ОПТИМАЛЬНОЕ ПЛАНИРОВАНИЕ РЕМОНТНЫХ РАБОТ ПО КРИТЕРИЮ РАВНОМЕРНОСТИ ЗАГРУЗКИ

Я.Г. Генис[5]
Нью-Йорк, США.

И.А. Ушаков[6]
Сан-Диего, США.

Аннотация. Предлагается алгоритм для оптимального расписания ремонтов (обслуживания) технических устройств, когда прерывание процесса обслуживания невозможно, но имеется определенное «окно» для проведения этого обслуживания. Ресурсы ремонтной базы ограничены. Оптимальность понимается в том смысле, что расписание обеспечивает максимальную равномерность загрузки ремонтных бригад.

1. ФОРМУЛИРОВКА ЗАДАЧИ

Имеется n заявок на работы с объемами $v_1, v_2, \ldots, v_n$ (см. рис.1). Каждая k-я работа должна быть выполнена в течение некоторого интервала $[s_k, e_k]$, который лежит между разрешенным моментом времени начала, S_k, и допустимым моментом ее окончания, E_k, т.е.

$$[s_k, e_k] \subseteq [S_k, E_k]. \qquad (1)$$

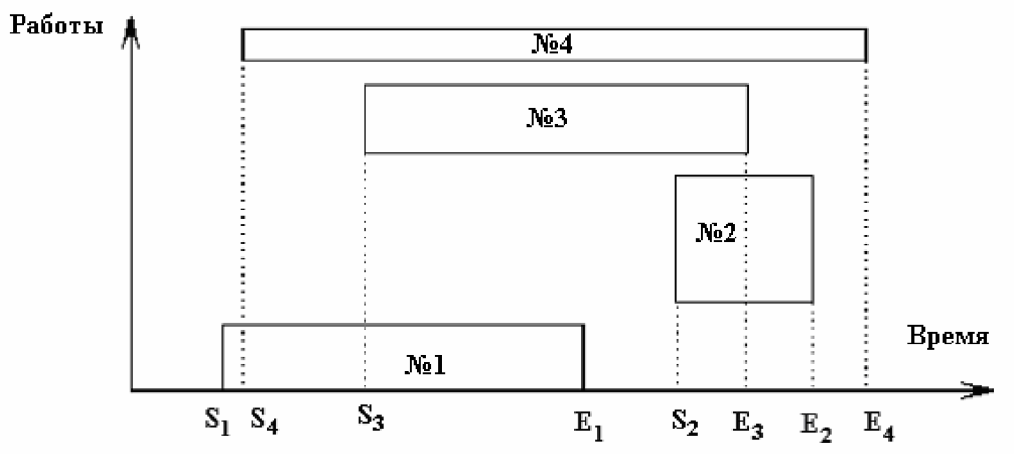

Рис. 1 Разрешенные интервалы выполнения работ и их объемы

[5] Яков Генис <yashag5@yahoo.com>
[6] Игорь Ушаков <iushakov2000@yahoo.com>

Прерывание работы или изменение интенсивности ее выполнения не допускается. Таким образом, интенсивность r_k выполнения работы k определяется, как

$$r_k(t) = \begin{cases} \dfrac{v_k}{e_k - s_k} & \text{если } t \in [e_k - s_k] \\ 0, \textit{ в противном случае} \end{cases} . \qquad (2)$$

Понятно, что суммарная интенсивность работы при некотором заданном размещении работ, назовем его G, есть

$$R(t) = \sum_{k \in G} r_k(t), \qquad (3)$$

где G – выбранное размещение работ.

Заметим, что для любого выбранного расписания функция $R(t)$ представляет собой ступенчатую функцию типа той, которая представлена на Рис. 2.

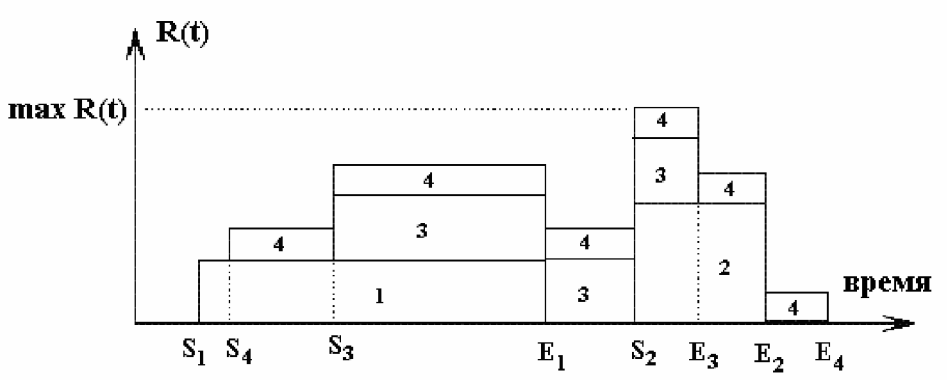

Рис. 2. Начальное распределение интенсивности работ

Задача заключается в том, чтобы найти такое разбиение подинтервалов $[s_k, e_k]$, чтобы

а) максимум распределения суммарной интенсивности работ был минимален

$$\min_g \max_t R(t) \qquad (4)$$

и/или минимумы суммарной интенсивности работ должны быть максимальны

$$\max_g \min_t R(t) \qquad (5)$$

где g – некоторое размещение интервалов $[s_k, e_k]$;

б) распределение суммарной интенсивности работ $R(t)$ на всем интервале работы системы обслуживания должно быть наиболее равномерным (минимум разницы между max $R(t)$ и min $R(t)$).

