Лекция 6. Эволюционные методы

Описание идей эволюционных методов

Эволюционные методы относятся к числу эффективных средств решения задач оптимизации и структурного синтеза проектных решений. Они основаны на использовании принципов оптимального приспособления организмов в живой природе к условиям окружающей среды. К числу эволюционных относятся методы генетические, колонии муравьев, поведения толпы. Наиболее развиты и востребованы в настоящее время генетические алгоритмы.

По мере развития техники и технологий растет доля сложных задач проектирования и управления, для решения которых применение традиционных методов проблематично. Поэтому все большее внимание уделяется применению методов искусственного интеллекта.

В настоящее время изучаются и развиваются четыре основные группы эволюционных методов – генетические методы (Genetic Algorithms — GA), генетическое программирование, методы поведения "толпы" (Particles Swarm Optimization — PSO) и методы "колонии муравьев" (Ant Colony Optimization — ACO). В настоящее время в системах автоматизированного проектирования и управления наибольшее внимание среди них уделяется генетическим методам, часто именуемыми по традиции генетическими алгоритмами.

Эволюционные методы (ЭМ) являются приближенными (эвристическими) методами решения задач оптимизации и структурного синтеза. Большинство ЭМ основано на статистическом подходе к исследованию ситуаций и итерационном приближении к искомому решению.

Эволюционные вычисления составляют один из разделов искусственного интеллекта. При построении систем ИИ по данному подходу основное внимание уделяется построению начальной модели, и правилам, по которым она может изменяться (эволюционировать). Причем модель может быть составлена по самым различным методам, например, это может быть и нейронная сеть и набор логических правил. К основным эволюционным

методам относятся методы отжига, генетические, поведения "толпы" (PSO), колонии муравьев (ACO), генетического программирования.

В отличие от точных методов математического программирования ЭМ позволяют находить решения, близкие к оптимальным, за приемлемое время, а в отличие от других эвристических методов оптимизации характеризуются существенно меньшей зависимостью от особенностей приложения (т.е. более универсальны) и в большинстве случаев обеспечивают лучшую степень приближения к оптимальному решению. Универсальность ЭМ определяется также применимостью к задачам с неметризуемым пространством управляемых переменных (т.е. среди управляемых переменных могут быть и лингвистические величины, т.е. не имеющие количественного выражения).

В методе отжига (Simulated Annealing) имитируется процесс минимизации потенциальной энергии тела во время отжига деталей. В текущей точке поиска происходит изменение некоторых управляемых параметров. Новая точка принимается всегда при улучшении целевой функции и лишь с некоторой вероятностью при ее ухудшении.

Важнейшим частным случаем ЭМ являются генетические методы и алгоритмы. Генетические алгоритмы (ГА) основаны на поиске лучших решений с помощью наследования и усиления полезных свойств множества объектов определенного приложения в процессе имитации их эволюции.

Свойства объектов представлены значениями параметров, объединяемых в запись, называемую в ЭМ хромосомой. В ГА оперируют подмножеством хромосом, называемом популяцией. Имитация генетических принципов — вероятностный выбор родителей среди членов популяции, скрещивание их хромосом, отбор потомков для включения в новые поколения объектов на основе оценки целевой функции — ведет к эволюционному улучшению значений целевой функции (функции полезности) от поколения к поколению.

Генетические алгоритмы

Для применения ГА необходимо:

1. выделить совокупность свойств объекта, характеризуемых внутренними параметрами и влияющих на его полезность, т.е. выделить множество управляемых параметров$X = (x_1, x_2, \dots, x_n)$ среди x_i могут быть величины различных типов (**real**, **integer**, **Boolean**, **enumeration**). Наличие нечисловых величин (enumeration) обусловливает возможность решения задач не только параметрической, но и структурной оптимизации;

2. сформулировать количественную оценку полезности вариантов объекта — функцию полезности F. Если в исходном виде задача многокритериальна, то такая формулировка означает выбор скалярного (обобщенного) критерия;

3. разработать математическую модель объекта, представляющую собой алгоритм вычисления F для заданного вектора X;

4. представить вектор X в форме хромосомы — записи следующего вида (см. рисунок 1).

