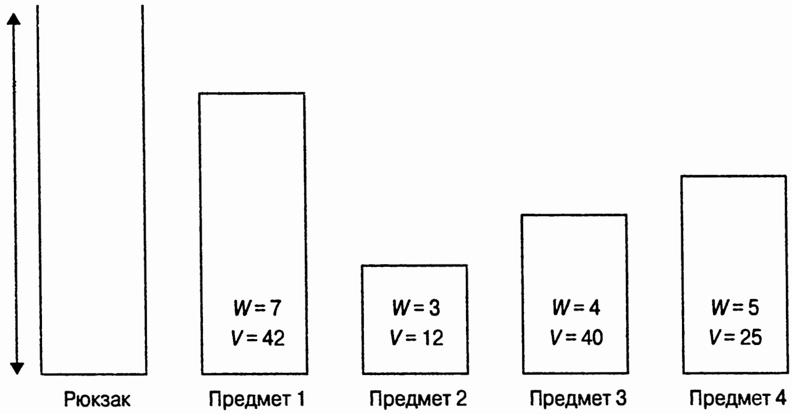
Задача о рюкзаке

Вот еще одна известная алгоритмическая задача. Дано n предметов весом $w_1, \ldots, w_n$ и ценой $v_1, \ldots, v_n$, а также рюкзак, выдерживающий вес W. Требуется найти подмножество предметов, которое можно разместить в рюкзаке, и которое имеет при этом максимальную стоимость. Если вам не нравится представлять себя вором, пытающимся запихнуть в свой рюкзак самые ценные вещи, представьте

себя транспортным чиновником, которому надо перевезти максимально ценный груз, не превышающий определенного веса. На рис. 3.8*а* приведен небольшой экземпляр задачи о рюкзаке.

Исчерпывающий перебор в этой задаче приводит к рассмотрению всех подмножеств данного множества из n предметов, вычислению общего веса каждого из них для того, чтобы выяснить, допустим ли такой набор предметов (т.е. не превосходит ли его общий вес возможности рюкзака), и выбору из допустимых подмножества с максимальным весом. В качестве примера решение экземпляра задачи, представленного на рис. 3.8*а*, показано на рис. 3.8*б*. Поскольку общее количество подмножеств n-элементного множества равно 2^n, исчерпывающий перебор приводит к алгоритму со временем работы $\Omega\left(2^n\right)$, вне зависимости от того, насколько эффективным методом генерируются рассматриваемые подмножества.

Таким образом, применение метода исчерпывающего перебора к задачам коммивояжера и о рюкзаке приводит к исключительно неэффективным алгоритмам для любых входных данных. На самом деле эти две задачи представляют собой наиболее известные примеры так называемых ***NP*-сложных задач** (NP-hard problems). Ни для одной из NP-сложных задач не известен алгоритм, решающий их за полиномиальное время. Более того, большинство ученых-кибернетиков сходятся



а)

Подмножество	Общий вес	Общая стоимость
$\varnothing$	0	0
$\{1\}$	7	42
$\{2\}$	3	12
$\{3\}$	4	40
$\{4\}$	5	25
$\{1,2\}$	10	36
$\{1,3\}$	11	Недопустим
$\{1,4\}$	12	Недопустим
$\{2,3\}$	7	52
$\{2,4\}$	8	37
$\mathbf{\{3,4\}}$	**9**	**65**
$\{1,2,3\}$	14	Недопустим
$\{1,2,4\}$	15	Недопустим
$\{1,3,4\}$	16	Недопустим
$\{2,3,4\}$	12	Недопустим
$\{1,2,3,4\}$	19	Недопустим

б)

Рис. 3.8. *а*) Экземпляр задачи о рюкзаке. *б*) Решение путем исчерпывающего перебора (оптимальное решение выделено полужирным шрифтом)

во мнении, что такие алгоритмы не существуют вообще, хотя это важное предположение никем не доказано. Более интеллектуальные подходы, рассматриваемые в разделах 11.2 и 11.2, позволяют решить некоторые (но не все) экземпляры этих (и подобных) задач за время, меньшее экспоненциального. Можно также воспользоваться одним из приближенных алгоритмов, подобных рассмотренным в разделе 11.3.