Замечание: Полученное размещение g не является единственным, в силу дискретности квантов времени.

2. СЛОВЕСНОЕ ОПИСАНИЕ АЛГОРИТМА FV&CH

Название алгоритм FV&CH образовано из сокращения его полного английского имени: «FILL THE VALLEYS & CUT THE HILLS». Смешно сказать, но буквальное словесное описание этого алгоритма приводится в следующих словах Евангелия от Луки (гл.3, стих 5): « ...всякий дол да наполнится, и всякая гора и холм да понизятся».

Но переходя от Библии к математике, дадим строгое (хотя и словесное) описание алгоритма, реализованного на языке Visual Basic.

Итак, мы должны найти такое разбиение подинтервалов (1), чтобы выполнить (4) и / или (5) и сделать *R(t)* настолько равномерным, насколько это возможно при заданных ограничениях. Разработанная на Visual Basic программа FV&CH имеет две кнопки «Cut Hills» (Срежь Холмы) и «Fill Valleys» (Заполни Долины). Первая кнопка выполняет (4), а вторая – (5). Выбирая последовательность нажатия этих кнопок мы можем найти разбиение подинтервалов (1), которое наибольшим образом спрямляет распределение *R(t)*.

Объясним алгоритм на иллюстративном примере. Пусть имеется пять работ с заданными объемами, v_k, и соответствующими допустимыми интервалами исполнения $[S_k, E_k]$, $k=\overline{1,5}$. Ради простоты введем в рассмотрение дискретные кванты времени, с точностью которых измеряется время. Они могут быть равны, например, часу, или 15 минутам, или 1 минуте и т.д.. Исходные данные приведены в Табл. 1, где d_1, d_2, d_3, d_4 и d_5 – это некоторые временные интервалы. Например, работа №1 первоначально должна была начаться в начале интервала d_1 и закончиться в конце интервала d_4

Табл. 1. Первоначальное размещение работ в допустимых интервалах

Работа №	**Объем**	d_1	d_2	d_3	d_4	d_5
1	8	2	2	2	2	
2	9	3	3	3		
3	15		5	5	5	
4	12			4	4	4
5	3		1	1	1	
	9.4^{opt}	5	11	15^{*}	12	4_{*}

Последнее значение в столбце «Объем» дает оптимальное значение нагрузки для идеального случая: когда все работы удается равномерно «размазать» на всем интервале времени, начиная с d_1 и кончая d_5. В последней строке звездочкой сверху обозначена наибольшая необходимая интенсивность обслуживания, а звездочкой снизу – наименьшая интенсивность для начальной расстановки работ. Начальное распределение работ представлено на Рис. 3.

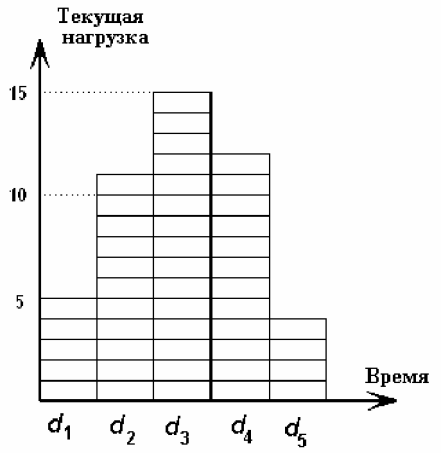

Рис. 3. Начальное распределение нагрузки в иллюстративном примере

Табл. 1 и Рис. 3 показывают, что максимальная нагрузка приходится на интервал d_3. Найдем те работы, которые предназначались для выполнения в этом интервале в исходном расписании, и сдвиг которых мог бы уменьшить максимальную нагрузку.

Шаг 1. Начнем с работы №1. Поскольку каждая работа может выполняться только непрерывно и с постоянной нагрузкой, мы можем сдвинуть выполнение всей этой работы либо левее интервала d_3, либо правее. В данном конкретном случае видно, что только сдвиг влево ведет к уменьшению пиковой нагрузки.

Замечание: На первом шаге могла бы быть выбрана и любая другая работа, выполнение которой намечалось в этом интервале в соответствии с исходным расписанием работ. При этом не должна увеличиваться пиковая нагрузка и не должна уменьшаться минимальная нагрузка.

Новое расписание приведено в Табл. 2.

