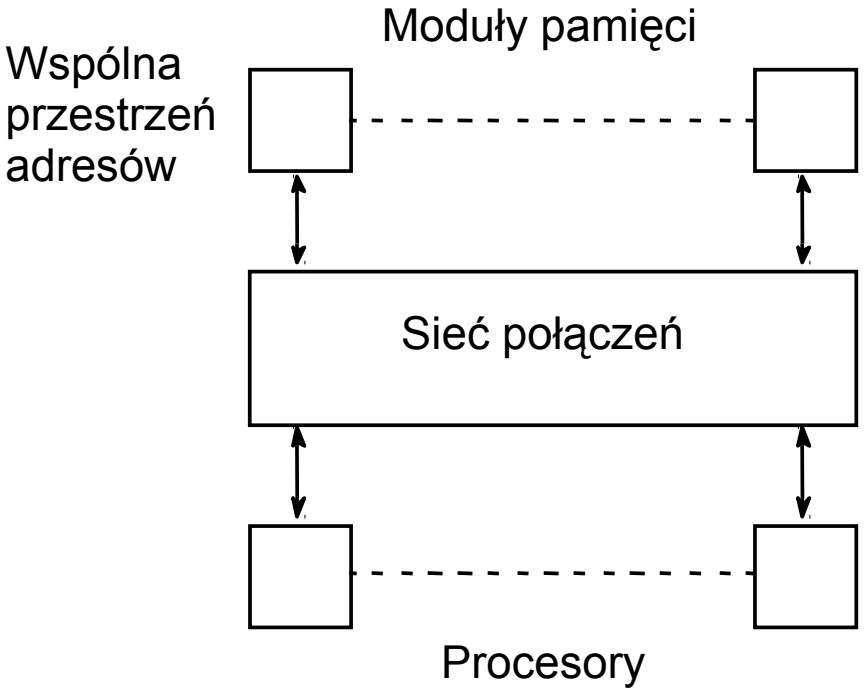
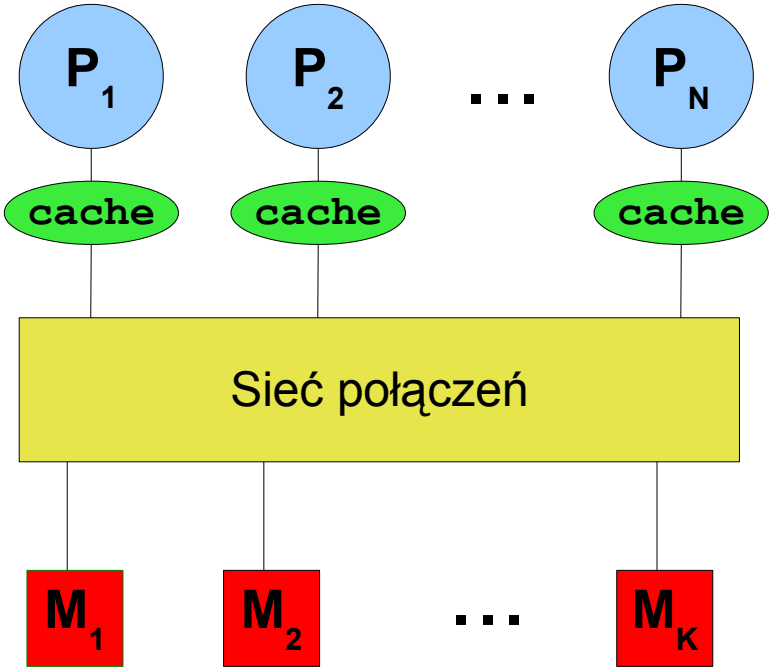
Architektury maszyn równoległych

System wieloprocesorowy ze wspólną pamięcią

- Każdy procesor ma dostęp do dowolnego modułu pamięci
- Bajt pamięci widoczny przez wszystkie procesory pod tym samym adresem
- Pamięci cache (indywidualne dla poszczególnych procesorów) komplikują sytuację.
- System SMP (Symmetric Multiprocessing), UMA (Uniform Memory Access) – czas dostępu dowolnego procesora do dowolnej komórki pamięci jest w przybliżeniu taki sam.

