

# 09, сентябрь

УДК 004.023

## **Использование генетического алгоритма в задаче составления личного расписания**

*Секирин П.А., студент*

*Россия, 105005, г. Москва, МГТУ им Н. Э. Баумана,  
кафедра «Системы обработки информации и управления»*

*Научный руководитель: Гапанюк Ю.Е.,*

*Россия, 105005, г. Москва, МГТУ им Н. Э. Баумана,  
кафедра «Системы обработки информации и управления»*

[gapyu@bmstu.ru](mailto:gapyu@bmstu.ru)

### **Введение**

Организациям требуется эффективно распределять время своих сотрудников, каждый человек хочет использовать своё время с максимальной пользой – успевать выполнять задачи вовремя и распределять нагрузку. Для решения этих задач используют расписание, откуда следует актуальность задачи составления расписания. Задача составления расписания решается за неполиномиальное время, поэтому она сложна в решении человеком. Чтобы составить расписания для большого количества начальных данных используют вычислительные средства, автоматизируя этот процесс. Существует ряд алгоритмов решения этой задачи.

### **Задача составления расписания**

Составление расписания – это типичная задача исследования операций, где нужно определить оптимальное решение  $X^*$  из множества допустимых решений  $G$ . Причём в данной задаче также требуется найти само допустимое множество решений. Тогда задачу можно определить следующим образом. Пусть  $S = \{s_i \mid i = 1:k\}$  – набор из  $k$  потребителей, требующих получения ресурсов  $R(s_i) = \{r_j \mid j = 1:n_i\}$ , каждый из которых может располагаться в некотором множестве мест  $P(r_j) = \{p_l \mid l = 1:q_j\}$ . Есть непрерывное или дискретное множество  $T$ , определяющее собой моменты времени. (Причём есть множество различных способов, чтобы представить это множество, например, в случае если время принимается непрерывным, то его можно представить себе как вектор  $\{[t_1 \ t_2]^T \mid t_2 > t_1, t_1 \in \mathfrak{R}, t_2 \in \mathfrak{R}\}$ , где  $t_1$  – начальное время события в секундах,  $t_2$  – конечное время

события в секундах, а за ноль принимается некоторая точка отсчёта). Тогда решением задачи будет отображение кортежей  $\langle s_i, r_j, p_l \rangle$  на множество  $T$ , причём таким образом, что если

$$T_1(s_{i1}, r_{j1}, p_{l1}) \cap T_2(s_{i2}, r_{j2}, p_{l2}) \neq \emptyset, (1)$$

то обязательно выполнится:

$$\{(s_{i1} \neq s_{i2} \wedge r_{j1} \neq r_{j2} \wedge p_{l1} \neq p_{l2}) \vee (s_{i1} = s_{i2} \wedge r_{j1} = r_{j2} \wedge p_{l1} = p_{l2})\}. (2)$$

Т.е. в один и тот же момент времени не будет такого, что один и тот же потребитель должен быть в разных местах или пользоваться разными ресурсами, также не будет такого, что один ресурс пользуется разными потребителями или должен быть в разных местах. Кроме того в одном месте будет только один ресурс и один потребитель.

Например, в случае расписания вуза  $S$  – множество студенческих групп,  $R$  – множество пар дисциплина–преподаватель, соответствующих выбранной учебной группе (если в указанный промежуток времени требуется больше чем одно занятие по этому предмету, то объект дублируется),  $P$  – множество аудиторий, лабораторий или компьютерных залов, пригодных для данной дисциплины. Множество  $T$  – дискретно и представляет собой ячейки сетки расписания вуза.

В данной работе рассматривается случай расписания для одного человека. Отличие состоит в том, что  $S$  – не множество потребителей, а один потребитель, т.е.  $k = 1$ . Тогда задача упрощается и сводится к отображению  $\langle r_j, p_l \rangle$  на множество  $T$  с ограничениями (1) и (2). Тогда  $R$  – это набор задач/событий, которые нужно сделать человеку, а  $P$  – множество мест, где он может их сделать. Однако следует учесть, что у обычного человека есть ряд обязательств, таких как работа, учёба и др. Таким образом есть ряд задач, время выполнения которых не зависят от человека, составляющего расписание. Тогда следует разбить задачи на «гибкие» и «жёсткие», где для жёстких задач отображение на  $T$  уже задано, а для гибких задач его требуется найти. Таким образом, оптимальное решение будет выбираться на основе наилучшего размещения «гибких» задач. Для «жёстких» задач определено начальное и конечное время и место, где выполняется задача. Для «гибких» определены длительность, место проведения и крайний рок выполнения. Тогда чтобы решить поставленную задачу следует определить начальное и конечное время для «гибких» задач. Визуальное представление задачи приведено на рисунке 1.

