КОМБИНИРОВАННЫЙ И ВОЛНОВОЙ АЛГОРИТМЫ РЕШЕНИЯ ЗАДАЧИ УПАКОВКИ: ПРИНЦИПЫ ПОСТРОЕНИЯ И ОСОБЕННОСТИ

М.В. Ульянов,
доктор технических наук, профессор кафедры «Прикладная математика и моделирование систем» Московского государственного университета печати (МГУП),
Адрес: 127550, Москва, ул. Прянишникова, 2а, кафедра «Прикладная математика и моделирование систем», Ульянов М.В. Тел: 8-916-589-9404. E-mail: muljanov@mail.ru
О.А. Наумова,
студентка кафедры «Персональная электроника» Московского государственного университета приборостроения и информатики (МГУПИ),
E-mail: helga_13_07@mail.ru

В статье рассмотрены базовые алгоритмы точного решения задачи одномерной оптимальной по стоимости упаковки: рекурсивный и табличный. На основе их рационального совмещения предлагаются комбинированный и волновой алгоритмы, обладающие лучшими ресурсными характеристиками. Указаны особенности их применения и различия в ресурсных требованиях.

Ключевые слова: Задача оптимальной упаковки; комбинированный алгоритм; волновой алгоритм; ресурсная эффективность.

Введение

Начиная с конца 50-х годов XX века, когда Р. Беллман предложил и обосновал идеи метода динамического программирования, задача оптимальной по стоимости одномерной упаковки привлекает внимание программистов и ученых, занимающихся разработкой и анализом алгоритмов. Именно этой задачей открывается ряд иллюстративных примеров в классической книге Р. Беллмана и Р. Дрейфуса «Прикладные задачи динамического программирования» [1]. Повышенный интерес к этой задаче, равно как и к её многочисленным модификациям, связан как с разнообразными практическими областями применения, так и с отсутствием «быстрых» алгоритмов её решения. Постановки, сводимые к задаче оптимальной упаковки, возникают при решении целого ряда экономических задач, задач планирования бизнеса и логистики. Мощности современных компьютеров позволяют в настоящее время получать точное решение этой задачи в практически значимом сегменте длин входов, однако требование повышения временной эффективности её решения в настоящее время остаётся актуальным.

Точный классический алгоритм Беллмана для задачи одномерной упаковки может быть реализован двумя принципиально разными алгоритмами:

- ✦ табличным алгоритмом, опирающимся на вычисление и хранение векторов оптимальной упаковки для всех последовательных целочисленных значений объема;
- ✦ рекурсивным алгоритмом, непосредственно реализующим основное функциональное уравнение Белламна.

В статье рассматриваются комбинированный и волновой алгоритмы, в основу создания которых положен принцип сочетания достоинств рекурсивного и табличного алгоритмов. Предлагаемые решения направлены на повышение временной эффективности решения задачи одномерной упаковки, и, следовательно, на расширение сегмента длин входов, на котором точное решение может быть получено за реальное время.

Математическая постановка задачи одномерной упаковки

Постановка задачи: пусть задано множество

$$Y = \{y_i\}, \quad y_i = \{v_i, c_i\}, \quad i = \overline{1,n},$$

где каждый элемент y_i обладает целочисленным линейным размером – v_i, или «объёмом» в общепринятых терминах задачи упаковки, и ценовой характеристикой – c_i, которая содержательно отражает практически значимые предпочтения для загрузки объектов данного типа. Также целочисленным значением задан основной объём упаковки V. В классической постановке элементы y_i называются типами грузов. Для описания количества загружаемых в объем V элементов y_i введём в рассмотрение характеристический вектор:

$$x = \{x_i\}, \quad i = \overline{1,n},$$

где x_i – неотрицательное целое.

