УДК519.87

АВТОМАТИЗИРОВАННАЯ СИСТЕМА ПОСТРОЕНИЯ ОПТИМАЛЬНЫХ РАСПИСАНИЙ ДЛЯ ДВУХСТАДИЙНЫХ СИСТЕМ ОБСЛУЖИВАНИЯ С НЕОПРЕДЕЛЕННЫМИ ДЛИТЕЛЬНОСТЯМИ ОПЕРАЦИЙ

Д.А. Жукова *, Н.М. Матвейчук**

** Белорусский государственный университет информатики и радиоэлектроники, Беларусь, Минск, darya.hatsura@gmail.com*

*** Белорусский государственный аграрный технический университет, Беларусь, Минск, matsveichuk@tut.by*

Аннотация. Разработан алгоритм построения оптимального (при выполнении достаточных условий) расписания обслуживания требований с различными маршрутами и неопределенными длительностями операций. Разработано web-приложение с использованием облачных технологий для автоматизации процесса составления расписаний, а также сбора статистической информации.

Ключевые слова: оптимальность, расписание, двухстадийная система, различные маршруты обслуживания, неопределенные длительности операций, автоматизированная система.

AUTOMATED SYSTEM OF CONSTRUCTING OPTIMAL SCHEDULE FOR TWO-STAGE JOB-SHOP WITH INTERVAL PROCESSING TIMES

D.A. Zhukova*, N.M. Matsveichuk**

** Belarusian State University of Informatics and Radioelectronics, Belarus, Minsk, darya.hatsura@gmail.com*

*** Belarusian State Agrarian Technical University, Belarus, Minsk, matsveichuk@tut.by*

Abstract. An algorithm is developed for constructing an optimal (if sufficient conditions) schedule for servicing requirements with different routes and undetermined duration of operations. A web application was developed using cloud technologies to automate the process of scheduling, as well as collecting statistical information.

Keywords: optimality, schedule, two-stage system, various routes of service, indefinite duration of operations, automated system.

В работе рассматривается автоматизированная система, позволяющая строить оптимальные (при выполнении достаточных условий оптимальности) или близкие к оптимальному расписанию для задачи оптимизации быстродействия двухстадийных систем обслуживания требований с различными маршрутами обслуживания в условиях неопределенности длительностей операций.

Постановка задачи: два прибора $M = \{M_1, M_2\}$ должны обслужить n требований $\mathfrak{J} = \{\mathfrak{J}_1, \mathfrak{J}_2, \dots, \mathfrak{J}_n\}$ с различными маршрутами: $\mathfrak{J} = \mathfrak{J}_1 \cup \mathfrak{J}_2 \cup \mathfrak{J}_{12} \cup \mathfrak{J}_{21}$, где $\mathfrak{J}_{12} \subseteq \mathfrak{J}$ – множество требований с маршрутом (M_1, M_2); $\mathfrak{J}_{21} \subseteq \mathfrak{J}$ – множество требований с маршрутом (M_2, M_1); $\mathfrak{J}_m \subseteq \mathfrak{J}$ $(m \in \{1,2\})$ – множество требований, которые обслуживаются только одним прибором $M_m \in M$, $n_l = |\mathfrak{J}_l|, l \in \{1, 2, 12, 21\}$. Прерывания операции по обслуживанию требования прибором запрещены. Длительность p_{jm} операции по обслуживанию требования $\mathfrak{J}_j \in \mathfrak{J}$ прибором $M_m \in M$ не фиксирована на момент составления расписания и может принимать любое действительное значение из заданного отрезка $[a_{jm}, b_{jm}]$, $0 \le a_{jm} < b_{jm}$. Такая задача обозначается $\mathfrak{J}2|a_{jm} \le p_{jm} \le b_{jm}, n_j \le 2|C_{max}$, при условии, что требуется минимизировать общее время C_{max} обслуживания требований множества $\mathfrak{J}$.

Пусть $T = \{p|a_{jm} \le p_{jm} \le b_{jm}, \mathfrak{J}_j \in \mathfrak{J}, M_m \in M\}$ – множество векторов (сценариев) $p = (p_{1,1}, p_{1,2}, \dots, p_{n1}p_{n2})$ допустимых длительностей операций.