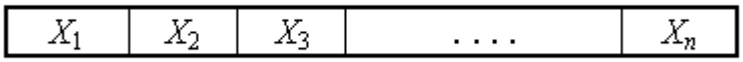

Рисунок 1 – Хромосома

В ГА используется такая терминология:

– ген – управляемый параметр x_i;

– аллель – значение гена;

– локус (позиция) – позиция, занимаемая геном в хромосоме;

– генотип – экземпляр хромосомы, генотип представляет совокупность внутренних параметров проектируемого с помощью ГА объекта;

– генофонд – множество всех возможных генотипов;

– функция полезности (приспособленности) F – целевая функция;

– фенотип – совокупность значений критериев, получаемых после декодирования хромосомы, под фенотипом часто понимают совокупность выходных параметров синтезируемого с помощью ГА объекта.

Генетический алгоритм представлен на рисунке 2.

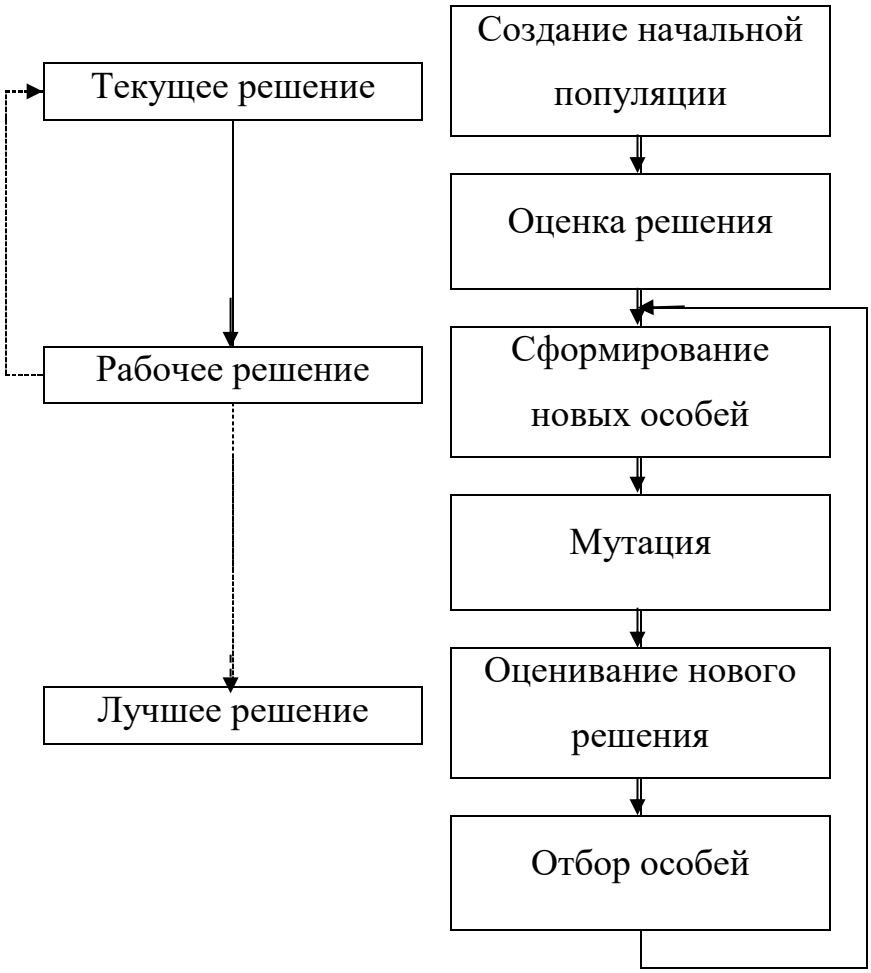

Рисунок 2 – Генетический алгоритм

Вычислительный процесс начинается с генерации исходного поколения – множества, включающего N хромосом, N – размер популяции. Генерация выполняется случайным выбором аллелей каждого гена.

Далее организуется циклический процесс смены поколений:

```
for (k=0; k<G; k++)
{ for (j=0; j<N; j++)
  { Выбор родительской пары хромосом;
    Кроссовер;
    Мутации;
    Оценка функции полезности F потомков;
```

Селекция;
}
Замена текущего поколения новым;
}

Для каждого витка внешнего цикла генетического алгоритма выполняется внутренний цикл, на котором формируются экземпляры нового (следующего за текущим) поколения. Во внутреннем цикле повторяются операторы выбора родителей, кроссовера родительских хромосом, мутации, оценки приспособленности потомков, селекции хромосом для включения в очередное поколение.

Рассмотрим алгоритмы выполнения операторов в простом генетическом алгоритме.

1. Выбор родителей.