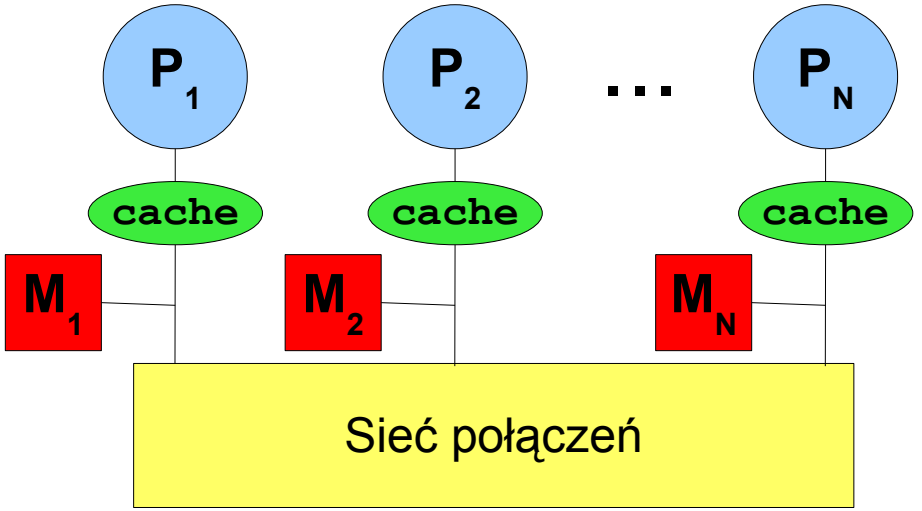


Architektura UMA



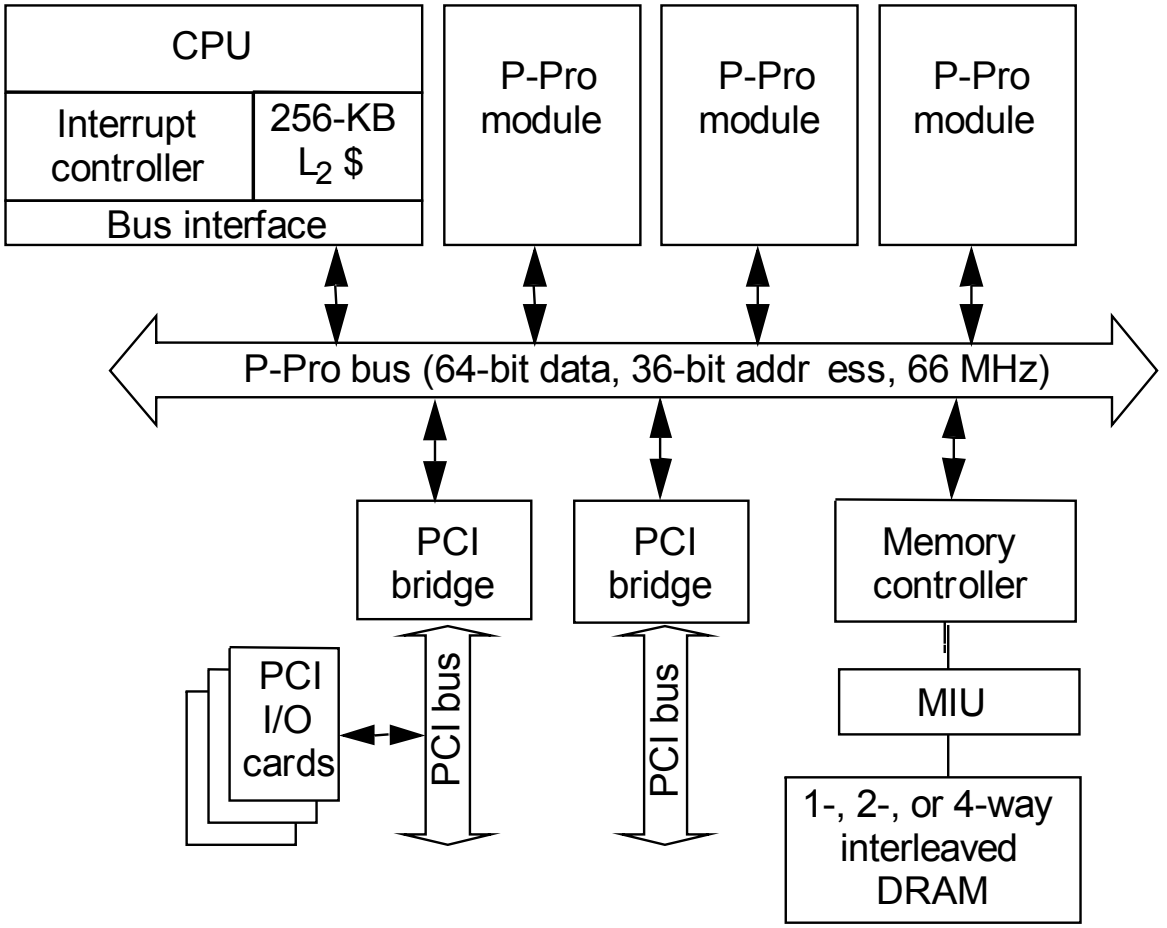
- N procesorów i K modułów pamięci.
- Każdy procesor posiada własną pamięć cache (istnieją architektury ze wspólną pamięcią L2 – o ile dwa procesory, pamięć L2 i dodatkowy poziom sieci połączeń zmieszczą się na chipie; patrz Intel Core 2).
- Sieć połączeń – wybór pomiędzy niską ceną (szyna) a wysoką skalowalnością (przełącznik krzyżowy).