Как известно, оптимальное расписание для задачи $\mathfrak{J}2||C_{max}$ (с детерминированными длительностями выполнения операций) определяется парой перестановок Джексона (π', π''), где π' – перестановка выполнения работ множества $\mathfrak{J}_1 \cup \mathfrak{J}_{12} \cup \mathfrak{J}_{21}$ прибором M_1, а π'' – перестановка выполнения работ множества $\mathfrak{J}_2 \cup \mathfrak{J}_{12} \cup \mathfrak{J}_{21}$ прибором M_2.

По условиям, описанным в работе [2], решение задачи $\mathfrak{J}2|a_{ij} \le p_{ij} \le b_{ij}, n_i \le 2|C_{max}$ достаточно искать среди пар перестановок $\pi' = (\pi_{12}^i, \pi_1, \pi_{21}^j)$ и $\pi'' = \left(\pi_{21}^j, \pi_2, \pi_{12}^i\right), 1 \le i \le n_{12}!, 1 \le j \le n_{12}!$ (работа $\mathfrak{J}_j$ содержится в перестановке π_l, если $\mathfrak{J}_j \in \mathfrak{J}_l, l \in \{1, 2, 12, 21\}$).

В силу неопределенности длительностей операций в общем случае не существует перестановки, оптимальной для задачи $\mathfrak{J}2|a_{ij} \le p_{ij} \le b_{ij}|C_{max}$ при традиционном подходе к решению задачи (оптимальной для всех векторов длительностей операций $p \in T$). Задача $\mathfrak{J}2|a_{ij} \le p_{ij} \le b_{ij}|C_{max}$ с математической точки зрения некорректна.

В работе используется следующее допущение: точное значение p_{ij}^* длительности p_{ij} становится известным лишь в момент завершения обслуживания требования $\mathfrak{J}_i$ прибором M_j (это условие выполняется для большинства реальных процессов).

Это допущение определяет необходимость решать не только вышеперечисленные задачи построения оптимального расписания до момента t_0 начала реализации процесса (в режиме off-line), но и задачи корректировки расписания в реальном времени (в режиме on-line) на основе информации, получаемой в ходе выполнения работ из множества $\mathfrak{J}$.

Поэтому в работе предлагается осуществлять построение расписания в два этапа: до начала выполнения работ (этап off-line) и во время выполнения работ (этап on-line). На этапе off-line вначале производится попытка построения пары перестановок, оптимальных для всех векторов длительностей операций $p \in T$, для чего производится проверка достаточных условий из работы [2]. Если эти условия не выполняются, то на множестве требований производится построение графа частичного строгого порядка на множестве требований $G = (\mathfrak{J}, A_\prec)$, что помогает сократить количество рассматриваемых перестановок. После чего, используя достаточные условия из работы [3], делается вывод о возможности построения оптимального расписания на этапе off-line. Если эти достаточные условия не выполняются, то далее построение расписания будет производиться на этапе on-line.

Этап on-line: в момент времени $t = 0$ оба прибора M_1 и M_2 начинают обслуживание упорядоченных требований множеств $\mathfrak{J}_{12}$ и $\mathfrak{J}_{21}$, соответственно. Оба прибора обслуживают требования без неоправданных простоев (рассматриваются только активные расписания). Обслуживание продолжается вплоть до первого момента времени t принятия решения, т.е. до момента, когда на одном из приборов не будут обслужены все упорядоченные требования, и возникнет необходимость выбора порядка обслуживания конфликтных требований.

Множеством конфликтных требований называется минимальное по включению подмножество $\mathfrak{J}_x \subseteq \mathfrak{J}_{12}$, такое, что любое требование $J_y \in \mathfrak{J}_{12} \backslash \mathfrak{J}_x$ находится с каждым требованием $J_x \in \mathfrak{J}_x$ в отношении $(J_x, J_y) \in A_\prec$ (находится с каждым требованием $J_x \in \mathfrak{J}_x$ в отношении $(J_y, J_x) \in A_\prec$). Очевидно, в множестве $\mathfrak{J}_{12}$ может быть несколько конфликтных подмножеств. Аналогично определяются подмножества конфликтных требований для множества $\mathfrak{J}_{21}$.

На этом этапе проверка достаточных условий из работы [2] производится в процессе обслуживания, когда известны точные значения длительностей операций, выполнение которых завершено к этому моменту t. В этот момент становятся известными точные значения p_{ij}^* некоторых длительностей p_{ij} обслуживания требований (тех. обслуживание которых завершено к моменту времени t).

Знание точных значений длительностей операций позволяет уменьшить множество T. При проверке достаточных условий используются известные фактические значения времени выполнения завершенных работ. В случае невыполнения этих условий, производится построение пары перестановок, оптимальных для средних значений времени выполнения работ.

