

КЛАССЫ АЛГОРИТМОВ

Для того, чтобы сравнивать алгоритмы между собой необходимо четко определить понятие *класса алгоритмов*. Тогда можно говорить об алгоритме оптимальном в своем классе, т. е. об алгоритме, который работает по крайней мере так же хорошо (относительно некоторого свойства, характеризующего класс), как любой другой алгоритм из рассматриваемого класса. Знание того, что имеющийся алгоритм является оптимальным, может предотвратить поиск не существующего лучшего алгоритма. Рассмотрим, как можно определить класс алгоритмов на примере задачи о фальшивой монете.

Задача о фальшивой монете формулируется следующим образом. Имеется n монет, о которых известно, что $n-1$ из них являются настоящими и не более, чем одна монета фальшивая (легче или тяжелее остальных монет). Дополнительно к группе из n сомнительных монет дается еще одна монета, причем известно, что она настоящая. Имеются весы, с помощью которых мы можем сравнивать общий вес любых m монет с общим весом любых других m монет и тем самым устанавливать, имеют ли две группы монет одинаковый вес, либо одна из групп легче другой. Задача состоит в нахождении фальшивой монеты или установление факта ее отсутствия за наименьшее число взвешиваний.

Пусть сомнительные монеты занумерованы числами $1, 2, \dots, n$, настоящая монета имеет номер 0 . Пусть теперь $S = \{0, 1, 2, \dots, n\}$ – множество монет. Пусть S_1, S_2 – непересекающиеся непустые подмножества множества S , запись $S_1 : S_2$ обозначает операцию сравнения весов множеств S_1 и S_2 . При сравнении возможны три исхода, которые обозначаются как $S_1 < S_2$, $S_1 = S_2$ или $S_1 > S_2$ в зависимости от того, является ли вес S_1 меньшим, равным или большим веса S_2 .

Рассматриваемые алгоритмы можно представить в виде тернарного дерева решений. Пример дерева для случая $n = 4$ приведен на Рис. 4. Корень дерева изображен в виде кружка и помечен $1 : 2$, что означает, что алгоритм начинает работу со сравнения монет с номерами 1 и 2. Три исходящие из корня ветви ведут к поддеревьям, определяющим продолжение работы алгоритма после каждого из трех возможных исходов первого сравнения. Квадраты, называемые листьями дерева, означают, что работа алгоритма заканчивается, метки в квадратах соответствуют исходам: “1L” – монета 1 легкая, “1H” – монета 1 тяжелая, “G” – все монеты настоящие, пустой квадрат означает, что при наших предположениях этот случай возникнуть не мог.

Алгоритм, приведенный на Рис.4, требует 3 сравнений в худшем случае, т.е. максимальное число сравнений равно 3. Считая все исходы равновероятными можно найти среднее число сравнений, которое требует данный алгоритм:

$$T_{cp} = \frac{6}{9} \cdot 2 + \frac{3}{9} \cdot 3 = \frac{7}{3}.$$

Мы учли, что 6 из 9 исходов соответствуют 2 сравнениям и 3 из 9 исходов требуют 3 сравнений.

На одну чашку весов можно положить больше одной монеты. Пример дерева решений, которое начинается со сравнения $1, 2 : 3, 4$, приведен на Рис.5. Если все монеты настоящие, задачу можно решить за одно сравнение. В худшем случае, независимо от того, как дополняется дерево решений, потребуется 3 сравнения, так как одно сравнение не может идентифицировать каждый из 4 исходов, которые возможны на ветви “<” и ветви “>”.

Используя монету 0 (настоящая монета) можно получить дерево решений, приведенное на Рис. 6. В худшем случае и в среднем оно требует двух сравнений.

