовательностях π_1^l и π_2^l не определяются, для них по формулам (5) вычисляются значения $t_{i_{\text{пр}}}^{0l}$ ($l = \overline{1, L}$) при учете $d_{i_{\text{пр}}} > 0$.

Таким образом, если для данных i_j^l, входящих в π_1^l (в π_2^l), выполняется условие $t_{i_{\text{пр}}}^{0l} \in \left[t_{i_j}^{0l}, \overline{t_{i_j}^l}\right]$, то длительности обработки данных i_j^l модифицируются в соответствии с (6). Если для данных i_j^l, входящих в π_1^l (в π_2^l), выполняются условия $t_{i_{\text{пр}}}^{0l} \notin \left[t_{i_j}^{0l}, \overline{t_{i_j}^l}\right]$ и $t_{i_{\text{пр}}}^{0l} \in \left[\overline{t_{i'_{j-1}}^l}, t_{i_j}^{0l}\right]$, то значение $t_{i_j}^{0l}$ для данных i_j^l модифицируется по (7).

Построению динамического расписания с учетом обработки данных $i_{\text{пр}}$ предшествует построение статического расписания для данных i-х типов с $d_i = 0$ ($i = \overline{1, n}$). Для этого используется предложенный в [1] метод формирования расписаний обработки данных в конвейерных системах, который основывается на "жадных" стратегиях.

Перед построением динамического расписания выполняется реализация предварительных этапов:

- в статическом расписании для последовательностей π^l ($l = \overline{1, L}$) в соответствии с выражениями (8)–(10) и (12)–(14) определяются для каждого l-го сегмента составы последовательностей π_1^l и π_2^l ($l = \overline{1, L}$);
- в последовательностях π_1^l в соответствии с условием (10) определяются данные $i_q^l \in \pi_1^l$, являющиеся граничными для этих последовательностей.

Для обоснования алгоритма построения динамического расписания в рассмотрение введено обозначение множеств N^l ($l = \overline{1, L}$) данных, порядок которых в последовательностях π_2^l ($l = \overline{1, L}$) на каждом l-м сегменте будет определяться. Для каждого сегмента в соответствии с (12)–(14) формируется множество N^l ($l = \overline{1, L}$) данных, позиции которых в последовательностях π_2^l будут определены при построении динамического

расписания. Также введено множество $N_{\text{пр}}$ всех типов данных, которые в процессе построения динамического расписания будут размещены в последовательностях π_2^l ($l = \overline{1, L}$): $N_{\text{пр}} = \bigcup\limits_{l=1}^{L} N^l$.

Расчет значений критерия (4) возможен в случае, когда количество данных в последовательностях π^l ($l = \overline{1, L}$) является одинаковым, при этом в π^l размещены данные одинаковых типов. Поэтому способ формирования последовательностей π_2^l предусматривает реализацию двух этапов.

1. Размещение в последовательностях π_2^l ($l = \overline{1, L}$) данных $i \in N_{\text{пр}}$ таких, что $i \notin \pi_1^l$ и $i \in \pi_1^{l'}$, где $l \neq l'$ (данные i-го типа не входят в последовательность $\pi_1^{l'}$ l-го сегмента конвейера, но входят в последовательность $\pi_1^{l'}$ другого l'-го сегмента). Определение в π_2^l эффективного положения этих данных (на первом этапе определяется местоположение в π_2^l всех данных i, для которых перед началом построения динамического расписания выполняется условие $i \in N_{\text{пр}} \cap \left(\bigcup\limits_{l=1}^{L} \pi_1^l\right)$).
2. Размещение в последовательностях π_2^l данных i-х типов, для которых перед построением динамического расписания выполняется условие
$$(i \in N_{\text{пр}}) \,\&\, \left(i \notin \left(\bigcup_{l=1}^{L} \pi_1^l\right)\right),$$
определение эффективного местоположения этих данных в последовательностях π_2^l, т. е. размещение в π_2^l данных i-х типов, которые перед началом построения расписания не входили в последовательности π_1^l ни на одном l-м сегменте ($l = \overline{1, L}$).

Для реализации первого этапа определения эффективных порядков обработки данных в последовательностях π_2^l введем в рассмотрение множество $N'_{\text{пр}}$ данных i-го типа, таких, что $i \in N_{\text{пр}} \cap \left(\bigcup\limits_{l=1}^{L} \pi_1^l\right)$, а также i' для данных, которые будут рассматриваться на s-м шаге алгоритма, позицию которых в π_2^l обозначим как j'.