Поставленная задача реализована в программном продукте, который предоставляет возможность пользователю ввести или импортировать следующие исходные данные: период, на который строится расписание, количество заданных требований на этот период, границы длительностей работ, а также вид разрешения конфликтов на этапе on-line. В результате выполнения программы пользователь получает построенное расписание на указанный период вместе со статистическими данными построенных расписаний. Пользователь может работать с этими данными, производя выборки по ним, а также строить графики и сводные таблицы по выбранным параметрам.

Разработанная система позволяет составлять расписания обслуживания двумя приборами n требований с интервальными длительностями операций в два этапа (off-line и on-line), а также анализировать статистические данные построенных расписаний. В случае построения расписания, приближенного к оптимальному, система выполняет построение оптимального расписания постфактум (на основании точных значений длительностей выполнения операций, которые становятся известными после окончания выполнения операции), и рассчитывает расхождение во времени выполнения для оценки точности построения расписания.

Описание алгоритма составления оптимального расписания.

1. Ввод данных на весь рассматриваемый период.

2. Запуск цикла построения расписания на один элемент периода (цикл 1).

3. Производится проверка достаточных условий оптимальности любого порядка обслуживания требований множества $\mathfrak{I}$. Если любой порядок требований оптимален для любых допустимых реализаций длительностей операций, составляется оптимальная перестановка (по порядку номеров требований), происходит сохранение результата с пометкой 1 (означает, что оптимальное расписание составлено на этапе off-line) и проводится переход дальше по циклу 1.

4. В противном случае проверяются достаточные условия существования перестановки, оптимальной для любых допустимых реализаций длительностей операций. Если такая перестановка существует (достаточное условие проверяется для обоих множеств требований $\mathfrak{I}_{12}$ и $\mathfrak{I}_{21}$), строится пара перестановок Джексона, оптимальная для любых допустимых реализаций длительностей операций, и происходит сохранение результата с пометкой 1 и дальше по циклу 1.

5. Если не получен оптимальный порядок обслуживания требований, происходит построение графа частичного строгого порядка $A_{\prec}$ на множестве $\mathfrak{I}_{12}$ путем сравнения не более $n(n-1)$ пар требований множества $\mathfrak{I}_{12}$ и строгий порядок $A_{\prec}$ на множестве $\mathfrak{I}_{21}$ сравнением не более $n(n-1)$ пар требований множества $\mathfrak{I}_{21}$. При этом может возникнуть ситуация, когда на одном из множеств $\mathfrak{I}_{12}$ или $\mathfrak{I}_{21}$ построен линейный порядок.

6. Производится проверка достаточных условий оптимальности любого порядка обслуживания требований множества конфликтных требований. Если любой порядок обслуживания конфликтных требований оптимален для любых допустимых реализаций длительностей операций, составляется оптимальная перестановка конфликтных требований (по порядку номеров требований), происходит сохранение результата с пометкой 1 и проводится переход дальше по циклу 1.

7. В противном случае проверяются достаточные условия существования оптимального порядка обслуживания конфликтных требований (при проверке составляются две перестановки). При выполнении достаточных условий хотя бы для одной перестановки, происходит построение оптимальной перестановки и сохранение результата с пометкой 1; проводится переход дальше по циклу 1.

8. Если не существует оптимального порядка обслуживания конфликтных требований хотя бы на одном из множеств $\Im_{12}$ или $\Im_{21}$, оптимальное расписание не удалось составить на этапе off-line. Далее будет проходить составление расписание на этапе on-line.

9. Запуск цикла разрешения конфликтов на этапе on-line (назовем его цикл 2). Определение оптимального порядка обслуживания множества конфликтных требований производится на основании достаточных условий. Для каждого множества конфликтных требований следует построить две перестановки и проверить два достаточных условия оптимальности, первое условие оптимальности – для первой перестановки, и второе условие оптимальности – для второй.

10. Производится обслуживание линейно упорядоченного подмножества требований с использованием фактического вектора $p^* \in T$ длительностей операций одновременно приборами M_1 и M_2 вплоть до момента, когда необходимо определить порядок обслуживания подмножества конфликтных требований. Вычисляется время начала очередного конфликта (на основании известного фактического значения) и какие работы выполняются в этот момент.

