Простой алгоритм заполнения расписания

Метод круга является стандартным [алгоритмом](https://en.wikipedia.org/wiki/Algorithm) для создания расписания турнира с круговым приемом . Все участники присваиваются номера, а затем соединяются в парном раунде:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Раунд 1. (1 играет 14, 2 играет 13, ...)** | | | | | | |
| **1** | **2** | **3** | **4** | **5** | **6** | **7** |
| **14** | **13** | **12** | **11** | **10** | **9** | **8** |

Затем один из участников в первом или последнем столбце таблицы фиксируется (номер один в этом примере), а остальные поворачиваются по часовой стрелке на одну позицию

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Раунд 2. (1 игра 13, 14 игра 12, ...)** | | | | | | |
| **1** | **14** | **2** | **3** | **4** | **5** | **6** |
| **13** | **12** | **11** | **10** | **9** | **8** | **7** |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Раунд 3. (1 играет 12, 13 играет 11, ...)** | | | | | | |
| **1** | **13** | **14** | **2** | **3** | **4** | **5** |
| **12** | **11** | **10** | **9** | **8** | **7** | **6** |

Это повторяется до тех пор, пока вы не окажетесь почти в исходной позиции:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Раунд 13. (1 игра 2, 3 игры 14, ...)** | | | | | | |
| **1** | **3** | **4** | **5** | **6** | **7** | **8** |
| **2** | **14** | **13** | **12** | **11** | **10** | **9** |

Если количество участников нечетное, можно добавить фиктивного участника, когда число конкурентов нечетное, оно работает не так хорошо. В качестве альтернативы широко используется в планировании турниров. Это график, в котором игрок 14 имеет фиксированную позицию, а все остальные игроки вращаются по часовой стрелке.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Раунд 1** | 1-14 | 2-13 | 3-12 | 4-11 | 5-10 | 6-9 | 7-8 |
| **Раунд 2.** | 14-8 | 9-7 | 10-6 | 11-5 | 12-4 | 13-3 | 1-2 |
| **Раунд 3** | 2-14 | 3-1 | 4-13 | 5-12 | 6-11 | 7-10 | 8-9 |
| **...** | ... | | | | | | |
| **Раунд 13** | 7-14 | 8-6 | 9-5 | 10-4 | 11-3 | 12-2 | 13-1 |

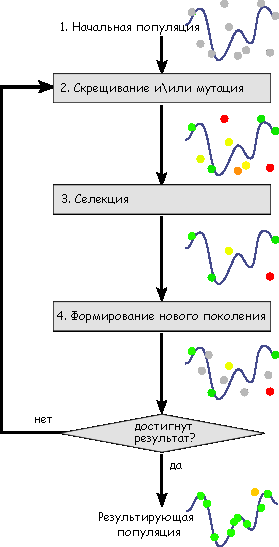
Чтобы построить следующий раунд, последний игрок, номер 8 в первом раунде, перемещается к главе стола, затем игрок 9 против игрока 7, игрок 10 против 6, пока игрок 1 против игрока 2.

Это расписание также может быть представлено в виде (n-1, n-1) таблицы, выражающей раунд, в котором игроки встречаются друг с другом. Например, игрок 7 играет против игрока 11 в раунде 4. Все игры в раунде составляют диагональ в таблице.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | **Диагональная схема** | | | | | | | | | | | | | | | | | | | | | | | | | | | **×** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | | **1** |  |  |  |  |  |  |  |  |  |  |  |  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | | **2** |  |  |  |  |  |  |  |  |  |  |  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 |  | | **3** |  |  |  |  |  |  |  |  |  |  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 |  |  | | **4** |  |  |  |  |  |  |  |  |  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 |  |  |  | | **5** |  |  |  |  |  |  |  |  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 |  |  |  |  | | **6** |  |  |  |  |  |  |  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 |  |  |  |  |  | | **7** |  |  |  |  |  |  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 |  |  |  |  |  |  | | **8** |  |  |  |  |  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 |  |  |  |  |  |  |  | | **9** |  |  |  |  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 |  |  |  |  |  |  |  |  | | **10** |  |  |  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 |  |  |  |  |  |  |  |  |  | | **11** |  |  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 |  |  |  |  |  |  |  |  |  |  | | **12** |  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 |  |  |  |  |  |  |  |  |  |  |  | | **13** | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 |  |  |  |  |  |  |  |  |  |  |  |  | |

Генетический алгоритм

Генети́ческий алгори́тм (англ. genetic algorithm) — это эвристический алгоритм поиска, используемый для решения задач оптимизации и моделирования путём случайного подбора, комбинирования и вариации искомых параметров с использованием механизмов, аналогичных естественному отбору в природе. Отличительной особенностью генетического алгоритма является акцент на использование оператора «скрещивания», который производит операцию рекомбинации решений-кандидатов, роль которой аналогична роли скрещивания в живой природе.