Этот оператор имитирует естественный отбор, если отбор в родительскую пару хромосом с лучшими значениями функции полезности F более вероятен. Например, пусть F требуется минимизировать. Тогда вероятность P_i выбора родителя с хромосомой C_i можно рассчитать по формуле

$$P_i = \frac{(F_{max} - F_i)}{\sum_{j=1}^{N}(F_{max} - F_j)},$$

где F_{max} – наихудшее значение целевой функции F среди экземпляров (членов) текущего поколения, F_i – значение целевой функции i-го экземпляра.

Правило, представленное ранее называют правилом колеса рулетки. Если в колесе рулетки выделить секторы, пропорциональные значениям $F_{max} - F_i$, то вероятности попадания в них суть P_i, определяемые в соответствии с этим правилом.

Пример 1

Пусть $N = 4$, значения F_i и P_i приведены в таблице 1.

Таблица 1 – Начальные данные

i	F_i	$F_{max}-F_i$	P_i
1	2	5	0,5
2	7	0	0
3	6	1	0,1
4	3	4	0,4

Кроссовер (скрещивание)

Кроссовер, иногда называемый кроссинговером, заключается в передаче участков генов от родителей к потомкам. При простом (одноточечном) кроссовере хромосомы родителей разрываются в некоторой позиции, одинаковой для обоих родителей, выбор места разрыва равновероятен, далее происходит рекомбинация образующихся частей родительских хромосом, как это показано в таблице 2, где разрыв подразумевается между пятым и шестым локусами.

Таблица 2 – Результаты кроссовера

Хромосома	Ген 1	Ген 2	Ген 3	Ген 4	Ген 5	Ген 6	Ген 7	Ген 8
Родитель A	f	a	c	d	g	k	v	e
Родитель B	a	b	c	d	e	f	g	h
Потомок C	f	a	c	d	g	f	g	h
Потомок D	a	b	c	d	e	k	v	e

Мутации

Оператор мутации выполняется с некоторой вероятностью P_m, т.е. с вероятностью P_m происходит замена аллелей случайным значением, выбираемым с равной вероятностью в области определения гена. Именно благодаря мутациям расширяется область генетического поиска.

Селекция

После каждого акта генерации пары потомков в новое поколение включается лучший экземпляр пары.

Внутренний цикл заканчивается, когда число экземпляров нового поколения станет равным N. Количество повторений G внешнего цикла чаще всего определяется автоматически по появлению признаков вырождения (стагнации) популяции, но с условием не превышения заданного лимита машинного времени.

Метод имитации отжига

Алгоритм отжига – это метод оптимизации, который называется *отжигом,* или *симуляцией восстановления* (Simulated annealing). Как ясно из названия, метод поиска моделирует процесс восстановления. Восстановление – это физический процесс, который заключается в нагреве и последующем контролируемом охлаждении субстанции. В результате получается прочная кристаллическая структура, которая отличается от структуры с дефектами, образующейся при быстром беспорядочном охлаждении. Структура здесь представляет собой кодированное решение, а температура используется для того, чтобы указать, как и когда будут приниматься новые решения.

Естественная мотивация

Свойства структуры зависят от коэффициента охлаждения после того, как субстанция была нагрета до точки плавления. Если структура охлаждалась медленно, будут сформированы крупные кристаллы, что очень полезно для строения субстанции. Если субстанция охлаждается скачкообразно, образуется слабая структура.

Чтобы расплавить материал, требуется большое количество энергии. При понижении температуры уменьшается и количество энергии. Чтобы яснее представить процесс восстановления, рассмотрим следующий пример. «Взбалтывание при высокой температуре сопровождается высокой молекулярной активностью в физической системе. Представьте себе, что вы взбалтываете емкость, в которой находится какая-то поверхность сложной формы. Внутри емкости также имеется шарик, который пытается найти точку

равновесия. При высокой температуре шарик может свободно перемещаться по поверхности, а при низкой температуре взбалтывание становится менее интенсивным и передвижения шарика сокращаются. Задача заключается в том, чтобы найти точку минимального перемещения при сильном «взбалтывании». При снижении температуры уменьшается вероятность того, что шарик выйдет из точки равновесия. Именно в таком виде процесс поиска заимствуется из восстановления. Алгоритм отжига очень прост и представлен на рисунке 3.