Формирование начального решения по порядку обработки данных в π_2^l ($l = \overline{1, L}$) предполагает добавление данных i-х типов, для которых выполняются условия $i \in N'_{\text{пр}}$ и $i \notin \pi_1^l$ ($l = \overline{1, L}$), т. е. данных, которые не входят в последовательность π_1^l на этом сегменте, но содержатся в последовательности $\pi_1^{l'}$ ($l \neq l'$). В силу приведенных рассуждений последовательность шагов алгоритма, реализующего формирование начального решения по составам последовательностей π_2^l ($l = \overline{1, L}$) для данных $i \in N'_{\text{пр}}$, имеет следующий вид.

1. Из $N'_{\text{пр}}$выбираются данные i'-го типа в соответствии с условием $i' = \min\left\{i | i \in N'_{\text{пр}}\right\}$.
2. Для данных i'-го типа выполняется проверка условия $i' \in \pi_1^l$ ($l = \overline{1, L}$), в случае $i' \notin \pi_1^l$ они добавляются в последовательность π_2^l: $\pi_2^l = \pi_2^l \cup \{i'\}$, в результате данные занимают в π_2^l некоторую j'-ю позицию ($q < j'$).
3. Для данных i'-го типа в j'-й позиции в π_2^l (данных $i'^l_{j'}$) реализуется проверка условия прерывания их обработки на l-м сегменте данными $i_{\text{пр}}$: $t^{0l}_{i_{\text{пр}}} \in \left[t^{0l}_{i'_{j'}}, \overline{t^{l}_{i'_{j'}}}\right]$. При его выполнении значение длительности $t^{l}_{i'_{j'}}$ модифицируется в соответствии с (6). Для данных $i''^l_{j'}$ реализуется проверка условий $t^{0l}_{i_{\text{пр}}} \notin \left[t^{0l}_{i'_{j'}}, \overline{t^{l}_{i'_{j'}}}\right]$ и $t^{0l}_{i_{\text{пр}}} \in \left[\overline{t^{l}_{i_{j-1}}}, t^{0l}_{i'_{j'}}\right]$, при их выполнении модифицируется значение $t^{0l}_{i'_{j'}}$ в соответствии с (7).

4. После добавления данных i'-го типа в последовательности π_2^l они удаляются из множеств $N'_{\text{пр}}$ и $N_{\text{пр}}$: $N'_{\text{пр}} = N'_{\text{пр}} \setminus \{i'\}$; $N_{\text{пр}} = N_{\text{пр}} \setminus \{i'\}$. Проверяется условие $N'_{\text{пр}} \neq \varnothing$. В случае его выполнения реализуется переход на шаг 1, при невыполнении все данные i-х типов ($i \in N'_{\text{пр}}$) размещаются в π_2^l и происходит останов алгоритма.

После определения начального решения для данных $i \in N'_{\text{пр}}$ осуществляется определение на его основе эффективных порядков их обработки. При этом последовательности π_2^l содержат данные в порядке возрастания значений идентификаторов этих типов. После формирования начального решения для последовательностей π_2^l количество данных в π^l ($l = \overline{1, L}$) одинаково.

Для определения эффективных порядков обработки данных в последовательностях π_2^l на основе сформированного начального решения выполним повторную инициализацию множества $N'_{\text{пр}}$ следующим образом: $N'_{\text{пр}} = \bigcup_{l=1}^{L} \pi_2^l$, тогда множество $N'_{\text{пр}}$ — это типы данных, которые размещены в последовательностях π_2^l и порядок которых в π_2^l будет изменен. Полученное множество $N'_{\text{пр}}$ упорядочивается по возрастанию идентификаторов типов данных $i \in N'_{\text{пр}}$.

Для определения порядков обработки данных в последовательностях π_2^l использован подход, предложенный в [11–14]. В текущем локально оптимальном решении π рассматриваемые данные i'-го типа занимают в последовательностях π_2^l некоторую начальную позицию j', которая в процессе поиска эффективного решения будет изменяться. В результате изменений j'-х позиций данных i'-го типа в π_2^l определяется окрестность $O_1(\pi)$ текущего решения π (расписания обработки данных), состоящая из решений, отличающихся от исходного решения π тем, что в одной из последовательностей π_2^l данные i'-го типа (если они входят в эти последовательности) перемещаются на одну позицию ближе к началу. При одновременном изменении положения данных i'-го типа сразу в нескольких последовательностях π_2^l [1] выполнен переход к окрестности $O_k(\pi)$ ($k = \overline{2, k_{\max}}$). По аналогии с [1] на s-м шаге алгоритма данные i' перемещаются на одну позицию к началу в каждой последовательности π_2^l ($i' \in \pi_2^l$) либо в совокупности последовательностей π_2^l, количество которых определяется значением k ($k = \overline{1, k_{\max}}$).