8.4 Задача о рюкзаке и функции с запоминанием

Этот раздел мы начнем с разработки алгоритма динамического программирования для решения задачи о рюкзаке: даны n предметов с известными весами $w_1, \ldots, w_n$ и стоимостями $v_1, \ldots, v_n$ и рюкзак вместимостью W. Требуется найти наиболее ценное подмножество предметов, помещающееся в рюкзаке. (Эта задача упоминалась в разделе 3.4, где мы рассматривали ее решение методом исчерпывающего перебора.) Здесь мы считаем, что все веса и емкость рюкзака представляют собой положительные целые числа; стоимости предметов — не обязательно целые числа.

Для разработки алгоритма динамического программирования мы должны вывести рекуррентное соотношение, которое выражает решение экземпляра задачи о рюкзаке через решения его меньших подэкземпляров. Рассмотрим экземпляр, определяемый первыми i предметами, $1 \leqslant i \leqslant n$, весами $w_1, \ldots, w_i$, стоимостями $v_1, \ldots, v_i$ и емкостью рюкзака $1 \leqslant j \leqslant W$. Пусть $V[i,j]$ — значение оптимального решения этого экземпляра, т.е. стоимость наиболее ценного подмножества из первых i предметов, которое помещается в рюкзак емкостью j. Мы можем разделить все подмножества первых i предметов, которые помещаются в рюкзак емкостью j, на две категории: те, которые не включают i-ый предмет, и те, которые его включают. Заметим следующее.

1. Среди подмножеств, которые не включают i-ый предмет, стоимость оптимального подмножества по определению равна $V[i-1,j]$.
2. Среди подмножеств, которые включают i-ый предмет (следовательно, $j-w_i \geqslant 0$), оптимальное подмножество составляется из этого предмета и оптимального подмножества первых $i-1$ предметов, которое размещается в рюкзаке емкостью $j - w_i$. Стоимость такого оптимального подмножества равна $v_i + V[i-1, j-w_i]$.

Таким образом, стоимость оптимального решения среди всех допустимых подмножеств из первых i предметов представляет собой большее из этих двух значений. Конечно, если i-ый предмет не помещается в рюкзак, стоимость оптимального подмножества, выбранного из первых i предметов, оказывается той же, что и стоимость оптимального подмножества, выбранного из первых $i-1$ предметов. Это наблюдение приводит нас к следующему рекуррентному соотношению:

$$V[i,j] = \begin{cases} \max\{V[i-1,j], v_i + V[i-1,j-w_i]\} & \text{если } j - w_i \geqslant 0, \\ V[i-1,j] & \text{если } j - w_i < 0. \end{cases} \qquad (8.12)$$

Начальные условия удобно определить следующим образом:

$$V[0,j] = 0 \text{ при } j \geqslant 0, \text{ и } V[i,0] = 0 \text{ при} i \geqslant 0. \qquad (8.13)$$

Наша цель состоит в том, чтобы найти $V[n,W]$ — максимальную стоимость подмножества из n предметов, которое помещается в рюкзаке емкостью W, и само это подмножество.

На рис. 8.12 показаны значения, входящие в (8.12) и (8.13). При $i,j > 0$ для вычисления элемента таблицы на пересечении i-ой строки и j-го столбца $V[i,j]$ мы берем значение элемента в предыдущей строке и том же столбце и сумму значений v_i и элемента в предыдущей строке и столбце, отстоящем на w_i столбцов слева, и находим максимальное из них. Таким образом мы заполняем таблицу либо строка за строкой, либо столбец за столбцом.

		0	$j-w_i$	j	W
	0	0	0	0	0
w_i, v_i	$i-1$	0	$V[i-1, j-w_i]$	$V[i-1, j]$	
	i	0		$V[i, j]$	
	n	0			Целевое значение

Рис. 8.12. Таблица для решения задачи о рюкзаке методом динамического программирования

Пример 1. Рассмотрим экземпляр задачи, определяемый следующими данными. Емкость рюкзака $W = 5$.

