Деревья решений, метод ветвей и границ

Деревья используются для решения многих задач. Это и задачи поиска, и задачи разбора выражений, и задачи получения кодов с заданными свойствами. В этой лекции мы рассмотрим использование деревьев в играх и задачах оптимизации.

Игровые деревья

Начнём с так называемых деревьев решений. Многие реальные задачи можно смоделировать с помощью таких деревьев. Каждый узел будет представлять собой один шаг решения задачи, ветвь в дереве соответствует решению, которое ведёт к более полному решению, листы представляют собой окончательное решение. Наша цель – найти в дереве лучший путь от корня до листа.

Деревья решений обычно невероятно велики. Дерево для простой игры в крестики-нолики содержит более полумиллиона узлов, а реальные задачи на порядок сложнее. Но, тем не менее, на примере детской игры можно рассмотреть методы, позволяющие находить оптимальные решения в очень больших деревьях.

Теоретически любую настольную стратегическую игру можно (шашки, шахматы, крестики-нолики) можно смоделировать с помощью игровых деревьев. Каждая ветвь, выходящая из узла, соответствует ходам игроков. Если у игрока n возможных ходов, то соответствующий узел будет иметь n ветвей (потомков).

Например, в крестиках-ноликах корневой узел соответствует пустому полю. Первый игрок может поставить крестик на любой из 9 квадратиков. Каждому возможному ходу будет соответствовать ветвь, исходящая из корневого узла. Когда игрок поставил крестик, второй игрок может поставить нолик в любом из оставшихся восьми квадратов. Каждому из этих ходов соответствует ветвь, исходящая из узла, представляющего текущую позицию крестика на доске. На рисунке показан небольшой фрагмент исследуемого дерева.

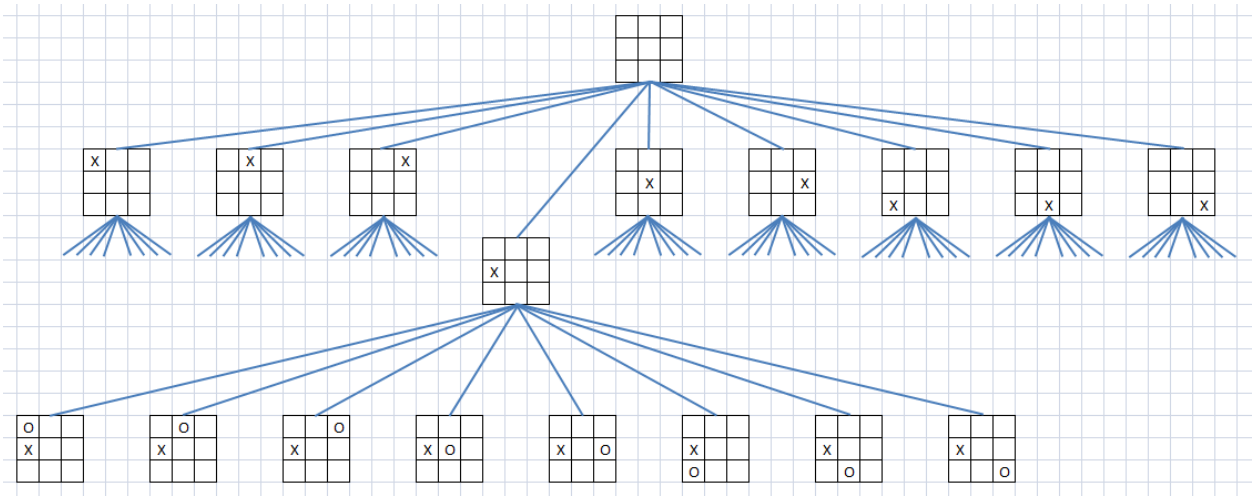

Рисунок 1

Как можно видеть из рисунка, дерево игры растёт чрезвычайно быстро. Если дерево продолжит расти таким образом, то во всём дереве будет содержаться 9!=362880 листьев.

