Эвристики

Рассмотренный ранее метод ветвей и границ позволяет значительно сократить перебор дерева решений, но зачастую даже с использованием этого метода нам не удастся найти оптимальное решение. Например, дерево для формирования портфеля из 65 элементов содержит более $7 * 10^{19}$ узлов. Предположим, что метод ветвей и границ перебирает десятую часть процента этих узлов, а компьютер проверяет миллион узлов в секунду. В этих условиях для решения задачи потребовалось бы более 2 тысяч лет, чтобы решить эту задачу. В задачах, где метод ветвей и границ работает недостаточно быстро, можно использовать эвристику.

Если качество решения не критично, то приемлемым можно считать результат, данный эвристикой. В некоторых случаях вы не можете знать входные данные с абсолютной точностью. Тогда хорошее эвристическое решение может иметь такую же силу, как и лучшее теоретическое решение.

Очень часто метод ветвей и границ используется для выбора инвестиционных компаний. Однако вложения могут быть рискованными, и точные результаты чаще всего неизвестны. Мы не можем знать точную прибыль, а зачастую даже стоимость инвестиций. В этом случае эффективное эвристическое решение может быть столь же надёжно, как и лучшее точно вычисленное решение.

Восхождение на холм

Эвристический метод восхождения на холм вносит изменения в текущее решение, продвигая его максимально близко к цели. Этот процесс называется восхождением на холм, потому что похож на то, как заблудившийся путешественник пытается ночью добраться до вершины горы. Даже если уже слишком темно, чтобы разглядеть что-то вдали, он может попробовать достигнуть вершины горы, постоянно двигаясь вверх.

Конечно, существует вероятность, что путник остановится на вершине меньшего холма и не достигнет пика. Эта проблема существует и при использовании данного эвристического метода. Алгоритм может найти решение, которое будет локально-оптимальным, но не будет лучшим возможным решением.

В задаче формирования портфеля инвестиций цель состоит в том, чтобы выбрать набор позиций с общей стоимостью не более допустимого предела, а общая прибыль при этом должна быть как можно больше. Эвристика восхождения на холм для этой задачи выбирает позицию, которая даёт максимальную прибыль на каждом шаге. При этом решение будет всё лучше соответствовать цели – получению максимальной прибыли.

Алгоритм сначала добавляет к решению позицию с максимальной прибылью, затем добавляется следующая позиция с максимальной прибылью (при условии, что полная цена остаётся в допустимых пределах). Присоединение позиций с максимальной прибылью будет продолжаться, пока не будет исчерпан лимит стоимости.

Для списка инвестиций, приведённого ниже, программа сначала выберет сделку A, потому что она имеет самую большую прибыль (9 млн. долларов). Затем выбирается сделка C, потому что она имеет самую большую прибыль из оставшихся (8 млн. долларов). В этот момент времени из допустимых 100 млн. потрачено уже 93 млн., и алгоритм больше не сможет выбирать какие-либо

сделки. Решение, вычисленное с помощью этой эвристики, включает элементы A и C, стоит 93 млн. и даёт прибыль в 17 млн.

Таблица 1

Инвестиция	Стоимость (млн.)	Прибыль (млн.)
A	63	9
B	35	7
C	30	8
D	27	7
E	23	3

Эвристика восхождения на холм заполняет портфель очень быстро. Если элементы изначально упорядочены в порядке уменьшения прибыли, то сложность алгоритма составляет порядка $O(N)$. Программа просто перемещается по списку, добавляя позиции с максимальной прибылью, пока не будет исчерпан лимит средств. Если список не отсортирован, то сложность этого алгоритма составляет $N * \log(N)$. Это много лучше, чем $O(2^N)$ шагов, необходимых для полного перебора всех узлов дерева. Для 20 позиций эта эвристика использует около 400 шагов, метод ветвей и границ – несколько тысяч, а полный перебор – более чем 2 млн.

