Задача о назначениях

...

Что это за задача?

Задача о назначениях - это задача о наилучшем распределении рабочих задач между исполнителями. Эффективное распределение должно минимизировать затраты на производство (время/деньги), максимизируя качество выполнения. Если количество задач и исполнителей не равно друг другу, задача превращается в обобщенную задачу о назначениях.

С математической точки зрения задача о назначениях является частным случаем задачи о нахождении максимального паросочетания в двудольном графе.

С чем мы работали

Подразделение						
№	Должность	ФИО	Языки	Качество работы (мах. 10)	Скорость работы (листов в час)	Примечания
1	Начальник отдела	ИВАНОВ Иван Иванович	Английский Немецкий			
2	Заместитель начальника отдела	ПЕТРОВ Петр Петрович	Английский Французский	9 8	1 0,75	
3	Начальник группы	ЕГОРОВ Егор Егорович	Испанский Английский	7 6	1 0,5	
4	Начальник группы	ТИМОФЕЕВ Тимофей Тимофеевич	Немецкий Английский	5 5	1,5 1,5	
5	Старший сотрудник отдела	СИДОРОВ Сидр Сидорович	Английский Португальский	8 9	1 1	
6	Старший сотрудник отдела	АНДРЕЕВ Андрей Андреевич	Итальянский	8	1	
7	Сотрудник отдела	МИХАЙЛОВ Михаил Михайлович	Польский Английский	4 7	0,5 0,75	
8	Сотрудник отдела	ДАНИЛОВ Даниил Даниилович	Английский Французский Немецкий	7 6 8	0,75 0,75 1	
Усиление						
9	Замначальника отдела	АРТЕМОВ Артем Артемович	Французский Английский	9 3	1,25 0,5	
10	Старший сотрудник отдела	СЕРГЕЕВ Сергей Сергеевич	Английский Итальянский	6 7	1 0,75	
11	Сотрудник отдела	ГАВРИЛОВ Гавриил Гавриилович	Английский Испанский	8 4	1 1	

№	Тип	Язык	Объем (листов)	Сложность (мах. 10)	Важность (мах. 3)	Сроки (дней)	Примечания
1.	Перевод	Английский	5	5	3	1	
2.	Перевод	Английский	15	6	1	2	
3.	Перевод	Французский	10	9	1	5	
4.	Перевод	Немецкий	40	5	2	25	
5.	Перевод	Испанский	12	4	2	4	
6.	Перевод	Французский	17	6	1	15	
7.	Перевод	Итальянский	9	7	1	21	
8.	Перевод	Английский	187	4	2	7	
9.	Перевод	Английский	45	5	1	1	
10.	Перевод	Английский	3	5	1	2	
11.	Перевод	Английский	12	5	2	25	
12.	Перевод	Португальский	6	7	2	18	
13.	Перевод	Португальский	15	6	2	14	
14.	Перевод	Португальский	11	4	3	4	
15.	Перевод	Немецкий	87	6	2	30	

С чем мы работали

У нас были следующие целевые функции:

- минимальное время выполнения всех задач;
- минимальное время выполнения срочных задач;
- минимальное время выполнения важных задач;
- максимальное качество выполнения всех задач;
- максимально быстрое выполнение особо важных задач
- максимально качественное выполнение особо важных задач
- максимально равномерная нагрузка на всех сотрудников
- распределение в зависимости от важности задач;
- распределение в зависимости от срока задач;
- минимальное время выполнения всех задач без ограничения продолжительности рабочего дня (макс. 36 часов подряд, потом - перерыв 6ч.)
- минимальное время выполнения всех задач с привлечением заместителя начальника отдела (не более 4ч нагрузки в обычный рабочий день и максимальная нагрузка для выполнения задачи высокой важности)

Распределение обязанностей

По итогам оценки сложности данной задачи, наша команда выбрала следующее распределение обязанностей:

Соболев Матвей занимался разработкой решения с помощью аппроксимационного алгоритма, применения к нашей задаче и его программной реализацией

А Степовик Виктор и Кашапова Ольга, после нескольких попыток приспособить Венгерский алгоритм под нашу задачу, нашли собственный подход к её решению, а также разработали общий алгоритм для некоторых целевых функций.

