DOI: 10.6060/ivecofin.2021493.560
УДК: 519.688

РЕШЕНИЕ ЗАДАЧ ПЛАНИРОВАНИЯ РАБОЧЕГО ЦЕХА С МАШИНАМИ, ДОПУСКАЮЩИМИ ОДНОВРЕМЕННУЮ ОБРАБОТКУ ТРЕБОВАНИЙ

Д.И. Коровин, Л.Н. Чернышов

Дмитрий Игоревич Коровин (ORCID 0000-0001-9941-0322),
Финансовый Университет при Правительстве Российской Федерации, Ленинградский пр., 49, Москва, 125993, Россия
E-mail: dikorovin@fa.ru

Лев Николаевич Чернышов (ORCID 0000-0002-1512-4052)
Московский авиационный институт, Волоколамское ш., 4, Москва, 125993, Россия
Финансовый Университет при Правительстве Российской Федерации, Ленинградский пр., 49, Москва, 125993, Россия
E-mail: levchern@gmail.com

В статье приводится авторский алгоритм для решения одной задачи теории расписаний, которая возникла в реализации производственной деятельности. Метод основан на принципе раскраски графов и позволяет реализовать одновременную обработку нескольких деталей на одном рабочем месте. В кратком анализе задач теории расписаний определяется место рассматриваемой постановки задачи в общей классификации задач. Разработан алгоритм и программа, решающие эту задачу для различных критериев оптимальности. Реализованы два варианта программы. Первый непосредственно следует структурам данных и последовательности действий метода, основанного на раскраске графов. Во втором варианте используются структуры линейного представления графа, а также введены многотактовые операции, что позволило повысить эффективность алгоритма. Приведены временные характеристики выполнения программы на разном количестве деталей для двух вариантов программы. В заключение рассматриваются перспективы развития программы и сферы ее применения.

Ключевые слова: теория графов, теория расписаний, окраска графа, садка, алгоритм построения расписаний.

SOLUTION OF TASKS OF WORKSHOP PLANNING WITH MACHINES ALLOWING SIMULTANEOUS PROCESSING OF REQUIREMENTS

D.I. Korovin, L.N. Chernyshov

Dmitriy I. Korovin
Financial University under the Government of the Russian Federation, Leningradsky Ave, 49, 125993, Moscow, Russia
E-mail: dikorovin@fa.ru
Lev N. Chernyshov
Moscow Aviation Institute (National Research University), Volokolamskoe Road, 4, Moscow,125993, Russian Federation
Financial University under the Government of the Russian Federation, Leningradsky Ave, 49, 125993, Moscow, Russia
Email: levchern@gmail.com

The article presents the author's algorithm for solving one of the problems in the theory of scheduling that arose in the implementation of production activities. The method is based on the principle of graph coloring and allows simultaneous processing of several parts in one workplace. In a brief problem analysis of schedule theory, the place of the task in question is determined in the general classification of problems. An algorithm and program have been developed that solve this problem for various optimality criteria. Two versions of the program have been implemented. The first follows directly the data structures

and the sequence of actions of the graph coloring method. In the second version, the structures of the linear representation of the graph are used, as well as multi-stroke operations are introduced, which made it possible to increase the efficiency of the algorithm. The time characteristics of the program execution on a different number of parts for two versions of the program are given. Finally, the prospects for the development of the program and the scope of its application are discussed.

Keywords: graph theory, scheduling theory, graph coloring, scheduling algorithm.

Для цитирования:
Коровин Д.И., Чернышов Л.Н. Решение задач планирования рабочего цеха с машинами, допускающими одновременную обработку требований. *Известия высших учебных заведений. Серия «Экономика, финансы и управление производством» [Ивэкофин]*. 2021. № 03(49). С.135-143. DOI: 10.6060/ivecofin.2021493.560

For citation:
Korovin D.I., Chernyshov L.N. Solution of tasks of workshop planning with machines allowing simultaneous processing of requirements. *Ivecofin*. 2021. № 03(49). С.135-143. DOI: 10.6060/ivecofin.2021493.560 (in Russian)

ВВЕДЕНИЕ

В организации производственных процессов на современных российских предприятиях, которые продолжают функционировать в сложной экономической обстановке, происходят глобальные изменения. Если основы регулирования, режима организации технологических линий на предприятиях, построенных еще в советское время, реализовывались для решения одних оптимизационных задач (выполнения поставленного плана) в условиях ограничений одного типа (ограничения строительных проектов, определяющих соответствующее оборудование шагу колонн, ограничений логистического характера, технологических ограничений определенного оборудования), то современное положение дел диктует свои условия. Обрабатывающие и сборочные производства машиностроительных предприятий при получении производственного заказа на выпуск (что в наше время уже достижение) могут решать следующие задачи.