В формируемых окрестностях $O_k(\pi)$ ($k = \overline{1, k_{\max}}$) реализуется определение более эффективных решений, чем текущее решение π. Затем для полученного нового локально оптимального решения также рассматриваются окрестности $O_k(\pi)$ ($k = \overline{1, k_{\max}}$), состоящие из построенных в соответствии с описанным выше подходом расписаний. Если в сформированных окрестностях $O_k(\pi)$ более эффективное решение не найдено, реализуется переход к следующему типу данных $i \in N'_{\text{пр}}$, позиция j которых в последовательностях π_2^l будет изменяться.

Для обоснования метода формирования решений по порядкам обработки данных в π_2^l введено множество L_S номеров сегментов, в последовательностях которых находятся данные рассматриваемого типа i'. Изменению подлежат позиции данных i'-го типа в последовательностях π_2^l, для которых $i' \in \pi_2^l$. Вид множества L_S следующий: $L_S = \{l_1, l_2, \ldots, l_{L_S}\}$, каждый индекс l последовательностей π_2^l такой, что $i' \in \pi_2^l$, и имеет в множестве L_S соответствующий ему номер. Выполнено определение способа задания комбинаций индексов l ($l \in L_S$) последовательностей π_2^l, в которых на текущем шаге алгоритма изменяются позиции данных i' в соответствии с метрикой k окрестности O_k рассматриваемого решения $\pi(s)$. Определение индексов l последовательностей π_2^l ($l \in L_S$), в которых выполняется изменение положения данных i', реализуется путем изменения значений их номеров таким образом, чтобы осуществить последовательный

перебор всех комбинаций этих индексов с учетом метрики k. При достижении номерами индексов $l \in L_S$ последовательностей π_2^l максимальных значений, задаваемых с учетом значения метрики k, формирование новых решений, входящих в окрестность, заканчивается.

Расписание $\pi^*(s)$ характеризуется матрицами $P^l(s)$, $\left(t_{ji}^{0l}(s)\right)$ $(l = \overline{1,L})$, поэтому локально оптимальное решение обозначим как $\left(P^l(s); \left(t_{ji}^{0l}(s)\right) | l = \overline{1,L}\right)^*$. Алгоритм оптимизации видов в последовательностях π_2^l имеет следующий порядок шагов.

1. Решение по порядкам обработки данных в последовательностях π_2^l $(l = \overline{1,L})$, полученное после размещения в π_2^l данных i-х типов
$$\left(i \in N_{\text{пр}} \cap \left(\bigcup_{l=1}^{L} \pi_1^l\right)\right),$$
определяется как локально оптимальное. Обозначим его как $\pi^*(s)$ и вычислим значение критерия f.
2. Из множества $N'_{\text{пр}}$ выбирается тип данных i' по условию $i' = \min\left\{i | i \in N'_{\text{пр}}\right\}$.
3. Формируется множество L_S номеров сегментов l, для которых $i' \in \pi_2^l$. Изменению подлежат позиции данных i'-го типа в последовательностях π_2^l, для которых $l \in L_S$. Определяются значения j' позиций данных i' в последовательностях π_2^l $(l \in L_S)$.
4. Значение метрики k окрестности $O_k(\pi^*(s))$ решения $\pi^*(s)$, в которой будет выполнен поиск локально оптимального решения, задается равным 1.
5. Значение g индекса промежуточного решения инициализируется значением 1 ($s+g$ — номер промежуточного шага алгоритма, связанного с формированием решения по определению позиций данных i'-го типа в π_2^l на основе решения $\pi^*(s)$).
6. Для каждого сегмента l $(l \in L_S)$ выполняется проверка условия $(q+1) < j'$ (положение данных i' в π_2^l может быть изменено). При условии $(q+1) = j'$ для последовательности π_2^l $(l \in L_S)$ ее индекс исключается из L_S: $L_S = L_S \backslash \{l\}$.
7. В соответствии со сформулированным способом определения индексов l последовательностей π_2^l $(l \in L_S)$, в которых изменяется положение данных i'-го типа, в L_S выполняется идентификация индексов l тех последовательностей, которые рассматриваются на $(s+g)$-м шаге алгоритма. Количество индексов l последовательностей соответствует значению метрики k окрестности O_k решения $\pi^*(s)$.
8. Для определенных таким образом последовательностей π_2^l реализуется изменение положения данных i'-го типа на одну позицию ближе к их началу. Изменение j'-х позиций данных i' в π_2^l выполняется путем преобразования матриц P^l $(l \in L_S)$ следующим образом: $l = \overline{1,L}$, $p^l_{i',j'}(s+g) = 0$, $p^l_{k',j'-1}(s+g) = 0$, $p^l_{k',j'}(s+g) = 1$, где k' — индекс строки в матрице P^l, в которой элементы $p^l_{k',j'-1}$ в $(j'-1)$-м столбце равны 1 (на s-м шаге алгоритма — это предыдущая позиция для рассматриваемых данных, которую они занимают на $(s+g)$-м шаге). В результате формируются матрицы $P^l(s+g)$.
9. На основе матриц P_l $(l = \overline{1,L})$ выполняется расчет значений элементов матриц $\left(t_{ji}^{0l}\right)$ $(l = \overline{1,L})$ на $(s+g)$-м шаге алгоритма. Для всех данных i-го типа и их позиций j после вычисления значения t_{ji}^{0l} (соответственно значения $\overline{t_{ji}^l} = t_{ji}^{0l} + t_i^l$) выполняется проверка условия прерывания их обработки данными $i_{\text{пр}}$: $t_{i_{\text{пр}}}^{0l} \in \left[t_{ji}^{0l}, \overline{t_{ji}^l}\right]$. При его выполнении значение t_i^l (и значение $\overline{t_{ji}^l}$) модифицируется в соответствии с (6). Для данных i-го типа в j-й позиции в последовательностях π_2^l для вычислен-

