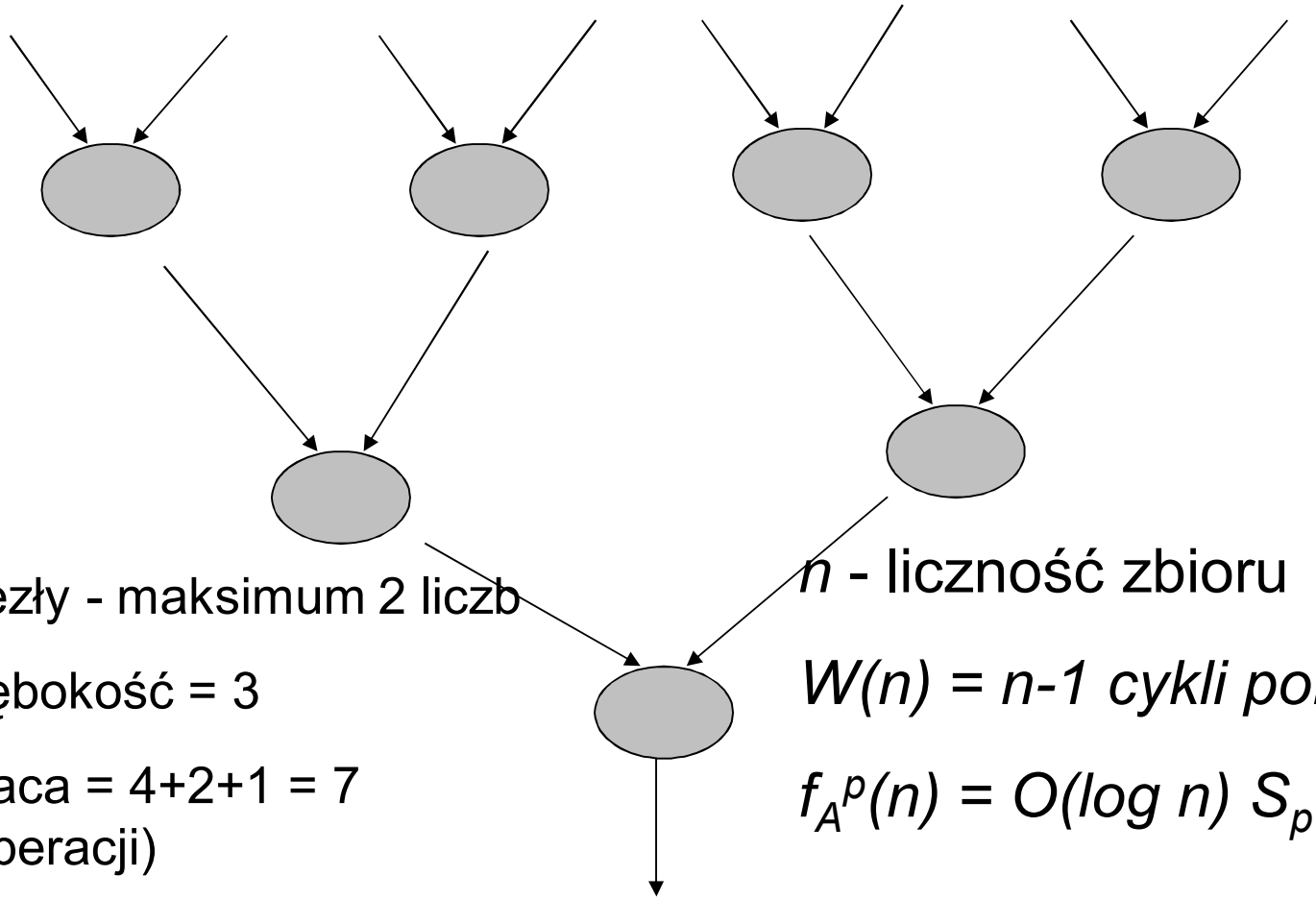
Algorytmy równoległe: ocena efektywności prostych algorytmów dla systemów wielokomputerowych

Rafał Walkowiak
Politechnika Poznańska
Studia inżynierskie
Informatyka 2015/16

Znajdowanie maksimum w zbiorze n liczb



- węzły - maksimum 2 liczb
- głębokość = 3
- praca = 4+2+1 = 7 (operacji)
- obliczenia w potoku