Подходы к решению

- Сначала было решено использовать венгерский алгоритм для решения задачи, но в процессе выяснилось, что его использование не обязательно.
- Аппроксимационный алгоритм был успешно применен к некоторым целевым функциям по отдельности
- Для решения были разработаны собственные алгоритмы

Аппроксимационный алгоритм

Аппроксимационный алгоритм находит решение, близкое к оптимальному, и решает обобщенную задачу о назначениях путём последовательного применения алгоритма для решения задачи о рюкзаке на каждом этапе своей работы.

Задача о рюкзаке называется от её конечной цели: положить как можно большее число ценных вещей в рюкзак при условии, что вместимость рюкзака ограничена. Задачу о рюкзаке можно сформулировать следующим образом: из заданного множества предметов со свойствами «стоимость» и «вес» требуется отобрать подмножество с максимальной полной стоимостью, соблюдая при этом ограничение на суммарный вес.

Алгоритм для решения задачи о рюкзаке

N - количество предметов, W - вместимость рюкзака, w - массив весов предметов, p - массив стоимостей предметов, A - рабочий массив.

Пусть A(k, s) есть максимальная стоимость предметов, которые можно уложить в рюкзак вместимости s, если можно использовать только первые k предметов, то есть {n_1, n_2, ..., n_k}. При этом A(k, 0) = 0 и A(0, s) = 0. Далее найдём A(k, s):

Если предмет k не попал в рюкзак, тогда A(k, s) равно максимальной стоимости рюкзака с такой же вместимостью и набором допустимых предметов {n_1, n_2, ..., n_{k−1}}, то есть A(k, s) = A(k − 1, s).

Если предмет k попал в рюкзак, то тогда A(k, s) равно максимальной стоимости рюкзака, где вес s уменьшаем на вес k-го предмета и набор допустимых предметов {n_1, n_2, ..., n_{k−1}} плюс стоимость k, то есть A(k − 1, s − w_k) + p_k.

То есть, A(k, s) = max(A(k − 1, s) (не берём), A(k − 1, s − w_k) + p_k (берём)), а стоимость искомого набора равна A(N, W).

Алгоритм для решения задачи о рюкзаке

Пример:

W = 13; N = 5;

$w_1 = 3$; $p_1 = 1$;

$w_2 = 4$; $p_2 = 6$;

$w_3 = 5$; $p_3 = 4$;

$w_4 = 8$; $p_4 = 7$;

$w_5 = 9$; $p_5 = 6$.

N W	0	1	2	3	4	5	6	7	8	9	10	11	12	13
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	1	1	1	1	1	1	1	1	1	1	1
2	0	0	0	1	6	6	6	7	7	7	7	7	7	7
3	0	0	0	1	6	6	6	7	7	10	10	10	11	11
4	0	0	0	1	6	6	6	7	7	10	10	10	13	13
5		0	0	1	6	6	6	7	7	10	10	10	13	13

Алгоритм для решения задачи о рюкзаке

Числа от 0 до 13 в первой строчке обозначают вместимость рюкзака. В первой строке как только вместимость рюкзака n>=3, добавляем в рюкзак один предмет. Рассмотрим k=3, при каждом s>=5 сравниваем A[k-1][s] и A[k-1][s-w3]+p3 и записываем в A[k][s] стоимость либо рюкзака без третьего предмета, но с таким же весом, либо с третьим предметом, тогда стоимость равна стоимости третьего предмета плюс стоимость рюкзака с вместимостью на w3 меньше. Максимальная стоимость рюкзака находится в A(5,13).

NW	0	1	2	3	4	5	6	7	8	9	10	11	12	13
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	1	1	1	1	1	1	1	1	1	1	1
2	0	0	0	1	6	6	6	7	7	7	7	7	7	7
3	0	0	0	1	6	6	6	7	7	10	10	10	11	11
4	0	0	0	1	6	6	6	7	7	10	10	10	13	13
5		0	0	1	6	6	6	7	7	10	10	10	13	13

Начиная с A(5,13) восстанавливаем ответ. Будем идти в обратном порядке по k. Синим цветом обозначим наш путь.

Аппроксимирующий алгоритм

Условия:

Пара (B, S), где B - это набор из M корзин (ранцев), а S - набор из N предметов. Каждая корзина $C_j \in B$ имеет емкость c (j), и для каждого элемента i и корзины C_j нам заданы размер s (i, j) и прибыль p (i, j).

Задача, которую должен решить алгоритм:

Найти подмножество $U \subseteq S$ предметов, которое может быть помещено в B, такое, чтобы прибыль была максимальной.