На самом деле многие возможные узлы в дереве игры запрещены правилами. Например, если один из игроков уже выиграл, то построение дерева на этом этапе можно завершить. Удаление всех невозможных узлов сократит дерево до четверти миллиона листов. Но это всё ещё очень и очень большое дерево и полный его обход займёт непозволительно много времени. Для более сложных игр деревья имеют просто невообразимый размер. Если бы в шахматах на каждом шаге у игрока было бы 16 возможных ходов, то в дереве было бы более триллиона узлов после пяти ходов.

Чтобы выполнить поиск в игровом дереве, нужно иметь возможность определить значение позиции игрового поля. В крестиках-ноликах для первого игрока большое значение имеют позиции, в которых три крестика расположены в ряд, так как при этом первый игрок выигрывает. Значение игрока, который ставит нолик, в этих позициях поля очень мало, потому что при этом он проигрывает.

Каждому игроку можно назначить четыре значения для конкретной позиции на поле:

- 4 – выигрыш
- 3 – не ясная ситуация
- 2 – ничья
- 1 – проигрыш

Для исчерпывающего исследования игрового дерева можно использовать стратегию минимакса. При этом мы будем пытаться минимизировать максимальное значение, которое может иметь позиция для противника после следующего хода. Сначала определяется максимальное значение, которое может набрать противник после каждого из ваших возможных ходов. Затем выбирается ход, при котором противник получает минимальное значение.

Процедура, поиска оптимального хода, вычисляет значение позиции поля. Эта процедура исследует каждый возможный ход. Для каждого хода она рекурсивно вызывает себя, чтобы определить значение которое будет иметь позиция противника. Затем она выбирает ход, который даёт противнику минимальное значение. Рекурсивное обращение процедуры к себе будет продолжаться, пока не наступит одно из трёх возможных событий. Во-первых, может быть найдена позиция, в которой игрок побеждает. В этом случае процедура устанавливает значение позиции поля в 4. В случае если игрок проигрывает, то значение поля будет установлено в 1. Во-вторых, может быть найдена позиция, в которой ни один игрок не может сделать ход. Игра заканчивается ничьей, поэтому процедура устанавливает значение позиции в 2. И, наконец, процедура может достичь заранее установленной глубины рекурсии. Если она превышает допустимую глубину, то значение поля устанавливается в 3, указывая на неопределённый исход. Максимальная глубина рекурсии предохраняет программу от слишком большой траты времени. Это особенно важно для сложных игр, вроде шахмат, где поиск может продолжаться практически бесконечно долго. Максимальная глубина рекурсии также позволяет задавать уровень мастерства. Чем глубже программа может исследовать дерево, тем лучше будут её ходы.

На рисунке показано дерево игры в конце партии. Первый и третий ходы приводят к победе игрока, играющего ноликами, поэтому эти ходы имеют для него значение 4. Второй возможный ход приводит к ничьей и имеет значение 2. Программа выбирает второй ход, потому что он даёт игроку ноликов самое маленькое значение поля.