ного значения t_{ji}^{0l} выполняется проверка условий $t_{i_{\text{пр}}}^{0l} \notin \left[t_{ji}^{0l}, \overline{t_{ji}^{l}}\right]$ и $t_{i_{\text{пр}}}^{0l} \in \left[\overline{t_{j-1,i}^{l}}, t_{ji}^{0l}\right]$, при их выполнении модифицируется значение t_{ji}^{0l} (и значение $\overline{t_{ji}^{l}}$) с использованием (7). В итоге формируется решение $\left(P^l(s+g), \left(t_{ji}^{0l}(s+g)\right) | l = \overline{1, L}\right)$, для которого рассчитывается значение $f(s+g)$.

10. Для решений $\left(P^l(s), \left(t_{ji}^{0l}(s)\right) | l = \overline{1, L}\right)^*$ и $\left(P^l(s+g), \left(t_{ji}^{0l}(s+g)\right) | l = \overline{1, L}\right)$ вычисляются значения градиентов критерия f [15]: либо левого ${}^-\nabla_g f(s) = [f(s+g) - f(s)] \le 0$, либо правого ${}^+\nabla_g f(s) = [f(s+g) - f(s)] > 0$. В случае выполнения на $(s+g)$-м шаге алгоритма условия ${}^-\nabla_g f(s) \le 0$ индекс g группы последовательностей π^l добавляется в множество N_s^p индексов направлений отрицательных левых градиентов: $N_s^p = N_s^p \cup \{g\}$. В результате формируется множество N_s^p направлений g, для которых ${}^-\nabla_g f(s) \le 0$). Реализуется изменение значения шага g: $g = g + 1$.
11. Реализуется проверка условия окончания формирования решений, входящих в окрестность $O_k(\pi^*(s))$. Это условие сформулировано при обосновании способа определения индексов l последовательностей π_2^l, в которых изменяется положение данных i'-го типа. При его выполнении реализуется переход к шагу 12, в противном случае (при возможности формирования дополнительных решений в окрестности $O_k(\pi^*(s))$ с метрикой k) осуществляется переход на шаг 7.
12. Если $N_s^p = \varnothing$, то реализуется переход к шагу 15. При $N_s^p \ne \varnothing$ в множестве N_s^p выбирается направление g', для которого модуль градиента ${}^-\nabla_g f(s) \le 0$ максимальный: $g' \to \max\limits_{g} (|{}^-\nabla_g f(s)|)$ при ${}^-\nabla_g f(s) \le 0$. Полученное решение $\left(P^l(s+g'), \left(t_{ji}^{0l}(s+g')\right) | l = \overline{1, L}\right)$ фиксируется как локально оптимальное, на основе которого формируются последующие решения:

$$\left(P^l(s),\ \left(t_{ji}^{0l}(s)\right) | l = \overline{1, L}\right)^* = \left(P^l(s+g'),\ \left(t_{ji}^{0l}(s+g')\right) | l = \overline{1, L}\right).$$

Модифицируется значение номера шага алгоритма $s = s + g$. Значения индексов j' столбцов матриц $\left(P^l\right)$, идентифицирующих местоположение в π_2^l данных i'-го типа для последовательностей, рассматриваемых на текущем шаге алгоритма, модифицируются: $j' = j' - 1$.