Аппроксимационный алгоритм

Дано:

Пусть M - количество ящиков, а N - количество элементов. Пусть p - матрица прибыли размером N × M. Значение p[i, j] - есть прибыль элемента i, когда он выбран для корзины (ранца) C_j. Пусть A - алгоритм задачи о рюкзаке. Теперь построим из A алгоритм для общей задачи о назначениях (далее - GAP). Поскольку наш алгоритм изменяет функцию прибыли, мы используем обозначение p_j для обозначения матрицы прибыли при j-ом рекурсивном вызове. Первоначально мы устанавливаем новую матрицу p_1 ← p и вызываем процедуру корзины (рюкзака) при j = 1!

Аппроксимационный алгоритм

1. Запустите алгоритм A (решение задачи о рюкзаке) для столбца C_j, используя текущую функцию прибыли p_j, используя S_j в качестве набора выбранных (возвращенных) номеров предметов.

Вместимость рюкзаков (bin sizes), матрица C:

C_1	C_2	C_3
2	3	4

Ценность предметов (profit function), матрица p:

	C_1	C_2	C_3
i_1	3	1	5
i_2	1	1	1
i_3	5	15	25
i_4	25	15	5

Веса предметов (item sizes), матрица s:

	C_1	C_2	C_3
i_1	1	1	1
i_2	2	3	3
i_3	2	3	4
i_4	1	2	3

Аппроксимационный алгоритм

2. Разложите функцию прибыли (матрицу) p_j на две части прибыли: функции (матрицы) p_j^1 и p_j^2 такие (1 и 2 - коэффициенты, а не возведение в квадрат), В p_j^1 оставляем текущий столбец значений, а также дублируем его значения на те строки, которые были выбраны (наглядно это показано на примере).

2.1. p_j^2 = p_j - p_j^1

Аппроксимационный алгоритм

Ценность предметов (profit function), матрица p_1:

	C_1	C_2	C_3
i_1	3	1	5
i_2	1	1	1
i_3	5	15	25
i_4	25	15	5

Ценность предметов (profit function), матрица p_1^1:

	C_1	C_2	C_3
i_1	3	3	3
i_2	1	0	0
i_3	5	0	0
i_4	25	25	25

Ценность предметов (profit function), матрица p_1^2:

	C_1	C_2	C_3
i_1	0	-2	2
i_2	0	1	1
i_3	0	15	25
i_4	0	-10	-20

S_1 = {i_1, i_4}.

Аппроксимационный алгоритм

3. Пока алгоритм не закончился (j < M):

3.1 p_{j + 1} = p_j^2;

3.2. Запустить алгоритм заново (S_{j} остаётся таким же за исключением элементов, которые были добавлены в S_{j + 1}).

Аппроксимационный алгоритм

Ценность предметов (profit function), матрица p_2:

	C_1	C_2	C_3
i_1	0	-2	2
i_2	0	1	1
i_3	0	15	25
i_4	0	-10	-20

Ценность предметов (profit function), матрица p_2^1:

	C_1	C_2	C_3
i_1		-2	0
i_2		1	0
i_3		15	15
i_4		-10	0

Ценность предметов (profit function), матрица p_2^2:

	C_1	C_2	C_3
i_1		0	2
i_2		0	1
i_3		0	10
i_4		0	-20

** - красным выделены клетки (столбцы), которые не используются.
S_2 = {i_3}.

Аппроксимационный алгоритм

Ценность предметов (profit function), матрица p_3:

	C_1	C_2	C_3
i_1	0	0	2
i_2	0	0	1
i_3	0	0	10
i_4	0	0	-20

Ценность предметов (profit function), матрица p_3^1:

	C_1	C_2	C_3
i_1			2
i_2			1
i_3			10
i_4			-20

Ценность предметов (profit function), матрица p_3^2:

	C_1	C_2	C_3
i_1			0
i_2			0
i_3			0
i_4			0

* - красным выделены клетки (столбцы), которые не используются.
S_3 = {i_3}.

Аппроксимационный алгоритм

Ответом будет следующий набор:

1. i_1, i_4;
2. -
3. i_3.

Важной особенностью данного алгоритма является то, что он не гарантирует выбор всех предметов, поэтому важно сделать акцент на матрицах размера и прибыли!

Аппроксимационный алгоритм

Существует также и второй аппроксимационный алгоритм, который работает идентично. Отличие лишь в том, что второй является итерационной версией первого.