Необходимо организовать режим подачи заготовок, ресурсов, оснастки на технологические линии так, чтобы время исполнения заказа соответствовала контрактным обязательствам. Специфика процесса в том, что на многопредметных поточных линиях технологические режимы могут организовываться с простоями, так как, либо соответствующий станок (рабочее место) будет ожидать прибытие необходимого для производственной операции ресурса, который находится в процессе на другом рабочем месте, либо рабочее место находится в состоянии переналадки для обработки на этом станке другого типа детали. Частными подзадачами можно считать создание таких расписаний подачи деталей, которые: а) реализуют все необходимые плановые операции в соответствующем порядке и время их полной реализации минимизируется; б) реализуется такой режим расписания подачи деталей на рабочие места, при котором суммарные простои оборудования минимальны; в) реализуется такой режим расписания подачи деталей на рабочие места, при котором число переналадок минимально.

Проблема нахождения расписаний в общем виде не решена. К сожалению, вычислительная сложность алгоритмов, реализующих расписания слишком высокая, а входные условия (количество и типы станков, характер обработки деталей, время обработки, количество предметов – типов деталей, и конечно, объемы планов производства) слишком вариабельны.

Обзор задач. В теории расписаний [1] в общем случае задача планирования формулируется так: Задано некоторое множество работ (требований) $J = \{J_1, \ldots, J_n\}$ с определённым набором характеристик: длительность обработки требования (простейший случай), стоимость обработки требования, момент поступления требования, директивный срок окончания обслуживания требования. Задано некоторое множество машин (приборов, станков) $M = \{M_1, \ldots, M_m\}$, на которых требования должны обслуживаться в соответствии с некоторым порядком. Необходимо построить расписание, которое минимизирует время выполнения работ, стоимость работ и т. п. Расписание – указание, на каких машинах и в какое время должны обслуживаться требования (выполняться работы).

По дисциплине выполнения работ на машинах можно выделить четыре основные класса задач: 1) открытая линия – порядок обслуживания на этих машинах произвольный (Open shop), 2) рабочий цех – для каждого требования задан маршрут обработки (Job shop), 3) потоковая линия – все машины упорядочены и каждое требование проходит все машины в этом

порядке (Flow shop), 4) задача с директивными сроками – для каждого требования задан момент поступления, время обслуживания и директивный срок окончания обслуживания. Задачи также классифицируются по возможности прерывания процесса выполнения требования. В каждом классе задач также рассматриваются различные критерии, по которым производится оптимизация. Рассматриваемая в статье задача относится к классу рабочего цеха без прерываний.

Решением задач «Job shop» даже с одной машиной без прерываний является некоторая перестановка требований и для произвольной целевой функции эти задачи NP-полны. Этот результат был получен в одной из ранних публикаций еще в 1976 г. [2] при числе машин не менее трех.

Полная классификация постановок задач и публикаций по теории расписаний размещена на сайте [3]. Классификация учитывает типы задач: параллельно работающие идентичные машины (тип P); машины, работающие с определенными скоростями (тип Q); машины с фиксированным временем обработки отдельных требований (тип R); машины, для которых не важен порядок обработки требований (тип O); потоковая линия, где каждое требование выполняется на фиксированной последовательности машин (тип F) и, наконец, рабочий цех, в котором требования состоят из некоторой последовательности операций, выполняемых на разных машинах (тип J). Целью задачи планирования является, как правило, минимизация общего времени обработки. Для каждого класса задач отдельно рассматриваются случаи с разным количеством машин. Также задачи отличаются добавлением различных ограничений: отношений приоритетов, группировкой требований в пакеты, ограничений по времени обработки и т.п.

Рассматриваемый в статье тип задачи относится к типу J. При этом в качестве дополнительного условия включена возможность наличия машин, которые одновременно обрабатывают несколько требований. Количество таких требований может задаваться диапазоном значений. В приведенной в работе [3] классификации таких условий нет.

ПОСТАНОВКА ЗАДАЧИ

Нами рассматривалась следующая задача. Необходимо было создать алгоритм организации подачи деталей в экспериментальном цеху термомеханической обработки, в котором N1 рабочих мест с N2 типами операций (каждая операция это или станок, или печь, или позиция контрольной операции). Каждая операция характеризуется количественными показателями одновременной обработки деталей. Это актуально для печей, которые могут вмещать разное количество деталей одного типа $a_{(\text{номер печи-операции})(\text{тип детали})}$, для многих операций $a_{ij}= 1$.