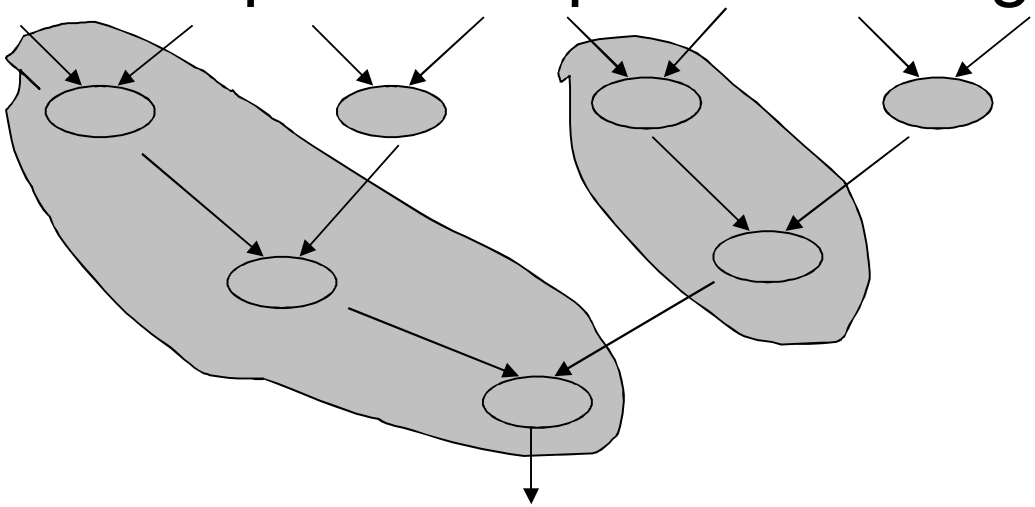
n - liczność zbioru

$W(n) = n-1$ *cykli porównań*

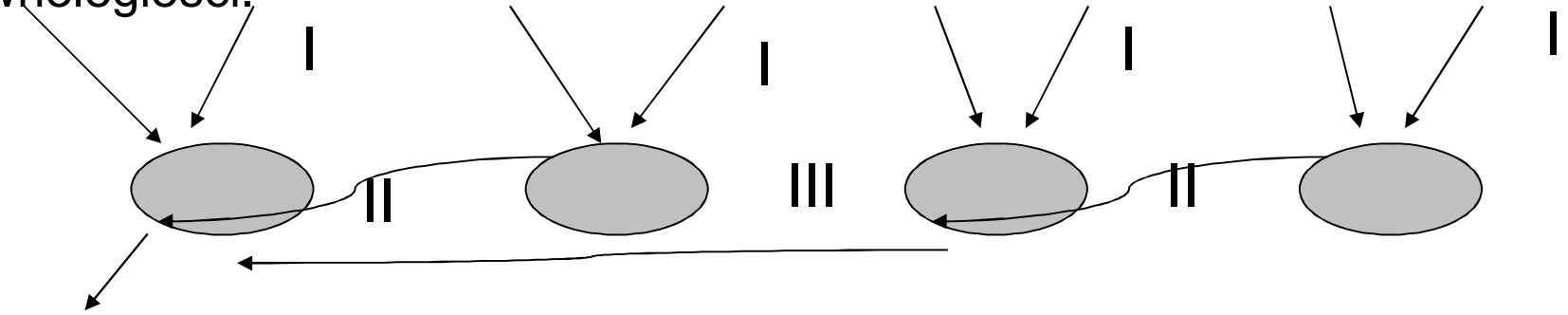
$f_A^p(n) = O(\log n)\ S_p(n), e_p(n)$

Analiza efektywności - aglomeracja (1)

Czy aglomeracja (połączenie zadań) jest możliwa bez spadku stopień równoległości ?



Zgrupowanie operacji realizowanych sekwencyjnie - możliwy przydział do jednego węzła przetwarzającego, mniej komunikacji, ten sam stopień równoległości.



Analiza efektywności - aglomeracja (2)

- Mniej przesłań między węzłami
- Niższa złożoność obliczeniowa
- Ta sama ilość pracy (7 porównań)
- Wzrost stopnia wierzchołka (3 do 5)
- Mniejsza liczba procesorów (4 zamiast 7)

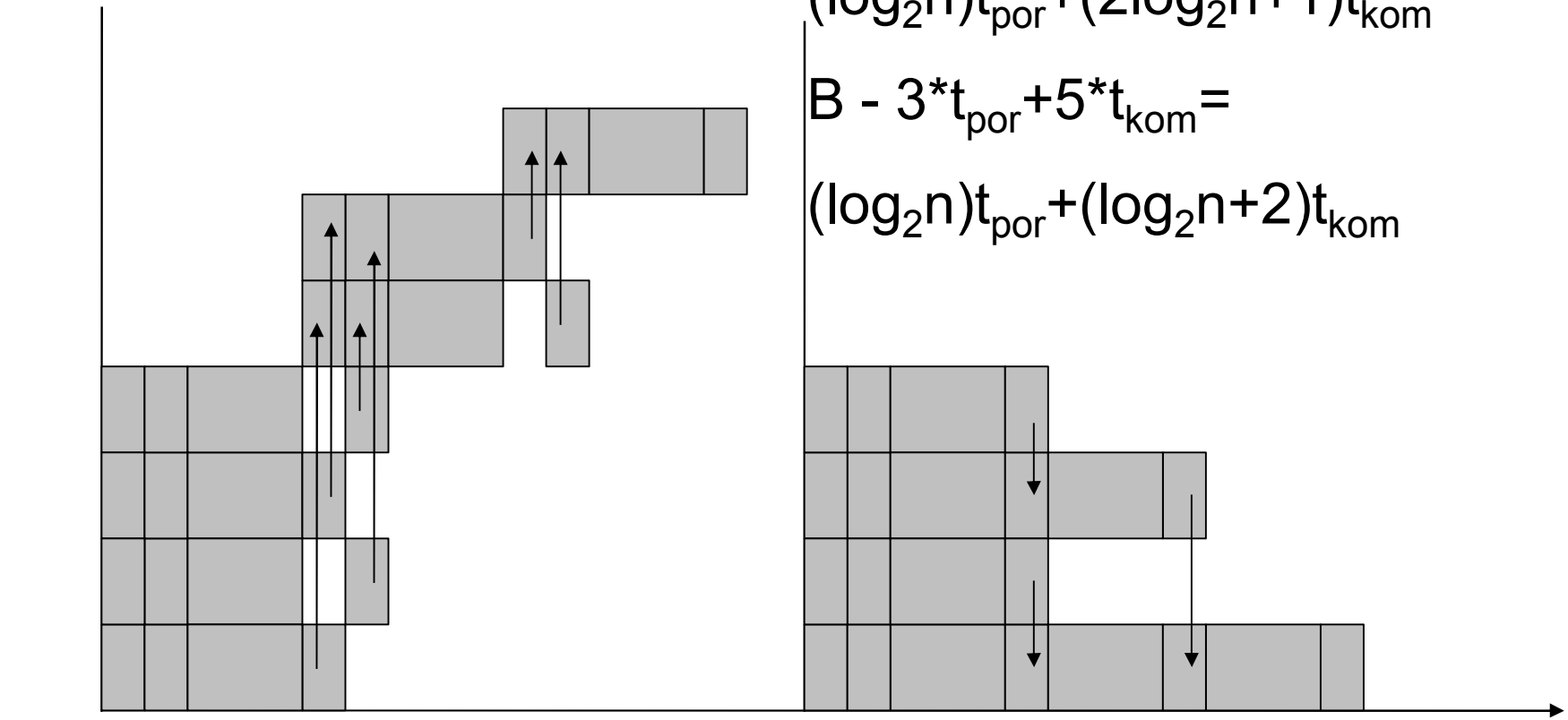
Analiza efektywności -aglomeracja (3) wykres praca-czas