Значение компонента вектора $x_i = k$ соответствует загрузке k элементов типа y_i в объём V. Таким образом, описание некоторой упаковки грузов представляет собой точку в n-мерном целочисленном пространстве E_z^n. Среди всех возможных упаковок объема V грузами из Y должна существовать по крайней мере одна упаковка, максимизирующая суммарную стоимость, что приводит к постановке задачи как задачи максимизации линейного функционала [1]:

$$P_n(\mathbf{x}) = \sum_{i=1}^{n} C_i(x_i) = \sum_{i=1}^{n} x_i \cdot c_i \to \max,$$
$$\text{при} \quad \sum_{i=1}^{n} x_i \cdot v_i \le V. \qquad (1)$$

Содержательно ограничение в (1) означает, что суммарный объём, занимаемый грузами всех типов в количествах, указанных компонентами характеристического вектора **x**, не должен превышать общего объёма упаковки. Поскольку и целевая функция и ограничение линейны, а значения x_i – неотрицательные целые числа, то задача (1) является задачей линейного целочисленного программирования.

Оценку вычислительной сложности решения данной задачи методом полного перебора можно получить, если рассматривать прямоугольный параллелепипед в пространстве E_z^n, размеры сторон которого определяются типами грузов. Это приводит к верхней оценке количества точек перебора в виде [2]

$$\prod_{i=1}^{n} \left(\lfloor V/v_i \rfloor + 1 \right) \le (m+1)^n, \quad m = \max_{i=1,n} \{V/v_i\}.$$

При значительном разбросе значений v_i оценка по произведению значительно лучше, чем $(m + 1)^n$, но всё равно неприемлема для практически значимых постановок задачи упаковки.

Отметим, в этой связи, что рассматриваемая задача является *NP*-трудной [1], и для неё в общем виде отсутствуют сегодня полиномиальные по n точные алгоритмы решения. Это приводит к необходимости использования более эффективных точных методов решения задачи одномерной оптимальной упаковки, существенно сокращающих полный перебор, одним из которых является метод динамического программирования.

Функциональное уравнение Беллмана

Опираясь на терминологию метода динамического программирования, будем считать, что в рассматриваемой задаче распределяемым ресурсом является объём упаковки V, а максимизизация дохода, заданного линейным функционалом $P_n(\mathbf{x})$ производится путём распределения ограниченного ресурса V между грузами указанных типов. Поскольку целевой функционал $P_n(\mathbf{x})$ обладает свойством аддитивности, то метод динамического программирования применим, и основное функциональное уравнение Беллмана имеет вид [1]

$$\begin{cases} f_0(v) = 0; \\ f_m(v) = \max\limits_{x_m} \{x_m \cdot c_m + f_{m-1}(v - x_m \cdot v_m)\}, \\ m = \overline{1,n},\ v = \overline{0,V},\ x_m = 0,1,\ldots,\left\lfloor \dfrac{v}{v_m} \right\rfloor. \end{cases} \qquad (2)$$

Таким образом, метод предусматривает последовательное решение одномерных задач целочисленной оптимизации с использованием информации об оптимальной упаковке объёма v предыдущими типами грузов. Решением поставленной задачи является значение $f_n(V)$ и соответствующий вектор оптимальной упаковки. Поскольку значения $f_1(v)$ могут быть элементарно вычислены на основе (2), то в дальнейшем будет рассматриваться следующее основное функциональное уравнение для задачи одномерной оптимальной упаковки, записанное в виде рекуррентного соотношения, определяющего рекурсивно заданную функцию $f_m(v)$ [2]

$$\begin{cases} f_1(v) = \left\lfloor \dfrac{v}{v_k} \right\rfloor \cdot c_1, \quad m = 1; \\ f_m(v) = \max\limits_{x_m} \{x_m \cdot c_m + f_{m-1}(v - x_m \cdot v_m)\}, \\ m = \overline{2,n},\ x_m = 0,1,\ldots,\left\lfloor \dfrac{v}{v_m} \right\rfloor. \end{cases} \qquad (3)$$

Рекурсивный алгоритм решения задачи упаковки

Рекурсивная реализация функционального уравнения Беллмана для задачи одномерной упаковки достаточно проста. Рекурсивный алгоритм, напрямую реализующий рекуррентное соотношение (3) – алгоритм **A1**, останавливается при значении $m = 1$, когда значение функции $f_1(v)$ может быть вычислено непосредственно. Рекурсия при $m \geq 2$ выполняется для вычисления оптимальной по стоимости упаковки текущего объёма v грузами типа m совместно с грузами предыдущих $m - 1$ типов.