Особенности программной реализации

Мы попытались перевести на математический язык исходные данные, чтобы их можно было применить к задаче. В реальной жизни намного больше факторов, которые нужно учитывать, но мы попытаемся систематизировать их и применить к программе.

1. Скорость первого переводчика (начальника отдела) равно “0”, а заместители начальников отдела работают как обычные сотрудники (кроме отдельных случаев);
2. Считаем срок работы каждого переводчика равным “30” дням (и считаем, что сроки исчисляются только рабочими днями), а также считаем 8-часовой рабочий день (кроме отдельных случаев);
3. Деление работ между сотрудниками в данном алгоритме и смешивание целевых функций применяться не будет (к примеру, мы находим работу по срокам, либо по качеству, но не одновременно);
4. Полученное время округляем всегда в большую сторону (как условие из реальной жизни и как необходимость для работы программы в целых числах);
5. Алгоритм не гарантирует распределение всех работ, поэтому мы будем подбирать матрицы прибыли и весов так, чтобы выполнялись все работы.

Особенности программной реализации

Для отдельных целевых функций мы также интерпретируем имеющиеся условия.

1. Для целевых функций 5 и 6 мы будем использовать только самые важные задачи (3 важность);
2. Целевая функция 7 не реализуема;
3. В целевой функции 8 мы будем выполнять задачи по важности с возможным ущербом для сроков выполнения;
4. В целевых функциях 8 и 9 нам важно только распределение работ;
5. В целевой функции 10 будем считать, что рабочий день длится 18 часов (18 часов + 6 часов перерыв);
6. В целевой функции 11 заместители будут работать как обычные сотрудники только при важности 3.

Также стоит отметить, что мы будем использовать “инвертированные” матрицы стоимостей.

Реализация аппроксимационного алгоритма

Для реализации аппроксимирующего алгоритма была разработана программа на языке C++. Примеры работы программы:

```
Chosed items:
Translator 1 jobs: (0 days)
Translator 2 jobs: 3 6 19 (15 days)
Translator 3 jobs: 18 20 (23 days)
Translator 4 jobs: 1 4 8 10 (22 days)
Translator 5 jobs: 11 12 13 14 21 30 (10 days)
Translator 6 jobs: 7 26 (4 days)
Translator 7 jobs: 2 22 27 (8 days)
Translator 8 jobs: 15 16 17 23 28 29 31 (23 days)
Translator 9 jobs: 9 25 (23 days)
Translator 10 jobs: (0 days)
Translator 11 jobs: 5 24 32 (6 days)

Answer: 23 days
Total 32/32 jobs
Program ended with exit code: 0
```

```
Chosed items:
Translator 1 jobs: (0 days)
Translator 2 jobs: 1 2 9 10 11 21 22 27 30 (18 days)
Translator 3 jobs: 5 24 32 (6 days)
Translator 4 jobs: 16 (3 days)
Translator 5 jobs: 12 13 14 18 20 25 (23 days)
Translator 6 jobs: 7 26 (4 days)
Translator 7 jobs: (0 days)
Translator 8 jobs: 4 15 17 31 (18 days)
Translator 9 jobs: 3 6 19 23 28 29 (12 days)
Translator 10 jobs: (0 days)
Translator 11 jobs: 8 (24 days)

Answer: 268 quality
Total 32/32 jobs
Program ended with exit code: 0
```

Выводы по аппроксимирующему алгоритму

По результатам работы программы аппроксимирующего алгоритма мы выяснили:

1. Алгоритм хорошо подходит для максимизации “выгоды” от определённого критерия, например, качества, важности или даже скорости работы для отдельно взятого работника.
2. Алгоритм не нацелен на получение лучшего результата, который связан с изменением функций во времени, следовательно, приходится получать решение изменением функции весов или максимальной вместимости.

Поэтому мы решили придумать собственное решение для целевых функций и найти обобщённый алгоритм.

Венгерский алгоритм

Одним из методов решения задачи о назначениях является венгерский алгоритм. Для его применения необходимо составить матрицу, содержащую какие-либо показатели рабочих задач или работников. Рассмотрим данный алгоритм на простом примере. Имеется домашнее задание по 4 предметам, и 4 студента собрались, чтобы коллективно его выполнить. Все обладают разными навыками по разным предметам (максимальный показатель навыка - 5), поэтому необходимо определить, кому какое задание доверить.