Заказ представляет собой набор V= (V1, V2, …,Vs) s деталей, которые должны быть выполнены в соответствии с технологическими планами. Представим их в виде таблицы с полями:номер операции; норма времени исполнения; назначенная группа оборудования (номер): предельный показатель садки $a_{(\text{номер печи-операции})(\text{тип детали})}$).

Указывается время переналадок s ой операции с обработки i-ой детали на обработку j-ой детали t_{sij}.

Нетривиальной задачей при даже не столь существенных значения Vi, N1 и N2 организация хотя бы одного непротиворечивого расписания. Нами предлагается алгоритм «умного» перебора создаваемых непротиворечивых расписаний, в результате которого отбираются варианты с оптимальными показателями:

F1 – полного времени выполнения заказа,

F2 – времени простоя оборудования,

F3 – количеством переналадок оборудования.

Мы предполагаем, что все операции однотактовые, и, если технологическое действие занимает больше времени, мы разделим его на несколько однотактных операций. Необходимо указать алгоритм для создания возможных расписаний использования рабочих мест и выбора оптимального.

МАТЕМАТИЧЕСКИЕ МЕТОДЫ, НЕОБХОДИМЫЕ ДЛЯ РЕАЛИЗАЦИИ АЛГОРИТМА

Мы применим некоторые методы, используемые в теории графов [5]. Граф называется вершинно-К-раскрашенным, если его вершины можно раскрасить в К цветов, так что никакие две соседние вершины не будут одного цвета. Минимальное число К, для которого граф G является вершинным К-раскрашенным, называется хроматическим числом графа. Раскрасить граф можно по следующему алгоритму:

1. Найдем количество соединений всех вершин в карте (степени вершин).

2. Рассмотрим вершины в порядке убывания количества связей.

3. Раскрасим цвет №1. Рассмотрим вершины последовательно. Если выбранная вершина не окрашена, и также нет связи с вершинами, окрашенными в этот цвет, покрасим ее в этот цвет.

4. Если все вершины пройдены, но не окрашены, повторим шаг 3 с другим цветом, пока не будут окрашены все вершины.

Этот алгоритм можно использовать для построения расписания. Вершины, для которых

существует связь – это действия, которые нельзя выполнять одновременно из-за использования одних и тех же заданий или работы над одним и тем же объектом.

В нашем случае возникает дополнительное условие: некоторые действия не могут быть выполнены, если другие работы не были выполнены с партией деталей, которая предшествует выполнению деталей, рассматриваемых в этой партии. Для этого мы вводим в алгоритм понятие направленной связи – две угловые точки А и В соединяются направленной связью А→В, если событие В не может произойти до события А. На графе мы показываем это стрелкой, преобразуя исходный граф в ориентированный граф.

Алгоритм окраски графика следующий. Помним, что вершины – это операции, нарисованные одним цветом, должны выполняться одновременно.

1. Выберите вершину, на которой нет стрелки (они говорят, что их полустепени входа равна нулю) и раскрасьте ее в цвет номер один.

2. Производим обход вершин в заранее определенном порядке.

3. Начинаем процесс окрашивания в текущий цвет. Если рассматриваемая вершина не окрашена, не имеет соединений с вершинами, окрашенными в этот цвет, и полустепень входа равна нулю, то красим ее в этот цвет. Удаляем (если есть) все ребра-стрелки, исходящие из этой вершины.

4. Если все вершины показаны, но не окрашены, повторяем шаг 3 с другим цветом, пока не будут окрашены все вершины.

Порядок операций обработки деталей должен регулироваться следующим правилом: каждая операция не должна предшествовать любой другой операции, выполненной ранее по технологическим планам.

Таким образом, после окраски графа мы получим следующую интерпретацию алгоритма. Так как каждая вершина – это необходимая операция с определенной заготовкой, реализуемая на заданном станке, то набор вершин, окрашенных в один цвет, является множеством операций, выполняемых одновременно. Построенная последовательность цветов четко определяет порядок выполнения операций с заданными заготовками на определенных станках.

В известных алгоритмах возникала проблема одновременной обработки заготовок. В нашей модели эта проблема решена следующим образом.

В поставленной задаче одновременной обработке подвергаются одинаковые детали при их обжиге в печах. Количество деталей, одновременно обрабатываемых (садка) задано.

Такая печь в нашем алгоритме – это последовательность вершин, последовательно соединенных в цепь дугами. Количество вершин в цепи соответствует тактам обработки деталей данного типа. Первая вершина характеризуется n-входящими ребрами, где n-садка, последняя вершина цепи – n-выходящими ребрами – уже обработанными деталями.