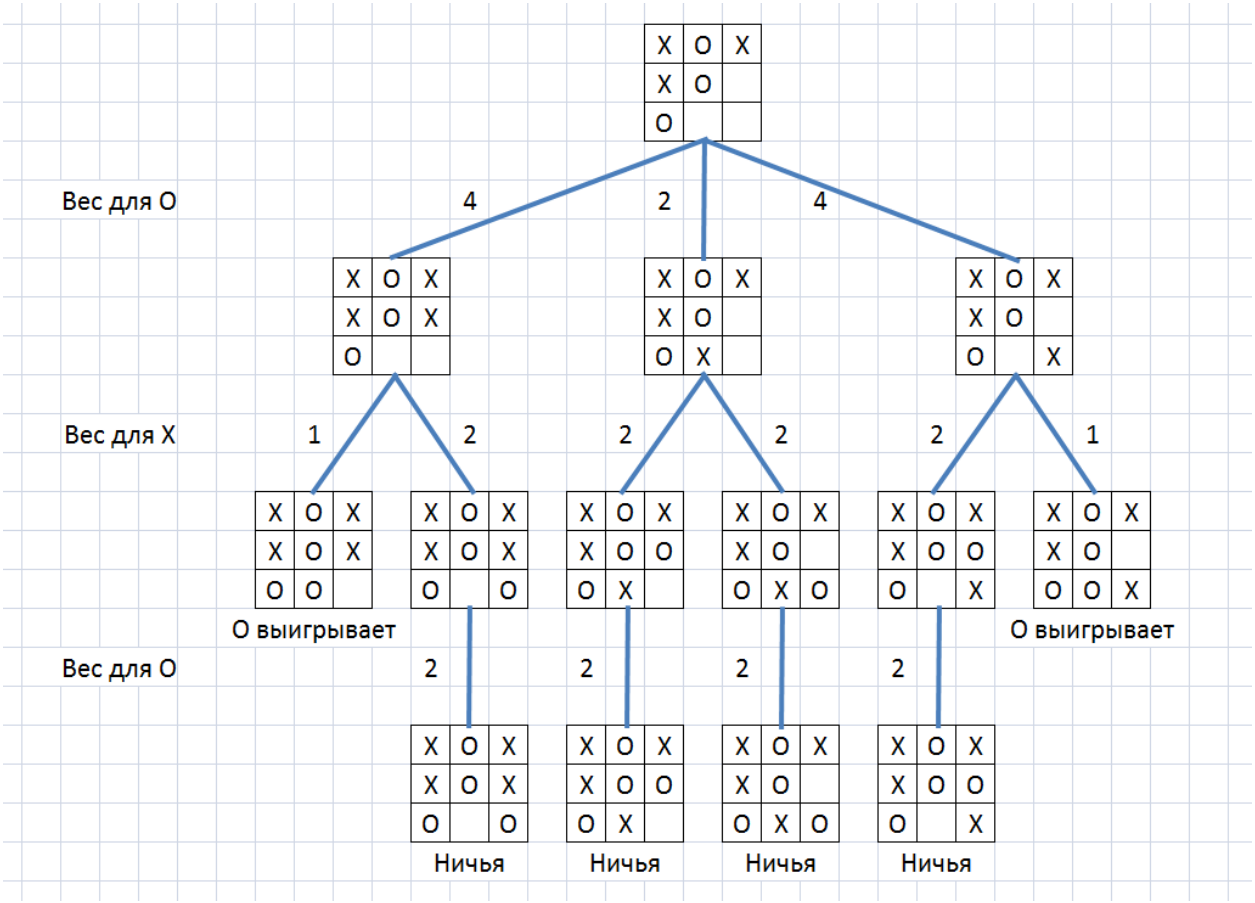

Рисунок 2

Если бы минимаксный алгоритм был бы единственным инструментом для исследования игровых деревьев, то перебор больших деревьев был бы довольно сложным. К счастью существует несколько приёмов, которые можно использовать для поиска в больших деревьях.

Во-первых, можно запомнить несколько первых ходов, выбранных экспертами игры. Если выбрать первый ход за программу, то дерево поиска будет сокращено в 9 раз. Фактически, программа не должна перебирать дерево, пока противник не сделал ход. В этом случае и компьютер, и его противник уже выбрали ветви, поэтому дерево будет содержать всего 7!=5040 ветвей.

Использование дополнительной памяти для хранения нескольких ходов значительно сокращает время, необходимое для поиска в дереве.

Другой способ улучшить поиск – указать важные шаблоны. В шахматах такими шаблонами могут быть позиции, угрожающие нескольким фигурам сразу, королю или ферзю. В случае если шахматная программа замечает шаблон обмена, она может временно изменить глубину рекурсии, чтобы просмотреть серию обменов до конца и решить будет ли обмен выгодным. Если обмен всё же происходит, количество оставшихся фигур становится меньше и поиск в дереве упрощается.