11. Производится проверка достаточных условий оптимальности любого порядка обслуживания требований множества конфликтных требований для on-line этапа. Если достаточные условия выполняются, составляется оптимальная перестановка конфликтных требований (по порядку номеров требований), и переходим к следующему конфликту.

12. В противном случае проверяются достаточные условия существования оптимального порядка обслуживания конфликтных требований для on-line этапа (при проверке составляются две перестановки). При выполнении достаточных условий хотя бы для одной перестановки, происходит определение оптимального порядка обслуживания конфликтных требований и переходим к следующему конфликту. Если достаточные условия не выполнены, определяется порядок обслуживания конфликтных требований (с помощью того способа, который задан пользователем), и система переходит к следующему конфликту в цикле 2. Например, обслуживание подмножества конфликтных требований может производиться в порядке, оптимальном для требований, длительности операций которых равны серединам заданных интервалов длительностей операций конфликтных требований.

13. После окончания цикла 2 производится проверка: если все подмножества конфликтных требований упорядочены оптимальным образом на этапе on-line (для каждого подмножества конфликтных требований выполнены достаточные условия оптимальности), то на этапе on-line построена оптимальная для фактического вектора длительностей операций пара перестановок, с доказательством ее оптимальности в процессе обслуживания требований до завершения обслуживания всех требований. Происходит сохранение результата с пометкой 2 и проводится переход дальше по циклу 1.

14. Если не все подмножества конфликтных требований упорядочены оптимальным образом на этапе on-line (не выполнены достаточные условия оптимальности), то вычисляется оптимальная длина расписания для фактического вектора длительностей операций (длина расписания, определяемого парой перестановок Джексона), и сравнивается с полученной длиной расписания.

15. Если фактический результат равен оптимальному, то происходит сохранение результата с пометкой 3 (означает, что оптимальное расписание составлено без доказательства оптимальности), в противном случае с пометкой 4 (не составлено оптимального расписания).

16. После окончания цикла 1 происходит обработка результатов из «буфера».

17. На последнем этапе сохраняются эти результаты.

Заметим, что сохранение промежуточного результата происходит в хранилище данных. Эти данные хранятся только в рамках составления одного расписания. После завершения составления расписания «буферные» данные агрегируются, обрабатываются и сохраняются в удобочитаемом виде в базу данных.

Таким образом, по завершении работы алгоритма может возникнуть одна из следующих ситуаций:

- на этапе off-line (до начала обслуживания требований) построена пара перестановок, оптимальная для всех допустимых векторов длительностей операций;
- на этапе on-line построена пара перестановок, заведомо оптимальная для фактического вектора длительностей операций;
- на этапе on-line построена пара перестановок, оптимальная для фактического вектора длительностей операций, без доказательства ее оптимальности;
- на этапе on-line построена пара перестановок, не являющаяся оптимальной для фактического вектора длительностей операций.

В программе предусмотрен сбор статистики. В статистические данные вносится информация о том, на каком этапе построения составлено оптимальное расписание, какое количество конфликтов было разрешено на каждом этапе построения расписания и то, какое время занимает проведение подсчетов у программы. Это время подсчитывается для того, чтобы узнать скорсть отклика программы в процессе составления расписания.

Схема алгоритма построения расписания представлена на рис. 1 и 2.

Сервер системы разработан с использованием облачных технологий, предоставляемых Azure, и написан на языке C#.

В качестве базы данных использовалась Azure SQL – это специальная версия базы данных MS SQL, оптимизированная для масштабирования в облаке. База данных использовалась для хранения пользователей, их прав доступа, а также паролей и исходных данных для составленных расписаний. Построенные расписания хранятся в облачном хранилище Azure Blob Storage. Такое разделение в хранении данных использовалось для того, чтобы улучшить скорость работы программы и ускорить ее отклик в процессе построения расписаний.

Клиентская часть написана на языке TypeScript с использованием фреймворка Angular 4. Аутентификация пользователей осуществляется через протокол OAuth 2.0 на основе JWT (JSON Web Token).

Для корректной работы сервера необходима ЭВМ, оснащенная одноядерным процессором с тактовой частотой не менее 1,0ГГц и оперативной памятью не менее 1,75Гб. Для работы с клиентской частью приложения пользователю необходима ПЭВМ с наличием доступа с сеть Интернет, а также браузер Google Chrome не ниже 52-й версии. Работа в других браузерах возможна, но при этом нет гарантии корректной работы приложения.

