Programowanie współbieżne i rozproszone

WYKŁAD 14

Notacja *O*

- f, g – funkcje o wartościach nieujemnych określone w dziedzinie nieujemnych liczb całkowitych. Piszemy:

$$f(n)=O(g(n))$$

i mówimy, że funkcja ***f*(*n*) jest co najwyżej rzędu *g*(*n*)** jeśli istnieją stałe $c>0$ oraz n_0 takie, że:

$$f(n)\leqslant c\cdot g(n) \;\; dla\, każdego \;\; n\geqslant n_0$$

Mówimy, że g jest asymptotycznym ograniczeniem górnym dla f.

Notacja θ

- f, g – funkcje o wartościach nieujemnych określone w dziedzinie nieujemnych liczb całkowitych. Piszemy:

$$f(n)=\Theta(g(n))$$

i mówimy, że ***f*(*n*) jest dokładnie rzędu *g*(*n*)** jeśli istnieją stałe $c_1>0$, $c_2>0$ oraz n_0 takie, że:

$$c_1 \cdot g(n) \leqslant f(n) \leqslant c_2 \cdot g(n) \;\; dla\, każdego \;\; n \geqslant n_0$$

Mówimy, że g jest asymptotycznym dokładnym oszacowaniem dla f.

Algorytmy równoległe

- **P1**, **P2**, ..., **Pn** - procesory.
- **parfor** - równoległa wersja instrukcji for.

```
parfor Pi, <warunek> do
    <ciąg instrukcji>
end parfor
```

Obliczanie minimum

- Dane:

```
a[1 ... n]
```

- Algorytm sekwencyjny:

```
min=a[1];
for i=2 to n do
   if a[i]<min then
      min=a[i];
   end if
end for
```

Obliczanie minimum

- Algorytm równoległy (założenie: $n=2^r$, gdzie r - liczba naturalna, n procesorów):

```
k=n;
for j=1 to log(n) do
   parfor Pi, 1≤i≤k/2 do
      if a[i]>a[i+k/2] then
         a[i]=a[i+k/2];
      end if
   end parfor
   k=k/2;
end for
min=a[1];
```

Obliczanie minimum

- Złożoność czasowa:
  - algorytm sekwencyjny: $O(n)$
  - algorytm równoległy: $O(\log n)$
- Efektywność wykorzystania procesorów:
  - algorytm równoległy: $O(1/\log n)$
    - jeśli $n \to \infty$, to efektywność $\to 0$ - nieefektywne wykorzystanie procesorów

Obliczanie sumy

- Dane:

```
a[1 ... n]
```

- Algorytm sekwencyjny:

```
suma=0;
for i=1 to n do
   suma=suma+a[i];
end for
```

Obliczanie sumy

- Algorytm równoległy (dowolne *n*, *n* procesorów):

```
k=n;
for j=1 to ceil(log(n)) do
    parfor Pi, 1≤i≤floor(k/2) do
        a[i]=a[i]+a[k+1-i];
    end parfor
    k=k/2;
end for
suma=a[1];
```

Obliczanie sumy

- Algorytm równoległy (dowolne *n*, *p* procesorów, $p \leq n$):

```
parfor Pi, 1≤i≤p do
    g=ceil(n/p)(i-1)+1;
    b[i]=a[g];
    for j=1 to ceil(n/p)-1 do
        if g+j≤n then
            b[i]=b[i]+a[g+j];
        end if
    end for
end parfor
sumowanie w tablicy b[1 ... p] za pomocą p procesorów
```

Obliczanie sumy

- Złożoność czasowa:
  - algorytm sekwencyjny: $O(n)$
  - algorytm równoległy: $O(n/p+\log p)$
- Efektywność wykorzystania procesorów:
  - algorytm równoległy: $O(n/(n+p \log p))$ - lepsze wykorzystanie procesorów

Obliczanie sumy

Jeśli $p=n/\log n$

- Złożoność czasowa:
  - algorytm równoległy: $O(\log n)$
- Efektywność wykorzystania procesorów:
  - algorytm równoległy: $\Theta(1)$