студент/предмет	Математический анализ	Дискретная математика	Английский язык	Правоведение
Петров	5	4	1	2
Смирнов	2	3	2	3
Сидоров	4	5	4	3
Егоров	1	4	5	2

Венгерский алгоритм

Так как мы ищем **максимум**, при котором студенты справятся с заданиями лучше всего, найдем в каждой строке максимальный элемент и вычтем его из этой же строки, а затем умножим матрицу на -1, чтобы избавиться от отрицательных значений.

5	4	1	2	5
2	3	2	3	3
4	5	4	3	5
1	4	5	2	5

0	-1	-4	-3
-1	0	-1	0
-1	0	-1	-2
-4	-1	0	-3

0	1	4	3
1	0	1	0
1	0	1	2
4	1	0	3

Некоторые значения, естественно, “занулились”. Далее нам необходимо выбрать нули таким образом, чтобы в каждой строке и каждом столбце был только один выбранный ноль. Выделяем их в матрице:

0	1	4	3
1	0	1	0
1	0	1	2
4	1	0	3

Венгерский алгоритм

Подставляем в изначальную матрицу местоположения выбранных нулей. Получили оптимальный план, при котором студенты получат наилучшие оценки за свои домашние задания.

5	4	1	2
2	3	2	3
4	5	4	3
1	4	5	2

студент/предмет	Математический анализ	Дискретная математика	Английский язык	Правоведение
Петров	5	4	1	2
Смирнов	2	3	2	3
Сидоров	4	5	4	3
Егоров	1	4	5	2

Целевая функция: минимальное время

Почему при решении мы обошлись без венгерского алгоритма? Наши задачи имеют неравный объем и другие показатели, то есть мы не можем решить задачу, опираясь лишь на показатели исполнителей. Поэтому было построен следующий алгоритм нахождения минимального времени выполнения работы (рабочий день в данном решении = 8 часам):

сотруд\задача	Англ (60л)	Франц(8л)	Немецкий(12л)	Итальян(30л)
Петров	1	0,75	~	~
Егоров	0,5	~	~	1
Данилов	0,75	0,75	1	~
Тимофеев	1,5	~	1,5	~

Целевая функция: минимальное время

1)Назначение сотрудников с наибольшей скоростью на самые объемные задачи

2)По формуле время = объем работы/скорость определяем, кто из сотрудников освобождается, и назначаем их на следующие подходящие им по квалификации работы, попутно фиксируя загруженность исполнителей в часах

3)Распределяем загруженность исполнителей равномерно: если один из сотрудников отдела выполняет все поставленные перед ним задачи раньше остальных, то он помогает наиболее загруженному работнику и оставшаяся работа последнего делится в равных частях между этими работниками

С помощью данного решения было найдено минимальное время выполнения всех задач, равное 15 суткам.

Таблица **“Y”. Время выполнения каждой из назначенных сотруднику задач**

сотр\№задачи	1	2	3	4	5
Петров	45	23	10	~	
Егоров	23	12	9	10	~
Тимофеев	124	32	~	~	~
Сидоров	42	15	6	~	~
Андреев	12	9	~	~	~
Михайлов	20	20	10	~	~
Данилов	87	40	11	7	2
Артёмов	48	8	8	8	~
Сергеев	55	~	~	~	~
Гаврилов	34	12	3	3	2

Массив **“Z”. Общее время выполнения сотрудниками задач**

Сотрудник	П	Е	Т	С	А	М	Д	Ар	Сер	Га
Загруженность(ч.)	78	54	156	63	21	50	147	72	55	54

Распределительный алгоритм:

Начало

Создадим переменную “answ” в которую записываем значение минимального времени в таблице, отличное от нуля (по окончании работы алгоритма данная переменная будет хранить ответ)

ПОКА в массиве “Z” есть значения больше нуля выполняем:

Находим минимальную занятость в массиве “Z”;

answ = answ + минимальная занятость;

Из всех ячеек массива отнимаем answ;

ПОКА есть ёмкие по времени задачи за которые могут приняться освободившиеся работники:

- Находим рабочих, у которых в массиве загруженность = 0 и рабочего , которого загруженность максимальна. - Среди задач последнего находим максимально емкую по времени задачу, за которую они могут приступить.