Предмет	Вес	Стоимость
1	2	12
2	1	10
3	3	20
4	2	15

Таблица динамического программирования после заполнения в соответствии с формулами (8.12) и (8.13) показана на рис. 8.13.

		Емкость j					
	i	0	1	2	3	4	5
	0	0	0	0	0	0	0
$w_1 = 2, \quad v_1 = 12$	1	0	0	12	12	12	12
$w_2 = 1, \quad v_2 = 10$	2	0	10	12	22	22	22
$w_3 = 3, \quad v_3 = 20$	3	0	10	12	22	30	32
$w_4 = 2, \quad v_4 = 15$	4	0	10	15	25	30	37

Рис. 8.13. Пример решения экземпляра задачи о рюкзаке при помощи алгоритма динамического программирования

Итак, максимальная стоимость $V[4,5] = 37$. Мы можем найти состав оптимального подмножества, отслеживая вычисления этого элемента таблицы. Поскольку $V[4,5] \neq V[3,5]$, предмет 4 был включен в оптимальное решение вместе с оптимальным подмножеством, заполняющим оставшиеся $5-2=3$ единицы емкости рюкзака. Последние представлены элементом $V[3,3]$. Поскольку $V[3,3] = V[2,3]$, элемент 3 не является частью оптимального подмножества. Далее, так как $V[2,3] \neq V[1,3]$, предмет 2 также является частью оптимального выбора, после чего элемент $V[1, 3-1]$ остается в качестве определения оставшейся части подмножества. Аналогично, так как $V[1,2] \neq V[0,2]$, делаем вывод, что предмет 1 является последней частью оптимального решения, которое представляет собой множество {Предмет 1, Предмет 2, Предмет 4}. ■

Как временнáя, так и пространственная эффективность данного алгоритма равна $\Theta(nW)$. Время, требующееся для поиска состава оптимального подмножества, равно $\Theta(n + W)$. Эти утверждения читателю предлагается доказать само-
смощи рекурсивного вызова, и полученный результат записывается в таблице.

Задача о рюкзаке

Давайте применим метод ветвей и границ к решению задачи о рюкзаке. С этой задачей мы также познакомились в разделе 3.4: дано n предметов с весами $w_1, \ldots, w_n$ и ценами $v_1, \ldots, v_n$, а также рюкзак, выдерживающий вес W. Требуется найти подмножество предметов, которое можно разместить в рюкзаке и которое имеет при этом максимальную цену. Оказывается удобным упорядочить предметы в убывающем порядке по их удельной цене (отношению цены к весу), с разрешением неоднозначностей произвольным образом:

$$v_1/w_1 \geqslant v_2/w_2 \geqslant \cdots \geqslant v_n/w_n.$$

Естественной структурой дерева пространства состояний для данной задачи является бинарное дерево, построенное следующим образом (рис. 11.8). Каждый узел на уровне $0 \leqslant i \leqslant n$ представляет все подмножества из n элементов, которые включают определенный выбор из первых i упорядоченных элементов. Этот частичный выбор однозначно определяется путем от корня к узлу: ветвь, идущая влево, указывает на включение очередного элемента в подмножество, в то время как правая ветвь указывает на отсутствие элемента в подмножестве. Мы записываем общий вес w и общую стоимость v выбора, соответствующего узлу, вместе с верхней границей ub значения для любого подмножества, которое может быть получено путем добавления некоторых элементов (возможно, никаких) к этому выбору.

