

Тема 3. ИСЧЕРПЫВАЮЩИЙ ПОИСК

3.1. Поиск с возвратом

3.1.4. Оценка сложности выполнения поиска с возвратом

Обычно поиск с возвратом приводит к алгоритмам, экспоненциальным по своим параметрам. Это является следствием того, что если все решения имеют длину не более n , то исследованию подлежат приблизительно $\prod_{i=1}^n |A_i|$ вершин дерева поиска, где $|A_i|$ – мощность множества A_i . Даже широкое использование ограничений и склеиваний позволяет в большинстве случаев добиться только того, что $|A_i|$ становится константой; при этом получаются деревья примерно с C^n вершинами для некоторой константы $C > 1$. Поскольку размеры дерева растут так быстро, можно попытаться определить возможность практического осуществления поиска (т. е. за приемлемое время) путем оценки числа вершин в дереве.

Аналитическое выражение для оценки удастся получить редко, так как трудно предсказать, как взаимодействуют различные ограничения по мере появления их при продвижении в глубь дерева поиска. В подобных случаях, когда построение аналитической модели является трудной или вовсе неосуществимой задачей, можно применить *метод Монте-Карло* (метод статистических испытаний). Смысл этого метода в том, что исследуемый процесс моделируется путем многократного повторения его случайных реализаций. Каждая случайная реализация называется *статистическим испытанием*.

Рассмотрим применение метода Монте-Карло для экспериментальной оценки размеров дерева поиска. Идея метода состоит в проведении нескольких испытаний, при этом каждое испытание представляет собой поиск с возвратом со случайно выбранными значениями a_i . Предположим, что имеется частичное решение $(a_1, a_2, \dots, a_{k-1})$ и что число выборов для a_k , основанное на том, вводятся ли ограничения или осуществляется склеивание, равно $x_k = |S_k|$. Если $x_k \neq 0$, то a_k выбирается случайно из S_k и для каждого элемента вероятность быть выбранным равна $1/x_k$. Если $x_k = 0$, то испытание заканчивается. Таким образом, если $x_1 = |S_1|$, то $a_1 \in S_1$ выбирается случайно с вероятностью $1/x_1$; если $x_2 = |S_2|$, то при условии, что a_1 было выбрано из S_1 , $a_2 \in S_2$ выбирается случайно с вероятностью $1/x_2$ и т. д. Математическое ожидание $x_1 + x_1x_2 + x_1x_2x_3 + x_1x_2x_3x_4 + \dots$ равно числу вершин в дереве поиска, отличных от корня, т. е. оно равно числу случаев, которые будут исследованы алгоритмом поиска с возвратом. Существует доказательство этого утверждения [21].

Общий алгоритм поиска с возвратом легко преобразуется для реализации таких испытаний; для этого при $S_k = \emptyset$ вместо возвращения просто заканчивается испытание. Алгоритм 3.4 оценки размера дерева поиска осуществляет N испытаний для подсчета числа вершин в дереве. Операция $a_k \leftarrow \text{rand}(S_k)$ реализует случайный выбор элемента a_k из множества S_k .

```

count  $\leftarrow$  0 // суммарное число вершин в дереве
for i  $\leftarrow$  1 to N do
    {
        sum  $\leftarrow$  0 // число вершин при одном испытании
        product  $\leftarrow$  1 // накапливаются произведения
        определить  $S_1 \subseteq A_1$ 
        k  $\leftarrow$  1
        while  $S_k \neq \emptyset$  do {
            {
                product  $\leftarrow$  product *  $|S_k|$ 
                sum  $\leftarrow$  sum + product
                ak  $\leftarrow$  rand ( $S_k$ )
                k  $\leftarrow$  k + 1
                определить  $S_k \subseteq A_k$ 
            }
        }
        count  $\leftarrow$  count + sum
    }
average  $\leftarrow$  count / N // среднее число вершин в дереве

```

Алгоритм 3.4. Метод Монте-Карло для поиска с возвратом

Таким образом, каждое испытание представляет собой продвижение по дереву поиска от корня к листьям по случайно выбираемому на каждом уровне направлению. Поскольку в методе Монте-Карло отсутствует возврат, оценка размеров дерева выполняется за полиномиальное время.

Вычисление по методу Монте-Карло можно использовать для оценки эффективности алгоритма поиска с возвратом путем сравнения его с эталоном, полученным для задачи с меньшей размерностью. Например, при выполнении алгоритма 3.3 решения задачи о ферзях в случае доски размером 10×10 требуется поиск на дереве с 35 538 вершинами (эталон). В случае размера 11×11 после 1000 испытаний по методу Монте-Карло оценка числа вершин была 161 668, и поэтому можно ожидать, что вычисления потребуют примерно в 4,5 раз больше времени, чем в случае с доской 10×10 . Фактически оказалось, что в случае с доской 11×11 дерево имеет 166 925 вершин и время вычисления в 4 раза больше.

Пример результатов экспериментальных исследований алгоритма решения задачи о ферзях представлены в таблице. В качестве эталона взят размер задачи 11×11 , для которого выполнен как поиск с возвратом (определен фактический размер дерева поиска), так и оценка размеров дерева поиска методом Монте-Карло. Для размера задачи 12×12 применен только метод Монте-Карло, который позволил определить, что ожидаемое время выполнения поиска для доски размера 12×12 составит примерно 451 мс. Для каждого из исследованных методом Монте-Карло размеров задачи проведено $N = 1000$ испытаний.