13. Выполняется проверка условия $(q+1) < j'$ для каждой последовательности π_2^l $(l \in L_S)$.
14. При невыполнении условия $(q+1) < j'$ для последовательностей π_2^l $(l \in L_S)$, в которых осуществлялось изменение порядка данных на $(s+g)$-м шаге алгоритма, индексы l исключаются из множества L_S: $L_S = L_S \backslash \{l\}$. При выполнении условия $L_S \ne \varnothing$ реализуется переход к шагу 4; при $L_S = \varnothing$ отсутствуют последовательности π_2^l $(l \in L_S)$, в которых может быть изменен порядок обработки данных, выполняется переход к шагу 17.
15. Если $N_s^p = \varnothing$, то в окрестности O_k решения $\left(P^l(s), \left(t_{ji}^{0l}(s)\right) | l = \overline{1, L}\right)^*$ (расписания $\pi^*(s)$) новое решение, уменьшающее значение критерия, не найдено; текущее значение метрики модифицируется: $k = k + 1$, если $k \le k_{\max}$, тогда выполняется переход к шагу 5.
16. При выполнении условия $k > k_{\max}$ в окрестностях O_k локально оптимального решения $\left(P^l(s), \left(t_{ji}^{0l}(s)\right) | l = \overline{1, L}\right)^*$ более эффективное решение не найдено, поэтому локально оптимальное решение $\left(P^l(s), \left(t_{ji}^{0l}(s)\right) | l = \overline{1, L}\right)^*$ интерпретируется как

эффективное решение по определению позиции данных i'-го типа в π^l $(l = \overline{1, L})$. При реализации условия $k > k_{\max}$ выполняется переход к шагу 17.

17. Выполняется модификация множества $N'_{\text{пр}}$: $N'_{\text{пр}} = N'_{\text{пр}} \backslash \{i'\}$. Если в результате $N'_{\text{пр}} = \varnothing$, то все данные $i \in N'_{\text{пр}}$ размещены в последовательностях π^l $(l = \overline{1, L})$ эффективным образом и реализуется переход на шаг 18. При $N'_{\text{пр}} \neq \varnothing$ не все данные размещены в последовательностях π^l $(l = \overline{1, L})$, реализуется переход на шаг 2.
18. Останов алгоритма.

При интерпретации алгоритма в последовательностях π^l $(l = \overline{1, L})$ (в последовательностях π_2^l) определены эффективные позиции данных i-го типа, для которых $i \in N_{\text{пр}} \cap \left(\bigcup\limits_{l=1}^{L} \pi_1^l \right)$. На втором этапе построения динамического расписания реализуется определение последовательностей π_2^l при размещении в них данных i-х типов, для которых перед началом построения динамического расписания выполнялось условие $i \in N_{\text{пр}} \& \left(i \notin \bigcup\limits_{l=1}^{L} \pi_1^l \right)$. Данные, типы которых входят в множество $N_{\text{пр}}$, полученное после реализации первого этапа алгоритма построения динамического расписания, должны быть добавлены в каждую из последовательностей π_2^l $(l = \overline{1, L})$, тогда для определения их позиций в этих последовательностях применен "жадный" алгоритм, предложенный в [1]. Отличием в использовании данного алгоритма при реализации второго этапа построения динамического расписания является то, что добавляемые на каждом шаге алгоритма в π_2^l данные могут занять любую позицию в них, начиная с $(q+1)$-й.

Рассмотренные первый и второй этапы алгоритма построения динамических расписаний с учетом поступления на обработку данных $i_{\text{пр}}$ с приоритетом $R_{\text{пр}} > R_i$ реализованы программно. С использованием разработанной программы выполнены исследования особенностей процесса формирования динамических расписаний. Возможность построения динамического расписания обусловливается различными порядками обработки данных в последовательностях π_2^l $(l = \overline{1, L})$ в исходном статическом расписании. Причиной формирования различных порядков обработки данных является несогласованность длительностей обработки на различных сегментах конвейера. На одном из сегментов максимальная длительность обработки данных должна превышать максимальную длительность обработки этих данных на предшествующем сегменте. Длительности обработки данных на остальных сегментах на формирование динамических расписаний влияния не оказывают.