Architektura NUMA



- ang. non-uniform memory access.
- Nadal każdy bajt widoczny przez wszystkie procesory pod tym samym adresem, Jednak czas dostępu procesora do lokalnej pamięci krótszy, niż do pamięci zdalnej.
- Przykłady:
  - Na dużą skalę SGI Origin, SGI Altix.
  - Na małą skale AMD Opteron (2,4,8 procesorów), Intel Xeon z QPI.

Architektura szyny – Pentium Pro

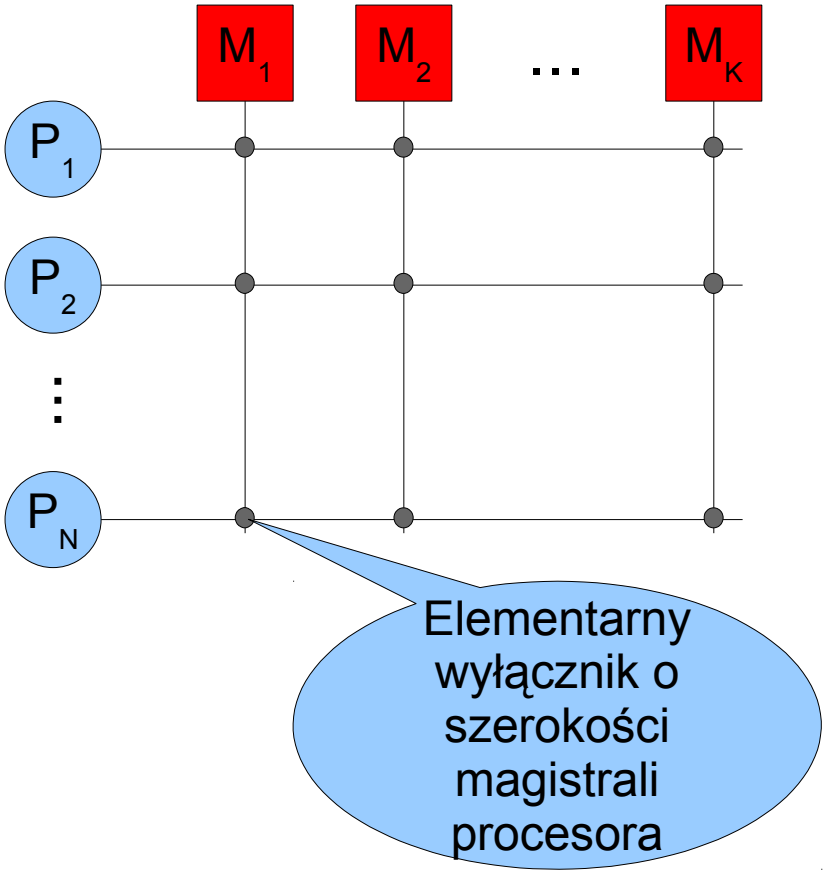


- Rozwiązanie dobrze skalujące się w sensie kosztu, bardzo źle w sensie wydajności.
- Rywalizacja o dostęp do szyny – w skrajnym przypadku system redukuje się do maszyny sekwencyjnej.
- Pamięci cache (L1 i zintegrowana L2) zmniejszają rywalizację o szynę.

Ograniczenia szyny

- Procesor komunikuje się z pamięcią w celu (a) pobrania kodu instrukcji (b) zapisu/odczytu danych.
