Zaawansowane algorytmy

Wstęp 1/2

dr Janusz Dybizbański

1. Założenia modelu PRAM

1. Założenia modelu PRAM

Model PRAM (Parallel Random Access Machine)

- p - liczba procesorów,

1. Założenia modelu PRAM

Model PRAM (Parallel Random Access Machine)

- p - liczba procesorów,
- każdy procesor ma swoją (małą) pamięć lokalną,

1. Założenia modelu PRAM

Model PRAM (Parallel Random Access Machine)

- p - liczba procesorów,
- każdy procesor ma swoją (małą) pamięć lokalną,
- każdy procesor zna swój unikalny numer identyfikacyjny,

1. Założenia modelu PRAM

Model PRAM (Parallel Random Access Machine)

- p - liczba procesorów,
- każdy procesor ma swoją (małą) pamięć lokalną,
- każdy procesor zna swój unikalny numer identyfikacyjny,
- każdy procesor zna ogólną liczbę procesorów,

1. Założenia modelu PRAM

Model PRAM (Parallel Random Access Machine)

- p - liczba procesorów,
- każdy procesor ma swoją (małą) pamięć lokalną,
- każdy procesor zna swój unikalny numer identyfikacyjny,
- każdy procesor zna ogólną liczbę procesorów,
- każdy procesor zna rozmiar danych wejściowych

1. Założenia modelu PRAM

Model PRAM (Parallel Random Access Machine)

- p - liczba procesorów,
- każdy procesor ma swoją (małą) pamięć lokalną,
- każdy procesor zna swój unikalny numer identyfikacyjny,
- każdy procesor zna ogólną liczbę procesorów,
- każdy procesor zna rozmiar danych wejściowych
- wspólna pamięć (służy między innymi do komunikacji),

1. Założenia modelu PRAM

Model PRAM (Parallel Random Access Machine)

- p - liczba procesorów,
- każdy procesor ma swoją (małą) pamięć lokalną,
- każdy procesor zna swój unikalny numer identyfikacyjny,
- każdy procesor zna ogólną liczbę procesorów,
- każdy procesor zna rozmiar danych wejściowych
- wspólna pamięć (służy między innymi do komunikacji),
- obliczenia są synchroniczne (wszystkie procesory w tym samym czasie wykonują tą samą instrukcję)

2. Dwa sposoby zapisu algorytmów równoległych

Algorytm 1. Koniunkcja elementów tablicy

wejście:

- tablica zmiennych logicznych $A[1...p]$ umieszczona w pamięci globalnej

Algorytm 1. Koniunkcja elementów tablicy

wejście:

- tablica zmiennych logicznych $A[1...p]$ umieszczona w pamięci globalnej

wyjście:

- zmienna logiczna C umieszczona w pamięci globalnej taka, że $C = \bigwedge_{i=1}^{p} A[i]$

Z perspektywy jednego procesora

Program dla procesora o identyfikatorze i:

```
1 if i=1 then
2     global_write(true, C)
3 global_read(A[i], a)
4 if not a then
5     global_write(false, C)
```

Z perspektywy jednego procesora

Program dla procesora o identyfikatorze i:

```
1 if i=1 then
2     global_write(true, C)
3 global_read(A[i], a)
4 if not a then
5     global_write(false, C)
```

Dla wszystkich procesorów jednocześnie

```
1 if i=1 then
2     global_write(true, C)
3 for 1 <= i <= p pardo
4     global_read(A[i], a)
5     if not a then
6         global_write(false, C)
```

Działanie algorytmu

id	1	2	3	4	5	6	7	8
A	T	T	T	F	F	T	F	T

Działanie algorytmu

id A	1 T	2 T	3 T	4 F	5 F	6 T	7 F	8 T
krok 1	C := T	∅	∅	∅	∅	∅	∅	∅

Działanie algorytmu

id A	1 T	2 T	3 T	4 F	5 F	6 T	7 F	8 T
krok 1	C := T	∅	∅	∅	∅	∅	∅	∅
krok 2	a := T	a := T	a := T	a := F	a := F	a := T	a := F	a := T

Działanie algorytmu

id A	1 T	2 T	3 T	4 F	5 F	6 T	7 F	8 T
krok 1	C := T	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$
krok 2	a := T	a := T	a := T	a := F	a := F	a := T	a := F	a := T
krok 3	nie	nie	nie	tak	tak	nie	tak	nie