Относительно небольшие требования для работы системы позволяют использовать ее не только для повышения эффективности и развития процессов по составлению расписаний обслуживания требований с различными маршрутами и решению практических задач календарного планирования на производстве, но и в других сферах, например, оптимизации рабочего времени сотрудников.

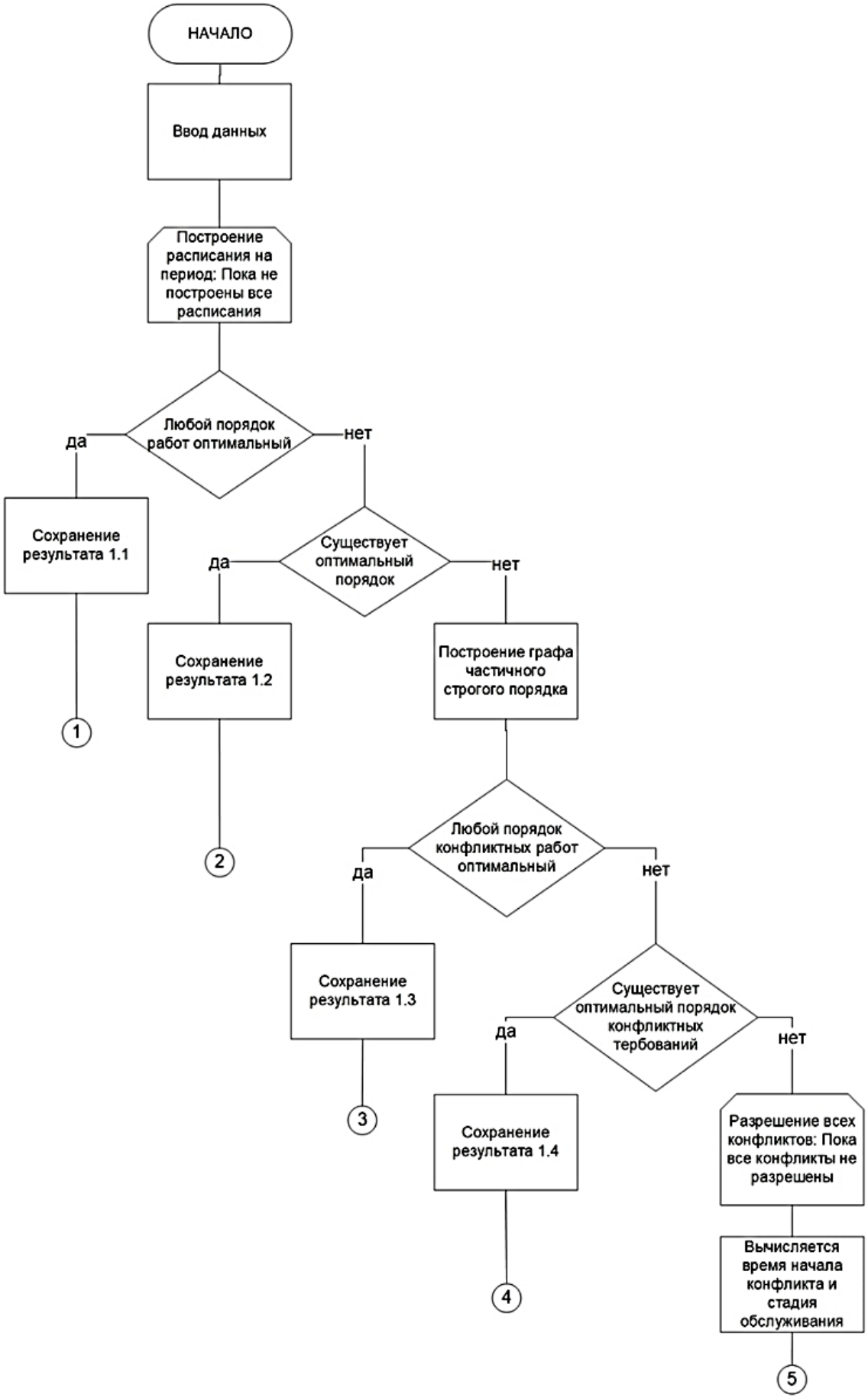

Рис. 1. Схема алгоритма построения расписания

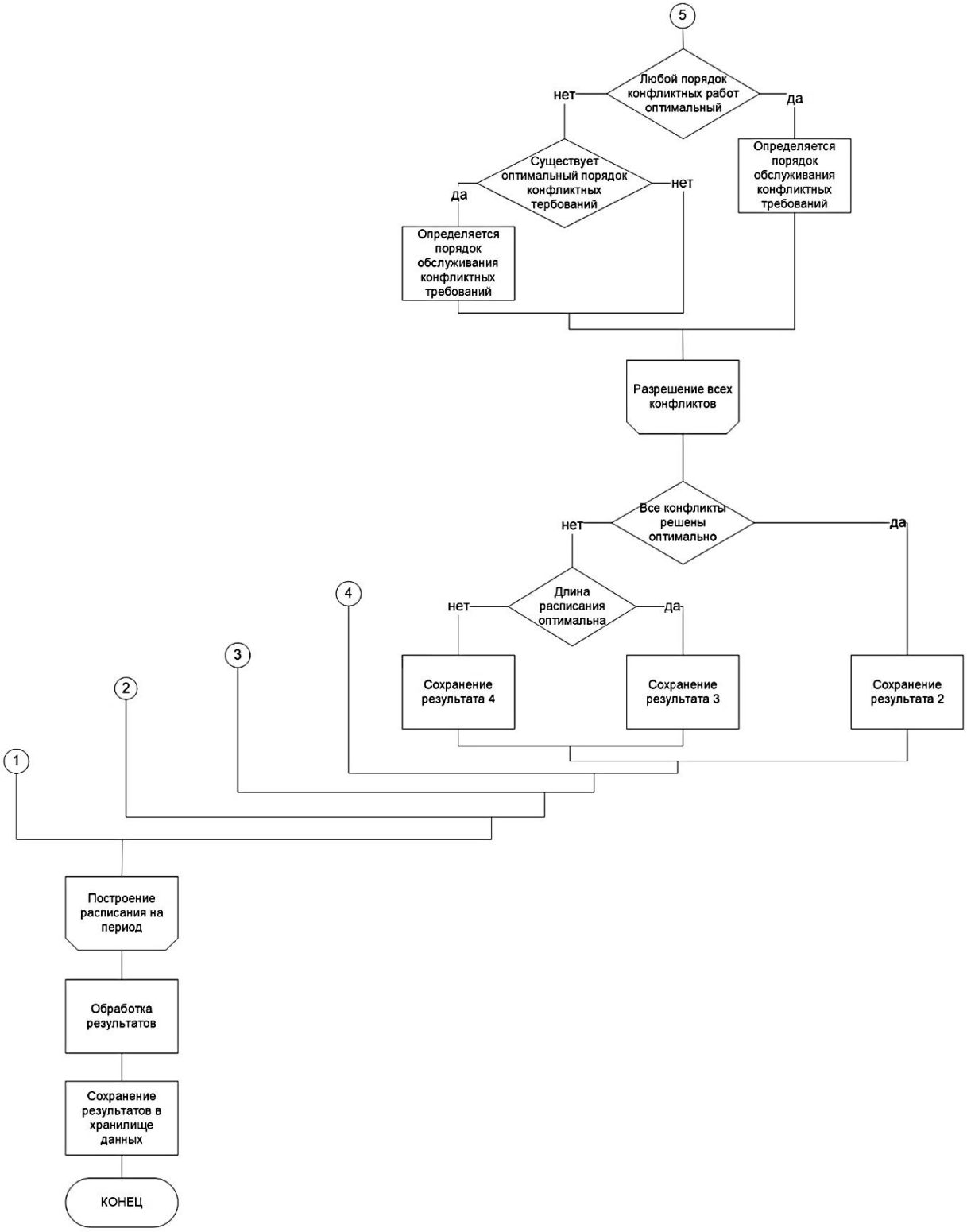

Рис. 2. Продолжение схемы алгоритма построения расписания

Библиографический список

1. Сотсков Ю.Н. и др. Модели и комплекс программ для планирования рабочего времени // Информатика. 2007. №4.
2. Матвейчук Н.М., Сотсков Ю.Н., Голами О. Достаточные условия оптимальности обслуживания двумя приборами множества требований с различными маршрутами с интервальными длительностями операций // Пятая междунар. науч. конф. Танаевские чтения: доклады. Минск, 28-29 марта 2012г. / ОИПИ НАН Беларуси; под науч. ред. к.ф.-м.н. Н.Н. Гущинского. Минск, 2012. С.69-75.
3. Шафранский Я.М. О существовании глобально оптимальных расписаний для задачи Беллмана-Джонсона для двух приборов в условиях неопределенности // Информатика. 2009. №3. С.100-110.