$A - 3*t_{por}+7*t_{kom}=$

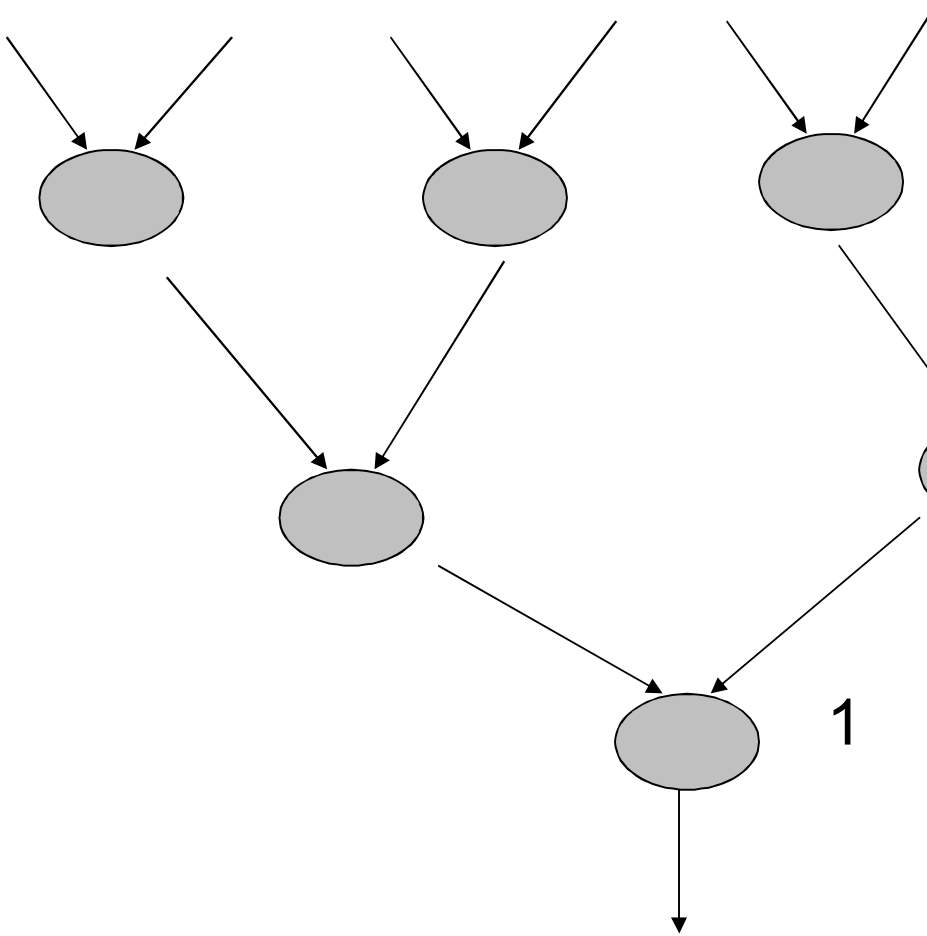
$(\log_2 n)t_{por}+(2\log_2 n+1)t_{kom}$

$B - 3*t_{por}+5*t_{kom}=$

$(\log_2 n)t_{por}+(\log_2 n+2)t_{kom}$



Znajdowanie maksimum w *k* zbiorach *n* liczb



k - krotne obliczenia w potoku

n - liczność zbioru

$W(n) = k*(n-1)$

$f_A^p(n) = O(k + \log n)$

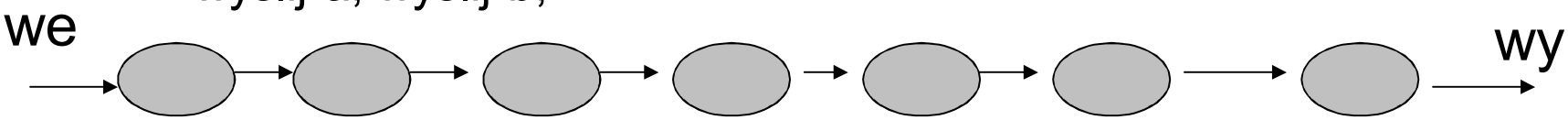
Sortowanie w łańcuchu procesorów

Wejście: ciąg liczb (minimum jedna) zakończonych „ -1”

Wyjście: posortowany ciąg zakończony -1

Kod dla wszystkich węzłów:

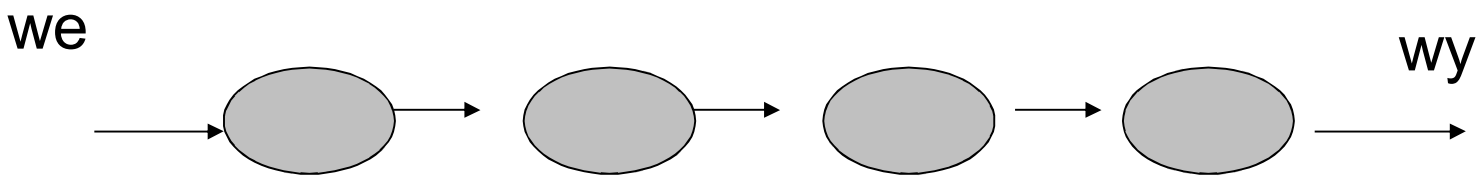
```
pobierz a
pobierz b
while b > 0 do {
    if a>b    then wyślij b
              else wyślij a; a:=b
    pobierz b }
wyślij a; wyślij b;
```