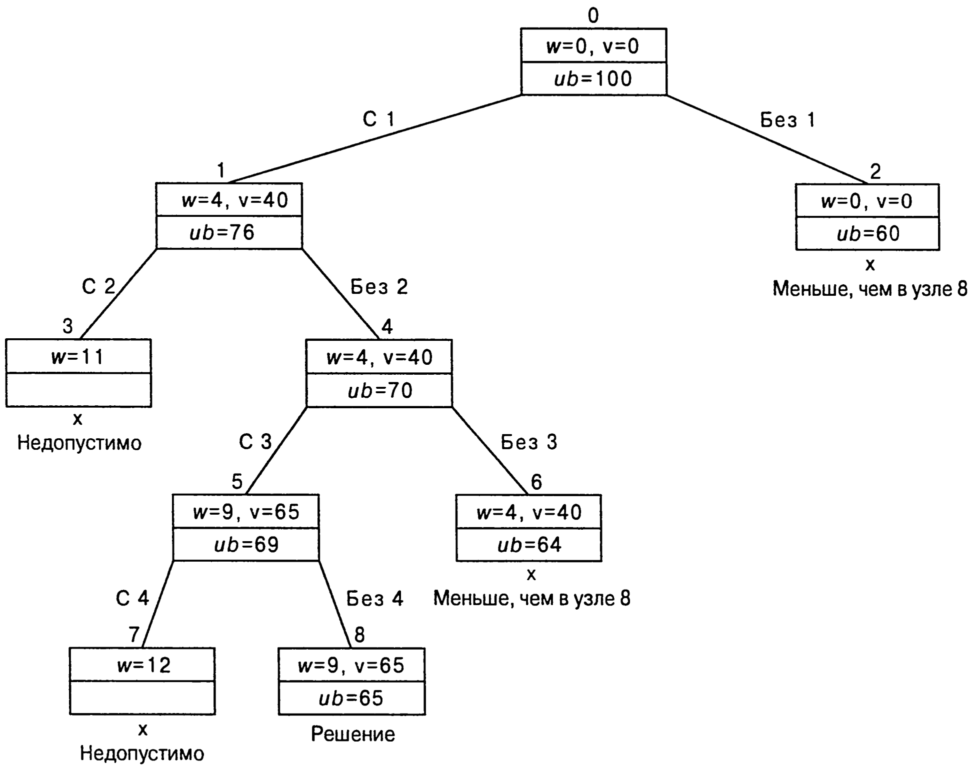


Рис. 11.8. Дерево пространства состояний алгоритма ветвей и границ для экземпляра задачи о рюкзаке

Простым способом вычисления верхней границы ub является добавление к общей стоимости уже выбранных элементов v произведения оставшейся емкости рюкзака $W - w$ и наибольшего значения удельной стоимости среди оставшихся элементов, которое равно v_{i+1}/w_{i+1}:

$$ub = v + (W - w)\left(v_{i+1}/w_{i+1}\right). \tag{11.1}$$

В качестве конкретного примера применим метод ветвей и границ к тому же экземпляру задачи о рюкзаке, который мы решали в разделе 3.4 методом исчерпывающего перебора (здесь мы переупорядочили элементы в порядке убывания их удельных стоимостей). Емкость рюкзака $W = 10$.

Предмет	Вес	Стоимость	Удельная стоимость
1	4	40	10
2	7	42	6
3	5	25	5
4	3	12	4

В корне дерева пространства состояний не выбран ни один элемент. Следовательно, как общий вес, так и общая стоимость выбранных элементов равны 0. Значение верхней границы, вычисленное по формуле (11.1), равно 100. Узел 1, левый дочерний узел корня, представляет подмножество, состоящее из одного предмета, 1; общий вес и стоимость в этом узле равны, соответственно, 4 и 40, а значение верхней границы — $40 + (10 - 4) \cdot 6 = 76$. Узел 2 представляет подмножество, которое не включает предмет 1, так что в этом узле $w = 0$, $v = 0$ и $ub = 0 + (10 - 0) \cdot 6 = 60$. Поскольку узел 1 имеет бóльшую верхнюю границу, чем узел 2, он является более обещающим для данной задачи максимизации, и мы начинаем ветвление с узла 1. Его дочерние узлы — 3 и 4 — представляют подмножества с элементом 1 и с и без элемента 2, соответственно. Поскольку общий вес любого подмножества, представленного узлом 3, превосходит емкость рюкзака, работа с этим узлом завершается. У узла 4 те же значения общего веса и общей стоимости, что и у родительского, так что значение верхней границы у этого узла $ub = 40+(10 - 4)\cdot 5 = 70$. Из узлов 2 и 4 для дальнейшего ветвления мы выбираем узел 4 (почему?) и получаем узлы 5 и 6, включающий и не включающий, соответственно, предмет 3. Общие вес и стоимость и значение верхней границы для этих узлов вычисляются точно так же, как и ранее. Ветвление из узла 5 дает узел 7, который дает недопустимое решение, и узел 8, представляющий подмножество $\{1, 3\}$. (Поскольку никаких дополнительных предметов нет, верхняя граница для узла 8 просто равна сумме стоимостей указанных предметов.) Оставшиеся живые узлы 2 и 6 имеют меньшие значения верхней границы, чем решение, представленное узлом 8. Следовательно, работа с этими узлами завершается, и множество $\{1, 3\}$ из узла 8 является оптимальным решением задачи.

