Wojciech Myszka

Laboratorium 3: Algorytm B

2021-03-12 20:43:20 +0100

1. Wprowadzenie

Pozostajemy przy algorytmie Euklidesa. Tym razem rozważana będzie wersja „binarna" tego algorytmu.

Nazwa binarna związana jest z tym, że jedyne operacje, które są tu wykonywane to:
— dzielenie przez dwa (liczb parzystych!),
— odejmowanie,
— sprawdzanie czy liczba jest parzysta,
— podnoszenie dwójki do całkowitej potęgi.
Wszystkie te operacje bardzo łatwo wykonać na komputerze dwójkowym używając (standardowo dostępnych) zupełnie podstawowych poleceń. Dzięki temu algorytm może być wykonywany bardzo szybko.

Niestety, okupione jest to nieco większym jego skomplikowaniem (zagmatawniem).

1.1. Problemy

Zasadniczym problemem związanym z tym algorytmem jest to, że w wyrysowanym schemacie blokowym, dające się zauważyć pętle „zachodzą na siebie", a linie krzyżują się, Sam schemat buduje się dosyć łatwo i warto podczas jego budowy zaznaczać kolejne kroki (1–6). Wówczas polecenia przejd do kroku 3 albo przejdź do kroku 4 będą łatwe do zrealizowania (za pomocą instrukcji **goto**.

Polecenie **goto** nie powinno być nadużywane (bo to nieładnie). Działa jakoś tak:

```
k3:   // To jest tak zwana etykieta związana
      // z miejscem "krok 3"
      ... // tu kolejne polecenia
```

```
goto k3; // Wykonanie tego polecenia spowoduje
         // że jako następne polecenie wykonane
         // będzie pierwsze polecenie po k3:
```

1.2. Zadania do wykonania

W poniższym algorytmie u i v to dane wejścowe. t i k to zmienne pomocnicze. Wszystkie typu całkowitego.

— Narysować schemat blokowy algorytmu B (bardzo ważne! bez tego będzie trudno):

1. Przyjmij $k \leftarrow 0$, a następnie powtarzaj operacje: $k \leftarrow k + 1$, $u \leftarrow u/2$, $v \leftarrow v/2$ zero lub więcej razy, do chwili gdy przynajmniej jedna z liczb u i v przestanie być parzysta.
2. Jeśli u jest nieparzyste to przyjmij $t \leftarrow -v$ i przejdź do kroku 4. W przeciwnym razie przyjmij $t \leftarrow u$.
3. (W tym miejscu t jest parzyste i różne od zera). Przyjmij $t \leftarrow t/2$.
4. Jeśli t jest parzyste to przejdź do kroku 3.
5. Jeśli $t > 0$, to przyjmij $u \leftarrow t$, w przeciwnym razie przyjmij $v \leftarrow -t$.
6. Przyjmij $t \leftarrow u - v$. Jeśli $t \neq 0$ to wróć do kroku 3. W przeciwnym razie algorytm zatrzymuje się z wynikiem $u \cdot 2^k$.

Uwaga: $t \neq 0$ oznacza różne!

— W jaki sposób zrealizować binarne operacje:
  — sprawdzenia parzystości liczby?
  — dzieleni aliczby przez 2?
  — podniesienia liczby 2 do potęgi całkowitej?

— Czy widzicie Państwo jakieś podobieństwa tego algorytmu do „klasycznego” algorytmu Euklidesa (w wersji z resztą lub odejmowaniem)?

— Zaprogramować algorytm B. W najgorszym wypadku można się wesprzeć instrukcją **goto** (krok opcjonalny)

— Zaprogramować go **bez** korzystania z instrukcji **goto**, używając wyłącznie instrukcji **if**, **while** i ewentualnie **do while**.

2. Wersja PDF tego dokumentu...

...pod adresem.

Wersja: 58 z drobnymi modyfikacjami! data ostatniej modyfikacji 2021-03-12 20:43:20 +0100