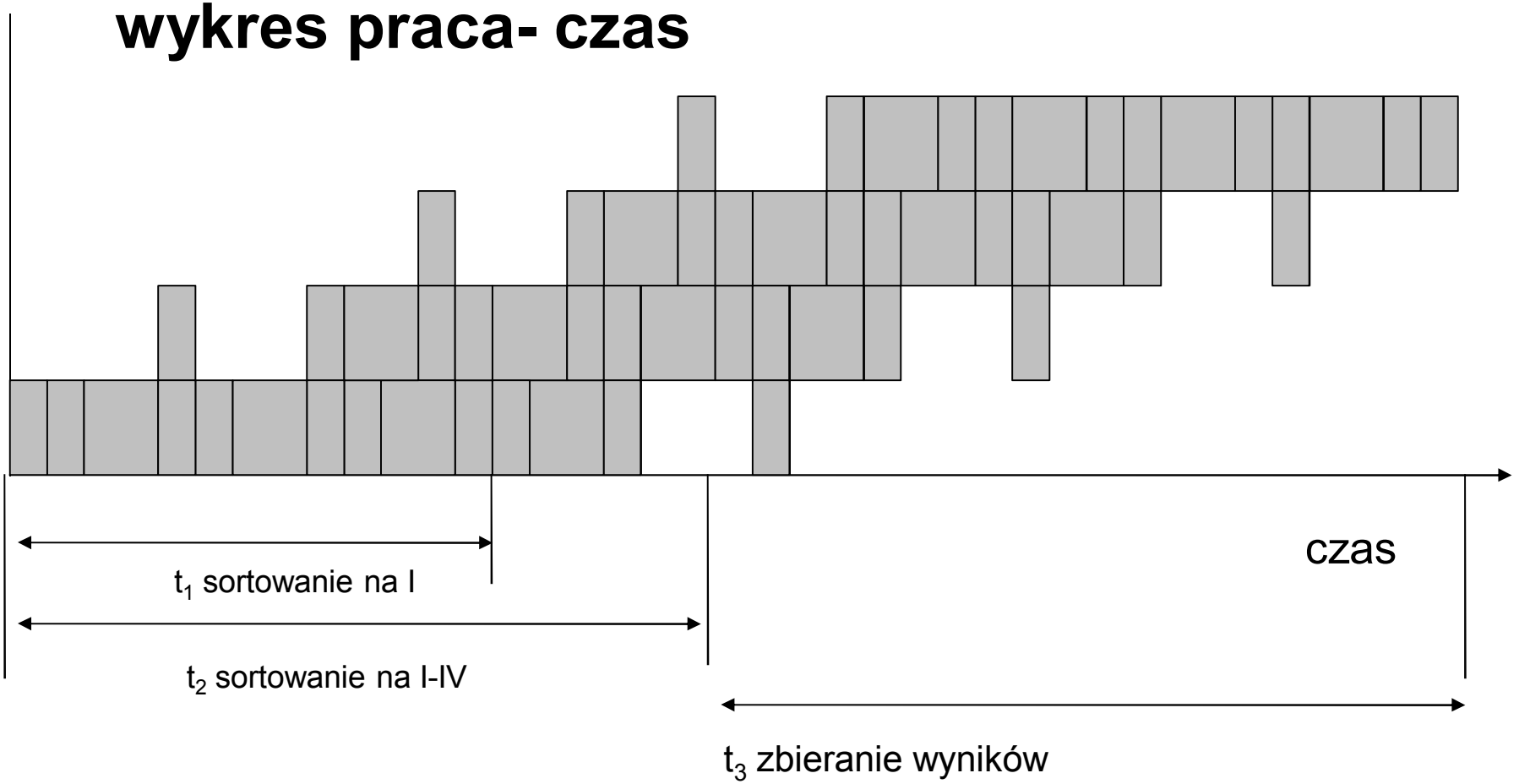
Działanie algorytmu

na wejście łańcucha podajemy kolejno wartości: 2,3,4,1,-1

proc1	proc2	proc3	proc4	czas
2,3				2k,p
3,4	2			2k,p
4,1	2,3			2k,p
4,-1	3,1	2		2k,p
-1	3,4	2,1		2k,p
	4,-1	2,3	1	2k,p
	-1	3,4	1,2	2k,p
		4,-1	2,3	2k,p
		-1	3,4	2k,p
			4,-1	2k,p
			-1	



Sortowanie liczb w łańcuchu
wykres praca- czas



$t_1 = (n-1)*t_p+(2n-1)* t_k \approx n* t_c$

$t_2 = t_1+(n-2)*(t_p+t_k) \approx 2n*t_c$

$t_3 = (n+2)*t_c$

$t_c = t_p+2t_k$

$t_2+t_3 \approx 3n*t_c$

$f_A^m = 3n*t_c = O(n)$

Złożoność sortowania liczb w łańcuchu

- Załóżmy, że liczba procesorów jest równa n - wielkości zbioru liczb do posortowania.
- Każdy procesor musi wykonać po n-1 cykli porównań (cykl - pobranie/wysłanie, porównanie), gdyż każda liczba przez każdy procesor przechodzi.
- Problemem jest wyznaczenie przesunięcia czasowego pomiędzy rozpoczęciem pracy kolejnego procesora Ważne są tutaj następujące spostrzeżenia:

1. Każdy kolejny **krok porównania** generuje **jedną liczbę** do wysłania na kolejny niewykorzystany jeszcze procesor;
2. Każdy krok porównania wymaga 2 transmisji blokujących = wysłanie i odbiór = RAZEM Tc=tp+2tk
3. Aby kolejny procesor rozpoczął porównywanie potrzebuje 2 liczb - konieczne jest wykonanie w systemie 2 następujących po sobie porównań na procesorze wcześniejszym.
4. Po ostatnim porównaniu (na ostatnim i każdym procesorze) mamy 2 komunikacje dla wysłania wyników porównania w odpowiedniej kolejności.

