CZAS OBLICZEŃ

Marek Szepski
Krakowska Akademia

Czas

Czas jest ważny. Program, który wykonuje się wolno, wzbudza naszą irytację. Często nie chcemy z takiego programu korzystać. Gdy jest to program komercyjny, to przeliczamy stracony czas na pieniądze.

Ważne jest, gdy piszemy program, aby mieć świadomość, jak przyjęte przez nas rozwiązania wpłyną na szybkość jego działania.

Tematy

- Biblioteka ctime
- Czas obliczeń podstawowych działań i typów zmiennych
- Czas działania algorytmu wykładniczego
- $O(n^2)$ i określenie współczynnika proporcjonalności

<ctime>

Jeśli chcemy mierzyć czas działania programu musimy skorzystać z odpowiedniej biblioteki, na przykład: `<ctime>`.

Biblioteka zawiera szereg elementów, ale do określenia czasu obliczeń potrzebne będzie tylko kilka z nich:

-typ zmiennych do przechowywania czasu systemowego

- funkcja pobierająca czas z systemu

-stała, pozwalająca wyrazić czas systemowy w sekundach

clock_t

W bibliotece `<ctime>` jest zdefiniowany specjalny typ danych clock_t, przeznaczony do zapisywania czasu systemowego. Aby określić czas działania potrzebujemy znać czas na początku i czas po zakończeniu działania programu lub jego części.

Będziemy więc mieć deklarację:

```
clock_t czas1, czas2;
```

clock()

Funkcja odczytująca czas systemowy nazywa się `clock()`. Zwraca ona wartość, która jest liczbą impulsów zegara systemowego, liczoną od uruchomienia komputera. Analizowany fragment programu, którego czas działania nas interesuje będzie zawarty pomiędzy instrukcjami:

```
czas1 = clock();
```

oraz

```
czas2= clock();
```

Czas będzie równy: `czas2-czas1`

CLOCKS_PER_SEC

Obliczony czas będzie wyrażony w liczbie impulsów zegara systemowego. Chcąc przeliczyć go na sekundy trzeba podzielić go przez stałą:
CLOCKS_PER_SEC.

Jest to stała systemowa (określona przez kompilator), może mieć różną wartość w różnych systemach i określa ile jednostek systemowych przypada na jedną sekundę.

Zadanie 1

Jaką wartość ma stała CLOCKS_PER_SEC w Twoim komputerze.

Wskazówka:
Wystarczy wypisać na ekranie wartość tej stałej.

Podaj otrzymaną wartość.

Sekundy

Aby otrzymać czas w sekundach wystarczy różnicę czasu systemowego podzielić przez stałą, czyli:

```
(czas2 - czas1) / CLOCKS_PER_SEC
```

Niestety taki zapis prowadzi do nieścisłości – jest to liczba całkowita, a więc czas zostanie zaokrąglony do pełnych sekund. Taka dokładność jest niewystarczająca. Chcemy mieć czas mierzony w ułamkach sekund. Musimy sprawić aby obliczenia i wynik były prowadzone na liczbach zmiennopozycyjnych.

..cd

Możemy do tego użyć operatora rzutowania:

```
(double)(czas2-czas1)/CLOCKS_PER_SEC
```

lub „trochę oszukać” kompilator każąc mu liczyć na liczbach ułamkowych:

```
1.0*(czas2 - czas1) / CLOCKS_PER_SEC
```

Problem 1

Zwykle, pisząc program nie zwracamy uwagi na to jak poszczególne instrukcje wpływają na czas jego działania. Ale, gdy w programie pewne działania trzeba wykonać miliony razy, warto wybrać szybsze rozwiązanie. Musimy porównać ze sobą czasy działania instrukcji, których efekt działania jest analogiczny ale różny sposób wykonania.

...

Podstawą analizy będzie wykonanie wielokrotne pętli. Kod może wyglądać tak:

```
int LIter = 1e8;
clock_t czas1 = clock();
for (i=0; i<LIter; i++){
continue; //tu będziemy wstawiać różne instrukcje
}
clock_t czas2 = clock();
```

Zadanie 2

Czas zależy od liczby iteracji czyli zmiennej LIter.

Dobierz wartość zmiennej LIter tak aby czas wykonania pustej pętli wynosił ok. 1 sekundy.

Jaka jest wartość LIter w Twoim komputerze. Podaj tę wartość.

Problem 1 - cd

Wstawiając w pętlę odpowiednie instrukcje możemy porównać je ze sobą. Interesujące może być czy zadeklarowany typ zmiennych ma wpływ na czas działania. Można ze sobą porównać int, long long, short, double, float itp..