И, наконец, в более сложных случаях необходимо использовать эвристики. Например, в шахматах обычной эвристикой является усиление преимущества. Когда у противника меньше сильных фигур и одинаковое с вами число слабых фигур, то следует идти на обмен при любой возможности.

Метод ветвей и границ

Метод ветвей и границ является одним из методов упрощения деревьев решений таким образом, чтобы не рассматривать все ветви дерева. Общая идея состоит в том, чтобы отслеживать границы уже обнаруженных и возможных решений. Если вы достигаете точки, где лучшее решение на данный момент эффективнее, чем возможное решение в нижних ветвях, вы можете проигнорировать все пути ниже данного узла.

Например, у нас есть 100 млн. долларов, которые нужно вложить в несколько возможных инвестиций. Каждое из этих вложений имеет различную стоимость и различный ожидаемый доход. Мы должны решить, как потратить деньги, чтобы получить максимальную прибыль.

Задачи такого типа называются задачами формирования портфеля. У нас есть несколько позиций (инвестиций), которые должны поместиться в портфель фиксированного размера (100 млн. долларов). Каждая позиция имеет свою прибыльность. Необходимо найти набор позиций, которые помещаются в портфель и дают максимальную прибыль.

Эту задачу можно смоделировать в виде дерева решений. Каждый узел в дереве соответствует определённым комбинациям инвестиций, помещённых в портфель. Каждая ветвь – это принятое решение о том, поместить элемент в портфель или извлечь его оттуда. Левая ветвь в первом узле соответствует расходу денег на первое вложение. Правая ветвь представляет отказ в первом вложении. На рисунке показано дерево решений для четырёх возможных вложений.

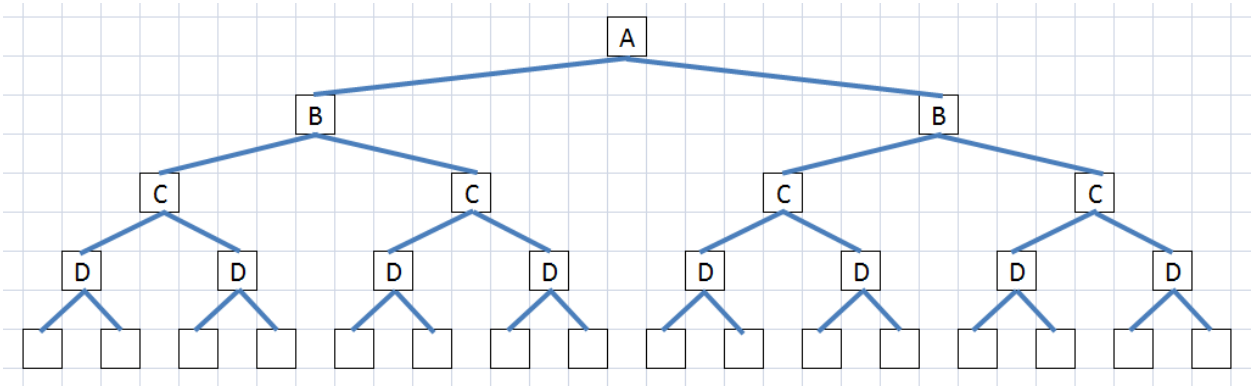

Рисунок 3

Дерево решений для этой задачи представляет собой полное двоичное дерево, глубина которого равна числу инвестиций. Легко показать, что это дерево очень быстро растёт с ростом числа возможных инвестиций. Если для 10 инвестиций в дереве будет всего $2^{10} = 1024$ листа, то для 20 инвестиций, дерево будет иметь более миллиона листьев. Полный поиск по такому дереву ещё возможен, но при дальнейшем росте числа инвестиций размер дерева станет слишком большим.