- Liczba kroków:
  - Liczba kroków (porównanie i 2 transmisje) do rozpoczęcia pracy procesora p to: 2*(p-1)
  - Liczba kroków pracy ostatniego procesora to n-1 cykli (porównanie i przesłania)
  - **Zatem czas przetwarzania T to: [2 * (p-1) + n-1] *Tc+ 2 tk**
  - Dla p1=n T= 3n -3 lub T= 3p1-3
  - Dla p2=n-1 T= 3n-5 lub T=3p2-2

Wyznaczanie sumy rozproszonych elementów

Drzewo binarne - algorytm:

1. Elementy każdym węźle – liczba węzłów K
2. Wysłanie liczby w kierunku korzenia
3. Wyznaczenie sum częściowych (suma 3 wartości) w węzłach pośrednich
4. Sumy częściowe w propagują w kierunku korzenia
5. Suma ostateczna wyznaczona w korzeniu
6. Propagacja sumy do wszystkich węzłów

Liczba operacji – praca:

- OBLICZENIA - dla każdego węzła oprócz liści - 2 komunikacje i 2 sumowania
- PROPAGACJA WYNIKU - dla każdego węzła oprócz liści – 2 komunikacjei w dół drzewa
- RAZEM: $(K-1)/2 * (2t_{kom} + 2 t_{sum}) + (K-1) t_{kom}$ - Obliczenia, propagacja wyniku,

Wyznaczanie sumy rozproszonych elementów – drzewo binarne (część 2)

Czas przetwarzania – liczba kolejno realizowanych kroków algorytmu:

- *liczba poziomów drzewa oprócz poziomu liści* - realizowany cykl - 2 odbiory komunikatu (synchroniczne) i 2 sumowania
- *liczba poziomów drzewa oprócz poziomu liści* – 2 przesłania
- RAZEM: $(\log_2(K+1) - 1) * (2t_{kom} + 2\, t_{sum}) + (\log_2(K+1) - 1) * 2t_{kom}$
  $\approx \log_2 K\, (t_{kom} + t_{sum})$

Uwagi :

- Liczba operacji sumowania równa minimalnej.
- Poziom równoległości obliczeń i równoległości komunikacji zmienny
- Możliwość dlaszej minimalizacji liczby przesłań (rozsyłanie wyniku) – w innej architekturze – prezentacja wkrótce.

Wyznaczanie sumy rozproszonych elementów
łańcuch dwukierunkowy

Łańcuch dwukierunkowy:

- Węzeł centralny, całkowita liczba węzłów K
- Analogiczna do drzewa praca – tym razem równoległa w 2 gałęziach- przesyłanie do węzła centralnego z sumowaniem

Liczba operacji – praca:

- K-3 – razy - dla każdego węzła oprócz węzłów końcowych i centralnego - odbiór komunikatu i sumowanie, w węźle centralnym 2 odbiory i 2 sumowania
- K-3 – razy – dla każdego węzła oprócz węzłów końcowych i centralnego - wysłanie informacji, węzeł centralny 2 przesłania
- RAZEM: $(K-3) * (t_{kom} + t_{sum}) + 2* t_{kom} + 2*t_{sum} + (K-3) * t_{kom} + 2* t_{kom} \approx 2 K t_{kom} + K t_{sum}$

Wyznaczanie sumy rozproszonych elementów
łańcuch dwukierunkowy (2)

Czas przetwarzania – liczba kolejno realizowanych kroków algorytmu:

- M = ½ (liczba węzłów oprócz końcowych i centralnego) razy realizowany cykl - odbiór komunikatu i sumowanie; węzeł centralny- 2 odbiory i 2 sumowania
- M razy – 1 przesłanie; węzeł centralny – 2 przesłania
- RAZEM: ½ (K-3) (t_{kom}+ t_{sum}) + 2 t_{sum}+ 2 t_{kom}+ ½* (K-3) * t_{kom}+ 2 * t_{kom} ≈ K t_{kom} + ½ K t_{sum}

Uwagi :

- Liczba operacji sumowania równa minimalnej.
- Poziom równoległości obliczeń i komunikacji bliski 2.

Wyznaczanie sumy rozproszonych elementów
- pierścień jednokierunkowy

- Węzeł centralny, całkowita liczba węzłów K
- Sekwencyjne sumowanie i przesyłanie sum częściowych do węzła centralnego; Rozsyłanie wyniku do wszystkich węzłów

Liczba operacji – praca:

K-1 – razy - dla każdego węzła oprócz węzła następnego po centralnym - odbiór komunikatu i sumowanie,

K-1 – razy – dla każdego węzła oprócz węzła przed centralnym - wysłanie wyniku – sumy

RAZEM: $(K-1) * (t_{kom} + t_{sum}) + (K-1) * t_{kom} \approx K\, t_{sum} + 2K\, t_{kom}$

Czas przetwarzania – liczba kolejno realizowanych kroków algorytmu:

- Jak wyżej, gdyż wszystkie operacje realizowane sekwencyjnie.

Uwagi :

- Liczba operacji sumowania równa minimalnej.
- Brak równoległości.
- Możliwość minimalizacji liczby przesłań w tej architekturze (uniknięcie rozsyłania wyniku) poprzez wyznaczanie sumy we wszystkich węzłach – porównaj następny algorytm.

Wyznaczanie sumy rozproszonych elementów
pierścień jednokierunkowy , komunikacja synchroniczna

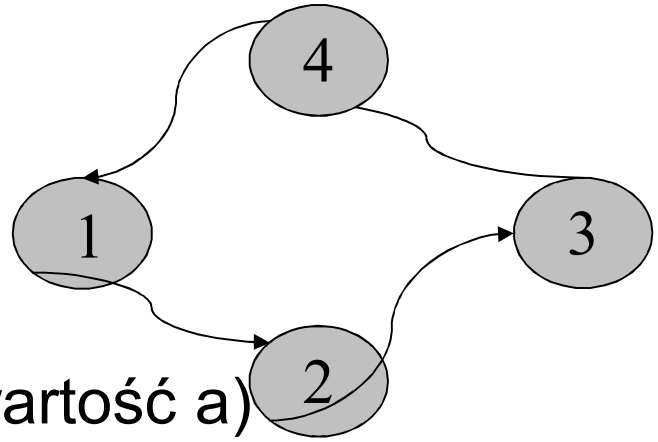
a – elementy sumowane; number_of_nodes=2n;n>1

```
k:=2;s := a
jeśli parzysty to odbierz b1; wyślij a
jeśli nieparzysty to wyślij a; odbierz b1
s:= s + b1
while k < number_of_nodes
      jeśli parzysty to odbierz b2; wyślij b1;b1:=b2
      jeśli nieparzysty to wyślij b1; odbierz b1
      s:= s + b1
      k:= k + 1
```

Optymalizacja liczby przesłań kosztem nadmiarowych obliczeń:

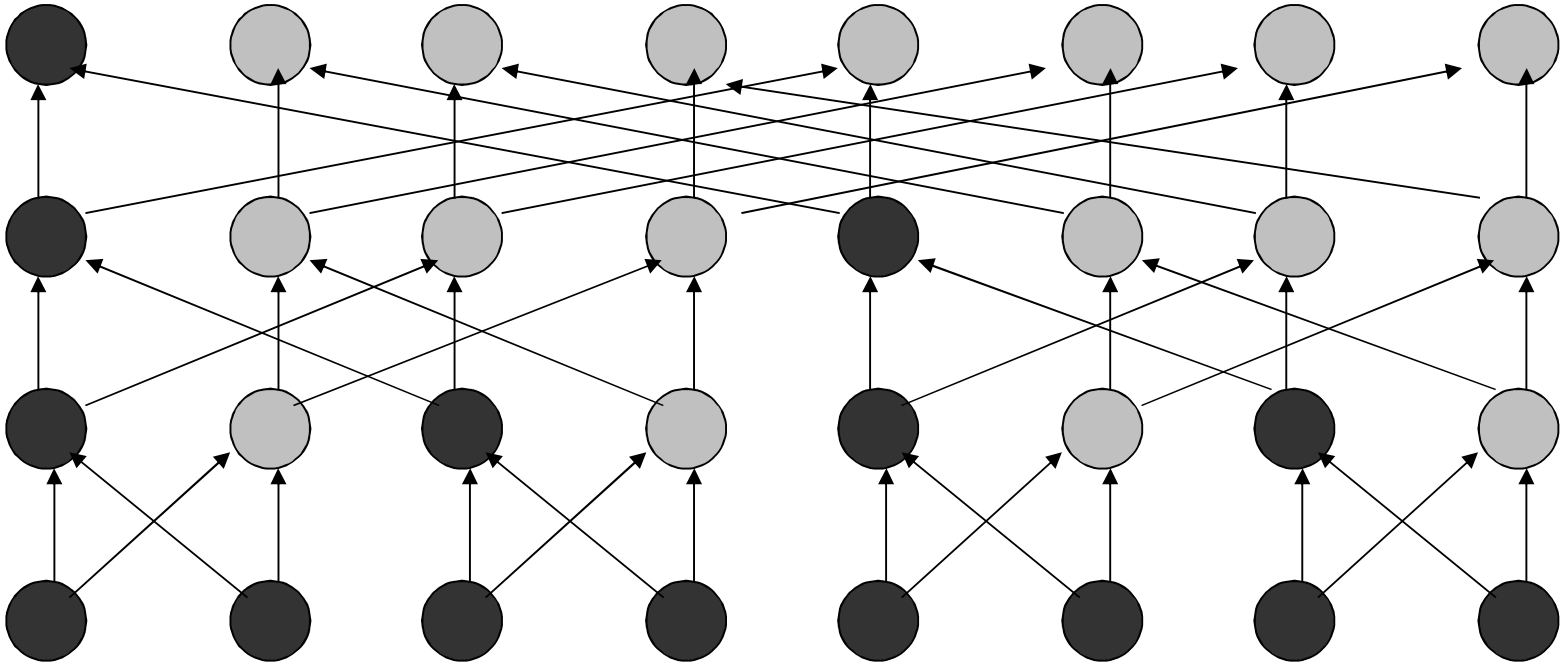
- n sumowań, n (2n synchronicznie) komunikacji
- zamiast – n sumowań, 2n komunikacji (zbieranie i wysyłanie) w rozwiązaniu z poprzedniej strony

Wyznaczanie sumy rozproszonych elementów pierścień (2) – przykład 4 węzły



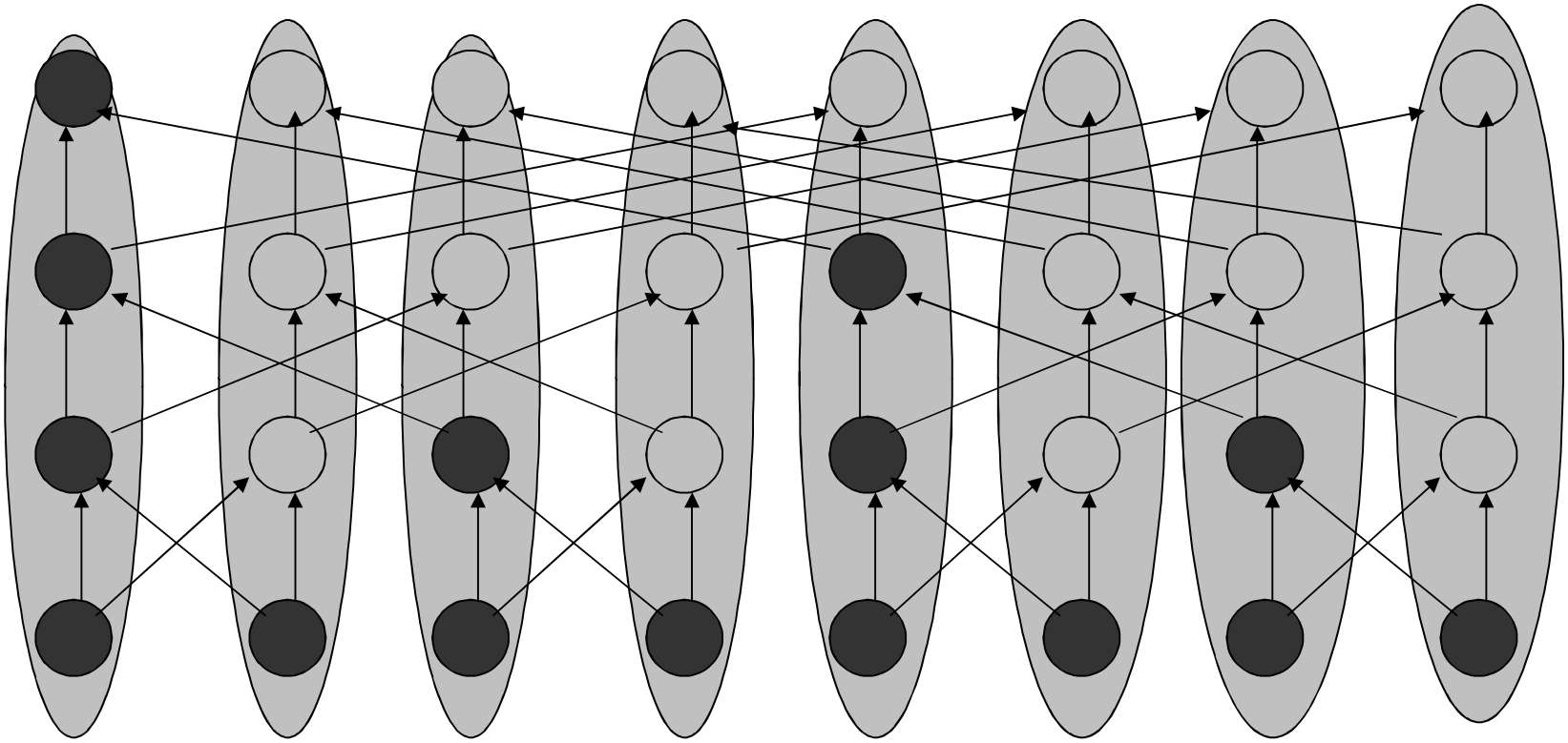
- Operacje węzła 3 (przechowuje wartość a)
  1. wyślij a;odbierz b; s = a+b;
  2. wyślij b; odbierz b; s = s+b;
  3. wyślij b;odbierz b; s = s +b;
- liczba węzłów *n*, liczba kroków przetwarzania *n-1*,
- każdy krok przetwarzania to: nadanie, odbiór, sumowanie
- ilość pracy n*(n-1) sumowań, n*(n-1) przesłań
- zmniejszenie czasu przetwarzania do $(n-1)t_s+2(n-1)t_k$