При исследовании метода построения динамических расписаний рассматривалась задача с пятью сегментами конвейера $(L = 5)$ и семью типами данных $(n = 7)$. При увеличении количества типов данных тенденция изменения эффективности формирования динамического расписания сохраняется. В качестве сегмента, максимальная длительность обработки данных на котором превышает максимальные длительности обработки данных на других сегментах, задан $(l = 3)$-й сегмент. Для задания степени несогласованности длительностей обработки данных на $(l = 3)$-м и предшествующем ему сегментах в рассмотрение введено отношение $\max\limits_i (t_i^3) \,/\, \max\limits_i \left(t_i^l\right)$, где t_i^3 — длительность обработки данных i-го типа на $(l = 3)$-м сегменте конвейера, t_i^l — длительность обработки данных i-го типа на l-х сегментах конвейера $(l \neq 3)$. Для задания степени несогласованности длительностей обработки данных на сегментах конвейера $(l \neq 3)$ в рассмотрение введено отношение $\max\limits_i \left(t_i^l\right) \,/\, \min\limits_i \left(t_i^l\right)$. При проведении экспериментов значения отношений $\max\limits_i (t_i^3) \,/\, \max\limits_i \left(t_i^l\right)$ и $\max\limits_i \left(t_i^l\right) \,/\, \min\limits_i \left(t_i^l\right)$ изменялись от 2 до 10 с шагом 2.

При этом $\min\limits_{i}(t_i^3) = \min\limits_{i}\left(t_i^l\right)$. Для задания несогласованности длительностей обработки данных $i_{\text{пр}}$-х и i-х типов на l-х сегментах конвейера в рассмотрение введено отношение $\min\limits_{l}\left(t_{i_{\text{пр}}}^l\right) / \min\limits_{i,l}\left(t_i^l\right)$. Для задания несогласованности длительностей обработки данных $i_{\text{пр}}$ на l-х сегментах конвейера в рассмотрение введено отношение $\max\limits_{l}\left(t_{i_{\text{пр}}}^l\right) / \min\limits_{l}\left(t_{i_{\text{пр}}}^l\right)$. При проведении эксперимента значения отношений $\min\limits_{l}\left(t_{i_{\text{пр}}}^l\right) / \min\limits_{i,l}\left(t_i^l\right)$ и $\max\limits_{l}\left(t_{i_{\text{пр}}}^l\right) / \min\limits_{l}\left(t_{i_{\text{пр}}}^l\right)$ изменялись от 2 до 10 с шагом 2.

Для определения работоспособности алгоритма выполнены исследования задач модификации статических расписаний с учетом поступления данных $i_{\text{пр}}$-го типа на обработку в моменты времени $d_{i_{\text{пр}}} > 0$ и формирования динамических расписаний обработки данных i-х и $i_{\text{пр}}$-х типов (с разными приоритетами). Модификация статических расписаний предполагает, что порядок обработки данных в последовательностях π_1^l $(l = \overline{1, L})$ не изменяется. Для данных i-го типа выполняется только модификация значений t_i^l в соответствии с выражением (6) либо модификация значений t_{ji}^{0l} в соответствии с выражением (7) при выполнении введенных условий, а также модификация значений t_{ji}^{0l} для следующих за ними данных.

Реализация метода построения динамических расписаний позволяет изменить виды последовательностей π^l $(l = \overline{1, L})$ с учетом поступления данных $i_{\text{пр}}$ на обработку. Выполнялось построение динамических расписаний при поступлении единственных данных $i_{\text{пр}}$ на обработку и поступлении пары данных $i_{\text{пр}}$ в различные моменты времени $d_{i_{\text{пр}}} > 0$. Анализ полученных результатов показал, что построение динамических расписаний (в сравнении с модифицированными статическими расписаниями) позволяет сократить время простоя сегментов конвейера на 5–45 % при поступлении единственных данных $i_{\text{пр}}$ и на 5–30 % при поступлении пары данных $i_{\text{пр}}$.

Также реализовано исследование применения известных эвристических правил, учитывающих приоритеты данных для построения динамических расписаний [3, 4]. В этом случае при поступлении данных $i_{\text{пр}}$ на обработку в каждой из последовательностей π^l $(l = \overline{1, L})$ определяются последовательности π_2^l, данные в которых упорядочиваются в соответствии со значениями весов w_i^l (где w_i^l — вес данных i-го типа в последовательности π^l при $i = \overline{1, n}$, $l = \overline{1, L}$) , определяемых по одному из указанных ниже способов:

1) $w_i^l = \dfrac{R_i}{t_i^l}$, где R_i — приоритет данных, а t_i^l — длительность их обработки на l-м сегменте;
2) $w_i^l = \dfrac{R_i}{t_i^l}\left[1 - \dfrac{\left(D_i - \hat{t}_i^l - t\right)^+}{h\hat{t}_i^l}\right]^+$, где D_i — директивный срок окончания обработки данных i-го типа $(i = \overline{1, n})$, задаваемый в качестве входного параметра, $\hat{t}_i^l$ — оставшееся время обработки данных i-го типа на сегментах конвейера, начиная с текущего l-го и заканчивая L-м (определяется следующим образом: $\hat{t}_i^l = \sum\limits_{q=l}^{L} t_i^q$), t — значение текущего времени, h — весовой коэффициент, $[A]^+$ — операция, определяемая как $\max(0, A)$.