Такой механизм позволяет включать печь, только в момент загрузки последнего элемента садки и заставляет переходить к дальнейшей обработке всех n деталей, так как вершина не будет удалена, пока не будет «стерта» последняя исходящая дуга.

АЛГОРИТМ ПОСТРОЕНИЯ РАСПИСАНИЙ

Для решения задачи построения расписания бала создана программа на языке программирования C# в среде MSVisualStudio, реализующая вышеописанный алгоритм. В первом варианте программы структуры данных и последовательность действий в точности соответствовали алгоритму. Так, граф однотактовых операций был представлен матрицей смежности, поиск вершин производился путем перебора и в программе уровень вложенности циклов достиг четырех. По результатам апробации на данных разного объемы были сделаны усовершенствования, снизившие время обработки. Во-первых, вместо однотактовых операций были введены многотактовые. Во-вторых, в раскраску стало входить не только номер такта, но и номер станка среди станков одного типа. Введены также дополнительные структуры данных, использование которых позволило снизить уровень вложенности циклов для расчета одного расписания с четырех до двух.

Усовершенствованный алгоритм описан ниже. Приведенные примеры соответствуют входным и выходным данным программы.

ПОСТАНОВКА ЗАДАЧИ

Имеется множество типов станков (машин) $M = \{M_1, \ldots, M_m\}$, предназначенных для обработки деталей. Рабочий цех состоит из станков разного типа общей численностью $Ns = ks_1 + \ldots + ks_m$, где ks_i– число станков типа M_i. Каждый станок типа M_iможет обрабатывать N_iдеталей. Для станков, обрабатывающих больше одной детали (печей), задается время обработки T_i в тактах (такт – некоторый фиксированный промежуток времени). На этих станках обрабатывается множество деталей разных ти-

пов D = {D_1, …, D_n}. Общее число деталей Nd = kd_1 + … + kd_n, где kd_j– число деталей типа D_j.

Для каждого типа деталей D_j задана последовательность операций (*план обработки*), выполняемых на станках определенных типов, P_j = ((M_{j1}, dt_{j1}), …, (M_{jq}, dt_{jq})), где M_{ji}– тип станка, dt_{ji}– число тактов, за которое деталь обрабатывается. Для операции, выполняемой на печи, число тактов dt_{ji}должно совпадать с длительностью T_{ji}, указанной для станка.

Задача составления плана обработки всех деталей заключается в построении расписания работы каждой машины. На каждом такте от 1 до tmax (общая длительность обработки всех деталей) каждая машина либо простаивает, либо обрабатывает какую-либо деталь. Таким образом, для каждого станка m_i необходимо построить последовательность $Plan_i$= [(d_{i1}, t_{i1}), (d_{i2}, t_{i2}), …], где d_{ij}, – деталь, обрабатываемая на такте t_{ij}, причем $t_{i1} < t_{i2} <$ …tmax и траектория движения (перенос детали от станка к станку) каждой детали соответствует плану обработки для ее типа и на каждом такте число задействованных машин типа M_i не превышает ks_i.

Наборы станков и наборы деталей, подлежащих изготовлению, назовем *конфигурацией рабочего цеха*. Для удобства в описании конфигураций для программы типам машин и деталей могут даваться имена.

Пример S1 входных данных программы – описание конфигурации рабочего цеха:

S1: m1, P(2, 3), m2, m3(2)
d1,1 (m1, P, m2/2)
d2,1 (m2, m3, P, m2)
d3,2 (m3, m1, P, m1, m2)
d4,2 (m1, P, m3, m1)

Первая строка содержит список типов станков и их количества. Станки m1, P, m2 представлены в единственном экземпляре, станок типа m3 – 2 экземпляра. Станок P (печь) обрабатывает по 2 детали одновременно за 3 такта. Далее описываются типы деталей, их количество и планы обработки. Деталей типов d1 и d2 по одной, а d3 и d4 – по две. Все операции на станках, которые обрабатывают по одной детали, однотактовые, кроме m2/2, выполняющейся за 2 такта.