Wyznaczanie sumy rozproszonych elementów - podział pracy na zadania



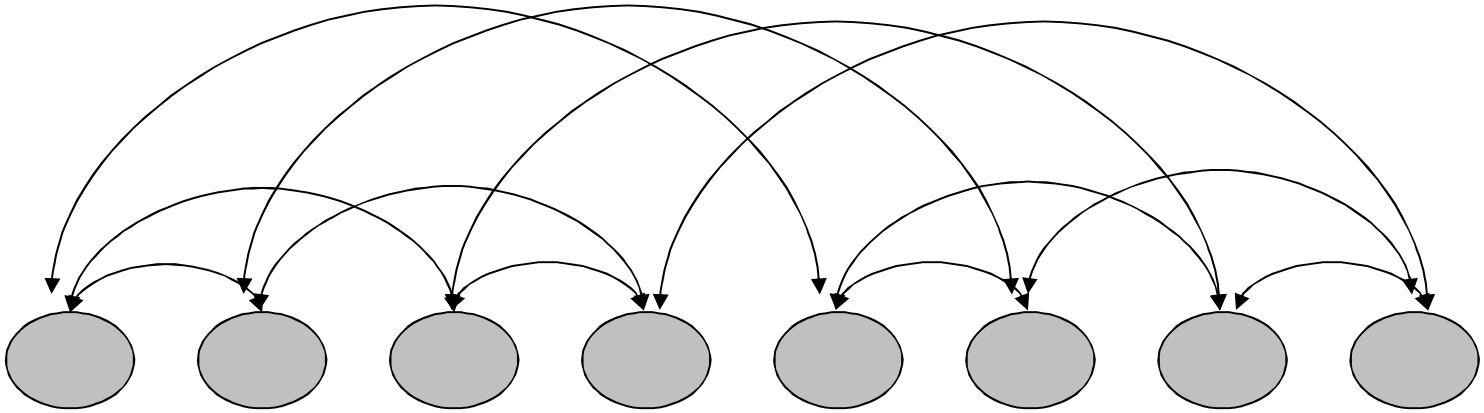
- węzeł - operacja dodawania
- łuk - operacja przesłania jednej liczby
- liczność zbioru n, W= 3*n, $f_A^p(n)= \log_2 n$

Wyznaczanie sumy rozproszonych elementów - aglomeracja – połączenie zadań



- zadania-operacje realizowane sekwencyjnie połączone w jedno zadanie, minimalizacja komunikacji, wzrost lokalności danych

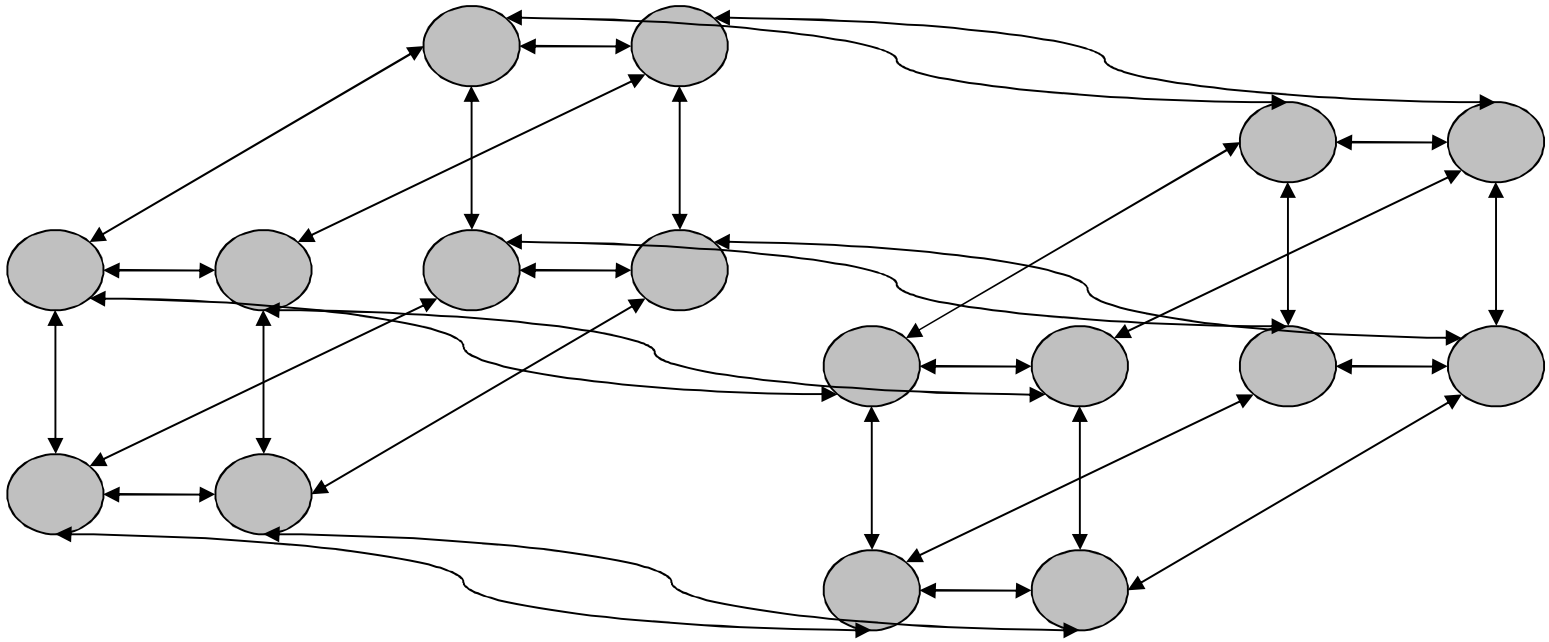
Wyznaczanie sumy rozproszonych elementów - wynik aglomeracji