Этапы генетического алгоритма:

1. Задать целевую функцию (приспособленности) для особей популяции
2. Создать начальную популяцию

* (Начало цикла)

1. Размножение (скрещивание)
2. Мутирование
3. Вычислить значение целевой функции для всех особей
4. Формирование нового поколения (селекция)
5. Если выполняются условия остановки, то (конец цикла), иначе (начало цикла).

Проблемы:

* Повторная оценка функции приспособленности (фитнесс-функции) для сложных проблем, часто является фактором, ограничивающим использование алгоритмов искусственной эволюции. Поиск оптимального решения для сложной задачи высокой размерности зачастую требует очень затратной оценки функции приспособленности.
* Генетические алгоритмы плохо масштабируемы под сложность решаемой проблемы.
* Решение является более пригодным лишь по сравнению с другими решениями. В результате условие остановки алгоритма неясно для каждой проблемы.
* Во многих задачах генетические алгоритмы имеют тенденцию сходиться к локальному оптимуму или даже к произвольной точке, а не к глобальному оптимуму для данной задачи.

Поиск с запретами

Поиск с запретами или табу-поиск является мета-алгоритмом поиска, использующим методы локального поиска, используемые для математической оптимизации.

Поиск с запретами использует процедуру локального поиска или поиска по соседям, чтобы итеративно продвигаться от одного потенциального решения x к улучшенному решению x' в окрестности x, пока не достигнем выполнения некоторого остановочного критерия (обычно, это количество итераций или порог целевой оценки). Процедуры локального поиска часто застревают в областях с плохими целевыми оценками или в областях, где оценка образует плато (ровную горизонтальную поверхность). Поиск с запретами тщательно исследует соседство каждого решения в процессе поиска. Решения, признанные новыми соседями, N\*(x), определяются с помощью структур в памяти. Используя эти структуры поиск прогрессирует путём итеративного перехода от текущего решения x к улучшенному решению x' из списка N\*(x). Эти структуры образуют так называемые табу-списки, множество правил и помеченных решений, используемых для фильтрации, какие решения из соседей N\*(x) обрабатывать при поиске. В простейшей форме, табу-список является краткосрочным набором решений, которые были посещены за последние итерации (менее чем за n итераций, где n равно числу запоминаемых решений и это число называется сроком действия табу). Более часто табу-лист состоит из решений, которые были изменены в процессе перехода от одного решения к другому. Удобно для простоты изложения, понимать «решение» закодированным и представленным некоторыми атрибутами.

Пример: задача коммивояжёра

Задача коммивояжёра иногда используется, чтобы показать работу поиска с запретами. Эта задача спрашивает, если дан список городов, каков кратчайший маршрут для посещения всех городов? Например, если город A и город B находятся близко друг от друга, а город C отстоит о них далеко, общая длина маршрута будет короче, если мы сначала посетим A и B, а затем отправимся в город C. Поскольку нахождение оптимального решения NP-трудно, для получения близкого к оптимальному решения полезны основанные на эвристике приближённые методы (такие как локальный поиск). Для получения хороших решений задачи коммивояжёра, важно исследовать структуру графа. Значение исследования структуры задачи является повторяющейся темой в мета-алгоритмических методах, а поиск с запретами хорошо подходит для этого. Класс стратегий, связанный с поиском с запретами и называемый методами извлечения цепочек (англ. ejection chain methods), дают возможность получить высококачественные решения задачи коммивояжёра эффективно.

Алгоритм имитации отжига

Алгори́тм имита́ции о́тжига (англ. Simulated annealing) — общий алгоритмический метод решения задачи глобальной оптимизации, особенно дискретной и комбинаторной оптимизации. Один из примеров методов Монте-Карло.

Применение

* Обучение нейронных сетей
* Решение комбинаторных задач, например, задачи о расстановке ферзей
* Решение задачи коммивояжера

Алгоритм основывается на имитации физического процесса, который происходит при кристаллизации вещества, в том числе при отжиге металлов. Предполагается, что атомы уже выстроились в кристаллическую решётку, но ещё допустимы переходы отдельных атомов из одной ячейки в другую. Предполагается, что процесс протекает при постепенно понижающейся температуре. Переход атома из одной ячейки в другую происходит с некоторой вероятностью, причём вероятность уменьшается с понижением температуры. Устойчивая кристаллическая решётка соответствует минимуму энергии атомов, поэтому атом либо переходит в состояние с меньшим уровнем энергии, либо остаётся на месте.

Жадный алгоритм

Жадный алгоритм — алгоритм, заключающийся в принятии локально оптимальных решений на каждом этапе, допуская, что конечное решение также окажется оптимальным.

Применение

* Размен монет
* Выбор заявок

Задачи, в которых жадные алгоритмы не дают оптимального решения

* Задача коммивояжера
* Задача минимальное раскраски графа
* Задача разбиения графа на подграфы
* Задача выделения максимальной клики
* **Задачи, связанные с составлением расписаний**