Метод наименьшей стоимости

Стратегия, которая в некотором смысле является противоположностью методу восхождения на холм, называется методом наименьшей стоимости. Вместо того, чтобы на каждом шаге приближать решение максимально близко к цели, можно попробовать уменьшить стоимость решения. В примере с формированием портфеля инвестиций на каждом шаге к решению добавляется позиция с минимальной стоимостью.

Данная стратегия будет помещать в решение максимально возможное число позиций. Это хорошо работает в случае, если все позиции имеют примерно одинаковую стоимость. Но если дорогая сделка приносит большую прибыль, эта стратегия может пропустить выпавший шанс, давая не лучший из возможных результатов.

Для инвестиций, приведённых выше, стратегия минимальной стоимости начинает с того, что сначала добавляет к решению сделку E стоимостью 23 млн. Затем она выбирает позицию D стоимостью 27 млн. и C стоимостью 30 млн. В этой точке алгоритм уже потратил 80 млн. из 100 млн. долларов и не может больше сделать ни одного вложения.

Полученное решение стоит 80 млн. и даёт прибыль 18 млн. Это на миллион лучше, чем решение, которое даёт нам эвристика восхождения на холм, но алгоритм минимальной стоимости далеко не всегда работает эффективнее, чем алгоритм восхождения на холм. Какой из методов даст лучшее решение зависит от конкретных данных.

Структура программ, реализующих эвристики минимальной стоимости и эвристики восхождения на холм, почти идентична. Единственная разница заключается в выборе следующей позиции, которая добавляется к имеющемуся решению. Метод минимальной стоимости вместо позиции с максимальной прибылью выбирает позицию, которая имеет самую низкую стоимость. Поскольку эти два метода очень похожи, сложность их одинаковы. Если позиции должным образом отсортированы, оба алгоритма имеют сложность порядка $O(N)$. При случайном расположении позиций их сложность составит порядка $O(N * \log(N))$.

Сбалансированная прибыль

Стратегия восхождения на холм не учитывает стоимости добавляемых позиций. Она выбирает позиции с максимальной прибылью, даже если они имеют большую стоимость. Стратегия минимальной стоимости не берёт в расчёт приносимую позицией прибыль. Она выбирает элементы с небольшими затратами, даже если они имеют маленькую прибыль.

Эвристика сбалансированной прибыли сравнивает как прибыль, так и стоимость позиций, чтобы определить, какие позиции необходимо выбрать. На каждом шаге эвристика выбирает элемент с самым большим отношением прибыли к стоимости (при условии, что после включения инвестиции в портфель, суммарная цена останется в допустимых пределах).

Включим в таблицу новый столбец – отношение $\frac{\text{прибыль}}{\text{стоимость}}$. При таком подходе сначала выбирается позиция C, потому что она имеет самое высокое отношение – 0.27. Затем добавляется D с отношением 0.26 и B с отношением 0.20. В этой точке потрачено 92 млн. из 100 млн., и в решение нельзя добавить ни одной позиции.

Таблица 2

Инвестиция	Стоимость (млн.)	Прибыль (млн.)	Прибыль / Стоимость
A	63	9	0.14
B	35	7	0.20
C	30	8	0.27
D	27	7	0.26
E	23	3	0.13

Это решение имеет стоимость 92 млн. и даёт прибыль в 22 млн. Это на 4 млн. лучше, чем решение, найденное с помощью метода минимальной стоимости и на 5 млн. лучше, чем решение найденное эвристикой восхождения на холм. Более того, найденное решение вообще будет наилучшим из всех возможных, что подтвердят результаты поиска полным перебором или методом ветвей и границ. Но важно понимать, что сбалансированная прибыль – это всё-таки эвристика, поэтому с помощью этого не всегда отыскивается лучшее возможное решение. Очень часто этот метод находит лучшие решения, чем решения, найденные методами восхождения на холм и минимальной стоимости, но это случается не всегда. Структура программы, реализующей эвристику сбалансированной прибыли, почти идентична структуре программ восхождения на холм и минимальной стоимости. Единственная разница заключается в способе выбора следующей позиции, которая добавляется к решению. Сложность этой эвристики пропорциональна $O(N)$, при условии предварительной сортировки. В случае, если позиции расположены произвольно, сложность алгоритма составит $O(N * \log(N))$.