КЦ;

ЕСЛИ такую задачу можно найти,ТО:

- Делим задачу между сотрудниками (тем, кто уже выполняет задачу, и теми, кто свободен и может помочь ему) по следующей формуле: Vx=V-vw*answ; t= Vx / (vw + СУММ(vi));

(где vw - скорость работника, трудящегося над задачей; V- объем работы для данного сотрудника(из таблицы №3); Vx-объем работы на текущий момент; vi- скорости работы помощников; t - добавленные помощникам часы работы)

- В массиве “Z” добавляем “помощникам” t, а уже работавшему над этой задачей сотруднику прибавляем время = t + время выполнения остальных задач этим сотрудником (которое можно найти по таблице “Y”);

-В таблице “Y” меняем время выполнения этой задачи у работавшего над ней сотрудника на t+answ, а “помощникам” добавляем в новую ячейку задач время t;

ИНАЧЕ:

- Получили ситуацию, когда работник(и) более некомпетентен(ы) выполнять другие задачи и он(и) отдыхают (~)

КЦ;

Конец.

Сотрудник	П	Е	Т	С	А	М	Д	Ар	Сер	Га
Загруженность (ч.)	78	54	156	63	21	50	147	72	55	54
ШАг 1 answ=21	78-21=57	54-21=33	156-21=135	63-21=42	21-21=0	50-21=29	147-21=126	72-21=51	55-21=34	54-21=33
ШАГ 2 answ = 50	28	4	135 - 29=106 (106-32)* 1,5=112/ 2,25=50 50+32 = 82	13	~	0 + 50	97	22	5	4
ШАГ 3 answ = 54	28-4=24	0 + 23	82-4=78 (78-32)* 1,5=69/ 3 = 23 23+32= 55	9	~	46	93	18	1	0+ 23
ШАГ 9 answ=75	3	2	24	0	~	3	72	3	0	2
ШАГ 10 answ=77	1	0	22	0	~	1	70	1	0	0
ШАГ 11 answ=69	0	~	21	~	~	0	69	0	~	~
ШАГ 12 answ=90	~	~	0+20	~	~	~	48*1 / 2,5= 20	~	~	~
ШАГ 13 answ=110	~	~	0	~	~	~	0	~	~	~

Изменения таблицы **“Y”**

сотр\№задачи	1	2	3	4	5	6
П	45	23	10	~	~	~
Е	23	12	9	10	23	~
Т	50+4+1+8+4	32	20	~	~	~
С	42	15	6	4+2+6	~	~
А	12	9	~	~	~	~
М	20	20	10	17+5+6	~	~
Д	39 + 20	40	11	7	2	~
Ар	48	8	8	8	6	~
Сер	55	14+6	~	~	~	~
Га	34	12	3	3	2	23

Итоговая таблица распределений

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	t
П																																	78
Е																																	77
Т																																	119
С																																	75
А																																	21
М																																	78
Д																																	119
Ар																																	78
Сер																																	75
Га																																	77

этот цвет - собственные задачи сотрудника

этот цвет - задачи, которые он помогает выполнить другим

Подключение других целевых функций

Теперь попробуем подключить ещё целевую функцию: минимальное время выполнения всех задач без ограничения продолжительности рабочего дня (макс. 36 часов подряд, потом - перерыв 6ч.)

По предыдущему решению мы выяснили, что отдел справится с работой за 119 ч => 119 - 36*3=11 => 119+3*6 = минимальное время выполнения всех задач = 6 суток (5 дней и 8 часов)
Т.е. мы просто переводим количество часов работы в новый “рабочий день” и считаем сколько нормальных суток эта работа займёт.

Важно добавить, что при таком рабочем графике выполняется и целевая функция **минимальное время выполнения срочных задач,** т.к расчёт распределения не зависел от порядка выполнения сотрудником задач.