Результат работы программы представляется в виде таблицы, где по строкам выводятся расписания станков, столбцы отмечены номерами тактов:

	1	2	3	4	5	6	7	8	9	10	11	12	13	14
m1/1	**d1/1**	d3/1	d3/2	d4/1	d4/2				d3/1	d3/2			d4/1	d4/2
P/1			**d1/1-d1/1-d1/1**			d3/1-d3/1-d3/1			d4/1-d4/1-d4/1					
P/1			d2/1-d2/1-d2/1			d3/2-d3/2-d3/2			d4/2-d4/2-d4/2					
m2/1	d2/1					**d1/1-d1/1**	d2/1			d3/1	d3/2			
m3/1	d3/1	d2/1										d4/1		
m3/2	d3/2											d4/2		

Расписание станка m1, например, будет таким: $Plan_1$ = [($d1_1$,1), ($d2_1$,2), ($d3_2$,3), ($d4_1$,4), ($d4_2$,5), ($d3_1$,9), ($d2_2$,10), ($d4_1$,13), ($d4_2$,14)]. Также по таблице можно проследить обработку деталей. Например, для m1/1 последовательность операций (выделено в таблице) : [(m1, 1, 1, 1), (P, 1, 3, 2), (m2, 1, 6, 2)].

ПРЕДСТАВЛЕНИЕ ДАННЫХ

Каждая деталь представляется кортежем sD = **(**td, nd, ti, p), где td – тип детали, nd – номер детали, ti – номер такта, p – индекс в последовательности операций, с которого начинается цепочка операций. Номер такта ti принимает значения от 0 до dt, и определяет сколько тактов обработала машина. Последовательность операций для обработки деталей определенного типа задается массивом кортежей Op = [(st, kst, col, dt), ...], где st – тип машины, kst – номер машины среди машин типа st, col – номер такта в общем процессе обработки, dt – число тактов, которое должен отработать машина. Порядок операций в этой последовательности соответствует порядку в упорядоченном множестве типов деталей, т.е. сначала располагаются планы обработки деталей типа D_1, затем планы деталей типа D_2 и т.д. Цепочка операций для каждой детали m_i с кортежем sD_i– это подпоследовательность в Op, с индексами от sD_i.p до sD_{i+1}.p-1.

Пример S2 конфигурации с тремя станками и тремя деталями:

S2: M1, M2(2)
D1, 2 (M1, M2/2)
D2, 1 (M2, M1/3, M2)

Представление данных в начале работы алгоритма:

```
mD = [(D1,1,0,0), (D1,2,0,2), (D2,1,0,4)]
Op = [(M1,1,0,1), (M2,2,0,2), (M1,1,0,1), (M2,2,0,2),
      (M2,2,0,1), (M1,1,0,3), (M2,2,0,1)]
```

Фактически Op – это вершины «нераскрашенного» графа. Поле «kst» в начале устанавливается в количество станков типа «st». После построения расписания они оказываются

«раскрашенными», т.е. вместо нулей в полях «col» будут указаны номер тактов. Поля «kst» устанавливаются в порядковый номер станка среди станков одного типа. В данном примере вершины примут следующие значения:

```
Op = ((M1,1,1,1), (M2,1,2,2), (M1,1,2,1), (M2,2,3,2),
      (M2,1,1,1), (M1,1,3,3), (M2,1,6,1))
```

На основании полученной раскраски строятся расписания отдельных станков:

```
M1/1      D1/1 D1/2 D2/1-D2/1-D2/1
M2/1      D2/1 D1/1-D1/1           D2/1
M2/2                D1/2-D1/2
```

Поля «ti» и «p» в кортеже деталей меняются в процессе выполнения алгоритма. Поле «ti» указывает сколько тактов отработал станок в многотакотовой операции, а «pi» указывает на кортеж соответствующей операции в последовательность «Op».

Алгоритм построения планов пошагово для «tcol» = 1, 2, … записывает в поля «col» и «kst» кортежей из массива «Op» значения такта и номер станка соответственно, определяя тем самым, какие детали и на каких станках будут обрабатываться на этом такте.

Блок-схема алгоритма представлена на рис.1. Обозначения на блок-схеме: Nd – общее число деталей, Dn – число обработанных деталей, tcol – номер такта (текущий цвет) в общем процессе обработки, i – порядковый номер детали, ti – номер такта в обработке i-й детали, kst – номер станка при обработке i-й детали, col – номер такта в общем процессе для i-й детали.

Внешний цикл по такту tcol назначает цвет деталям (col = tcol), когда для них находится подходящий станок (kst = j1). Цикл завершается, когда все детали отказываются обработанными (Dn = Nd). Внутренний цикл по деталям проверяет, является ли деталь обработанной (ti = td), необработанной (ti = 0) или находится в ожидании размещения в печи (ti = -1). Если деталь не обработана, производится поиск подходящего станка. Для этого предусмотрен массив, где отмечается занятость станков. При успехе индекс станка j1 записывается в поле ksi-го станка. Номер станка ns определяется по текущей операции.

Если станок находится в обработке (ti>0), проверяется не последний ли этот такт. Если да, то происходит переход к следующей операции (увеличивается на 1 указатель p), а станок отмечается как свободный. Отметка проводится во вспомогательном массиве. Фактическое освобождения проводится после просмотра всех деталей, так как на текущем такте этот станок еще не может быть занят обработкой другой детали.

