АЛГОРИТМИ.
ОЦЕНКА И СЛОЖНОСТ НА АЛГОРИТМИТЕ

ВТУ, 09.10.2008 г.

АЛГОРИТМИ – ОСНОВНИ ПОНЯТИЯ

Думата 'алгоритъм' идва от името на арабския математик al-Khwarezmi, който през VIII в. описва откриването на десетичната бройна система и представя редица важни концепции в книгата си Al-jabr wa'l muqabala.

Алгоритъм - крайна последователност от действия (стъпки) за решаване на конкретен проблеми или даден вид проблеми.

Точното математическо определение свежда понятието 'алгоритъм' до добре дефинирано въображаемо изчислително устройство, например *машината на Тюринг*.

Съвременно понятие за алгоритъм – „ясно и точно предписание за изпълнение на последователност от елементарни операции, с цел решаването на клас еднотипни задачи" (акад.Макаров)

Алгоритъм в програмирането – пълно, точно и еднозначно предписание за изпълнение на даден клас задачи при входни данни, които могат да се променят в определени граници.

РАЗВИТИЕ НА АЛГОРИТМИТЕ

„Историческо“ развитие – условно се разделя на 3 етапа:

- Намиране на начин за символно представяне на информацията под формата на данни (например видовете числа);
- Формулиране на алгоритми, използващи данните за решаване на изчислителни задачи;
- Създаване на механични изчислителни машини, които могат да изпълняват машините ефективно.

Алгоритмите през вековете:

- ок.300 г. пр.н.е. – алгоритъм на Евклид;
- ок.250 г. пр.н.е. – решето на Ератостен;
- ок.800 г. – ал-Хорезми и книгата му „Алгебрата и уравненията“
- 1424 г. – Масуд ал-Каши изчислява π с точност до 16-я знак след запетаята.

АНАЛИЗ НА АЛГОРИТМИТЕ

- 1845 г. – *Габриел Лейм* показва, че алгоритъмът на Евклид извършва брой деления, не повече от 5 пъти броя на десетичните цифри на по-малкото число;

- 1900 г. – *Дейвид Хилберт* поставя въпроса за намирането на процедура за решаване на диофантови уравнения;

- 1910 г. – *Поклингтън* дефинира сложност на алгоритъм като полином, зависещ от броя на цифрите в двоичното кодиране на данните;

- 1920-1930 г. – *Пост* предлага прост унарен модел за изчисления, известен като машина на Пост;

- 1930 г. – *Алонсо Чърч* представя ламбда–смятането;

- 1936 г. – *Алън Тюринг* публикува статия, където представя машината на Тюринг (формален, но физически осъществим модел за изчислимост).

СВОЙСТВА НА АЛГОРИТМИТЕ

- Еднозначност;
- Детерминираност (определеност);
- Крайност;
- Резултатност;
- Масовост;
- Дискретност.

КОМПЮТЪРНИ АЛГОРИТМИ

Три основни свойства:

- Простота и елегантност;
- Коректност;
- Бързодействие;

Докато първото от тях може да се оцени интуитивно, за другите две е нужен много по-задълбочен анализ.

Пример:

```
1) n = 100;
2) sum = 0;
3) for (int i=0; i<n; i++)
4)      for (int j=0; j<n; j++)
5)           sum++;
```

АНАЛИЗ НА ПРИМЕРА

Редове 1) и 2): $n = 100; sum = 0;$ – статична инициализация (константно време a)

Редове 3) и 4): $(i = 0; i < n; i++)$ и $(j = 0; j < n; j++)$ – константни времена b, c и d (съответно e, f и g)

Ред 5): константно време h

За прозволно n имаме:
$a + b + n.c + n.d + n.(e + n.f + n.g + n.h) = a + b + n.c + n.d + n.e + n.n.f + n.n.g + n.n.h = n^2.(f + g + h) + n.(c + d + e) + a + b = \underline{i.n^2 + j.n + k}$

Нека имаме 2 функции $F(n)$ и $G(n)$ (време за изпълнение на два алгоритъма A_1 и A_2):
$F(n) = 2.n^2$ – квадратична; $G(n) = 200.n$ – линейна

За достатъчно големи n (в случая $n > 100$) имаме $F(n) > G(n)$.