8 zadań to:

- 3 fazy wymiany wartości między parami zadań
- 3 fazy dodawania – dodawanie 2 wartości, wyznaczana lokalnie suma 2,4 lub 8 elementów
- Pracę można zrzutować w sposób 1 zadanie na 1 procesor na 3 wymiarową kostkę procesorów.

Wyznaczanie sumy rozproszonych elementów - struktura przetwarzania w systemie 4 wymiarowej kraty podwójnej



Sumowanie danych rozproszonych w 16 węzłach –

$\log_2 16 = 4$ kroki

Znajdowanie liczb pierwszych

- Metody:
  - **dzielenie** badanej liczby przez liczby pierwsze i badanie wartości reszty z dzielenia
  - usuwanie wielokrotności - usuwanie ze zbioru badanych liczb liczb będących wielokrotnością liczb pierwszych
- Jakie liczby pierwsze uwzględniać dla badanej liczby (bądź górnego zakresu badanego przedziału) n?
- Wystarczy znaleźć dla każdej liczby złożonej minimalny podzielnik: dla 35 – 5, dla 77 – 7, dla 121 – 11.
- Czy istnieje warunek ograniczający maksymalną wartość najmniejszego podzielnika liczby n?
- Tak.
- Maksymalna wartość najmniejszego podzielnika liczby złożonej n wynosi $n^{1/2}$.
- Aby znaleźć zatem liczby pierwsze $x_i \in <k,l>$ należy:
  - usunąć liczby dzielące się bez reszty przez liczby pierwsze brane z przedziału $<2, x_i^{1/2}>$ lub
  - usunąć liczby będące wielokrotnością liczb pierwszych z przedziału $<2, l^{1/2}>$

Wykreślanie z tablicy

Z badanego zbioru (tablicy) usuwamy wielokrotności (jakie?) liczb pierwszych z przedziału <2,zakres górny$^{1/2}$>

Przykład dla zakresu : <2,65>

2 : 4,6,8,...64

3 : 9,15,21,27,33,39,45,51,57,63

5 : 25,35,55,65

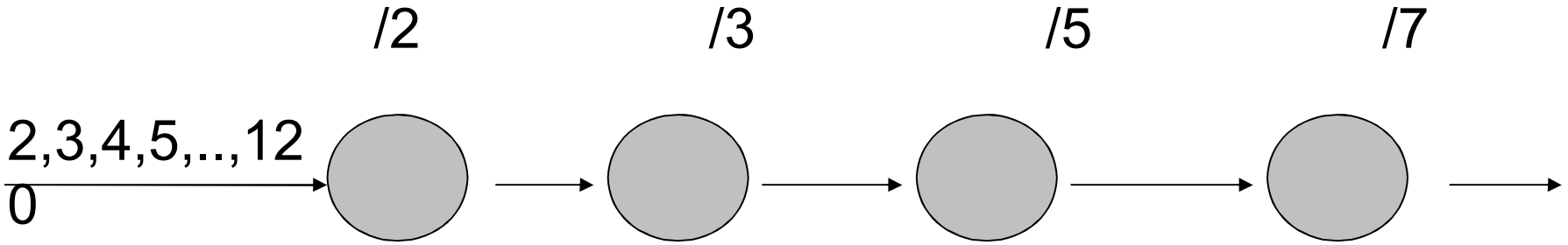
7 : 49

- nie jest konieczna do rozpoczęcia obliczeń znajomość wszystkich liczb-pierwszych z przedziału <2,zakres górny$^{1/2}$>;

- kolejno pojawiające się liczby pierwsze mogą być wykorzystane dopiero później, gdyż wyznaczanie wielokrotności mniejszych liczb pierwszych odbywa się dla całego badanego przedziału i zajmuje stosunkowo dużo czasu.

Sito Eratostenesa - koncepcja podejścia funkcjonalnego do podziału w architekturze z przekazywaniem komunikatów –

znaczenie koncepcyjne – niska efektywność



11,13,17,19,23,29,31,37,41,...,

Pierwsza liczba odebrana przez każdy z procesów jest traktowana jako dzielnik i jako liczba pierwsza. Liczby dzielące się z resztą są przesyłane dalej. Wynik przetwarzania – liczby pierwsze pojawiają się na wyjściu systemu oraz rezydują w procesach (należy je przesłać na wyjście).

Liczba procesów niezbędbych dla zakresu <n,k> jest równa liczbie liczb pierwszych w zakresie od <2, $k^{1/2}$> około $2k^{1/2}/\ln k$