Случайные методы

Случайный поиск

Случайный поиск выполняется в соответствии со своим названием. На каждом шаге алгоритм добавляет случайно выбранную позицию, которая удовлетворяет границам стоимости. Этот вид перебора называется методом Монте-Карло.

Поскольку случайно выбранное решение вряд ли окажется наилучшим, то для получения приемлемого результата необходимо повторить поиск несколько раз. Хотя на первый взгляд кажется, что вероятность нахождения хорошего решения очень мала, использование этого метода

иногда приносит удивительно хорошие результаты. В зависимости от исходных данных и числа проверенных случайных решений, эта эвристика часто работает лучше, чем методы восхождения на холм и минимальной стоимости.

Преимущество случайного поиска состоит в том, что этот метод лёгок для понимания и реализации. Иногда трудно представить, как реализовать для конкретной задачи метод восхождения на холм, минимальной стоимости или приведённой прибыли, но всегда легко генерировать решения наугад. Даже для решения крайне сложных задач случайный поиск является наиболее простым методом.

Последовательное приближение

Другая стратегия состоит в том, чтобы начать со случайного решения, а затем производить последовательное приближение. Начав с произвольно сгенерированного решения, программа делает случайный выбор. Если новое решение является улучшением предыдущего, программа закрепляет изменение и продолжает проверку других позиций. Если изменение не улучшает решение, программа отказывается от него и делает новую попытку.

Особенно просто реализовать метод последовательного приближения для задачи формирования портфеля инвестиций. Программа всего-навсего выбирает случайную позицию из пробного решения и удаляет её из текущего. Затем она случайным образом добавляет в решение позиции до тех пор, пока не будет исчерпан лимит средств. Если удалённая позиция имела очень высокую стоимость, то не её место программа может добавить несколько позиций.

Как и случайный поиск, эта эвристика проста для понимания и реализации. Для решения сложной задачи бывает нелегко создать алгоритмы восхождения на холм, минимальной стоимости и приведённой прибыли, но довольно просто написать эвристический алгоритм последовательного приближения.

Момент остановки

Существует несколько хороших способов определить момент, когда необходимо прекратить случайные изменения. Например, допускается выполнить фиксированное число изменений. Для задачи из N элементов можно выполнить N или N^2 случайных изменений и затем остановить выполнение программы.

Другая стратегия состоит в том, чтобы делать изменения до тех пор, пока последовательных изменения будут приносить улучшения. Как только несколько подряд идущих изменений не дают улучшений, программу можно остановить.

Локальный оптимум

Если программа заменяет случайно выбранную позицию в пробном решении, она может найти решение, которое уже нельзя улучшать, но оно всё же не будет лучшим возможным решением. В качестве примера рассмотрим следующий набор возможных инвестиций.

Таблица 3

Инвестиция	Стоимость (млн.)	Прибыль (млн.)
A	47	9
B	43	8
C	35	5
D	32	7

E	31	6

Предположим, что алгоритм случайно выбирает позиции A и B в качестве начального решения. Его стоимость будет равна 90 млн., оно принесёт прибыль в 17 млн.

Если программа удаляет или A, или B, то решение будет иметь достаточно большую стоимость, поэтому программа сможет добавить только одну новую позицию. Поскольку позиции A и B имеют самую большую прибыль, замена их другой позицией уменьшит полную прибыль. Случайное удаление одной позиции из этого решения никогда не приведёт к улучшению.

Лучшее решение содержит позиции C, D, E. Его полная стоимость равна 98 млн., полная прибыль – 18 млн. Чтобы найти это решение, алгоритм должен удалить из решения сразу обе позиции A и B и добавить на их место новые.

Такие решения, когда небольшие изменения не могут улучшить решения, называются локальным оптимумом. Есть два способа, при применении которых, программа не остановится в локальном оптимуме, а будет искать глобальный оптимум.