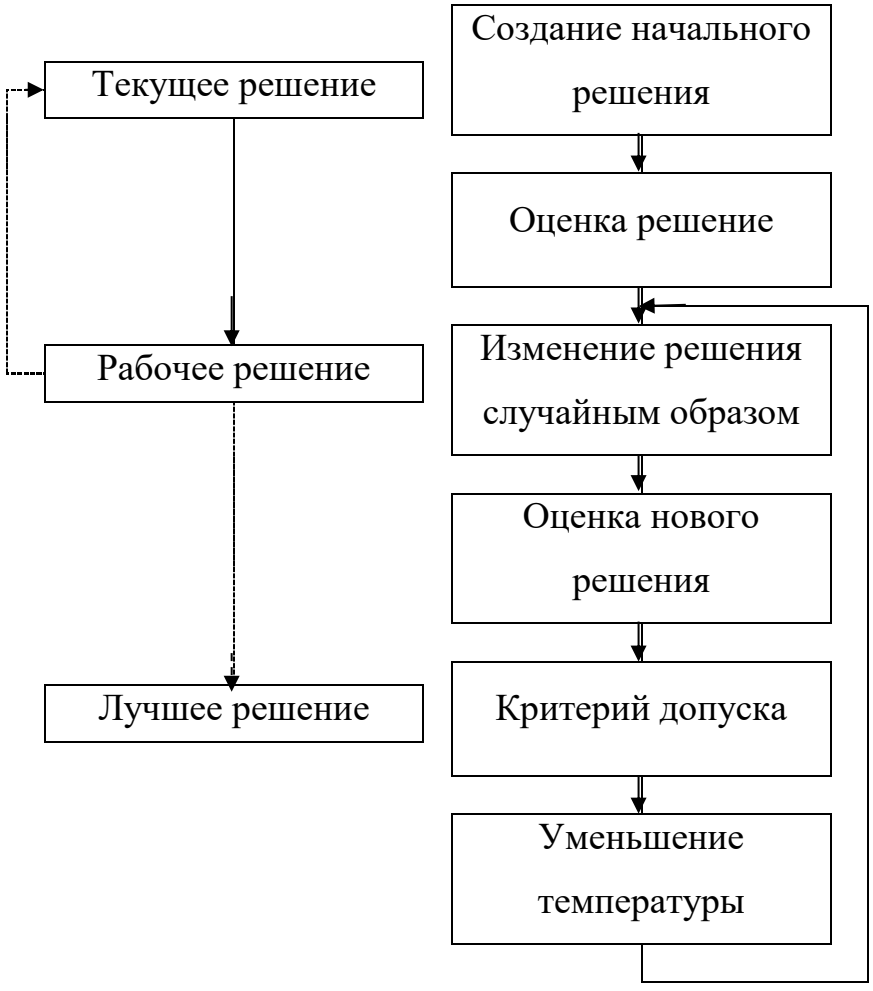

Рисунок 3 – Алгоритм метода имитации отжига

Начальное решение

Для большинства проблем начальное решение будет случайным. На самом первом шаге оно помещается в *текущее решение*. Другая возможность заключается в том, чтобы загрузить в качестве начального решения уже существующее, возможно, то самое, которое было найдено во время предыдущего поиска. Это предоставляет алгоритму базу, на основании которой выполняется поиск оптимального решения проблемы.

Оценка решения

Оценка решения состоит из декодирования текущего решения и выполнения нужного действия, позволяющего понять его целесообразность для решения данной проблемы. Обратите внимание, что закодированное решение может просто состоять из набора переменных. Они будут декодированы из существующего решения, а затем эффективность решения будет оценена на основании того, насколько успешно удалось решить данную задачу.

Случайный поиск решения

Поиск решения начинается с копирования текущего решения в рабочее решение. Затем мы произвольно модифицируем рабочее решение. Как именно модифицируется рабочее решение, зависит от того, каким образом оно представляется (кодируется). Представьте себе кодировку задачи коммивояжера, в которой каждый элемент представляет собой город. Чтобы выполнить поиск по рабочему решению, мы берем два элемента и переставляем их. Это позволяет сохранить целостность решения, так как при этом не происходит повторения или пропуска города. После выполнения поиска рабочего решения мы оцениваем решение, как было описано ранее. Поиск нового решения основан на методе Монте-Карло (то есть случайным образом).