Рассмотрим пример решения задачи одномерной упаковки с использованием данного алгоритма – нас интересует порождённое дерево рекурсии. Пусть необходимо решить задачу для трёх типов грузов при общем объёме упаковки $V = 10$. Информация об объёмах v_i и стоимостях c_i для типов грузов приведена в *табл. 1.*

Таблица 1

Описание типов грузов

i	v_i	c_i
1	2	3
2	3	5
3	4	7

Решением поставленной задачи будет значение функции Беллмана $f_3(10)$, при этом алгоритм порождает дерево рекурсии, показанное на *рис. 1.*

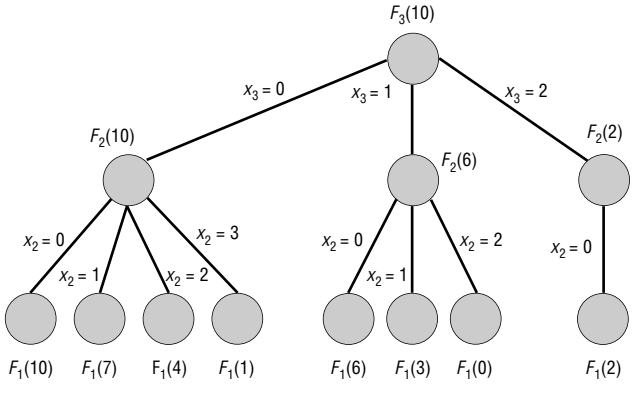

Рис. 1. Дерево рекурсии, порождаемое рекурсивным алгоритмом упаковки.

Характерным недостатком данного алгоритма является повторное вычисление значений функции $f_m(v)$ в различных фрагментах дерева рекурсии. При этом чем дерево шире, а глубина рекурсии больше, тем выше вероятность повторных вычислений. В общем случае трудоёмкость этого алгоритма асимптотически определяется биномиальными коэффициентами при специальной параметризации [1], что приводит ряда входов к значительному времени получения решения.

Табличный алгоритм решения задачи упаковки

Рассмотрим другой алгоритм точного решения этой задачи методом динамического программирования – табличный алгоритм **A2**, позволяющий получить оптимальное решение не только для заданного объёма V, но и набор оптимальных решений для всех промежуточных дискретных значений этого объёма. Обозначим вектор оптимальной упаковки для объёма v элементами y_i, $i = \overline{1,m}$, $m \leq n$, через $\mathbf{x}_v^m$, а через $f_m(v)$ стоимость оптимальной упаковки в этом объёме. В силу принципа оптимальности [1]

$$f_m(v) = \max_{x_m} P_m(\mathbf{x}),$$

а порядок предъявления элементов не является существенным. Результатом решения задачи является набор оптимальных значений целевой функции $f_n(v)$ и соответствующих векторов оптимальной упаковки $\mathbf{x}_v^n$ для всех объемов v от 0 до V исходными элементами (грузами различных типов) из множества Y. Отметим, что поскольку табличный способ позволяет получить оптимальные решения для всех промежуточных объемов упаковки, то эту информацию можно использовать, например, для исследования чувствительности целевой функции $f_n(v)$ по изменению объёма упаковки. Заметим также, что рекурсивный вызов в функциональном уравнении (3) в табличном алгоритме есть не что иное, как обращение к предыдущей таблице оптимальной упаковки для определённого значения объёма.

Пример формирования таблицы оптимальной упаковки для исходных данных, приведённых в *табл. 1*, представлен в *табл. 2.*

Очевидным недостатком данного алгоритма являются «пустые» вычисления в тех строках таблицы, которые не будут задействованы в будущем. Таким образом, алгоритм выполняет лишние вычисления, объём которых может быть значительным. Явная зависимость трудоёмкости данного алгоритма от объёма упаковки приводит к тому, что для больших значений объём лишних вычислений возрастает.