- Przyjmijmy procesor o zegarze 1GHz z 32-bitowymi instrukcjami. 4GB/s potrzebne na pobieranie instrukcji. Zakładając, że 30% instrukcji to instrukcje odwołujące się do pamięci potrzeba dodatkowo 1.2GB/s na transfer danych. Łącznie 5.2 GB.
- Przyjmijmy, że procesor jest wyposażony w pamięć cache (dzisiaj co najmniej dwu-poziomową) zapewniającą współczynnik trafień 98% dla instrukcji i 95% dla danych.
- Zapotrzebowanie na przepustowość spada do 140 MB/s.
- Jednakże jeżeli szyna ma przepustowość 1 GB/s, to osiem procesorów (8*140=1120) nasyci szynę.

Przełącznik krzyżowy (ang. crossbar switch)



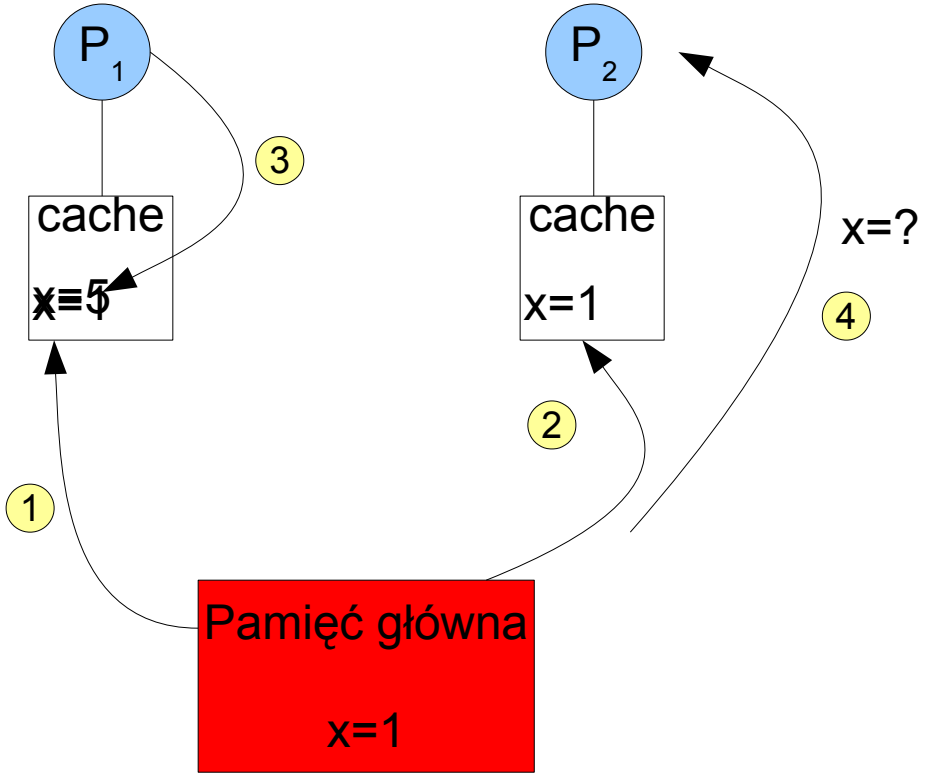
- Jeden wyłącznik w wierszu i jeden w kolumnie może być włączony.
- Sieć nieblokująca: procesor i nie blokuje procesora j, o ile żądają dostępu do różnych banków pamięci.