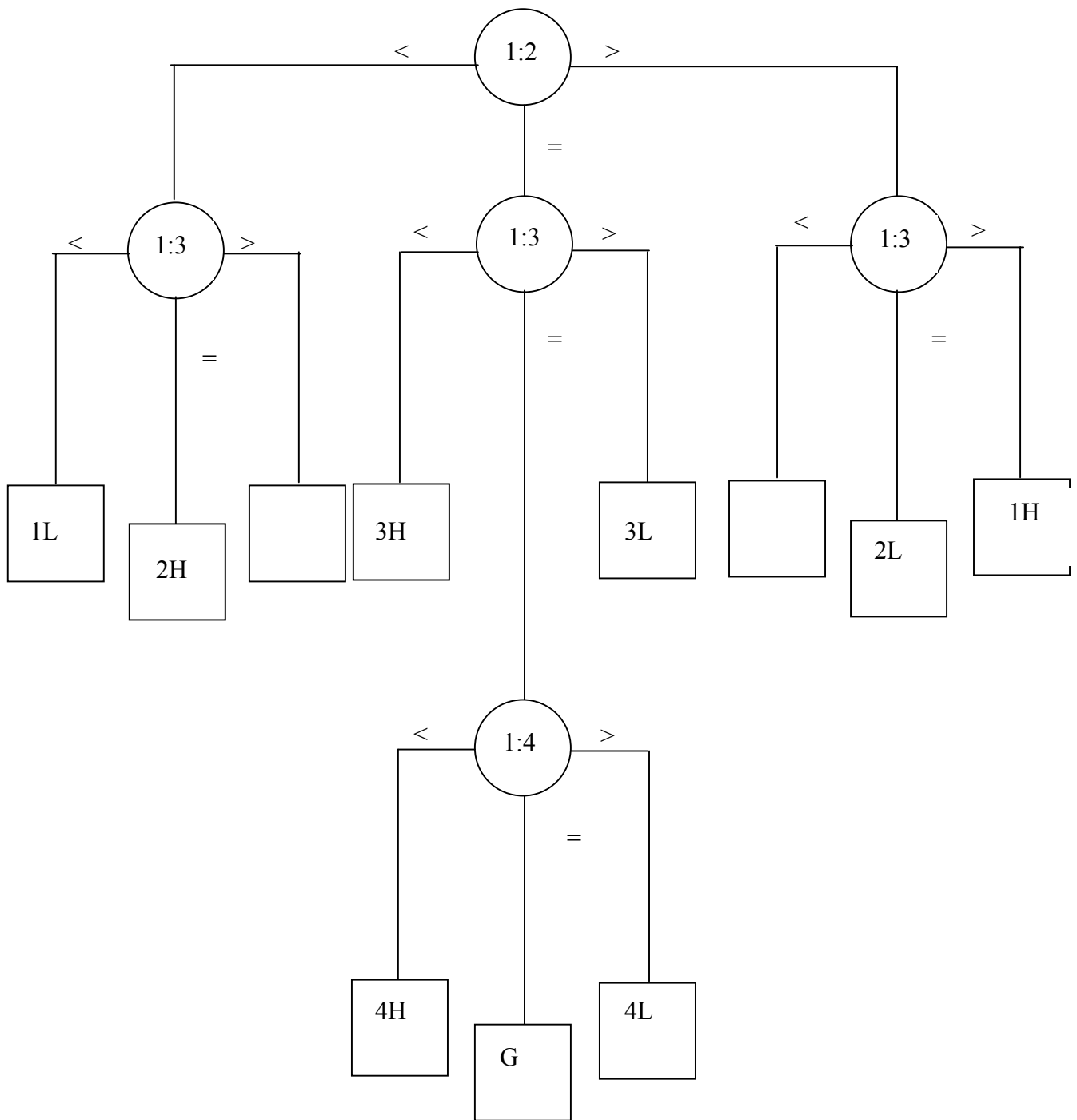


Рис. 4. Пример дерева решений для случая четырех монет

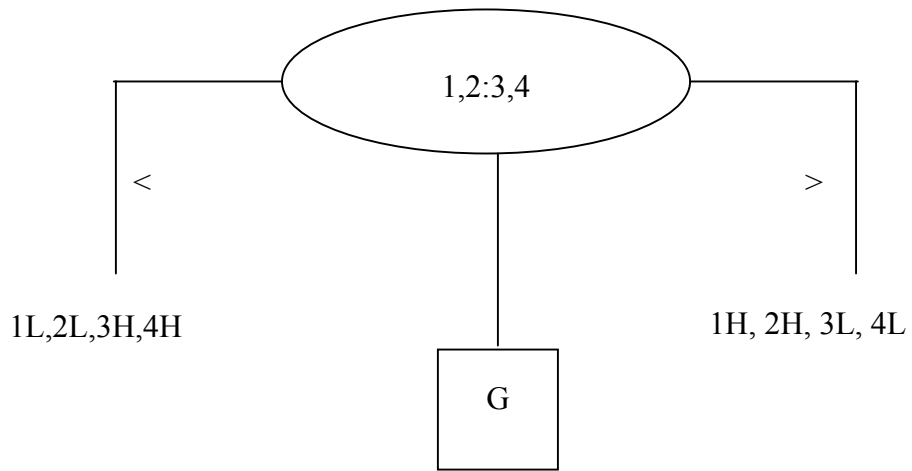


Рис. 5. Другое дерево решений для задачи о четырех монетах

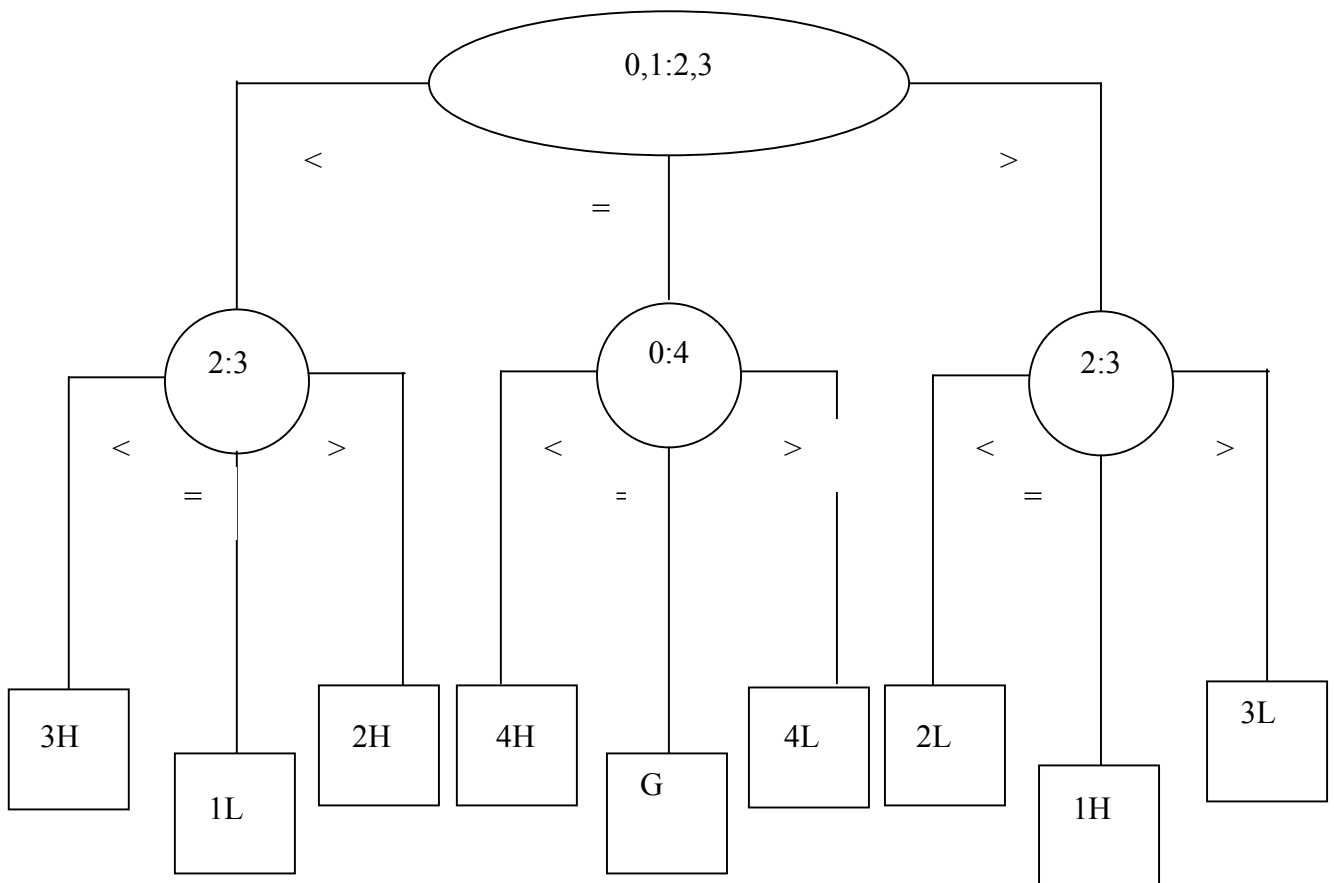


Рис. 6. Оптимальное дерево решений для задачи о четырех монетах

Рассматриваемый класс алгоритмов решения задачи о фальшивой монете есть множество *тернарных деревьев решений*, обладающих следующими свойствами:

1. Каждый узел помечен сравнением $S_1 : S_2$, где S_1 и S_2 – непересекающиеся непустые подмножества множества $S = \{0,1,2,\dots,n\}$ монет.
2. Каждый лист не помечен (что соответствует невозможному исходу в предположении существования не более чем одной фальшивой монеты), либо помечен одним из исходов iL , iH , G , означающем соответственно, что монета с номером i является легкой или тяжелой или что все монеты настоящие.

Теперь попытаемся найти в этом классе алгоритмов оптимальный. Начнем со случая $n = 4$. Общее число исходов в случае четырех монет равно 9 (каждая из 4 монет может быть легкой или тяжелой, и все монеты могут быть настоящими). Таким образом, любое дерево решений в этом случае должно иметь по крайней мере 9 листьев и не менее двух ярусов. Отсюда следует, что дерево на Рис.6 является оптимальным и в смысле худшего случая и в среднем. Постараемся ответить на вопрос о существовании других оптимальных деревьев решений для случая $n = 4$. Для этого нам необходимо рассмотреть все возможные деревья решений для задачи о четырех монетах. Заметим, что задачу можно упростить, исключив из рассмотрения часть возможного множества решений. Нетрудно заметить, что путем перестановки множества сомнительных монет из дерева, приведенного на Рис.6 можно получить другие оптимальные деревья решений. Одно из них приведено на Рис.7. Все эти оптимальные деревья *изоморфны* дереву на Рис. 6. Исключим из рассмотрения все изоморфные деревья, т.е. деревья, которые могут быть получены одно из другого перенумерацией монет, и будем рассматривать только попарно неизоморфные деревья.