Таблица 2

Таблица оптимальной упаковки

v	x_1	$f_1(v)$
1	0	0
2	1	3
3	1	3
4	2	6
5	2	6
6	3	9
7	3	9
8	4	12
9	4	12
10	5	15

v	x_1	x_2	$f_2(v)$
1	0	0	0
2	1	0	3
3	0	1	5
4	2	0	6
5	1	1	8
6	0	2	10
7	2	1	11
8	1	2	13
9	0	3	15
10	2	2	16

v	x_1	x_2	x_3	$f_3(v)$
1	0	0	0	0
2	1	0	0	3
3	0	1	0	5
4	0	0	1	7
5	1	1	0	8
6	0	2	0	10
7	0	1	1	12
8	0	0	2	14
9	0	3	0	15
10	0	2	1	17

Комбинированный алгоритм

Совместное рассмотрение недостатков рекурсивного и табличного алгоритмов позволяет сделать вывод о том, что в крайних случаях, например, при значительном объёме упаковки и больших значениях объёмов грузов рекурсия будет существенно эффективнее табличных расчётов, а при малых объёмах грузов мы будем наблюдать противоположную ситуации. Таким образом, мы можем говорить, что эти два классических алгоритма являются в определённом смысле комплементарными – они дополняют друг друга, обладая противоположным поведением функции трудоёмкости [3]. Эта особенность позволяет создать комбинированный алгоритм, использующий табличный расчёт и рекурсию в их рациональных диапазонах.

Комбинированное алгоритмическое решение состоит в выборе одного из двух алгоритмов, что может быть реализовано при заданных значениях n, V, k, на основе сравнения их трудоёмкостей [3]. При определённых условиях, зависящих от исходных данных, более рациональным является построение комбинированного алгоритма, основной идеей которого является предвычисление оптимальных упаковок для некоторого числа типов грузов табличным алгоритмом и дальнейшее решение задачи рекурсивным алгоритмом с сокращённым числом типов грузов. Каков оптимальный по трудоемкости порог, т.е. оптимальное число типов грузов обрабатываемых табличным алгоритмом? Ответ на этот вопрос можно получить на основе следующих рассуждений.

Предположим, что табличным алгоритмом находится оптимальное решение для упаковки первыми (в порядке предъявления) m типами грузов, причем $m \geq 1$, при этом для остальных $n-m$ типов задача решается рекурсивным алгоритмом. В листьях порождённого дерева рекурсии на уровне $n-m+1$ происходит копирование вектора оптимальной упаковки, полученного табличным алгоритмом. Это приводит к необходимости модификации рекурсивного алгоритма. Этот модифицированный алгоритм, обозначим его через **A1M**, отличается от алгоритма **A1** фрагментом останова рекурсии, в котором происходит копирование строки результатов табличного алгоритма.

Параметризуем исходные данные числом грузов среднего объёма, размещаемых в общем объёме упаковки. Обозначим этот параметр через k. Поскольку на основе исходных данных значения (n, V, k) известны, то можно построить функцию $g(m)$, значением которой является совокупная трудоёмкость комбинированного алгоритма, при этом предполагается, что значения n, V, k, определяемые входом, фиксированы:

$$g(m) = f_{A1M}(n-m+1,\ k) + f_{A2}(m,\ V,\ k).$$

Оптимальное значение может быть найдено как

$$m^* = \arg\min_{1 \le m \le n-1} g(m). \qquad (4)$$

Отметим, что значение $m^* = m^*(n, V, k)$. Поскольку значение m является целым числом, то m^* может быть найдено простым перебором значений функции $g(m)$ при $1 \le m \le n-1$. Тогда трудоёмкость комбинированного алгоритма, назовём его **AC**, составит

$$f_{AC}(n,V,k,m^*) = g(m^*) = \\ = f_{A1M}(n-m^*+1,\ k) + f_{A2}(m^*,\ V,\ k).$$

В общем случае возникает задача построения комбинированного алгоритмического решения – рационального выбора одного из трёх алгоритмов на основе анализа текущего входа. Мы сравниваем между собой рекурсивный, табличный и комбинированный алгоритмы – **A1, A2, AC**. Критерием выбора является минимум значения функций трудоёмкости. Понимая под аргументом оптимизации индекс алгоритма, и фиксируя значения (n, V, k), определённые текущим входом, можно записать

$$A_{opt}:\ f_{A_{opt}}(n,V,k) = \\ = \arg\min_{A1,A2,AC} \left\{ f_{A1}(n,k),\ f_{A2}(n,V,k),\ f_{AC}(n,V,k,m^*) \right\}, \qquad (5)$$

при этом возможны три следующих случая, возникновение которых определяется текущим входом алгоритма:

1. Случай 1 –

$$f_{A_{opt}}(n,V,k) = f_{A1}(n,k).$$

В этом случае рекурсивный алгоритм **A1** обеспечивает минимальную трудоёмкость, и комбинации с табличным алгоритмом не требуется.