Обработка в печи на блок-схеме показана как процедура. В ней также используются дополнительные структуры для отслеживания заполненности. Если печь еще не заполнена, деталь ставиться в режим ожидания (ti = -1), а когда число деталей становиться достаточной, производятся соответствующие отметки в деталях, а печь освобождается.

ПОИСК ОПТИМАЛЬНОГО РАСПИСАНИЯ

Приведенный алгоритм строит только один вариант расписания. Результат существенно зависит от порядка описания как станков, так и типов деталей. Это можно увидеть даже на очень простых примерах:

```
d2,1 (s1/2,s2)
d1,1 (s1/3,s2/2)
                1    2    3    4    5    6    7
s1/1      d2/1-d2/1 d1/1-d1/1-d1/1
s2/1                d2/1           d1/1-d1/1

d1,1 (s1/3,s2/2)
d2,1 (s1/2,s2)
                1    2    3    4    5    6
s1/1      d1/1-d1/1-d1/1 d2/1-d2/1
s2/1                     d1/1-d1/1 d2/1
```

Программа может делать перебор перестановок типов деталей и рассчитывать некоторые показатели. В данном варианте программа рассчитывает следующие: общее число тактов (T) для обработки всех деталей, количество тактов простоя (P), количество переналадок (N) и средневзвешенный показатель (T+P+N). Так, в примере конфигурации S2 T=6, P=7, N=2. При переборе находятся конфигурации, для которых один из этих показателей минимален.

На каждом шаге внешнего цикла алгоритма подбор подходящих станков осуществляется в том порядке, как они описаны в конфигурации. Тем самым станки, находящиеся в начале списка, имеют приоритет. Понятно, что это не всегда окажется лучшим выбором. Поэтому должен быть предусмотрена перестановка типов станков в этом списке. Этот перебор назовем внутренним, в отличие от перестановок типов деталей, внешним перебором.