- Przykład Sun Ultra HPC Server.
- Bardzo dobrze skaluje się pod względem wydajności, bardzo źle pod względem kosztu budowy [O(N*K)]

Pamięć podręczna – przypomnienie i problemy

- Przechowuje najczęściej używane dane (lub kod). Operuje na wierszach o dużej długości (32,64, 128 bajtów). W przypadku odczytu jeżeli odczytywane dane są w wierszu będącym w pamięci cache nie trzeba sięgać do pamięci głównej.
- Co zrobić w przypadku zapisu:
  - write-through – zapis jednocześnie do pamięci głównej i podręcznej (niska wydajność)
  - write-back – zapis do pamięci jedynie podręcznej, zapis do pamięci głównej w momencie opróżnienia wiersza w pamięci podręcznej.
- Strategia write-back zapewnia wyższą wydajność, prowadzi do problemu ze spójnością pamięci.
  - Przez pewien czas zmodyfikowane dane są w pamięci podręcznej procesora a nie ma ich w pamięci głównej.
  - A co się stanie gdy ***inny procesor w systemie wieloprocesorowym zechce przeczytać te dane ?***

Problem ze spójnością pamięci cache

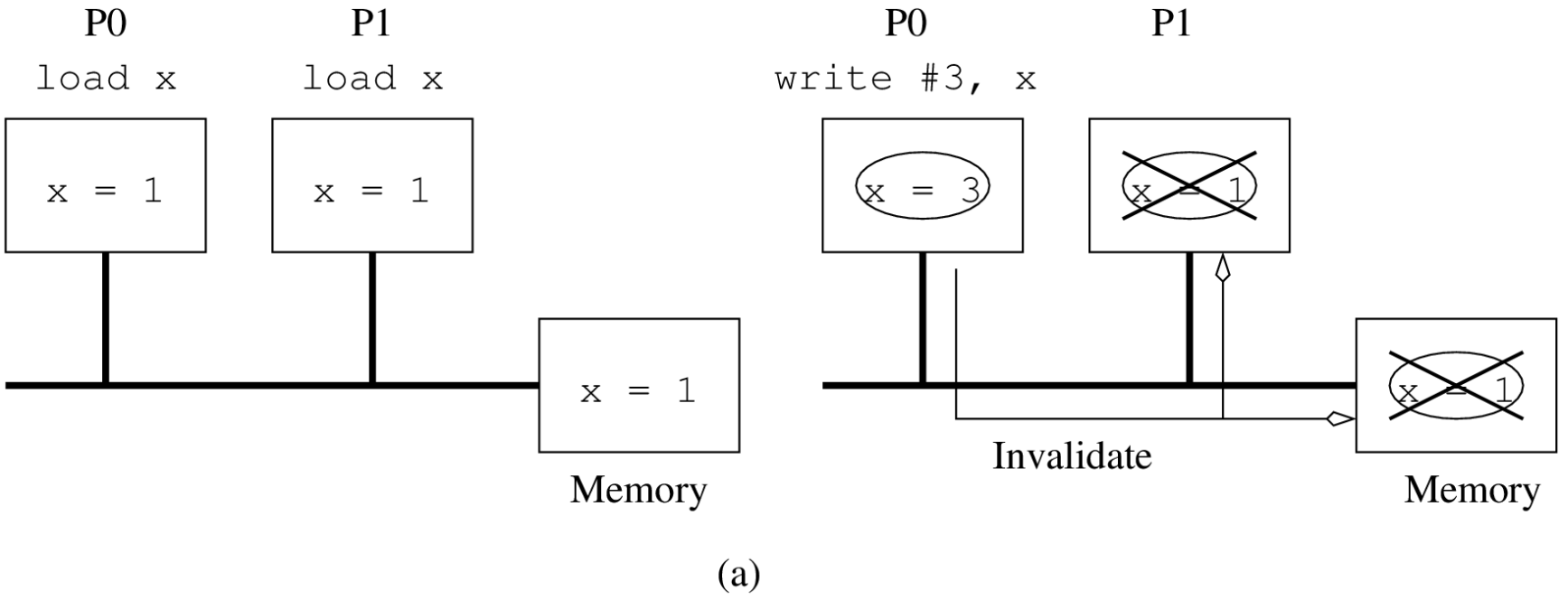


- Procesor 1 odczytuje dane z pamięci głównej.
- Procesor 2 odczytuje dane z pamięci głównej.
- Procesor 1 modyfikuje dane w swojej pamięci cache, zgodnie ze strategią write-back nie modyfikując pamięci głównej.
- *Co odczyta procesor 2 ?*