2. Случай 2 –

$$f_{A_{opt}}(n,V,k) = f_{A2}(n,V,$$

В этом случае табличный алгоритм **A2** имеет минимальную трудоёмкость для данного входа, и комбинации с рекурсивным алгоритмом не требуется.

3. Случай 3 –

$$f_{A_{opt}}(n,V,k) = f_{AC}(n,V,k,m^*), \quad 1 \le m^* \le n-1.$$

В этом случае рациональным является применение комбинированного алгоритма **AC** с порогом переключения равным m^*. При этом на первом этапе табличный алгоритм получает таблицу оптимальной упаковки заданного объёма грузами типов $1,\overline{m}^*$, а рекурсивный алгоритм на втором этапе, порождает дерево рекурсии от груза с номером n до груза с номером m^*, останавливает рекурсию на уровне $n-m^*+1$, копирует результаты табличного алгоритма в свою структуру данных, и цепочкой рекурсивных возвратов получает вектор оптимальной упаковки и значение целевого функционала для исходной задачи.

Программная реализация комбинированного алгоритма содержит управляющий фрагмент, который по параметрам текущего входа решаемой задачи (n, V, k) определяет значение оптимального порога переключения m^* по формуле (4) и запускает табличный, а затем рекурсивный алгоритмы, – это реализация статически-адаптивного подхода к созданию комбинированных алгоритмов.

Совокупно программная реализация, представляющая собой комбинированное алгоритмическое решение будет содержать табличный, рекурсивный и комбинированный алгоритмы, а также управляющий модуль, который выбирает в соответствии с формулой (5) алгоритм, наиболее рациональный для текущего входа.

Дополнительно отметим, что хотя окончательный результат, получаемый как табличным, так и рекурсивным алгоритмами, не зависит от порядка предъявления типов грузов, а теоретическая формула нивелирует порядок предъявления типов грузов, тем не менее, порожденный фрагмент дерева рекурсии в комбинированном алгоритме будет меньше (в смысле общего числа вершин), если исходные типы грузов будут первоначально отсортированы по убыванию их объёмов. Такой приём позволяет несколько сократить ширину уровней дерева рекурсии, порожденного рекурсивным алгоритмом, и получить лучшие временные характеристики для рекурсивного и комбинированного алгоритмов.

Волновой алгоритм

В основе волнового алгоритма также лежит идея как эффективного сочетания достоинств обоих классических решений, так и их комплементарности. Главное отличие волнового алгоритма от комбинированного – это отсутствие предвычисления порога переключения. Ни до, ни во время поиска решения мы не анализируем функции трудоёмкости табличного и рекурсивного алгоритмов. Вместо этого, после расчёта упаковки очередным грузом, значение некоторого параметра определяет, каким из алгоритмов лучше обрабатывать последующий тип грузов.

Так как табличный алгоритм на каждом шаге вычисляет оптимальную упаковку для всех дискретных значений объёма от 0 до V, то его разумно использовать, когда следующий шаг рекурсии будет порождать значительное число листьев с различными значениями объёма упаковки. К тому же эффективность рекурсии при обработке каждого последующего груза в общем случае падает, т.к. возрастает число вершин, имеющих одинаковый аргумент целевой функции. Исходя из этого, на первом этапе волнового алгоритма управление передаётся рекурсии. Но в отличие от комбинированного и рекурсивного алгоритма в чистом виде, здесь используется иная структура данных. Мы «разворачиваем» рекурсию в таблицу хранения рекурсивных обращений. Что это означает? Порождение очередной вершины дерева рекурсии обрабатывается как создание строки таблицы с номером, соответствующим значению оставшегося объёма упаковки, и записью дополнительной информации, позволяющей организовать быстрые вычисления на последнем этапе волны обратных расчётов, эквивалентном цепочке рекурсивных возвратов. Для следующего типа груза создаётся новая таблица, при этом рекурсивные вызовы, порождающие цепочку таблиц, будут продолжаться до передачи управления табличному алгоритму. Порог переключения между алгоритмами

