Функция Аккермана

Функция Аккермана — простой пример всюду определённой вычислимой функции, которая не является примитивно рекурсивной. Она принимает два неотрицательных целых числа в качестве параметров и возвращает натуральное число, обозначается $A(m,\ n)$. Эта функция растёт очень быстро, например, число $A(4,\ 4)$ настолько велико, что количество цифр в порядке этого числа многократно превосходит количество атомов в наблюдаемой части Вселенной.

1. История

В конце 20-х годов XX века математики-ученики Давида Гильберта — Габриэль Судан и Вильгельм Аккерман изучали основы вычислений. Судан и Аккерман известны[1] за открытие всюду определённой вычислимой функции (иногда называемой просто «рекурсивной»), не являющейся примитивно рекурсивной. Судан открыл менее известную функцию Судана, после чего, независимо от него, в 1928 Аккерман опубликовал свою функцию φ . Функция трёх аргументов Аккермана $\varphi(m, n, p)$ определялась так, что для $p = 0, 1, 2$, она проводила операцию сложения, умножения или возведения в степень:

$$\varphi(m, n, 0) = m + n,$$

$$\varphi(m, n, 1) = m \cdot n,$$

$$\varphi(m, n, 2) = m^n,$$

а для $p > 2$ она доопределяется с помощью стрелочной нотации Кнута, опубликованной в 1976 году:

$$\varphi(m, n, p) = m \uparrow^{p-1} (n + 1)$$

(Помимо её исторической роли как первой всюду определённой не примитивно рекурсивной вычислимой функции, оригинальная функция Аккермана расширяла основные арифметические операции за возведение в степень, хотя и не так хорошо, как специально предназначенные для этого функции вроде последовательности гипероператоров Гудстейна.)

В статье «On the infinite» Гильберт высказал гипотезу о том, что функция Аккермана не является примитивно рекурсивной, а Аккерман (личный секретарь и бывший студент Гильберта) доказал эту гипотезу в статье «On hilbert's construction of the real numbers». Роза Петер и Рафаэль Робинсон позже представили двухаргументную версию функции Аккермана, которую теперь многие авторы предпочитают оригинальной[2].

2. Определение

Функция Аккермана определяется рекурсивно для неотрицательных целых чисел m и n следующим образом:

$$A(m,\ n) = \begin{cases} n+1, & m = 0; \\ A(m-1,\ 1), & m > 0,\ n = 0; \\ A(m-1,\ A(m,\ n-1)), & m > 0,\ n > 0. \end{cases}$$

Может показаться неочевидным, что рекурсия всегда заканчивается. Это следует из того, что при рекурсивном вызове или уменьшается значение m , или значение m сохраняется, но уменьшается значение n . Это означает, что каждый раз пара $(m,\ n)$ уменьшается с точки зрения лексикографического порядка, значит, значение m в итоге достигнет нуля: для одного значения m существует конечное число возможных значений n (так как первый вызов с данным m был произведён с каким-то определённым значением n , а в дальнейшем, если значение m сохраняется, значение n может только уменьшаться), а количество возможных значений m , в свою очередь, тоже конечно. Однако, при уменьшении m значение, на которое увеличивается n , неограниченно и обычно очень велико.

3. Таблица значений

Подробнее о hyper см. гипероператор.

4. Обратная функция

Так как функция $f(n) = A(n,\ n)$ растёт очень быстро, обратная функция $f^{-1}(n) = \min\{k \geqslant 1 : A(k,\ k) \geqslant n\}$, обозначаемая α , растёт очень медленно. Эта функция встречается при исследовании сложности некоторых алгоритмов, например, системы непересекающихся множеств или алгоритма

Чазелла для построения минимального остовного дерева[3]. При анализе асимптотики можно считать, что для всех практически встречающихся чисел значение этой функции меньше пяти, так как A(4, 4) одного порядка с $2^{2^{10^{19729}}}$.

5. Использование в тестах производительности

Функция Аккермана в силу своего определения имеет очень глубокий уровень рекурсии, что можно использовать для проверки способности компилятора оптимизировать рекурсию. Первым функцию Аккермана для этого использовал Ингви Сандблад[4].

Брайан Вичман (соавтор теста производительности Whetstone) учёл эту статью при написании серии статей о тестировании оптимизации вызовов[5][6][7].