Działanie algorytmu

id A	1 T	2 T	3 T	4 F	5 F	6 T	7 F	8 T
krok 1	C := T	∅	∅	∅	∅	∅	∅	∅
krok 2	a := T	a := T	a := T	a := F	a := F	a := T	a := F	a := T
krok 3	nie	nie	nie	tak	tak	nie	tak	nie
krok 4	∅	∅	∅	C := F	C := F	∅	C := F	∅

Konflikt - jednoczesny odczyt/zapis z/do pamięci globalnej przez więcej niż jeden procesor

3. Podmodele modelu PRAM

Konflikt - jednoczesny odczyt/zapis z/do pamięci globalnej przez więcej niż jeden procesor

Podział na modele

Concurrent/**E**xclusive **R**ead **C**oncurrent/**E**xclusive **W**rite

- EREW PRAM - model nie dopuszcza żadnych konfliktów
- CREW PRAM - model doupszcza konflikty odczytu
- ERCW PRAM - model dopuszcza konflikty zapisu
- CRCW PRAM - model doupszcza konflikty odczytu i zapisu.

- EREW PRAM - model nie dopuszcza żadnych konfliktów
- CREW PRAM - model doupszcza konflikty odczytu
- ERCW PRAM - model dopuszcza konflikty zapisu
- CRCW PRAM - model doupszcza konflikty odczytu i zapisu.

Podział ze względu na sposób rozstrzygania konfliktów zapisu

- EREW PRAM - model nie dopuszcza żadnych konfliktów
- CREW PRAM - model doupszcza konflikty odczytu
- ERCW PRAM - model dopuszcza konflikty zapisu
- CRCW PRAM - model doupszcza konflikty odczytu i zapisu.

Podział ze względu na sposób rozstrzygania konfliktów zapisu

- **common** - zezwala na jednoczesny zapis tylko, gdy wszystkie procesory chcą zapisać tę samą wartość

- EREW PRAM - model nie dopuszcza żadnych konfliktów
- CREW PRAM - model doupszcza konflikty odczytu
- ERCW PRAM - model dopuszcza konflikty zapisu
- CRCW PRAM - model doupszcza konflikty odczytu i zapisu.

Podział ze względu na sposób rozstrzygania konfliktów zapisu

- **common** - zezwala na jednoczesny zapis tylko, gdy wszystkie procesory chcą zapisać tę samą wartość
- arbitrary - zapisuje jeden procesor (arbitralnie wybrany)

- EREW PRAM - model nie dopuszcza żadnych konfliktów
- CREW PRAM - model doupszcza konflikty odczytu
- ERCW PRAM - model dopuszcza konflikty zapisu
- CRCW PRAM - model doupszcza konflikty odczytu i zapisu.

Podział ze względu na sposób rozstrzygania konfliktów zapisu

- **common** - zezwala na jednoczesny zapis tylko, gdy wszystkie procesory chcą zapisać tę samą wartość
- arbitrary - zapisuje jeden procesor (arbitralnie wybrany)
- priority - zapisuje ten o największym priorytecie (np. najmniejszy identyfikator)

- EREW PRAM - model nie dopuszcza żadnych konfliktów
- CREW PRAM - model doupszcza konflikty odczytu
- ERCW PRAM - model dopuszcza konflikty zapisu
- CRCW PRAM - model doupszcza konflikty odczytu i zapisu.

Podział ze względu na sposób rozstrzygania konfliktów zapisu

- **common** - zezwala na jednoczesny zapis tylko, gdy wszystkie procesory chcą zapisać tę samą wartość
- arbitrary - zapisuje jeden procesor (arbitralnie wybrany)
- priority - zapisuje ten o największym priorytecie (np. najmniejszy identyfikator)

W jakim modelu działa Algorytm 1?

- EREW PRAM - model nie dopuszcza żadnych konfliktów
- CREW PRAM - model doupszcza konflikty odczytu
- ERCW PRAM - model dopuszcza konflikty zapisu
- CRCW PRAM - model doupszcza konflikty odczytu i zapisu.

Podział ze względu na sposób rozstrzygania konfliktów zapisu

- **common** - zezwala na jednoczesny zapis tylko, gdy wszystkie procesory chcą zapisać tę samą wartość
- arbitrary - zapisuje jeden procesor (arbitralnie wybrany)
- priority - zapisuje ten o największym priorytecie (np. najmniejszy identyfikator)

W jakim modelu działa Algorytm 1?

Algorytm 1 działa w modelu common ERCW PRAM.