Целевая функция: максимальное качество

№	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	часы работы
Т	А	А	Ф	Н	ИС	Ф	ИТ	А	А	А	А	П	П	Н	Н	Н	Н	А	Ф	А	А	А	Ф	ИС	А	ИТ	А	Ф	Ф	А	Н	ИС	
V	5	15	10	40	12	17	9	187	45	3	12	6	15	11	87	32	2	34	60	55	9	7	9	23	42	12	15	7	9	4	7	2	
Сложж	5	6	9	5	4	6	7	4	5	5	5	7	6	4	6	6	5	4	5	6	5	5	8	6	9	7	6	5	5	4	5	6	
П	9	9	8	~	~	8	~	9	9	9	9	~	~	~	~	~	~	9	8	9	9	9	8	~	9	~	9	8	8	9	~	~	433
Е	6	6	~	~	7	~	~	6	6	6	6	~	~	~	~	~	~	6	~	6	6	6	~	7	6	~	6	~	~	6	~	7	37
Т	5	5	~	5	~	~	~	5	5	5	5	~	~	5	5	5	5	5	~	5	5	5	~	~	5	~	5	~	~	5	5	~	0
С	8	8	~	~	~	~	~	8	8	8	8	9	9	~	~	~	~	8	~	8	8	8	~	~	8	~	8	~	~	8	~	~	21
А	~	~	~	~	~	~	8	~	~	~	~	~	~	~	~	~	~	~	~	~	~	~	~	~	~	8	~	~	~	~	~	~	21
М	7	7	~	~	~	~	~	7	7	7	7	~	~	~	~	~	~	7	~	7	7	7	~	~	7	~	7	~	~	7	~	~	0
Д	7	7	6	8	~	6	~	7	7	7	7	~	~	8	8	8	8	7	6	7	7	7	6	~	7	~	7	6	6	7	8	~	179
Ар	3	3	9	~	~	9	~	3	3	3	3	~	~	~	~	~	~	3	9	3	3	3	9	~	3	~	3	9	9		~	~	84
Сер	6	6	~	~	~	~	7	6	6	6	6	~	~	~	~	~	~	6	~	6	6	6	~	~	6	7	6	~	~	6	~	~	0
Га	8	8	~	~	4	~	~	8	8	8	8	~	~	~	~	~	~	8	~	8	8	8	~	4	8	~	8	~	~	8	~	4	0

Целевые функции: максимальное качество + минимальное время

Чтобы выполнить все задачи максимально качественно и быстро, необходимо распределить сотрудников так, чтобы качество выполнения ими задачи было >= сложности задачи. Использовался следующий алгоритм с последующим равномерным распределением работ:

Создаем таблицу “X” , таблицу “Y”, куда записываем время, затраченное каждым сотрудником на каждую из своих задач, и массив “Z”, где хранится общее время выполнения задач сотрудником.

ПОКА есть не распределенные задачи:

ПОКА есть свободные рабочие:

ЕСЛИ рабочий свободен и нашлась свободная задача для него, то:

- Ищем для него самую сложную работу, которую он может качественно выполнить
- Записываем время, затраченное данным сотрудником на эту задачу в таблицу “Y”(если это первая задача, то в первую колонку, если вторая - во вторую и т.д.)
- Прибавляем время, затраченное данным сотрудником на эту задачу к значению в массиве “Z” в соответствующей сотруднику ячейке.

КЦ.

Обновляем массив “Z”, вычитая из всех ячеек минимальное значение, и если в ячейке сотрудника после этого действия стоит “0”, то он считается свободным.

- В таблицу “X” добавляем заполненную соответствующими данным строку свободного сотрудника.

КЦ.

Целевые функции: максимальное качество + минимальное время

Мы пришли к той же ситуации, что и в 3-ем этапе первого решения - далее нам необходим “распределительный” алгоритм, который осложнится следующей проверкой: освободившийся работник может прийти на помощь только на ту задачу, сложность которой ниже качества работы этого сотрудника.

Данный алгоритм даст нам ответ на решение задачи относительно связки следующих целевых функций:

- минимальное время выполнения всех задач;
- минимальное время выполнения всех задач;
- максимальное качество выполнения всех задач;
- максимально равномерная нагрузка на всех сотрудников

Дополнительно можно подключить ещё:

- минимальное время выполнения всех задач без ограничения продолжительности рабочего дня (макс 36 часов подряд, потом - перерыв 6ч.)
- минимальное время выполнения всех задач с привлечением заместителя начальника отдела (не более 4ч нагрузки в обычный рабочий день и максимальная нагрузка для выполнения задачи высокой важности)

Сочетание нескольких целевых функций

Для алгоритма, сочетающего в себе несколько целевых функций был разработан следующий алгоритм: 1) Строим таблицу “X” зависимостей сотрудник-работа (каждая строка - сотрудник, каждый столбец - задача) и заполняем её значениями характеристик сотрудников в зависимости от целевой функции, где вся работа предварительно отсортирована по необходимым критериям (назовём их “актуальность”), а также строим таблицу “Y”, куда записываем время, затраченное каждым сотрудником на каждую из своих задач, и массив “Z”, где хранится общее время выполнения всех задач.

