Вычислительная сложность алгоритма

Подготовил: ученик 9 класса Хайновский И.Г.

- Вычислительная сложность алгоритма Евклида изучена полностью.[17] Эта сложность может быть описана произведением количества шагов деления, требуемых алгоритмом, на вычислительную сложность одного шага. Первый известный анализ алгоритма Евклида был предложен Рейнаудом в 1811.[18] Он показал, что число шагов алгоритма для пары чисел (u, v) ограничено v. Позже он улучшил оценку до v/2 + 2. Эмиль Леже, в 1837 году, изучил наихудший случай, когда для вычисления НОД подаются последовательные числа Фибоначчи.[19] Затем, в 1841 году, Пьер Джосеф Финк показал,[20] что количество шагов алгоритма не превышает 2 log2 v + 1. Следовательно алгоритм работает за полиномиальное время от размера меньшего из пары чисел (u, v).[19] Анализ Финка был уточнён Габриэлем Ламе в 1844 году.[21] Он показал, что количество шагов, необходимых для завершения алгоритма, не более чем в пять раз превышает н — количество цифр в десятичном представлении меньшего из пары чисел (u, v).[22][23]

- Когда НОД вычисляется для чисел, которые вписываются в одно машинное слово, каждый шаг алгоритма занимает постоянное время. В данном случае анализ Ламе предполагает, что вычислительная сложность оценивается как O(н). Однако в модели расчёта, подходящей для вычислений с числами больше одного машинного слова, оценка сложности вычисления одного остатка может быть O(н2).[24] В этом случае общее время для всех этапов алгоритма можно проанализировать с помощью телескопического ряда, показав что это также O(н2). Для ускорения алгоритма Евклида могут быть использованы современные алгоритмические методы, основанные на методе Шёнхаге — Штрассена для быстрого целочисленного умножения . Это приводит к квазиполиномиальному времени.[25]

Количество шагов

- Число шагов для вычисления НОД двух натуральных чисел a и b обозначим как T(a, b). Если g — это наибольший общий делитель a и b, тогда a = mg и b = ng для двух взаимно простых чисел m и n. Тогда T(a, b) = T(m, n), что можно заметить, если разделить уравнения, полученные при вычислении НОД для пары (a, b), на g.[26] Используя тот же принцип, число шагов алгоритма остаётся неизменным, если a и b умножаются на общий множитель w, что эквивалентно равенству T(a, b) = T(wa, wb). Следовательно, количество шагов T может сильно различаться между соседними парами чисел, такими как (a, b) и (a, b+1), так как данная величина зависит от НОД.

- Рекурсивный характер алгоритма Евклида даёт следующее уравнение T(a, b) = 1 + T(b, r0) = 2 + T(r0, r1) = … = N + T(rN−2, rN−1) = N + 1, где T(x, 0) = 0 по предположению.[17]

Наихудший случай

- Если для алгоритма Евклида требуются N шагов для пары натуральных чисел a > b > 0, наименьшие значения a и b, для которых это выполняется — числа Фибоначчи FN+2 и FN+1 соответственно.[27] Тогда, если алгоритм Евклида требует N шагов для пары чисел (a,b), где a > b, выполняются следующие неравенства a ≥ FN+2 и b ≥ FN+1. Доказать это можно по математической индукции. Если N = 1, тогда a делится на b без остатка. Наименьшие натуральные числа, для которых это верно, равны b = 1 и a = 2, соответственно F2 и F3. Предположим теперь, что результат выполняется для всех значений N до M − 1. Первый шаг алгоритма Евклида с M шагами a = q0b + r0, и алгоритм Евклида для пары чисел (b,r0), где b > r0, требует M − 1 шагов. По предположению индукции имеем b ≥ FM+1 и r0 ≥ FM. Следовательно, a = q0b + r0 ≥ b + r0 ≥ FM+1 + FM = FM+2, что является искомым неравенством. Это доказательство, опубликованное Габриэлем Ламе в 1844 году, представляет собой начало теории сложности вычислений,[28] а также первое практическое применение чисел Фибоначчи.[27]

Теорема Ламе

- Число делений с остатком в процессе применения алгоритма Евклида не превосходит упятеренного количества цифр меньшего числа {\displaystyle b}b, записанного в десятичной системе.[29]

Среднее

- Существуют различные способы вычисления среднего количества шагов алгоритма. Первый способ вычисления — среднее время T(a), необходимое для вычисления НОД заданного числа a и меньшего натурального числа b, выбранного с равной вероятностью из целых чисел от 0 до a − 1.[17]

- {\displaystyle T(a)={\frac {1}{a}}\sum _{0\leq b<a}T(a,b).}{\displaystyle T(a)={\frac {1}{a}}\sum _{0\leq b<a}T(a,b).}
- Однако, поскольку T(a, b) сильно колеблется в зависимости от НОД двух чисел, усреднённая функция T(a) также является «шумной».[30] Для того, чтобы уменьшить этот шум, второе среднее τ(a) берётся по всем взаимно простым числам с a.