АСИМПТОТИЧНИ НОТАЦИИ

$\boldsymbol{O}$–нотация:

$O(g(n)) = \{f(n)$: съществуват цели константи c и n_0 такива, че $0 \leq f(n) \leq c.g(n),\ \forall\ n \geq n_0\}$;

Ω–нотация:

$\Omega(g(n)) = \{f(n)$: съществуват цели константи c и n_0 такива, че $0 \leq c.g(n) \leq f(n),\ \forall\ n \geq n_0\}$;

Θ–нотация:

$\Theta(g(n)) = \{f(n)$: съществуват цели константи c_1, c_2, и n_0 такива, че $0 \leq c_1.g(n) \leq f(n) \leq c_2.g(n),\ \forall\ n \geq n_0\}$.

$O(F)$ е най-често използвана при оценка на сложност на алгоритми и програми.

ОСНОВНИ АСИМПТОТИЧНИ ФУНКЦИИ

- Сложност $O(1)$ – константна
- Сложност $O(\log_2 n)$ – логаритмична
- Сложност $O(n)$ – линейна
- Сложност $O(n^2)$ – квадратична
- Сложност $O(c^n)$ – експоненциална

Когато функцията F е полином, сложността $O(F)$ наричаме *полиномна*

Изпълнено е:
$O(1) \subset O(\log_2 n) \subset O(n) \subset O(n^2) \subset \ldots \subset O(n^k) \subset O(c^n) \subset O(n!) \subset O(n^n)$

От примера: $i.n^2 + j.n + k = O(n^2) + O(n) + O(1) = O(n^2)$

ВКЛЮЧВАНЕ В КЛАС НА СЛОЖНОСТ

Положителни примери	*Отрицателни примери*
$10n \in O(n)$	$10n \notin O(1)$
$10n \in O(n^2)$	$10n^2 \notin O(n)$
$10n + 3 \in O(n)$	$10n^2 + 3n \notin O(n)$
$4n^2 - 5n + 2 \in O(n^2)$	$4n^3 - 5n + 2 \notin O(n^2)$
$\sqrt{n} \in O(n)$	$5n + 1 \notin O(\sqrt{n})$
$\log_2 n \in O(\sqrt{n})$	$\sqrt{n} \notin O(\log_2 n)$
$n^{100} + 1000n^{99} \in O(2^n)$	$(\frac{3}{2})^n \notin O(n^{1000})$
$2^n \in O(3^n)$	$n! \notin O(4^n)$
$n! + 100^n \in O(n!)$	$n^n \notin O(n!)$

ОПРЕДЕЛЯНЕ СЛОЖНОСТ НА АЛГОРИТЪМ

- Елементарен оператор – сложност $O(1)$;
- Последователност от оператори – определя се от сложността на най-бавния от тях;
- Вложени оператори – сложността се пресмята като произведение на отделните сложности;
- Оператор *if* – $if(P)\ S_1; else\ S_2$; сложността е $\max\{O(P), O(S_1), O(S_2)\}$;
- Цикъл – сложност $O(n)$;
- Вложени цикли – сложността зависи от включените в циклите оператори;
- Рекурсия – to be continue.....

ПРИМЕРИ С ВЛОЖЕНИ ЦИКЛИ

Пример 1:

```
unsigned sum = 0;
for (int i = 0; i < n; i++)
    for (int j = 0; j < n; j++)
        sum++;
```

Сложност $O(n.n) = O(n^2)$

Пример 2:

```
unsigned sum = 0;
for (int i = 0; i < n-1; i++)
    for (int j = i; j < n; j++)
        sum++;
```

Сложност $O(\frac{(n.n-1)}{2}) = O(n^2)$

ПРИМЕРИ С ВЛОЖЕНИ ЦИКЛИ

Пример 3:

```
unsigned sum = 0;
for (int i = 0; i < n*n; i++)
    sum++;
```

Сложност $O(n^2)$

Пример 4:

```
unsigned sum = 0;
for (int i = 0; i < n; i++)
    for (int j = 0; j < n; j++)
        if (i == j)
            for (k = 0; k < n; k++)
                sum++;
```

Сложност $O(???)$