определяется динамически, исходя из условия рационального продолжения развёртывания рекурсии. После обработки очередного уровня рекурсии, т.е. создания таблицы для соответствующего типа груза, подсчитывается число заполненных строк текущей структуры данных – таблицы хранения рекурсивных обращений. Если число обращений достигло значения $(V+1)/2$, то мы предполагаем, что на следующем шаге число рекурсивных вызовов будет больше, чем число строк таблицы, и, следовательно, управление передаётся табличному алгоритму. Структура данных табличного алгоритма идентична структуре таблиц таблицы хранения рекурсивных обращений, что позволяет при обратном ходе волны непосредственно копировать результаты табличного алгоритма в соответствующую структуру. Схема волнового алгоритма показана на *рис. 2.*

Первый этап алгоритма – волна порождения цепочки таблиц, эквивалентных рекурсии по типам грузов. Первый этап продолжается до достижения рационального порога переключения на табличный алгоритм, т.е. до достижения половины заполненных строк таблицы хранения рекурсивных обращений. Второй этап – классический табличный алгоритм выполняется для оставшихся типов грузов. В точке переключения выполняется копирование из структуры данных табличного алгоритма в таблицу хранения рекурсивных обращений. Третий, заключительный этап – волна обратных расчётов, формирующая на последнем шаге решение поставленной задачи упаковки в первой порождённой таблице рекурсии.

Как и в случае комбинированного алгоритма, степень использования рекурсивного и табличного алгоритмов зависит от входных данных с той лишь разницей, что для всех входов волнового алгоритма обязательно будет построена таблица хранения рекурсивных обращений хотя бы для первого типа грузов. Таким образом, остаётся два варианта поведения волнового алгоритма:

- ✦ после обработки груза с номером заполненными оказываются более половины строк таблицы хранения рекурсивных обращений, и управление передаётся табличному алгоритму;
- ✦ число заполненных строк не достигает значения даже при рассмотрении предпоследнего груза. В этом случае табличный алгоритм не участвует в поиске решения.

Сравнение комбинированного и волнового алгоритмов

Рациональное совмещение рекурсивного и табличного алгоритмов решения задачи одномерной упаковки – общий принцип построения волнового и комбинированного алгоритмов. Этот принцип во многом обусловливает и схожесть структур этих алгоритмов. Так какой же из методов предпочтительнее использовать для получения быстрого решения? Ответ на этот вопрос не является вполне очевидным и зависит от аппаратных возможностей.