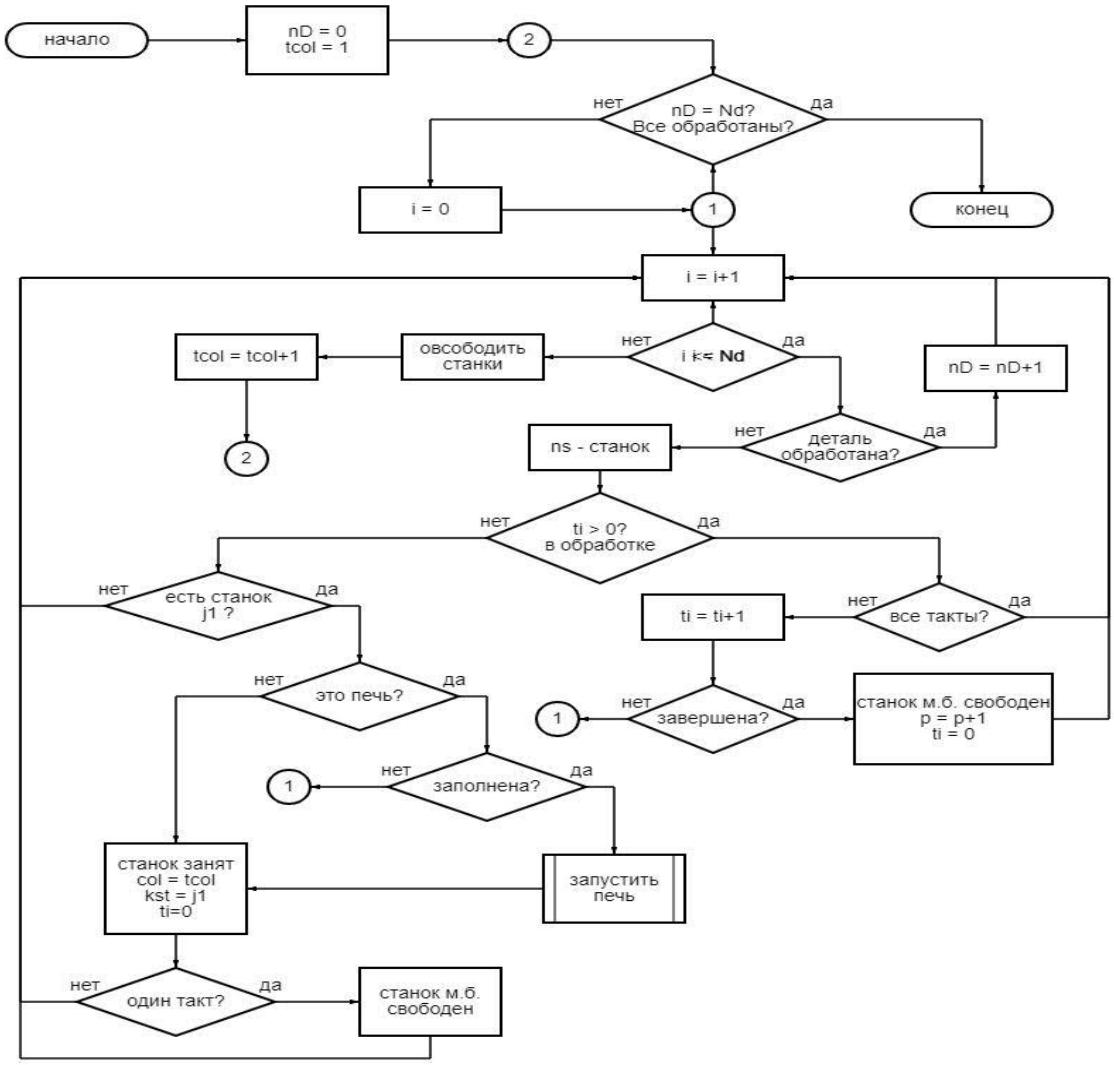

Рисунок 1. Блок-схема алгоритма построения расписания
Figure 1. Flowchart of the scheduling algorithm

Даже при относительно небольших количествах станков и деталей число перестановок может оказаться огромным. Поэтому в программе предусмотрено несколько режимов работы: 1) указание начальной перестановки максимального числа перестановок; 2) случайный выбор перестановок.

УЧЕТ ВРЕМЕНИ НА ПЕРЕНАЛАДКИ

Если время переналадки меньше одного такта, то полученные оптимальные расписания будут приемлемы. В противном случае это может существенно повлияет на общее время обработки. Число тактов для переналадки может различаться для разных типов деталей. В приведенном алгоритме это можно учесть путем добавления некоторого числа тактов (значение ti) к числу тактов обработки деталей, если станок на предыдущем такте обрабатывал деталь другого типа. Для этого необходимо добавить функцию Fn(ts, td1, td2), возвращающую числу тактов, необходимое станку типа ts перенастроиться с обработки деталей td1 на обработку деталей типа td2.

ПРОБЛЕМА ПЕЧЕЙ

В реальности время обработки в печи может варьироваться от некоторого минимального значения до максимального. Также может меняться загрузка – число одновременно обрабатываемых деталей. В принципе, для нахождения оптимального расписания это не очень важно. В данном варианте программы оба параметра фиксированы. Это приводит к проблеме запуска печи, когда печь еще недогружена, а необработанных деталей уже нет. Можно было бы запускать последнюю порцию при любой загрузке. Для этого требуется незначительная доработка алгоритма.

Другая проблема может возникнуть при наличии нескольких печей, когда для разных деталей требуется обработка в нескольких печах в разное время. Рассмотрим следующую простую конфигурацию:

S2: P1(2, 3), P2(2,4)
d1,1 (P1, P2)
d2,1 (P2, P1)

На первом шаге деталь d1 станет в очередь к печи P1 и печь будет ожидать второй детали. Аналогично, деталь d2 станет в очередь к P2, но печь не сможет запуститься, так как первая деталь находится в состоянии ожидания. Так же и P1 не может начать обработку. Общий процесс приостановится – возникает ситуация тупика. Эта проблема аналогична проблеме тупиков в теории операционных систем, подробно описанной в работе [4]. В терминах этой теории детали являются процессами, а станки – ресурсами. Одно из простых решений заключается в том, что процессам разрешается использовать ресурсы строго в одном порядке. Остается вопрос, является ли такое решение приемлемым для конфигураций рабочего цеха.

ПРОИЗВОДИТЕЛЬНОСТЬ АЛГОРИТМА

Для оценки производительности программы был проведен вычислительный эксперимент на нескольких конфигурациях с разным числом деталей. Данные по станкам и операциям взяты из реального рабочего цеха в упрощенном варианте. Число станков взято 18, по одному каждого типа (в реальности 30 станков). Типов деталей – 66. Всего деталей более 10000. Планы операций для деталей разных типов от 7 до 23, всего операций – 1112. Число перестановок по K типам деталей равно K! (10! = 3628800, 20! ~= 2.4*10^18, 30! = 2.6*10^32, …). Уже при K>10 время полного перебора превышает возможности современных ЭВМ. В табл. 1 приведены временные параметры для K= 2, 4 и 8.