Конкретизация метода Монте-Карло для задачи о ферзях

```

count ← 0 // суммарное число вершин в дереве
for i ← 1 to N do
    {
        sum ← 0 // число вершин при одном испытании
        product ← 1 // накапливаются произведения
        S1 ← {1, 2, ..., n}
        k ← 1
        while |Sk| > 0 do
            {
                product ← product * |Sk|
                sum ← sum + product
                ak ← rand(Sk)
                k ← k + 1
                // определить Sk ⊆ Ak
                Sk ← ∅
                j ← 1
                while j ≤ n do {if QUEEN(j, k) then Sk ← Sk ∪ {j}
                                j ← j + 1}
            }
        count ← count + sum
    }
average ← count / N // среднее число вершин в дереве

```

Оценка времени выполнения

Размер задачи	Метод Монте-Карло		Фактически		
	Число узлов	Порядок роста	Число узлов	Время	Порядок роста
8×8	—	—	2 056	1,14 мс	—
9×9	—	—	8 393	4,57 мс	в 4 раза
10×10	—	—	35 538	19,40 мс	в 4,2 раза
11×11	161 124	в 4,5 раза	166 925	92,04 мс	в 4,7 раза
12×12	825 987	в 4,9 раза	—	—	—

График, построенный по полученным экспериментальным данным, показан на рисунке (тонкой линией изображена аппроксимирующая функция). Ось абсцисс – размер задачи, ось ординат – размер дерева поиска. Наиболее близкой аппроксимирующей функцией является функция $y = 0,012e^{1,4982n} = O(c^n)$ с величиной достоверности аппроксимации $R^2 = 0,9992$. Полученные результаты подтверждают экспоненциальную вычислительную сложность решения задачи о ферзях. Что касается емкостной сложности, то она очевидна: для хранения вектора решений (a_1, \dots, a_n) требуется n ячеек памяти, для хранения вектора $S = (s_1, \dots, s_{n+1})$ требуется $n + 1$ ячеек и одна ячейка для k , т.е. всего требуется памяти $n + n + 1 + 1 = 2n + 2 = O(n)$.

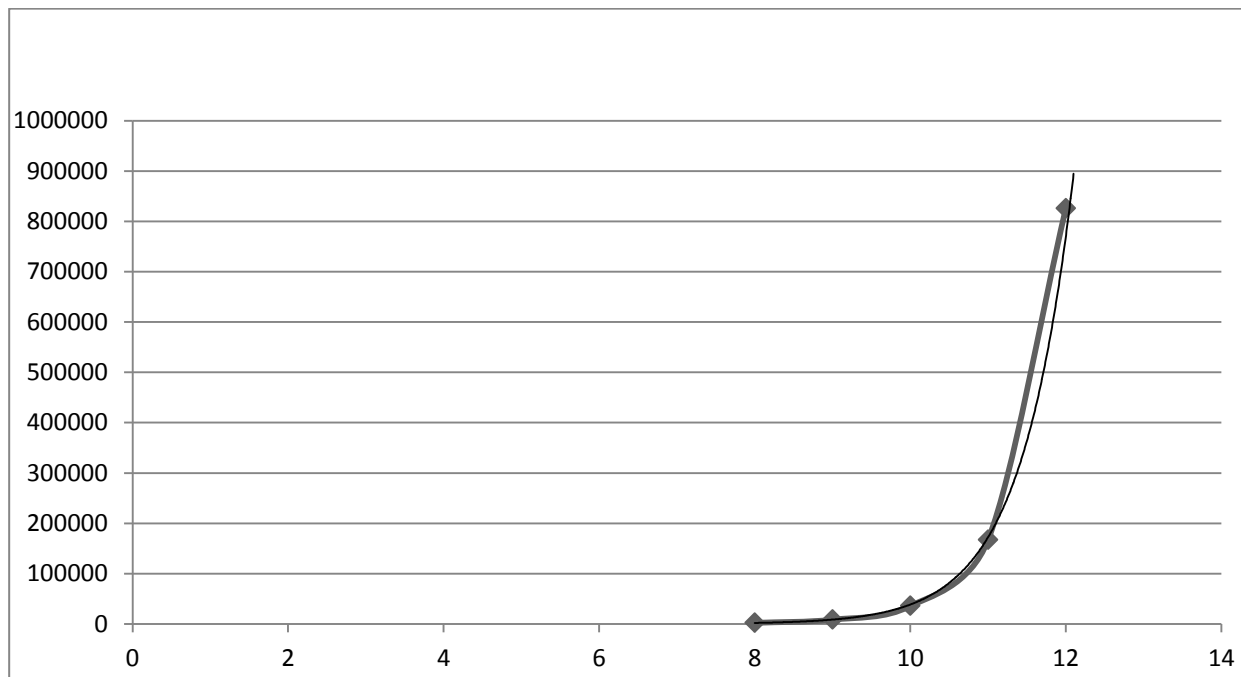


График функции вычислительной сложности