- {\displaystyle \tau (a)={\frac {1}{\varphi (a)}}\sum _{\begin{smallmatrix}0\leq b<a\\\gcd(a,b)=1\end{smallmatrix}}T(a,b)}{\displaystyle \tau (a)={\frac {1}{\varphi (a)}}\sum _{\begin{smallmatrix}0\leq b<a\\\gcd(a,b)=1\end{smallmatrix}}T(a,b)}
- где φ(a) функция Эйлера. Это среднее плавно растёт с ростом a.[31]
- {\displaystyle \tau (a)={\frac {12}{\pi ^{2}}}\ln 2\ln a+C+O(a^{-1/6-\varepsilon })}{\displaystyle \tau (a)={\frac {12}{\pi ^{2}}}\ln 2\ln a+C+O(a^{-1/6-\varepsilon })}
- Константа (константа Портера[32]) в этой формуле {\displaystyle C\approx 1.467}{\displaystyle C\approx 1.467}, а ε является бесконечно малым.
- Третье среднее значение Y(n) определяется как среднее число шагов, требуемых, когда a и b выбираются случайным образом (с равномерным распределением) от 1 до n.
- {\displaystyle Y(n)={\frac {1}{n^{2}}}\sum _{a=1}^{n}\sum _{b=1}^{n}T(a,b)={\frac {1}{n}}\sum _{a=1}^{n}T(a).}{\displaystyle Y(n)={\frac {1}{n^{2}}}\sum _{a=1}^{n}\sum _{b=1}^{n}T(a,b)={\frac {1}{n}}\sum _{a=1}^{n}T(a).}

Вычислительная сложность шага

- На каждом шаге алгоритма Евклида вычисляется коэффициент qk и остаток rk для заданной пары целых чисел rk−2 и rk−1. Эти величины связаны следующим соотношением:

- rk−2 = qk rk−1 + rk
- Вычислительная сложность каждого шага связана главным образом с нахождением qk, так как остаток rk можно быстро вычислить используя rk−2, rk−1, и qk

- rk = rk−2 − qk rk−1
- Вычислительная сложность операции деления чисел размером n бит оценивается как O(n(ℓ+1)), где ℓ размер частного.[24]

- Для сравнения, исходный алгоритм Евклида, с использованием вычитания, может быть намного медленнее. В большинстве случаев коэффициент qk является малым числом. Вероятность данного частного q примерно равна ln|u/(u − 1)|, где u = (q + 1)2.[33] Для иллюстрации вероятность частного 1, 2, 3 или 4 составляет примерно 41,5 %, 17,0 %, 9,3 % и 5,9 % соответственно. Так как операция вычитания быстрее, чем деление, особенно для чисел больше одного машинного слова,[34] алгоритм Евклида с использованием вычитания может быть более конкурентоспособным в сравнении с алгоритмом, использующим деление.[35] Это используется в бинарном алгоритме вычисления НОД.[36]

- Оценка сложности алгоритма вычисляется как произведение количества шагов на время выполнения одного шага. Она показывает, что алгоритм Евклида растёт квадратично O(n2), где n — среднее число цифр в двух начальных числах a и b в десятичной записи. Пусть n0, n1, …, nN−1 представляют число цифр в последовательных остатках r0, r1, …, rN−1. Так как число шагов N растёт линейно с n , время работы ограничено следующим выражением:

- {\displaystyle O{\Big (}\sum _{i<N}n_{i}(n_{i}-n_{i+1}+2){\Big)}\subseteq O{\Big (}n\sum _{i<N}(n_{i}-n_{i+1}+2){\Big)}\subseteq O(n(n_{0}+2N))\subseteq O(n^{2}).} {\displaystyle O{\Big (}\sum _{i<N}n_{i}(n_{i}-n_{i+1}+2){\Big)}\subseteq O{\Big (}n\sum _{i<N}(n_{i}-n_{i+1}+2){\Big)}\subseteq O(n(n_{0}+2N))\subseteq O(n^{2}).}