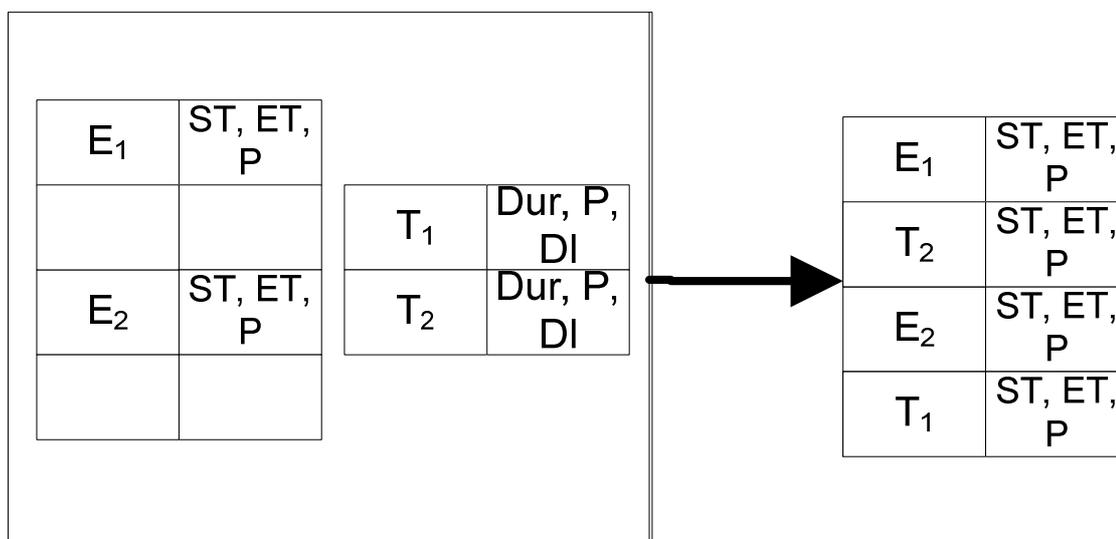


Рис. 1. Слева указано расписание до решения задачи, справа показано готовое расписание

### Методы решения задачи расписания.

За время существования задачи сформировано множество методов решения данной задачи.

Классическими методами решения задачи являются метод раскраски графа и симплекс-метод. Метод раскраски графа не применим в задаче составления расписания для одного человека, так как он предназначен для составления расписания, когда потребителей несколько. При этом методе строится дерево, каждая ветвь отражает состояние потребителя. Основная идея состоит в том, что требуется не допустить циклов (что равносильно нарушению условия (3) ) [5]. Такой алгоритм очень плохо учитывает пожелания отдельных пользователей, поэтому редко применим на практике. Симплекс метод заключается в нахождении решения с помощью максимизации целевой функции[3]. Такой подход намного лучше учитывает пожелания отдельных участников системы, так как их можно включить в целевую функцию.

Многоагентный подход заключается в разбиении задачи на подзадачи, где в каждой подзадаче есть агент (который представляет собой интересы пользователя системы), который ищет решение своей задачи, согласующееся с решениями других агентов. Агенты между собой обмениваются сообщениями, что приводит к решению задачи. В рамках задачи составления расписания для одного человека этот метод допустим, но не раскрывает свой полный потенциал, так как изначально метод подразумевает наличие большого числа пользователей в системе. Также, следует отметить, что при многоагентном подходе можно использовать многопроцессорную

обработку программы, что ускорит нахождение результата и может быть эффективным при большом количестве пользователей (по сравнению с другими методами).

Кроме того для решения данной задачи используют эвристические алгоритмы – алгоритм имитации отжига и генетический алгоритм. Алгоритм имитации отжига основывается на физическом процессе кристаллизации вещества. Вычисляется начальное решение и целевая функция этого решения. Задаётся начальная «температура», далее формируется новое допустимое решение. Вычисляется целевая функция нового решения, если её значение больше исходного, то решение заменяет старое, иначе вычисляется вероятность замены в зависимости от температуры и разности значений целевых функций. Затем изменяется температура, и цикл замыкается на нахождении нового решения и продолжается пока температура не понизится до нуля. Метод эффективен для составления небольших расписаний. [2]

Генетический алгоритм – это эвристический алгоритм, сходный по своему механизму с естественным отбором. Чтобы использовать этот алгоритм следует создать модель каждого решения как набор параметров, которые впоследствии мы будем менять. В результате мы получим то, что на языке биологии называлось бы «генотипом организма», а в нашем случае это будут значения параметров каждого решения. Далее определяется набор стартовых решений, которые, скорее всего, будут неудачными с точки зрения критерия оптимальности. Финальным шагом подготовки является создание целевой функции, которая играет роль критерия оптимальности. Далее в цикле симулируется производство нового «потомства» (решений) на основе старых, новые решения могут мутировать с некоторой вероятностью и далее следует естественный отбор, в ходе которого лучшие решения проходят в следующий этап, а худшие «погибают». Цикл заканчивается, если время на выполнение задачи вышло или количество итераций достигло заданного значения. Блок-схема работы такого алгоритма приведена на рисунке 2.