Зададимся вопросом, существует ли оптимальное дерево среди тех, у которых в корне не используется монета с номером 0. Возможны только два различных сравнения в корне. Первое $1:2$, а второе $1,2:3,4$. Деревья первого типа имеют разбиение исходов по ветвям – $(2,5,2)$, второго типа – $(4,1,4)$. Для того чтобы дерево решений содержало только 2 яруса, исходы должны быть распределены как $(3,3,3)$. Таким образом, мы можем сделать вывод, что задачу для четырех монет нельзя решить за два сравнения, не используя настоящую монету.

Рассмотрим деревья решений, которые используют настоящую монету. В этом случае в корне возможны только два различных сравнения: $0:1$ или $0,1:2,3$. Дерево решений для случая использования сравнения $0:1$ приведено на Рис.8. Независимо от дальнейших ветвлений распределение исходов по ветвям $(1,7,1)$, и, следовательно, это дерево не может быть оптимальным. Начальное сравнение $0,1:2,3$ приводит к оптимальному дереву с распределением исходов $(3,3,3)$. Аналогично устанавливается, что в первом от корня ярусе дерева сравнения для оптимального дерева определяются единственным образом. Таким образом, для задачи о четырех монетах существует только одно оптимальное дерево.

Поскольку число листьев в дереве должно быть не меньше числа возможных исходов $2n+1$, то для общего случая можно получить нижнюю оценку необходимого числа сравнений.

Теорема 1. Если $n > 1/2(3^l - 1)$, то задачу для n монет нельзя решить за l сравнений в худшем случае.

Так как $2n+1 \leq 3^l$, то получаем, что для того, чтобы решить задачу за l сравнений, необходимо выполнение условия $n \leq 1/2(3^l - 1)$.

Теорема 2. Если $n > 1/2(3^l - 1)$, то задачу для n монет нельзя решить за l сравнений в среднем.

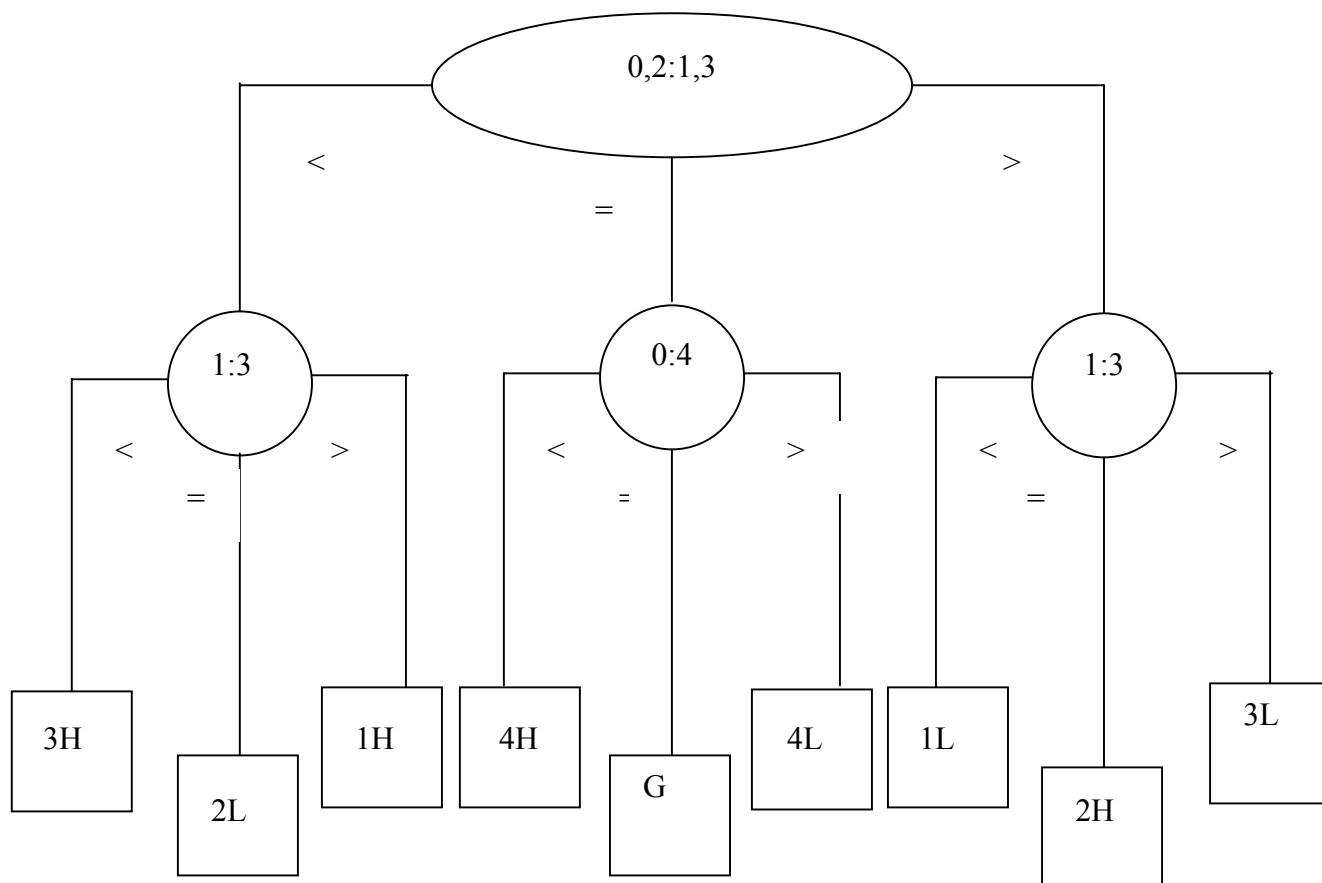


Рис. 7. Оптимальное дерево решений для задачи о четырех монетах, полученное путем перестановки множества сомнительных монет