Например, компилятор, который анализируя вычисление $A(3, 30)$ способен сохранять промежуточные значения, например, $A(3, n)$ и $A(2, n)$, может ускорить вычисление $A(3, 30)$ в сотни и тысячи раз. Также вычисление $A(2, n)$ напрямую вместо рекурсивного раскрытия в $A(1, A(1, A(1, \ldots A(1, 0)\ldots)))$ прилично ускорит вычисление. Вычисление $A(1, n)$ занимает линейное время n. Вычисление $A(2, n)$ требует квадратичное время, так как оно раскрывается в O(n) вложенных вызовов $A(1, i)$ для различных i. Время вычисления $A(3, n)$ пропорционально 4^{n+1}.

Значение $A(4, n)$ невозможно посчитать с помощью простого рекурсивного применения за разумное время. Вместо этого для оптимизации рекурсивных вызовов используются сокращённые формулы, например, $A(3, n) = 8\times2^n-3$.

Один из практичных способов вычисления значений функции Аккермана — мемоизация промежуточных результатов. Компилятор может применить эту технику к функции автоматически, используя memo functions[8][9] Дональда Мичи.

6. Примечания

[1] Cristian Calude, Solomon Marcus and Ionel Tevy (November 1979). «The first example of a recursive function which is not primitive recursive». *Historia Math.* **6** (4): 380–84. DOI:10.1016/0315-0860(79)90024-7.

[2] Raphael M. Robinson (1948). «Recursion and double recursion». *Bulletin of the American mathematical society* **54** (10): 987-993. DOI:10.1090/S0002-9904-1948-09121-2.

[3] Дискретная математика: алгоритмы. Минимальные остовные деревья

[4] Y. Sundblad (1971). «The Ackermann function. A theoretical, computational and formula manipulative study». *BIT* **11**: 107–119. DOI:10.1007/BF01935330.

[5] Ackermann's function: A study in the efficiency of calling procedures (1975). Архивировано из первоисточника 24 февраля 2012.

[6] How to call procedures, or second thoughts on Ackermann's function (1977). Архивировано из первоисточника 24 февраля 2012.

[7] Latest results from the procedure calling test. Ackermann's function (1982). Архивировано из первоисточника 24 февраля 2012.

[8] Michie, Donald, “Memo functions and machine learning, " *Nature*, No. 218, pp. 19-22, 1968.

[9] Example: explicit memo function version of Ackermann's function implemented in PL/SQL.

7. Ссылки

- *Weisstein, Eric W.* Ackermann Function (англ.) на сайте Wolfram MathWorld.

8. Источники текстов и изображения, авторы и лицензии

8.1. Текст

- **Функция Аккермана** *Источник:* https://ru.wikipedia.org/wiki/%D0%A4%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D1%8F_%D0%90%D0%BA%D0%BA%D0%B5%D1%80%D0%BC%D0%B0%D0%BD%D0%B0?oldid=81530127 *Авторы:* A5b, Nikolay Pogorskiy, Infovarius, Escarbot, Thijs!bot, Alexei Kopylov, Nyq, Alex Smotrov, KleverI, Evgeny Kapun, TXiKiBoT, РоманСузи, SieBot, AlleborgoBot, VVVBot, OKBot, Njarlatotep, Petrov Victor, Salak, VlsergeyBot, Nikita27, Luckas-bot, Rubinbot, X7q, Softwayer, Convallaria majalis, Mvk608, EmausBot, W2, KrBot, WebCite Archiver, MBHbot, Ikosarakt, Addbot, NapalmBot и Аноним: 23

8.2. Изображения

- **Файл:No_iwiki_template.svg** *Источник:* https://upload.wikimedia.org/wikipedia/commons/4/45/No_iwiki_template.svg *Лицензия:* CC BY-SA 3.0 *Авторы:* Wikipedia-logo-v2-en.svg: *Художник:* Urutseg, Amit6, Юкатан, Ain92
- **Файл:Wikipedia_interwiki_section_gear_icon.svg** *Источник:* https://upload.wikimedia.org/wikipedia/commons/c/c9/Wikipedia_interwiki_section_gear_icon.svg *Лицензия:* Public domain *Авторы:* Universal Language Selector extension of MediaWiki, https://github.com/wikimedia/mediawiki-extensions-UniversalLanguageSelector/blob/master/resources/images/cog-sprite.svg *Художник:* Wikimedia Foundation; Santhosh Thottingal <santhosh.thottingal@gmail.com> (according to File:Cog-ULS-gear-latest.png);

8.3. Лицензия

-