Решение задачи о рюкзаке методом ветвей и границ имеет весьма необычные характеристики. Обычно внутренние узлы дерева пространства состояний не определяют точку пространства поиска задачи, поскольку некоторые из компонентов решения остаются неопределенными (см., например, дерево для задачи о назначениях, рассматривавшейся в предыдущем подразделе). В задаче же о рюкзаке каждый узел дерева представляет подмножество данных предметов. Этот факт можно использовать для обновления информации о наилучшем подмножестве после генерации каждого нового узла дерева. Для рассмотренного экземпляра это означает, что мы могли бы прекратить работу с узлами 2 и 6 еще до генерации узла 8, так как значения верхних границ рассматриваемых узлов меньше стоимости подмножества в узле 5, равной 65.

Приближенные алгоритмы для задачи о рюкзаке

Еще одна широко известная NP-сложная задача — задача о рюкзаке, с которой мы познакомились в разделе 3.4. Дано n предметов с весами $w_1, \dots, w_n$ и ценами $v_1, \dots, v_n$, а также рюкзак, выдерживающий вес W. Наша задача — найти подмножество предметов, которое можно разместить в рюкзаке и которое имеет при этом максимальную цену. Мы видели, как можно решить эту задачу методом исчерпывающего перебора (раздел 3.4), динамического программирования (раздел 8.4) и ветвей и границ (раздел 11.2). Теперь мы будем решать эту задачу при помощи приближенных алгоритмов.

Жадные алгоритмы для задачи о рюкзаке

Можно рассмотреть несколько жадных подходов к данной задаче. Один из них состоит в выборе предметов в убывающем порядке по их весам; беда в том, что более тяжелые предметы могут не быть наиболее ценными в множестве. Другой вариант состоит в выборе предметов в порядке уменьшения их стоимости, однако он не гарантирует эффективное использование емкости рюкзака. Можно ли найти жадную стратегию, которая бы принимала во внимание как вес, так и стоимость предметов? Да, можно: вычисляя удельную стоимость предметов v_i/w_i, $i = 1, 2, \dots, n$ и выбирая предметы в порядке уменьшения удельной стоимости (в действительности мы уже использовали этот подход при разработке алгоритма ветвей и границ для задачи о рюкзаке в разделе 11.2). Вот как выглядит алгоритм, основанный на этой жадной эвристике.

Шаг 1. Вычислим удельные стоимости всех предметов множества $r_i = v_i/w_i$, $i = 1, 2, \ldots, n$.

Шаг 2. Отсортируем предметы в невозрастающем порядке по их удельным стоимостям, вычисленным на шаге 1 (неоднозначности разрешаются произвольным образом).

Шаг 3. До тех пор пока в отсортированном списке не останется ни одного предмета, повторяем следующие действия: если текущий предмет помещается в рюкзак, мы помещаем его туда; в противном случае переходим к следующему предмету.