Теорема 3. Если $n \geq 1/2(3^l - 1)$, то задачу для n монет нельзя решить за l сравнений, не используя монету 0.

Из теоремы 1 следует, что достаточно только показать, что если $2n + 1 = 3^l$, то не существует l уровневых деревьев решений для задачи об n монетах, если не используется настоящая монета. Для некоторого j , $1 \leq j \leq n/2$ в корне любого дерева мы сравнивали бы множество монет из j монет с номерами, скажем, $1, 2, \dots, j$ с другим множеством из j монет, например, с номерами $j + 1, j + 2, \dots, 2j$. Пример соответствующего дерева приведен на Рис.9. Каждой из ветвей “<” и “>” соответствует $2j$ исходов, а оставшиеся $2n + 1 - 4j$ исходов соответствуют ветви “=”. Для того, чтобы получить дерево $l = \log_3(2n + 1)$ уровнями, каждой из перечисленных ветвей должна соответствовать треть исходов. Тогда бы имело место равенство $2j = 2n + 1 - 4j$ или $6j = 2n + 1$, что невозможно для целых n и j так как $6j$ - четное число, а $2n + 1$ - всегда нечетно.

Теорема 4. Если $n \leq 1/2(3^l - 1)$, то задачу для n монет можно решить за l сравнений.

В качестве доказательства приведем оптимальный алгоритм для случая $n = 1/2(3^l - 1)$. Пусть $K_l = 1/2(3^l - 1)$. Положим $K_l = n$. Мы будем использовать два важных тождества:

$$\begin{aligned} 2K_l + 1 &= 3^l, \\ K_l &= 3K_{l-1} + 1. \end{aligned}$$

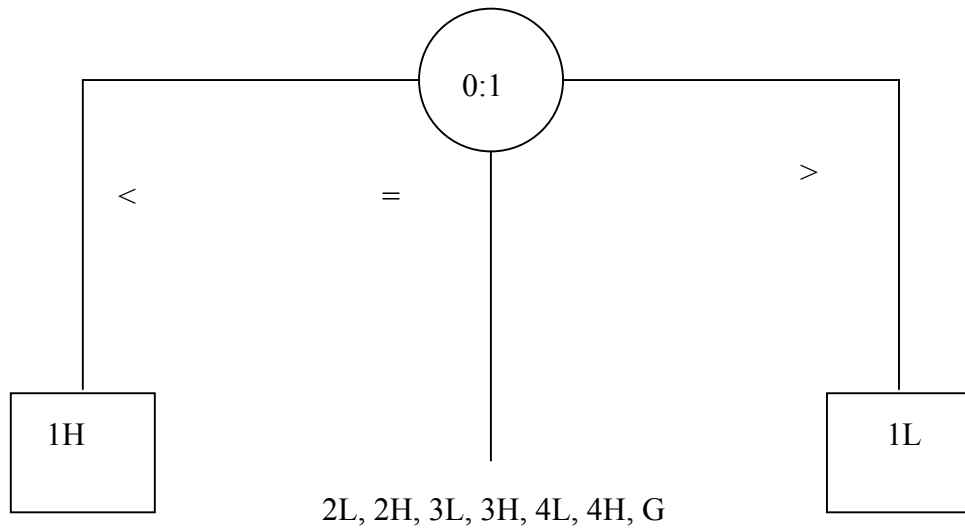


Рис.8. Дерево, использующее настоящую монету

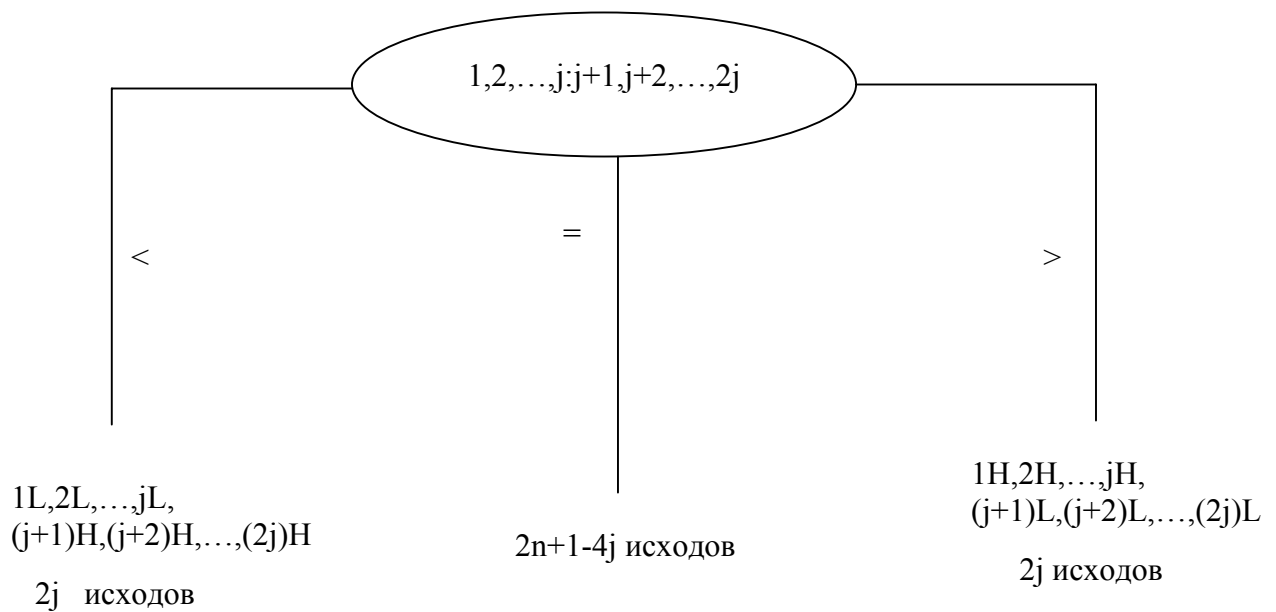


Рис. 9. Корень дерева для задачи об n монетах с ограничением (нельзя использовать монету 0)

Очевидно, что при $K_1 = 1$ и при $n = K_1$, т.е. для одной монеты задачу можно решить за одно сравнение. Мы хотим показать, что в общем случае задачу можно решить за l сравнений. Мы утверждаем, что оптимальный алгоритм решения задачи всегда находится в одном из трех состояний:

- Состояние 1: мы имеем K_i сомнительных монет и до принятия окончательного решения нам разрешается сделать i сравнений.

- Состояние 2: мы имеем множество из $K_i + 1$ монеты и другое из K_i монет. До окончания мы имеем право сделать i сравнений и мы знаем, что либо тяжелая монета находится в множестве из $K_i + 1$ монеты либо в множестве из K_i монет находится легкая монета.
- Состояние 3: то же, что 2, но мы знаем, что или в множестве из $K_i + 1$ монеты находится легкая монета или в множестве из K_i монет находится тяжелая.

Для каждого из состояний мы указываем, какое надо сделать следующее сравнение с помощью функции переходов, приведенной на Рис.10. При этом мы утверждаем, что задаваемое функцией переходов действие всегда будет приводить нас к одному из трех перечисленных выше состояний. При каждом переходе число сравнений i убывает на 1. Таким образом, алгоритм имеет структуру, представленную на Рис.11.