Чтобы использовать метод ветвей и границ необходимо создать массив, который будет отслеживать позиции наилучшего найденного до сих пор решения. При старте программы этот массив должен быть пуст. Для хранения значения наилучшего решения необходимо использовать вспомогательную переменную. Изначально значение этой переменной должно быть равно 0, чтобы первое же найденное решение было лучше исходного.

Если во время поиска мы вдруг достигнем точки, где рассматриваемое решение не может быть достаточно хорошим, чтобы конкурировать с текущим лучшим решением, то можно прекратить

исследование этого пути. То же самое относится и к ситуации, когда в выбранной точке позиции уже не вмещаются в портфель.

В качестве конкретного примера предположим, что вы можете вложить деньги в любую из инвестиций, приведённых в таблице.

Таблица 1

Позиция	Стоимость (млн.)	Прибыль (млн.)
A	45	10
B	52	13
C	46	8
D	35	4

Приведённое выше дерево прекрасно моделирует текущую задачу. Некоторые из возможных решений просто не укладываются в портфель. Например, крайний левый путь предполагает затраты равные 178 млн.

Предположим, что мы начали перебор дерева. Мы увидели, что можно потратить 97 млн. на сделках A и B при прибыли в 23 млн. Это соответствует 4 слева листу.

Продолжив поиск, можно дойти до второго узла, обозначенного как C. Это соответствует инвестиционным пакетам, которые включают сделку A, не включают сделку B, и могут включать или не включать сделки C и D. В этой точке пакет уже стоит 45 млн. для сделки A и даёт прибыль 10 млн. долларов.

Единственные оставшиеся сделки это C и D. Вместе они могут улучшить решение на 12 млн. Значение текущего решения равно 10 млн., поэтому лучшее возможное решение ниже этого узла стоит почти 22 млн. Это меньше уже найденного решения на 23 млн., поэтому не следует продолжать рассматривать этот путь.

По мере продвижения программы по дереву ей не нужно постоянно проверять, является ли частное рассматриваемое решение лучше, чем наилучшее найденное до сих пор. Если частное решение лучше, то лучше будет и самый правый узел внизу от него. Этот узел представляет собой ту же самую комбинацию позиций, что и частное решение, поскольку все остальные позиции в данном случае исключены. Следовательно, программе необходимо искать лучшее решение только тогда, когда она достигает листа.

Фактически, любой лист, которого достигает программа, всегда является улучшенным решением. Если бы это было не так, то ветвь, на которой находится лист, была бы отсечена, когда программа рассматривала родительский узел. В этой точке перемещение к листу уменьшит цену невыбранных позиций до нуля. Если значение решения не больше, чем лучшее решение на данный момент, проверка нижней границы остановит продвижение программы к листу. Используя этот факт, программа может модифицировать лучшее решение, когда достигает листа.

Перебор методом ветвей и границ исследует гораздо меньше узлов, чем полный перебор. Дерево решений для задачи о формировании портфеля с 20 элементами содержит 2097151 узел. В то время, как полный перебор всегда исследует все листы, метод ветвей и границ может перебрать примерно 1600 элементов.

Число исследуемых методом ветвей и границ узлов зависит от точных значений данных. Если стоимость элемента большая, то в правильном решении окажется немного элементов. Как только

эти элементы добавляются в исследуемое решение, оставшиеся элементы уже не вписываются в статью расходов, поэтому большая часть дерева будет отрезана.

С другой стороны, если элементы имеют низкую стоимость, многие из них смогут поместиться в правильном решении, поэтому программа должна исследовать множество допустимых комбинаций. В таблице приведены сравнительные оценки количества исследованных узлов методами полного перебора и методом ветвей и границ. В тесте использовались следующие параметры: количество инвестиций – 20, размер портфеля – 100 млн.

Таблица 2

Средняя стоимость элемента	**Полный перебор**	**Метод ветвей и границ**
60	2097151	203
50	2097151	520
40	2097151	1322
30	2097151	4269
20	2097151	13286
10	2097151	40589