Во-первых, программу можно изменить так, чтобы она удаляла из решения несколько позиций. Если программа удалит две случайно выбранные позиции, она сможет найти правильное решение для данного примера. Однако для задач большего размера удалить две позиции обычно недостаточно. Программе надо будет удалить три, четыре, а может большее количество позиций.

Более простой способ состоит в том, чтобы выполнить большее количество испытаний с различными исходными решениями. Некоторые из начальных решений могут привести к локальным оптимумам, но одно из них позволит достичь глобального оптимума.

Метод отжига

Метод отжига заимствован из термодинамики. При отжиге металл нагревается до высокой температуры. Молекулы в горячем металле совершают быстрые колебания. Если металл медленно охлаждать, то молекулы начинают выстраиваться в линии, образуя кристаллы. При этом молекулы постепенно переходят в состояние с минимальной энергией.

Когда металл остывает, соседние кристаллы сливаются друг с другом. Молекулы одного кристалла временно покидают свои позиции с минимальной энергией и соединяются с молекулами другого кристалла. Энергия получившегося кристалла большего размера будет меньше, чем сумма энергий двух исходных кристаллов. Если металл охлаждать достаточно медленно, кристаллы станут просто огромными. Конечное расположение молекул будет иметь очень низкую суммарную энергию, поэтому металл будет очень прочным.

Начиная с состояния с высокой энергией, молекулы в конечном счёте достигают состояния с низкой энергией. На пути к окончательному положению они проходят через множество локальных минимумов энергии. Каждая комбинация кристаллов представляет локальный минимум. Довести кристалл до минимального энергетического состояния можно только временным разрешением структуры меньших кристаллов, увеличивая тем самым энергию системы, в результате чего кристаллы могут объединяться.

Метод отжига использует аналогичный способ поиска лучшего решения задачи. Когда программа ищет путь решения, она может «застрять» в локальном оптимуме. Чтобы избежать этого, она время от времени вносит в решение случайные изменения, даже если очередной вариант не

приводит к мгновенному улучшению результата. Это позволяет программе выйти из локального оптимума и отыскать лучшее решение.

Чтобы программа не зацикливалась на этих модификациях, алгоритм через какое-то время изменяет вероятность внесения случайных изменений. Вероятность внесения одного изменения равна $P = \frac{1}{e^{\frac{E}{k*T}}}$, где E - количество «энергии», добавленной системе, k - константа, выбранная в зависимости от рода задачи и T – переменная, соответствующая «температуре».

Сначала величина T должна быть довольно высокой, поэтому вероятность изменений тоже будет довольно высокой. Через какое-то время значение величины T снижается, и вероятность случайных изменений тоже уменьшается. Как только процесс достиг точки, в которой никакие изменения не смогут улучшить решение и значение T станет настолько мало, что случайные изменения будут очень редкими, алгоритм закончит работу.

Для задачи формирования портфеля инвестиций E - величина, на которую сокращается прибыль в результате изменения. Например, если мы удаляем позицию, прибыль которой равна 10 млн. долларов, и заменяем её позицией, имеющей прибыль в 7 млн. долларов, добавленная к системе энергия будет равна 3. Если величина E велика, то вероятность изменений небольшая, поэтому вероятность больших изменений ниже.

Сравнение эвристических методов

Различные эвристические методы ведут себя по-разному в различных задачах. Для решения задачи о формировании портфеля инвестиций эвристика сбалансированной прибыли достаточно хороша, учитывая её простоту. Стратегия последовательного приближения тоже работает достаточно хорошо, но требует гораздо большего времени. Для других задач наилучшей может быть какая-нибудь другая эвристика, в том числе и не рассмотренная в этой главе.

Эвристики работают гораздо быстрее, чем методы полного перебора и ветвей и границ. Некоторые эвристические подходы (восхождение на холм, минимальная стоимость, сбалансированная прибыль и т.д.) работают чрезвычайно быстро, потому что рассматривают только одно возможное решение. Эти методы работают настолько быстро, что порой имеет смысл выполнить их все по очереди, а затем выбрать наилучшее из трёх полученных решений. Конечно, невозможно гарантировать, что это решение будет наилучшем, но оно точно будет достаточно хорошим.