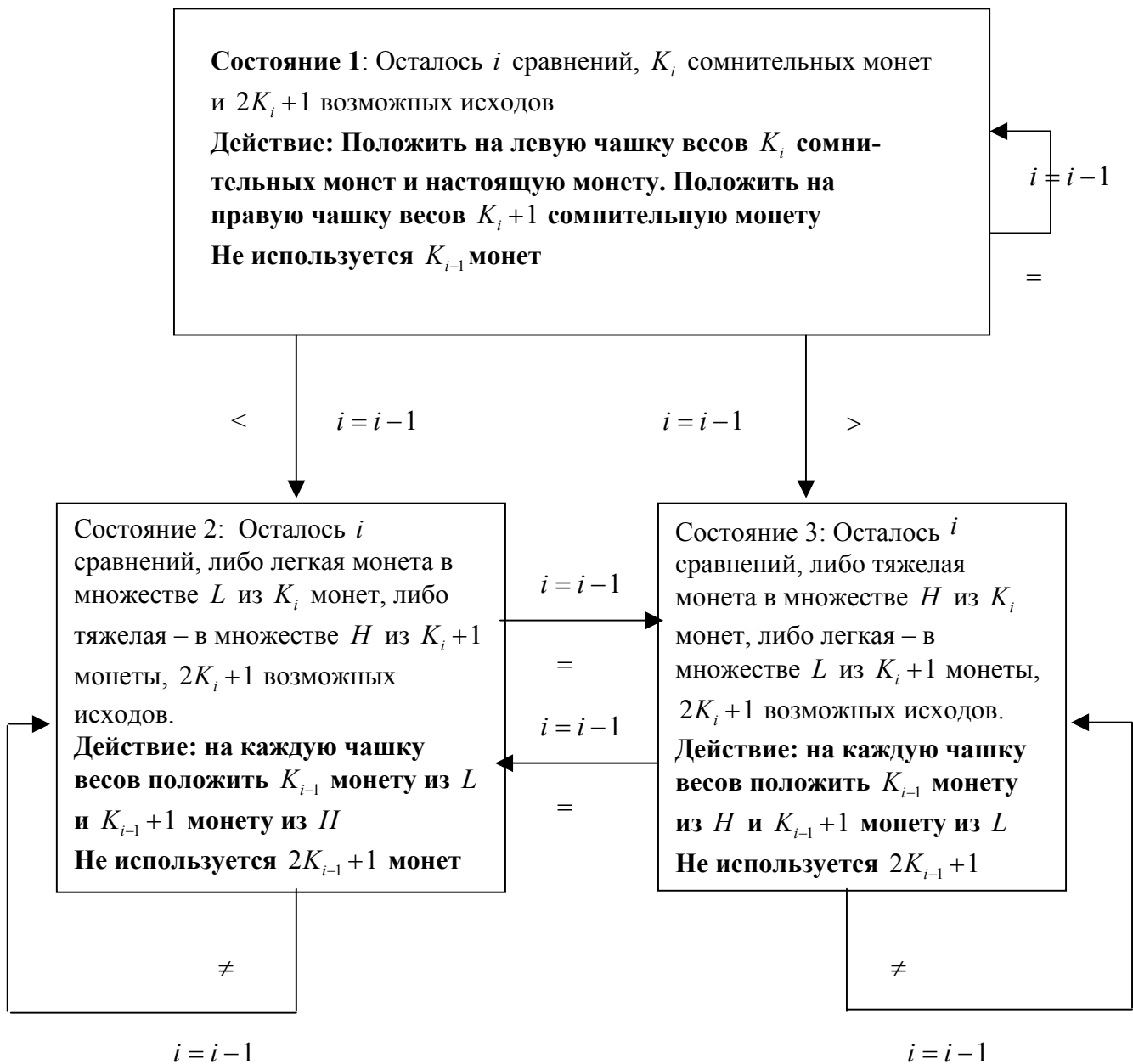


Рис. 10. Диаграмма состояний и функция переходов f оптимального алгоритма в виде дерева решений