Рис. 2. Блок-схема генетического алгоритма

Такой подход даёт ряд преимуществ. Во-первых, генетический алгоритм выдаёт корректный с точки зрения условий (1) и (2) результат на любой итерации цикла, следовательно получить допустимый результат можно в любой момент работы алгоритма, с течением времени решение лишь улучшается. Во-вторых, алгоритм очень хорошо учитывает пожелания пользователя, так как можно очень легко добавить любой критерий оптимальности в целевую функцию и учитывать его пропорционально важности этого критерия. В-третьих, данный алгоритм линейно зависит от объёма входных данных и даёт результат за время

$$T = O(\text{iter} * \text{popul} * V),$$

где  $\text{iter}$  – максимальное число итераций цикла,  $\text{popul}$  – размер популяции (число генерирующихся решений),  $V$  – число входных задач, избавляя нас от неполиномиального времени вычисления.

### Решения при генетическом подходе

Далее будут рассмотрены возможные методы определения основных этапов генетического алгоритма для задачи составления расписания.

При создании популяции сначала создаём расписание, которое заполнено «жёсткими» задачами. Далее дублируем это расписание до количества нашей популяции и в произвольном порядке заполняем «гибкими» задачами. Таким образом сформировано первое поколение решений.

Целевая функция является критерием оптимальности, т.е. с её помощью определяется насколько хорошо то или иное решение. В рамках задачи составления расписания наиболее простым решением будет использовать систему штрафов[4] и поощрений для целевой функции. В начале оценки расписания мы присвоим целевой функции некоторую константу. Затем будем вычитать из функции некоторое значение, если расписание нас не удовлетворяет по какому-то критерию (например, если срок выполнения «гибкой» задачи просрочен) или прибавлять некоторое значение, если расписание нас удовлетворяет (например, за каждый свободный час в выходные).

В начале цикла происходит селекция – выбор решений среди существующих, чтобы произвести новые решения. Применительно к задаче составления расписания селекция ничем не отличается от селекции в других задачах. Выбираются два решения с вероятностью прямо пропорциональной значению целевой функции этих решений.

После того как выбрана пара решений из них формируется новое. Коэффициент увеличения числа решений (во сколько раз увеличивается популяция). Чтобы сформировать новое решение соответствующие «гибкие» задачи переносятся в новое решение от одного из исходных решений случайным образом. Если такая вставка нарушает условие непересекаемости задач во времени, то вставляем «гибкую» задачу в ближайшее допустимое место во временной линии. Таким образом, создаётся новое решение, обладающее, скорее всего, новыми признаками и новым значением целевой функции, которое внутренне непротиворечиво (задачи не пересекаются во времени).

Чтобы обеспечить разнообразие решений и выйти из локального максимума (минимума) целевой функции применяют механизм мутации. С некоторой вероятностью новое решение, сформированное в процессе кроссовера, может измениться. В таком случае мы случайным образом меняем место «гибких» задач в этом решении.

В конце цикла мы выбираем наилучшие решения. Отсортировав по убыванию необходимо убрать плохие решения. Однако, если останется только одно решение, пусть даже наилучшее в данный момент, мы можем не достичь более лучших решений, так как в таком случае кроссовер не будет давать новых решений. На практике такая ситуация может возникнуть, если просто отсекал худшую часть массива решений. Поэтому необходимо регулировать число лучших решений, чтобы они не превысили некоторую часть.

## **Вывод**

Генетический алгоритм является наилучшим методом при решении задачи составления расписания для одного человека, так как является наиболее быстрым, и наиболее полно учитывает пожелания пользователя.

## **Список литературы**

1. Конькова И.С. Использование генетического алгоритма в задаче оптимизации расписания вуза // Вестник ТвГТУ. 2012. Вып. 22. С. 26-31.
2. Сидорин А.Б., Ликучева Л.В., Дворянкин А.М. Методы автоматизации составления расписания занятий. В 2 ч. Ч. 1. Классические методы. Режим доступа: <http://cyberleninka.ru/article/n/metody-avtomatizatsii-sostavleniya-raspisaniya-zanyatiy-chast-1-klassicheskie-metody> (дата обращения 14.05.2015.)
3. Сидорин А.Б., Ликучева Л.В., Дворянкин А.М. Методы автоматизации составления расписания занятий. В 2 ч. Ч. 2. Эвристические методы оптимизации. Режим доступа: <http://cyberleninka.ru/article/n/metody-avtomatizatsii-sostavleniya-raspisaniya-zanyatiy-chast-2-evristicheskie-metody-optimizatsii> (дата обращения 14.05.2015.)
4. Волков И.К., Загоруйко Е.А. Исследование операций. М.: Издательство МГТУ им. Баумана. 2000. 436 с.
5. Яндыбаева Н.В. Генетический алгоритм в задаче оптимизации учебного расписания вуза // Современные наукоемкие технологии. 2009. № 11. С. 97-98
6. Свиридова О.В., Михайлов А.В. Обобщенный анализ методов проектирования и разработки автоматизированного составления расписания образовательных учреждений // Молодой ученый. 2013. № 6. С. 214-216.