Kolejny problem to czy działania różnią się między sobą, tzn. czy mnożenie zajmuje tyle samo czasu ile wymaga dodawanie, czy czas zależy jak dużą liczbę dodajemy lub mnożymy albo dzielimy, czy lepiej użyć k+2*i czy k=i+i, czy lepiej x=x/2 czy x=x*0.5 itd. Takich pytań można postawić wiele.

Zadanie 3

Sformułuj 5 pytań tego typu i znajdź na nie odpowiedzi.

Podaj postać instrukcji, które wykonywane były w pętli i otrzymane czasy działania.

Sformułuj wnioski.

Problem 2
Algorytm wykładniczy

Na ćwiczeniach analizowaliśmy ciąg Fibonacciego. Przypomnijmy, rekurencyjna funkcja obliczająca wartość poszczególnych wyrazów tego ciągu wyglądała następująco:

```
int fib(int k){
  if(k<3) return 1;
  else
  return fib(k-1) + fib(k-2);
}
```

...

Zaobserwowaliśmy 2 zjawiska:

1. Ciąg bardzo szybko rośnie i szybko osiągaliśmy zakres zmiennej typu int. Przekroczenie go powodowało błędne wyniki.
2. Czas obliczeń dla większych wyrazów ciągu był wyraźnie coraz dłuższy. Analiza liczby wywołań skłaniała nas do wniosku, że rośnie on wykładniczo.

Zadanie 4

Oblicz kolejne wyrazy ciągu Fibonacciego i znajdź czas potrzebny na ich obliczenie.
Uwaga: czasy dla wyrazów mniejszych niż ok. 30 mogą być bardzo małe (komputer poda 0) i takie wyniki możemy pominąć. Pozostałe wyniki poddaj analizie.

Otrzymane wyniki zachowaj w tablicy i oblicz stosunek kolejnych wyrazów ciągu względem siebie, tzn. oblicz fib(n) / fib(n-1).

Taki sam stosunek określ dla czasów obliczeń kolejnych wyrazów ciągu.

Zadanie 5

A. Stosunek wartości wyrazów ciągu powinien być prawie stały i wynosić ok. 1.62.
Odpowiedz na pytanie co ciąg Fibonacciego ma wspólnego ze „Złotym podziałem” i dlaczego.

B. Stosunek czasów obliczeń też oscyluje wokół pewnej wartości. Określ jego średnią wartość.

Na tej podstawie sformułuj hipotezę na temat złożoności algorytmy, określ O( ).

Zadanie 5

A. Stosunek wartości wyrazów ciągu powinien być prawie stały i wynosić ok. 1.62.
Odpowiedz na pytanie co ciąg Fibonacciego ma wspólnego ze „Złotym podziałem” i dlaczego.

B. Stosunek czasów obliczeń też oscyluje wokół pewnej wartości. Określ jego średnią wartość.

Na tej podstawie sformułuj hipotezę na temat złożoności algorytmy, określ O( ).

Problem 3
Algorytm $O(n^2)$

Na zajęciach z algorytmów analizowaliśmy algorytmy sortowania o złożoności kwadratowej $O(n^2)$. Było to np. „Proste wybieranie” lub „Proste wstawianie”.

Złożoność kwadratowa oznacza, że czas obliczeń jest proporcjonalny do kwadratu liczby sortowanych danych.

Zadanie 6

Określ współczynnik proporcjonalności, który jest pomijany w notacji O().

Wskazówka.
Określ czasy sortowania dla różnych n np.. n=10 000, 20 000, … 50 000.

Jeżeli algorytm jest kwadratowy to czas ten powinien wynosić t = c * n * n.

Określ c. Z różnych powodów czas trochę się waha. Uśrednij wartość c. Czy możesz uznać, że algorytm jest kwadratowy? Podaj wartość c.

Zadanie 7

Czas zależy od komputera. Podaj dane komputera, na którym prowadzone były obliczenia – stacjonarny/laptop, procesor typ i zegar, pamięć.

Sprawozdanie

Sprawozdanie powinno być jednym plikiem tekstowym i zawierać odpowiedzi na pytania i problemy oraz kody napisanych programów. Zadbaj o zwartą formę sprawozdania. Prześlij plik na adres:
mszepski@afm.edu.pl

Plik ma mieć nazwę: **Nazwisko-Prog10.xdoc** (lub .doc)

TERMIN: 26 stycznia 2021 (studia stacjonarne)

TERMIN: 6 lutego 2021 (studia niestacjonarne)