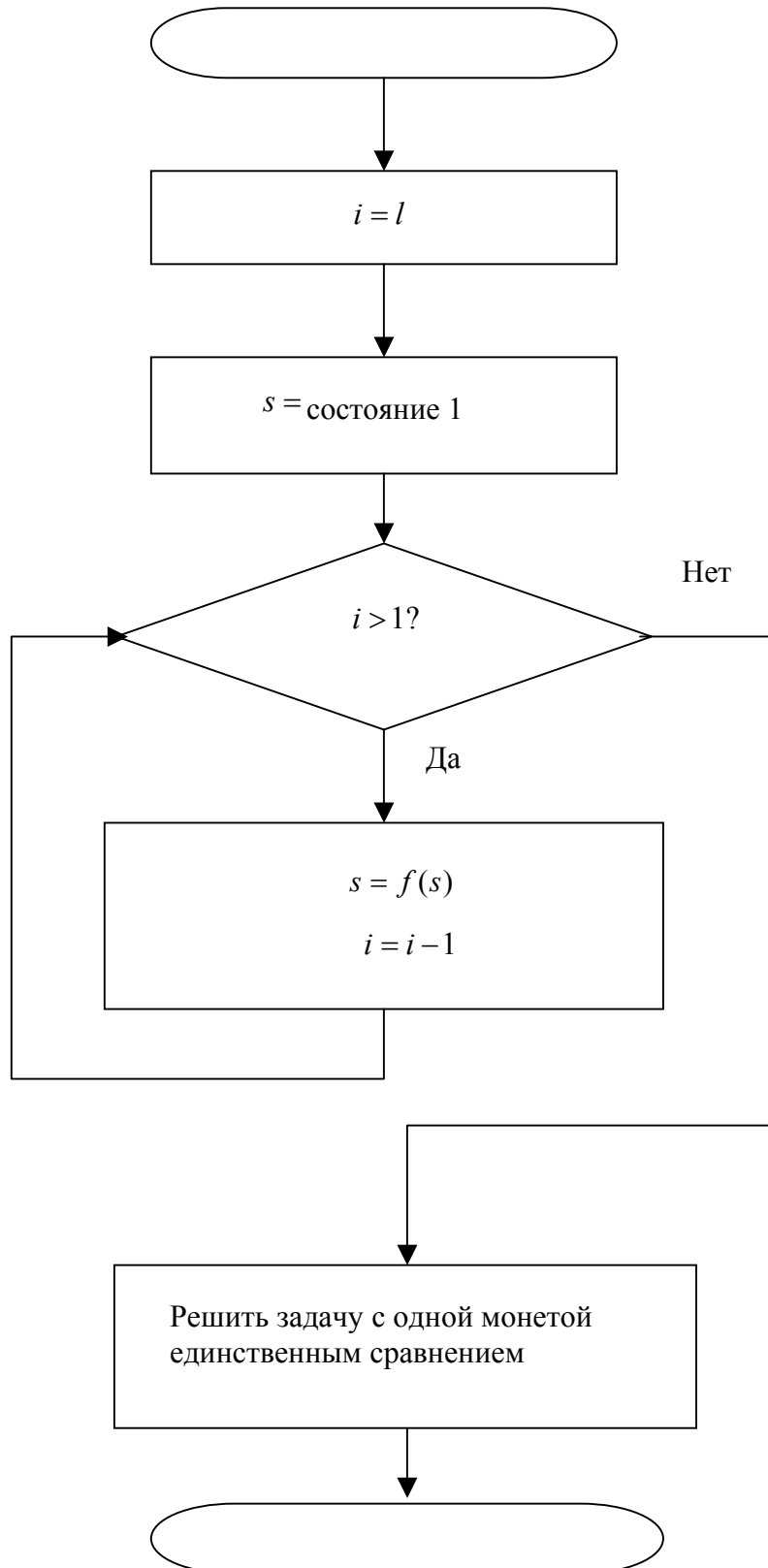


Рис. 11. Структура оптимального алгоритма решения задачи о фальшивой монете для случая $n = 1/2(3^l - 1)$

Применим описанный оптимальный алгоритм для решения задачи о четырех монетах. Начинаем с $K_2 = 4$, состояние 1. На левую чашку весов кладем $K_1 = (K_2 - 1)/3 = 1$ сомнительную монету и монету номер 0. (Например, 0,1). На правую чашку весов кладем $K_1 + 1 = 2$ сомнительных монет (например, монеты 2,3). Если результат сравнения $<$, то $i = i - 1 = 1$ и переходим в состояние 2. Кладем на каждую чашку весов $K_0 = (K_1 - 1)/3 = 0$ монет из легкого множества и $K_0 + 1 = 1$ монету из тяжелого множества (сравниваем монеты 2 и 3). Аналогично при исходе $>$ переходим в состояние 3 и получаем, что необходимо сравнивать монеты 2 и 3. В случае исхода $=$, $i = i - 1 = 1$ остаемся в состоянии 1. На левую чашку весов кладем $K_0 = 0$ сомнительных монет и настоящую монету, на правую чашку весов кладем $K_0 + 1 = 1$ сомнительную монету (сравнение 0 и 4). Нетрудно видеть, что полученное дерево решений совпадает с деревом решений на Рис. 6.

Отметим, что рассмотренный изящный алгоритм решения задачи о фальшивой монете не обобщается на случай задачи определения более, чем одной фальшивой монеты. Однако, он подсказывает очевидный эвристический подход и для этого случая. Для уменьшения числа сравнений необходимо сделать дерево решений как можно более широким. Тернарное дерево решений с данным числом решений можно сделать широким, если в каждой вершине исходы разбить на три группы примерно одинаковой мощности. Возникает вопрос: как оценить степень равномерности распределения? Рассмотрим два неоптимальных дерева для задачи о четырех монетах (Рис. 4, 5). Первое обеспечивает разбиение исходов (2,5,2), второе – (4,1,4). Какое разбиение более однородное и ближе к идеальному разбиению (3,3,3)? Введем меру однородности дерева.

Пусть (p_1, p_2, \dots, p_n) – это вектор вероятностей. Здесь $p_i \geq 0$ – неотрицательное вещественное число, $\sum_{i=1}^n p_i = 1$. Дерево на Рис. 4 порождает распределение вероятностей $p_1 = 2/9$, $p_2 = 5/9$ и $p_3 = 2/9$. В предположении, что все исходы равновероятны, эти величины представляют собой вероятности каждой из трех ветвей, выходящих из корня. Нам нужна функция, обладающая следующими свойствами:

- Симметрична относительно (p_1, p_2, \dots, p_n) , т.е. не зависит от взаимного расположения p_i
- Достигает максимума на векторе с максимальной однородностью:
 $p_1 = p_2 = \dots = p_n = 1/n$
- Достигает минимума, равного 0, на векторе с минимальной однородностью:
 $p_1 = 1, p_2 = \dots = p_n = 0$.
- Монотонно возрастает от точки минимума до точки максимума при изменении α от 0 до $1/n$, $H(1 - (n-1)\alpha, \alpha, \dots, \alpha)$.

Таковыми свойствами обладает хорошо известная в теории информации функция – энтропия:

$$H(p_1, p_2, \dots, p_n) = -\sum_{i=1}^n p_i \log_2 p_i.$$

Используя эту функцию для определения степени однородности деревьев решений, получаем:

$$H(2/9, 5/9, 2/9) = -(2/9 \log_2(2/9) + 5/9 \log_2(5/9) + 2/9 \log_2(2/9)) \approx 1.44$$

$$H(4/9, 1/9, 4/9) = -(4/9 \log_2(4/9) + 1/9 \log_2(1/9) + 4/9 \log_2(4/9)) \approx 1.39.$$

Та же самая мера, примененная к идеальному разбиению (3,3,3) дает

$$H(1/3, 1/3, 1/3) = \log_2 3 \approx 1.58.$$

Таким образом, согласно этой мере, разбиение $(2,5,2)$ ближе к однородному, чем $(4,1,4)$.