Таблица 1. Производительность исходного и усовершенствованного алгоритмов
Table 1. Performance of original and improved algorithms

Число деталей	Размер матрицы в исходном алгоритме	Время расчета **одного** расписания	Время расчета **100** расписаний в новом алгоритме
2	2224 x 2224	33 сек	25 сек
4	4448 x 4448	2.5 мин	54 сек
8	8896 x 8896	42.2 мин	2 мин 21 сек

ВОЗМОЖНЫЕ УСОВЕРШЕНСТВОВАНИЯ АЛГОРИТМА

Улучшить алгоритм возможно разными способами.

1. Ввести укрупненные операции, т.е. считать некоторую фиксированную последовательности операций, которая может встретиться для разных типов деталей, за одну операцию.
2. Группировать детали одного типа, т.е. рассчитывать их обработку как одну «укрупненную» деталь.
3. Фиксировать в списке станков некоторую подпоследовательность, внутри которой не производить перестановки.

Вообще говоря, многое зависит от конфигурации конкретного цеха и безусловно настройка алгоритма должна вестись совместно со специалистами, знающими особенности конкретной ситуации.

ЗАКЛЮЧЕНИЕ

В рамках описываемого исследования алгоритм создания расписания подачи деталей в задаче, допускающей одновременную обработку заготовок на одном рабочем месте, частично решена. Окончательное решение проблемы, по-видимому, лежит в сфере использования мощных вычислительных средств. Однако в приведенном решении присутствует ряд приемов, позволяющих существенно снизить потребность в вычислительных мощностях.

Реализация планов, в которых допускается одновременная обработка нескольких деталей на одном рабочем месте, использование нескольких одинаковых станков является сложной задачей и общих принципов подхода к решению таких задач не существует.

Разработанные алгоритм и программа решают поставленную задачу построения плана рабочего цеха с машинами, допускающими одновременную обработку нескольких деталей. Проведенные численные эксперименты подтверждают возможность практического применения в организации работы процессов с ограниченным количеством обрабатываемых заявок.

Дальнейшие исследования по развитию алгоритма предполагают определение конкретных условий его применения в технологическом процессе.

ЛИТЕРАТУРА

1. Теория расписаний https://ru.wikipedia.org/wiki/Теория расписаний.
2. **Garey M.R., Johnson D.S., Ravi Sethi.** The Complexity of Flowshop and Jobshop Scheduling. *INFORMS. Mathematics of Operations Research*. 1976. Vol. 1. N 2. P. 117-129.
3. The Scheduling Zoo. http://www-desir.lip6.fr/~durrc/query/.
4. **Шоу А.** Логическое проектирование операционных систем. Мир.1981. 360 с.
5. **Коровин Д.И.** Логические принципы в организации производства. Иваново: ИвГУ. 2006. 154 с.
6. **Береговых Ю.В., Васильев Б.А., Володин Н.А.** Алгоритм составления расписания занятий. *Искусственный интеллект*. 2009. №2. С. 50-56.
7. **Панкратьев Е.В., Череповский А.М., Черепанов Е.А., Чернышов С.В.** Алгоритмы и методы решения задач составления расписаний и других экстремальных задач на графах больших размерностей. *Фундаментальная и прикладная математика*. 2003. Т. 9. № 1, С. 235-251.

REFERENCES

1. Schedule Theory. https://ru.wikipedia.org/wiki/Теория расписаний. (in Russian).
2. **Garey M.R., Johnson D.S., Ravi Sethi.** The Complexity of Flowshop and Jobshop Scheduling. *INFORMS. Mathematics of Operations Research*. 1976. Vol. 1. N 2. P. 117-129.
3. The Scheduling Zoo. http://www-desir.lip6.fr/~durrc/query/.
4. **Shaw A.** The Logical Design of Operating System. Mir. 1981. 360 p. (in Russian).
5. **Korovin D.I.** Logical principles in the organization of production. Ivanovo: IvSU. 2006.154 p. (in Russian).
6. **Beregovyh U.V., Vasiliev B.A., Volodin N.A.** Algorithm for scheduling classes. *Artificial intelligence*. 2009. N 2. P. 50-56. (in Russian).
7. **Pankratiev E.V., Cherepovskij A.M., Cherepanov E.A., Chernyshov S.V.** Algorithms and methods for solving scheduling problems and other extremal problems on graphs of large dimensions. *Fundamental and Applied Mathematics*. 2003. Vol. 9. № 1, P. 235-251. (in Russian).

Поступила в редакцию 15.06.2021
Принята к опубликованию 28.06.2021

Received 15.06.2021
Accepted 28.06.2021