ПОКА есть не распределенные задачи:

ПОКА есть свободные рабочие:

ЕСЛИ рабочий свободен и нашлась свободная задача для него, то:

- Ищем для него самую актуальную работу, которую он может выполнить согласно целевой функции(качественно, быстро и т.п.) и выделяем соответствующую ячейку (так строим паросочетание)
- Записываем время, затраченное данным сотрудником на эту задачу в таблицу “Y” (если это первая задача, то в первую колонку, если вторая - во вторую и т.д.)
- Прибавляем время, затраченное данным сотрудником на эту задачу к значению в массиве “Z” в соответствующей сотруднику ячейке.

КЦ.

- Обновляем массив “Z”, вычитая из всех ячеек минимальное значение, и если в ячейке сотрудника после этого действия стоит “0”, то он считается свободным.
- В таблицу “X” добавляем заполненную соответствующими данным строку свободного сотрудника.

КЦ.

Сочетание нескольких целевых функций

-Обновляем массив “Z”, заполняя его ячейки суммой времени, затраченного сотрудником на каждую из задач (находим общее время работы сотрудника)

- Создадим переменную “answ” в которую записываем значение минимального времени в таблице, отличную от нуля.

ПОКА в массиве “Z” есть значения больше нуля выполняем:

- Находим минимальную занятость в массиве “Z”
- answ = answ + минимальная занятость;
- Из всех ячеек массива отнимаем answ;

ПОКА есть ёмкие по времени задачи за которые могут приняться освободившиеся работники:

- Находим рабочих, у которых в массиве “Z” загруженность = 0 и рабочего , которого загруженность максимальна. Среди задач последнего находим максимально емкую по времени задачу(таблица “Y”), за которую они могут приступить.

КЦ;

Сочетание нескольких целевых функций

ЕСЛИ такую задачу можно найти И на неё у сотрудников хватает **“компетенции” по подключенной целевой функции**, ТО:

- Делим задачу между сотрудниками (тем, кто уже выполняет задачу, и теми, кто свободен и может помочь ему) по следующей формуле:

  Vx=V-vw*answ;

  t=Vx /(vw+ СУММ(vi));

(где vw - скорость работника, трудящегося над задачей; V- объем работы для данного сотрудника(из таблицы №3); Vx-объем работы на текущий момент; vi- скорости работы помощников; t - добавленные помощникам часы работы)

-Добавляем “помощникам” в массив “Z” t, а уже работавшему над этой задачей сотруднику прибавляем время = t + время выполнения остальных задач этим сотрудником (которое можно найти по таблице “Y”);

-В таблице “Y” меняем время выполнения этой задачи у работавшего над ней сотрудника на t + answ, а “помощникам” добавляем в новую ячейку задач время t;

ИНАЧЕ:

Получили ситуацию, когда работник(и) более некомпетентен(ы) выполнять другие задачи и он(и) отдыхают (~)

КЦ;

Конец.

Сочетание нескольких целевых функций

Определим распределение, удовлетворяющие следующим функциям:

-распределение в зависимости от важности задач;
-распределение в зависимости от срока задач;
-максимально быстрое выполнение особо важных задач
-максимально качественное выполнение особо важных задач
-максимально равномерное распределение рабочей нагрузки

Для задач были определены следующие категории:

важность задачи	показатель
особо важная	3
важная	2
обычная	1

срочность задачи	показатель
срочная	1-3 дня
средней срочности	4-7 дней
несрочная	>7 дней

Сочетание нескольких целевых функций

Решение имело следующие этапы:

1) Распределение особо важных задач между исполнителями с наибольшим качеством, срочных - с наибольшей скоростью
2) Распределение важных задач средней срочности между исполнителями с высокой скоростью (важный принцип: при выборе исполнителей для важных задач предпочтение отдавалось кандидатам с не только высокой скоростью, но и высоким качеством работы)
3) Распределение обычных задач средней срочности между исполнителями с высокой скоростью
4) Распределение не срочных важных задач между оставшимися исполнителями по принципу из пункта 2
5) Распределение не срочных обычных задач между оставшимися исполнителями

На каждом шаге фиксировалась загруженность рабочих и выбиралось максимально равномерное распределение работы между ними.

Сочетание нескольких целевых функций

Итоговое распределение (в правом столбце - часы работы, разные цвета означают разные этапы решения):