Табл. 2. Шаг 1: Проверка размещения работы №1

Работа №	**Объем**	d_1	d_2	d_3	d_4	d_5	***Действие***
1	8	4	4				Сдвиг влево
2	9	3	3	3			
3	15		5	5	5		
4	12			4	4	4	
5	3		1	1	1		
	9.4	7	13^*	13^*	10	4_*	

Мы должны попробовать «пошевелить» и другие работы. В качестве решения на первом шаге выбирается наилучшее из решений. Компьютерная программа легко выполняет эти действия.

В данном примере, выполняя действия вручную, мы на интуитивном уровне отдаем предпочтение работе №4 в качестве следующей работы, подлежащей возможному перемещению. Она может быть вся перемещена для выполнения в интервал времени d_5.

Назовем эту передвижку «вправо-вправо». Это действие также приводит к улучшению расписания. Заметим, что это действие на первом шаге оказывается наилучшим (см. Табл. 3).

Табл. 3. Шаг 1: Проверка размещения работы №4

Работа №	**Объем**	d_1	d_2	d_3	d_4	d_5	**Действие**
1	8	2	2	2	2		
2	9	3	3	3			
3	15		5	5	5		
4	12			0	0	12	«вправо-вправо»
5	3		1	1	1		
	9.4	5*	11	11	8	12*	

Шаг 2. Этот шаг направлен на «заполнение ямы» в кванте d_1 . Вся работа №1 перемещается в этот квант. Назовем этот сдвиг «влево-влево». Новое расписание представлено в Табл. 4.

Табл. 4 . Шаг 2: Проверка размещения работы №1

Работа №	**Объем**	d_1	d_2	d_3	d_4	d_5	**Действие**
1	8	8					“left-left”
2	9	3	3	3			
3	15		5	5	5		
4	12			0	0	12	
5	3		1	1	1		
	9.4	11	9	9	6*	12*	

При этом действии мы одновременно «убили двух зайцев»: минимум возрос, а распределение стало более равномерным.

Шаг 3. Перенесем всю работу №5 в интервал времени d_4, т.е. произведем с этой работой операцию переноса «вправо-вправо». Результат переноса показан в Табл. 5.

Таблица 5. Шаг 3. Проверка размещения работы №5

Работа №	**Объем**	d_1	d_2	d_3	d_4	d_5	**Действие**
1	8	8					
2	9	3	3	3			
3	15		5	5	5		
4	12			0	0	12	
5	3		0	0	3		«вправо-вправо»
	9.4	11	8	8	8*	12*	

На этом шаге заканчивается построение наиболее равномерного расписания с минимально возможной «пиковой нагрузкой» и максимальной «недогрузкой».

Сравнение исходного распределения и распределения после оптимизации приведено на рис. 4.

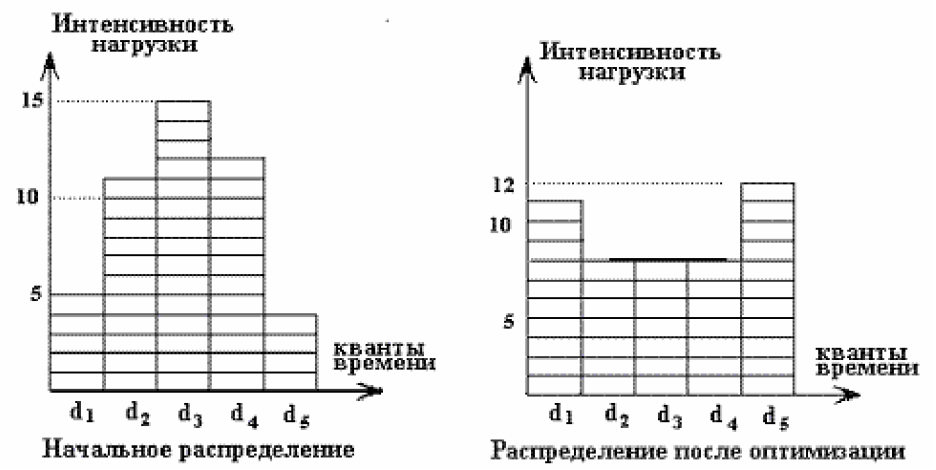

Рис. 4. Сравнение исходного распределения и распределения после оптимизации

ЗАКЛЮЧЕНИЕ

1. Предложенный алгоритм FV&CH дает строгое решение в смысле нахождения приближенного к равномерному оптимального расписания обслуживания
2. Полученное решение является оптимальным (в смысле выравнивания нагрузки за счет достижения минимума максимальной нагрузки и максимума минимальной нагрузки), хотя и не является единственным.
3. Описанный алгоритм прост для программирования.
4. Разработана программа на Visual Basic, использующая этот алгоритм. Программа имеет простой и удобный интерфейс и позволяет работать с неограниченным числом работ при сколь угодно малом временном кванте. Заинтересованных в программе просим обращаться к авторам.