Выполненные исследования показали, что эвристическое правило упорядочивания данных в последовательностях π^l $(l = \overline{1, L})$, использующее выражение первого вида, не позволяет при формировании динамических расписаний получать решение, лучшее,

чем модифицированное статическое расписание. Формирование расписания с использованием данного эвристического правила не обеспечивает уменьшения общих простоев сегментов конвейера по сравнению с модифицируемым статическим расписанием.

Использование эвристического правила на основе выражения второго вида позволяет получить решения, улучшающие значения критерия по сравнению с модифицированным статическим расписанием на 2–5 %. При этом решения, лучшие, чем модифицированное статическое расписание, получены только для отдельных комбинаций значений исходных данных. Не наблюдается общая тенденция улучшения решений с использованием эвристических правил. Таким образом, предложенный метод формирования динамических расписаний обработки данных с разными приоритетами позволяет получить результаты, которые на 25–40 % лучше, чем методы на основе эвристических правил.

Предложенный подход может быть использован при оперативном мониторинге поверхности Земли по данным дистанционного зондирования. В случае идентификации наличия источника загрязнения (пожара либо другого события, требующего особого внимания) запланированная штатная обработка снимков должна быть прервана для обработки данных (информации), детализирующих это событие, которые поступают с другого спутника.

Выводы

Предложен метод построения динамических расписаний в конвейерных системах с учетом поступления в моменты времени $d_{i_{\text{пр}}} > 0$ данных с приоритетами $R_{i_{\text{пр}}} > R_i$.

Определены условия формирования последовательностей обработки данных на соответствующих сегментах конвейера, которые оптимизируются предложенным методом составления расписаний. Разработан способ формирования окрестностей для текущего локально оптимального решения, в которых будет выполняться поиск лучших, чем текущее, решений.

Предложенный метод формирования динамических расписаний реализован программно, выполнены исследования целесообразности его применения в рассматриваемой задаче. Установлено, что использование метода позволяет уменьшить простои оборудования при обработке данных (в случае реализации рассматриваемых возмущающих воздействий) на 5–40 % по сравнению со модифицированным статическим расписанием, в котором порядок обработки данных в соответствии с возмущающими воздействиями не изменяется.

Список литературы / References

[1] **Кротов К.В.** Градиентный метод составления статических расписаний для конвейерных систем, основывающийся на жадных стратегиях // Вест. НГУ. Информ. технологии. 2015. Т. 13, № 1. С. 55–73.
Krotov, K.V. Gradient method of making static schedules for conveyor systems based on greedy strategies // Novosibirsk State Univ. J. of Information Technologies. 2015. Vol. 13, No. 1. P. 55–73. (In Russ.)

[2] **Топорков В.В.** Модели распределенных вычислений. М.: Физматлит, 2004. 320 с.
Toporkov, V.V. Models of distributed computing. Moscow: Fizmatlit, 2004. 320 p. (In Russ.)

[3] **Morton, T.E., Pentico, D.W.** Heuristic scheduling systems: with applications to production systems and project management. Wiley Series in Engineering & Technology Management. John Wiley & Sons, 1993. 720 p.

[4] **Jakobovic, D., Budin, L.** Dynamic scheduling with genetic programming // Genetic Programming, 9th European Conf., EuroGP2006. Lecture Notes in Computer Science, LNCS 3905. P. 238–249.

[5] **Barbosa da Silva, E., Costa, G.M., Fatima de Souza da Silva, M., Pereira, F.H.** Simulition study of dispatching rules in stochastic job shop dynamic scheduling // World J. of Modelling and Simulation. 2014. Vol. 10, No. 3. P. 231–240.

[6] **Frachtenberg, E., Feitelson, D.G., Ferrandez, Ju., Petrini, F.** Parallel job scheduling under dynamic workloads // 9th Workshop in Job Scheduling Strategies for Parallel Processing. JSSPP2003. Seattle, WA, USA. 24 June. 2003.

[7] **Lee, S., Kim, H., Lee, J.** A soft aperiodic task scheduling algorithm in dynamic priority system // 2nd Intern. Workshop on Real-Time Computing Systems and Applications. October 25–27, 1995. Japan. Tokyo, 1995. P. 68–72.