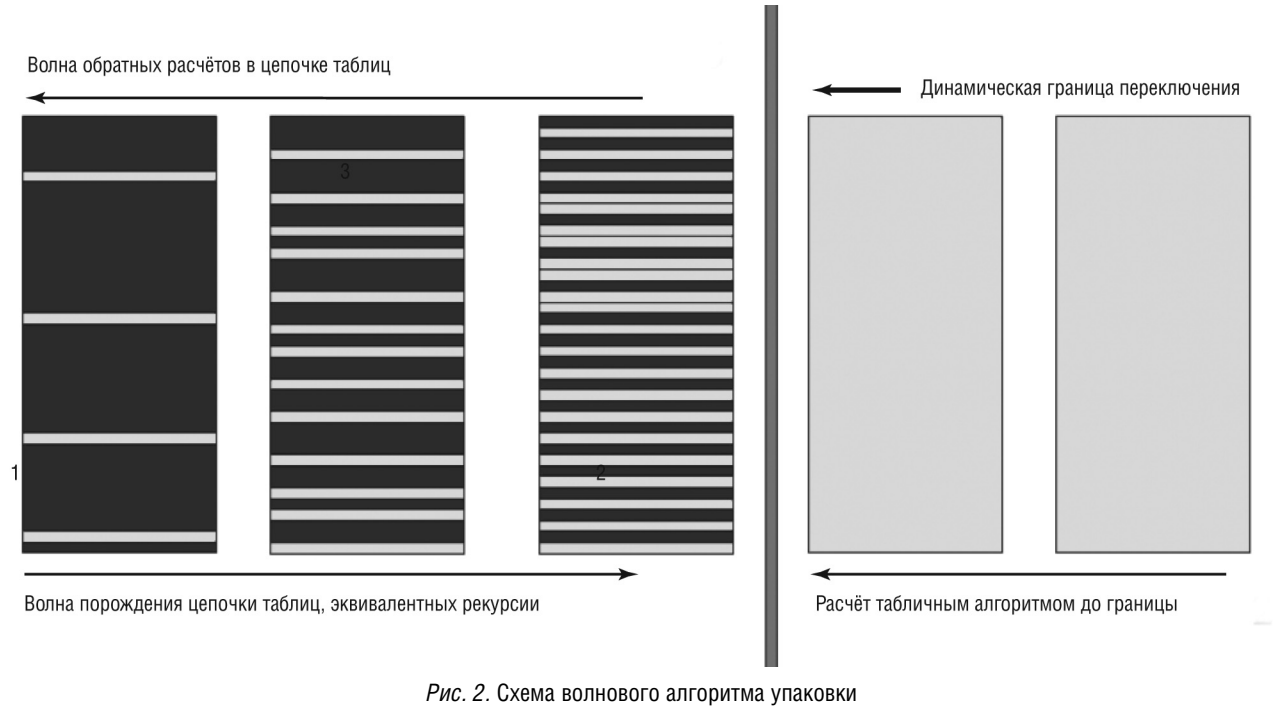

Рис. 2. Схема волнового алгоритма упаковки

Если определяющим фактором является время получения точного решения, то выбор за волновым алгоритмом. Он обеспечивает более быстрое решение за счёт:

- ✦ отсутствия модуля предвычисления порога переключения;
- ✦ отсутствия повторных расчётов целевой функции с одинаковым аргументом для вершин дерева рекурсии разных уровней;
- ✦ непосредственного копирования результатов работы табличного алгоритма в таблицу хранения рекурсивных обращений, благодаря идентичности структур.

Как известно, за всё приходится платить. В случае волнового алгоритма расплачиваться за временную эффективность приходится дополнительными затратами оперативной памяти на хранение полных ссылок рекурсивных вызовов. Волновой алгоритм демонстрирует классическую дилемму ресурсной эффективности – улучшение временной эффективности за счёт ухудшения ёмкостных характеристик.

Заключение

Таким образом, в статье предложены два алгоритма решения задачи одномерной оптимальной по стоимости упаковки, основанные на рациональном совмещении классических алгоритмов её решения – рекурсивном и табличном.

Комбинированный алгоритм использует результаты теоретического анализа трудоёмкости для предвычисления оптимального порога переключения между рекурсивным и табличным алгоритмами на основе данных конкретного входа.

Волновой алгоритм не выполняет статического расчёта порога переключения. Порог в данном алгоритме определяется динамически на основе подсчёта числа заполненных строк специальной таблицы хранения рекурсивных обращений. За счёт затрат оперативной памяти предлагаемый волновой алгоритм имеет лучшую временную эффективность.

Предложенные алгоритмы могут быть использованы при решении практических постановок задач бизнес-информатики, сводимых к задаче оптимальной одномерной упаковки. ■

Литература

1. Беллман Р., Дрейфус Р. Прикладные задачи динамического программирования: Пер. с англ. – М.: Наука, 1965, – 457 с.
2. Головешкин В. А., Ульянов М. В. Теория рекурсии для программистов. – М.: Издательство «Наука ФИЗМАТЛИТ», 2006. – 296 с.
3. Ульянов М. В., Гурин Ф. Е., Исаков А. С., Бударагин В. Е. Сравнительный анализ табличного и рекурсивного алгоритмов точного решения задачи одномерной упаковки // Exponenta Pro Математика в приложениях. 2004. №2(6). С. 64–70.