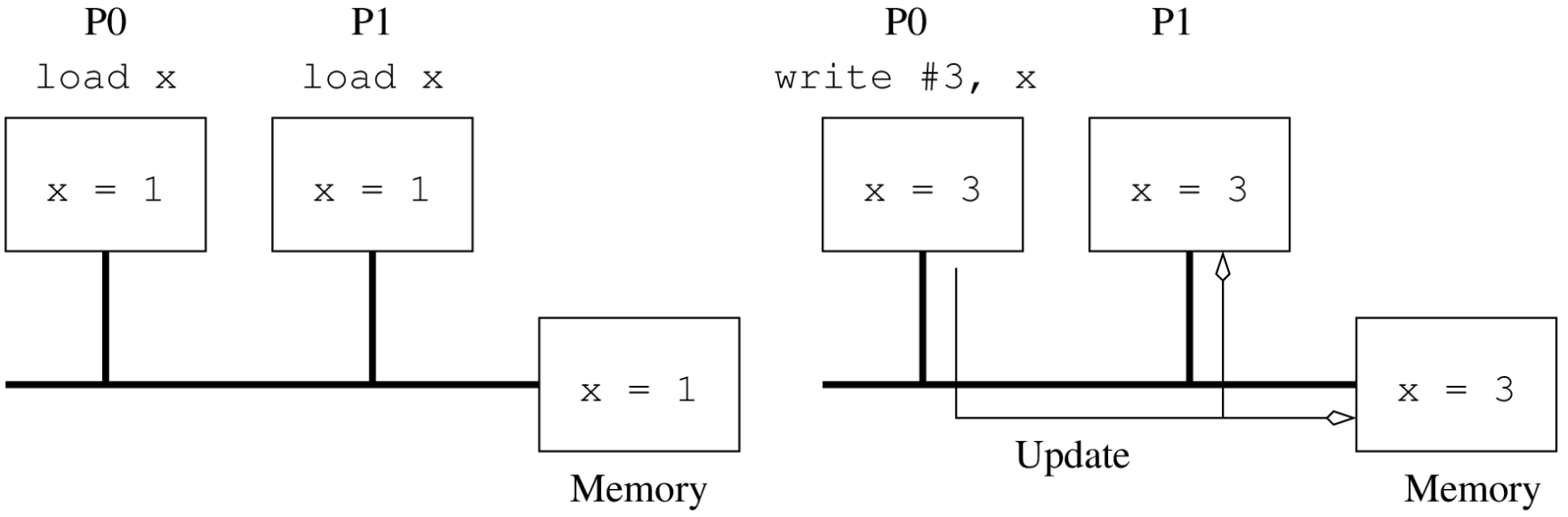
Utrzymywanie spójności

- Na nasze szczęście problem ze spójnością został rozwiązany przez specjalistów od hardware'u w postaci sprzętowego protokołu spójności.
- Programista nie musi znać wszystkich jego szczegółów wiedzieć, ale powinien wiedzieć jaki wpływa on na wydajność.
- System jest tak skonstruowany, że procesor jest w stanie wykryć fakt ***współdzielenia*** (ang. sharing) wiersza pamięci podręcznej, tzn. sytuacji w której inny procesor ma ten wiersz w swojej pamięci cache.
- Protokoły dzielą się na dwie grupy: (a) ***invalidate protocols*** (najczęściej stosowane)
  - w momencie zapisu do współdzielonego wiersza wyślij sygnał do procesorów współdzielących ten wiersz aby unieważniły go w swoich pamięciach podręcznych. Oznacza to, że przy spróbują one odczytu z pamięci głównej
  - Przy próbie odczytu z pamięci głównej danych, które zostały zmodyfikowane w swojej pamięci podręcznej procesor przesyła te dane czytającemu procesorowi zastępując pamięć
- (b) ***update protocols***. Przy zapisie do współdzielonego wiersza kopia jest wysyłana innym procesorom.

Porównanie protokołów spójności (Grama i wsp., 2003)



(a)



(b)

Koszt protokołu spójności

- Zapis do wiersza który nie jest współdzielony, odczyt z wiersza współdzielonego wykonywane są z maksymalną szybkością (np. jeden takt zegara do pamięci L1).
- Zapis do wiersza współdzielonego wykonywany jest o wiele wolniej.
  - Trzeba przesłać informację innym procesorom po magistrali procesora co zajmuje co najmniej kilkadziesiąt taktów zegara.
  - W przypadku protokołu typy invalidate inne procesory pobierają informacje przez swoje magistrale.
- Trzeba unikać sytuacji w której:

  a) dane są współdzielone

  b) często dokonujemy do nich zapisu.