[8] **Cosnard, M., Jeannot, E., Rougent, L.** Low memory cost dynamic scheduling of large coarse grain task graphs // Intern. Parallel Processing Symp./Symp. on Parallel and Distributed Processing (IPPS/SPDP'98). USA, Florida, Orlando, 1998. P. 524–530. Available at: http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=5485&punumber%3D5485%26filter%3DAND%28p_IS_Number%3A14764%29%26pageNumber%3D3&pageNumber=4

[9] **Dahal, K., Hossain, A., Varghese, B., Abraham, A., Xhafa, F., Daradoumis, A.** Scheduling in multiprocessor system using genetic algoritm // 7th Intern. Conf. on Computer Inform. System and Industrial Management Application. Ostrava. Czech Republic, June 26–28. 2008. P. 281–286.

[10] **Lin, S.-C., Goodman, E.D., Punch, W.F.** A genetic algorithm approach to dynamic job shop scheduling problems // 7th Intern. Conf. on Genetic Algoritm. East Lansing. USA, July 19–23. 1997. P. 481–488.

[11] **Кочетов Ю.А., Младенович Н., Хансен П.** Локальный поиск с чередующимися окрестностями // Дискретный анализ и исследование операций. 2003. Т. 10, № 1. С. 11–43.
Kochetov, Yu.A., Mladenovich, N., Khansen, P. Local search with alternating neighborhoods // Diskretnyi Analiz i Issledovanie Operatsii. Series 2. 2003. Vol. 10, No. 1. P. 11–43. (In Russ.)

[12] **Кононова П.А.** Нижние и верхние оценки длины оптимального расписания презентаций медиа-объектов // Дискретный анализ и исследование операций. 2012. Т. 19, № 1. С. 59–73.
Kononova, P.A. Lower and upper bounds for the optimal makespan in the multimedia problem // Diskretnyi Analiz i Issledovanie Operatsii. 2012. Vol. 19, No. 1. P. 59–73. (In Russ.)

[13] **Кононова П.А., Кочетов Ю.А.** Локальный поиск с чередующимися окрестностями для задачи Джонсона с пассивным буфером // Дискретный анализ и исследование операций. 2012. Т. 19, № 5. С. 63–82.
Kononova, P.A., Kochetov, Yu.A. Variable neighborhood search for two machine flowshop problem with a passive prefetch // Diskretnyi Analiz i Issledovanie Operatsii. 2012. Vol. 10, No. 5. P. 63–82. (In Russ.)

[14] **Кононова П.А.** Локальный поиск с чередующимися окрестностями для задачи Джонсона с пассивным буфером: Дис. ... канд. физ.-мат. наук. Новосибирск: Ин-т математики им. С.Л. Соболева, 2012. 106 с.

Kononova, P.A. Local search with alternating neighborhoods for the Johnson problem with passive buffer: Dissertatsionnaya Rabota Kandidata Fiz.-Mat. Nauk. Novosibirsk: Sobolev Institute of Mathematics, 2012. 106 p. (In Russ.)

[15] **Ковалев М.М.** Матроиды в дискретной оптимизации. М.: Едиториал УРСС, 2003. 224 с.
Kovalev, M.M. Matroids in discrete optimization. Moscow: Editorial URSS, 2003. 224 p. (In Russ.)

Поступила в редакцию 9 марта 2016 г.,
с доработки — 10 июня 2016 г.

Gradient method for construction dynamic data processing schedules in a conveyor system with various data arrival time and different priorities

Krotov, Kirill V.*, Krotova, Tatiana Yu.

Sevastopol State University, Sevastopol, 290053, Russia
*Corresponding author: Krotov, Kirill V., e-mail: krotov_k1@mail.ru

Nowadays the task of handling of pipelined data using various priority programs is an actual task. High priority data is received for processing by the system at later time points. It interrupts the processing of the previously received low priority data on the conveyor segments. As the disturbance, it violates the planned course of the computational process. Reducing the effect of these disturbances is possible by building dynamic scheduling that accounts for entering the high-data system. The model of the computer data processing with different priorities, type of endpoint of generated dynamic scheduling, as well as a method of constructing dynamic scheduling along with the search for locally optimal solutions within the neighbourhood with a variety of metrics are executed for the task. The formulated method for constructing dynamic scheduling is based on the conditions that allow the determination the important data in the sequences on each segment of the pipeline. The order in these sequences can be changed, and the data order may be not. For data, for which the order of the processing sequences can be changed, a dynamic schedule based on the received high-priority data is arranged. Comparison of the results based on the dynamic scheduling that uses this method and the results of the build schedules, which use the heuristic rules is completed. The heuristic rules take into account the priorities for organizing data in a sequence of processing them on the conveyor segments. Studies have shown that this method for construction of the dynamic scheduling is 25–40 % more efficient then the procedure that uses heuristic rules that take into account these priorities.

Keywords: conveyor system, data processing, programs execution schedule, greedy algorithm, dynamic scheduling, priorities.

Received 9 March 2016
Received in revised form 10 June 2016