Критерий допуска

На этом этапе алгоритма у нас имеется два решения. Первое - это наше оригинальное решение, которое называется текущим решением, а второе -

найденное решение, которое именуется рабочим решением. С каждым решением связана определенная энергия, представляющая собой его эффективность (допустим, что чем ниже Энергия, тем более эффективно решение). Затем рабочее решение сравнивается с текущим решением. Если рабочее решение имеет меньшую энергию, чем текущее решение (то есть является более предпочтительным), то мы копируем рабочее решение в текущее решение и переходим к этапу снижения температуры. Однако если рабочее решение хуже, чем текущее решение, мы определяем критерий допуска, чтобы выяснить, что следует сделать с текущим рабочим решением. При высокой температуре (свыше 60 *C) плохие решения принимаются чаще, чем отбрасываются. Если энергия меньше, вероятность принятия решения выше. При снижении температуры вероятность принятия худшего решения также снижается. При этом более высокий уровень энергии также способствует уменьшению вероятности принятия худшего решения. При высоких температурах симулированное восстановление позволяет принимать худшие решения для того, чтобы произвести более полный поиск решений. При снижении температуры диапазон поиска также уменьшается, пока не достигается равенство при температуре 0°.

Снижение температуры

После ряда итераций по алгоритму при данной температуре мы ненамного снижаем ее. Существует множество вариантов снижения температуры. Возможны разные стратегии снижения температуры, включая линейные и нелинейные функции.

Повтор

При одной температуре выполняется несколько итераций. После завершения итераций температура будет понижена. Процесс продолжится, пока температура не достигнет нуля.

Оптимизация алгоритма

Вы можете изменять параметры алгоритма в зависимости от сложности проблемы, которую нужно решить. В этом разделе описаны параметры,

которые допускается переопределять, а также результаты возможных изменений.

Начальная температура

Начальная температура должна быть достаточно высокой, чтобы сделать возможным выбор из других областей диапазона решений. По утверждению Грэхема Кендалла, если известно максимальное расстояние между соседними решениями, то легко рассчитать начальную температуру. Начальную температуру также можно изменять динамически. Если задать статистику по коэффициенту допуска худших решений и нахождению новых лучших решений, можно повышать температуру до тех пор, пока не будет достигнуто нужное количество допусков (открытий новых решений). Этот процесс аналогичен нагреву субстанции до перехода ее в жидкую форму, после чего уже нет смысла повышать температуру.

Конечная температура

В большинстве задач ноль является конечной температурой и может быть критерием окончания алгоритма. Если есть необходимость, то можно определить окончание алгоритма при достижении заданной точности или качества решения в соответствии с целевой функцией (функция полезности).

Функция изменения температуры

Используемую функцию изменения температуры можно модифицировать в зависимости от решаемой задачи. Снижение температуры допускается определять и с помощью многих других функций. Результатом использования этих функций может быть постепенное снижение температуры в первой половине графика или медленное снижение, за которыми следует резкий спад.

Количество итераций при одном значении температуры

При высоких температурах алгоритм отжига выполняет поиск оптимального решения во всем диапазоне решений. При снижении температуры движение уменьшается, и алгоритм ищет локальный оптимум, чтобы улучшить решение. Поэтому количество итераций, заданное для

каждой температуры, имеет большое значение. Чтобы правильно определить количество итераций, которое оптимально подходит для решения проблемы, необходимо поэкспериментировать.

Метод отжига может быть эффективным при решении задач различных классов, требующих оптимизации. Ниже приводится их краткий список:

1. создание пути;
2. реконструкция изображения;
3. назначение задач и планирование;
4. размещение сети;
5. глобальная маршрутизация;
6. обнаружение и распознавание визуальных объектов;
7. разработка специальных цифровых фильтров.

Поскольку метод отжига представляет собой процесс генерации случайных чисел, поиск решения с использованием данного алгоритма может занять значительное время. В некоторых случаях алгоритм вообще не находит решение или выбирает не самое оптимальное.

Алгоритм отжига как способ выполнения процедур поиска и оптимизации. Данный метод является аналогом процесса нагревания тела до состояния плавления с последующим постепенным охлаждением. При высоких температурах поиск ведется по всему диапазону. При снижении температуры диапазон поиска уменьшается до небольшой области вокруг текущего решения.

Контрольные вопросы по теме:

1. Поясните смысл понятия "генетические алгоритмы".
2. В чем заключается эволюционный поиск?
3. Приведите основные цели и задачи генетических алгоритмов.
4. Выделите основные отличительные особенности ГА.
5. Приведите основные понятия и определения генетических алгоритмов.

6. Что такое целевая функция в генетических алгоритмах?
7. Перечислите предварительные этапы работы генетических алгоритмов.
8. В чем суть метода имитации отжига?
9. Какие задачи можно решать методом имитации отжига?