Пример 3. Давайте рассмотрим экземпляр задачи о рюкзаке емкостью 10 и следующей информацией о предметах:

Предмет	Вес	Стоимость
1	7	42
2	3	12
3	4	40
4	5	25

Вычислим удельные стоимости и отсортируем предметы в невозрастающем порядке их удельных стоимостей, что даст нам следующую таблицу:

Предмет	Вес	Стоимость	$\dfrac{\text{Стоимость}}{\text{Вес}}$
3	4	40	10
1	7	42	6
4	5	25	5
2	3	12	4

Жадный алгоритм выбирает предмет 3 с весом 4, пропускает предмет 1 с весом 7, затем выбирает предмет 4 с весом 5 и пропускает предмет 2 с весом 3. Полученное решение оказывается оптимальным для данного экземпляра задачи (см. раздел 11.2, где этот же экземпляр задачи решен методом ветвей и границ). ■

Может быть, жадный алгоритм всегда приводит к оптимальному решению? Конечно же, нет — если бы это было так, мы бы имели полиномиальный алгоритм для NP-сложной задачи. Приведенный далее пример показывает, что при помощи этого алгоритма нельзя получить гарантированную конечную верхнюю границу точности.

ветвей и границ для задачи о рюкзаке в разделе 11.2). Вот как выглядит алгоритм, основанный на этой жадной эвристике.

Шаг 1. Вычислим удельные стоимости всех предметов множества $r_i = v_i/w_i$, $i = 1, 2, \ldots, n$.

Шаг 2. Отсортируем предметы в невозрастающем порядке по их удельным стоимостям, вычисленным на шаге 1 (неоднозначности разрешаются произвольным образом).

Шаг 3. До тех пор пока в отсортированном списке не останется ни одного предмета, повторяем следующие действия: если текущий предмет помещается в рюкзак, мы помещаем его туда; в противном случае переходим к следующему предмету.

Пример 3. Давайте рассмотрим экземпляр задачи о рюкзаке емкостью 10 и следующей информацией о предметах:

Предмет	Вес	Стоимость
1	7	42
2	3	12
3	4	40
4	5	25

Вычислим удельные стоимости и отсортируем предметы в невозрастающем порядке их удельных стоимостей, что даст нам следующую таблицу:

Предмет	Вес	Стоимость	$\frac{\text{Стоимость}}{\text{Вес}}$
3	4	40	10
1	7	42	6
4	5	25	5
2	3	12	4

Жадный алгоритм выбирает предмет 3 с весом 4, пропускает предмет 1 с весом 7, затем выбирает предмет 4 с весом 5 и пропускает предмет 2 с весом 3. Полученное решение оказывается оптимальным для данного экземпляра задачи (см. раздел 11.2, где этот же экземпляр задачи решен методом ветвей и границ). ■

Может быть, жадный алгоритм всегда приводит к оптимальному решению? Конечно же, нет — если бы это было так, мы бы имели полиномиальный алгоритм для NP-сложной задачи. Приведенный далее пример показывает, что при помощи этого алгоритма нельзя получить гарантированную конечную верхнюю границу точности.

Пример 4. Емкость рюкзака $W > 2$.

Предмет	Вес	Стоимость	$\frac{\text{Стоимость}}{\text{Вес}}$
1	1	2	2
2	W	W	1

Поскольку предметы уже расположены в требуемом порядке, алгоритм выбирает первый из них и пропускает второй; общая стоимость подмножества равна при этом 2. Оптимальным же является выбор второго предмета стоимостью W. Следовательно, отношение точности $r(s_a)$ этого приближенного решения равно $W/2$ — величине, не ограниченной сверху. ■

Этот алгоритм очень легко модифицировать, получив приближенный алго-ритм с конечным коэффициентом производительности. Все, что для этого надо, — выбирать лучшее из двух решений: одно из них получается при помощи жадного алгоритма, а второе — из одного предмета наибольшей стоимости, который может поместиться в рюкзаке (заметим, что второй вариант в последнем примере оказывается лучше первого). Нетрудно доказать, что коэффициент производительности такого ***усовершенствованного жадного алгоритма*** равен 2. Таким образом, стоимость оптимального подмножества s^* не более чем в два раза больше стоимости подмножества s_a, полученного при помощи усовершенствованного жадного алгоритма, причем 2 — наименьший множитель, для которого можно сформулировать такое утверждение.