Fałszywe współdzielenie (ang. false scharing)

```
int x;
int y;
```

- Niech procesor A cyklicznie odczytuje zmienną x, a procesor B cyklicznie zapisuje zmienną y. Formalnie współdzielenia nie ma.
- Ale pamięć podręczna operuje ***wierszami*** o typowym rozmiarze 64 bajtów. Dwie sąsiednie zmienne prawie na pewno znajdują się w tym samym wierszu. A wiersz ten jest wpółdzielony.
- Aby uniknąć współdzielenia pomiędzy x a y należy wstawić odstęp równy długości wiersza pamięci podręcznej.
  - Linuks to robi !
  - Nie jest to problem akademicki. Przykładowy program wykazuje duże spowolnienie związane ze współdzieleniem.

„Przywiązanie" procesu do procesora (ang. processor affinity)

- Problem dla twórcy systemu operacyjnego (planowanie procesów).
- Rozważmy sytuację w której proces wykonywał się na procesorze A. Następnie był wywłaszczony.
- Obecnie proces został wybrany do wykonania. W systemie wolne są procesory A i B. Na którym procesorze uruchomić proces ?
- Odpowiedź na procesorze A ponieważ w jego pamięci cache mogły pozostać dane/kod procesu.
- Gorzej jest w sytuacji gdy A jest zajęty, a B wolny.
  - Czekamy aż A się zwolni (i B jest bezczynny)
  - Przydzielamy procesor B (i na początku mały współczynn8ik trafień do pamięci cache).

Dygresja: Wpływ sposobu dostępu do pamięci na wydajność (Grama i wsp.)

- Rozważmy kod obliczający w wektorze b sumę wszystkich kolumn kwadratowej macierzy A o wymiarach nxn. n jest duże (powiedzmy ~1000).
- b[0] jest sumą elementów A[0][0],A[1][0],A[2][0], ...

```
for(int i=0;i<n;i++){
   b[i]=0.0;
   for(int j=0;j<n;j++)
      b[i]+=A[j][i];
}
```

- Pamięć jest tablicą jednowymiarową, zaś macierz A tablicą dwuwymiarową. Jak odwzorować ją w jednowymiarowej pamięci ? W języku C macierz jest umieszczona wierszami:

A

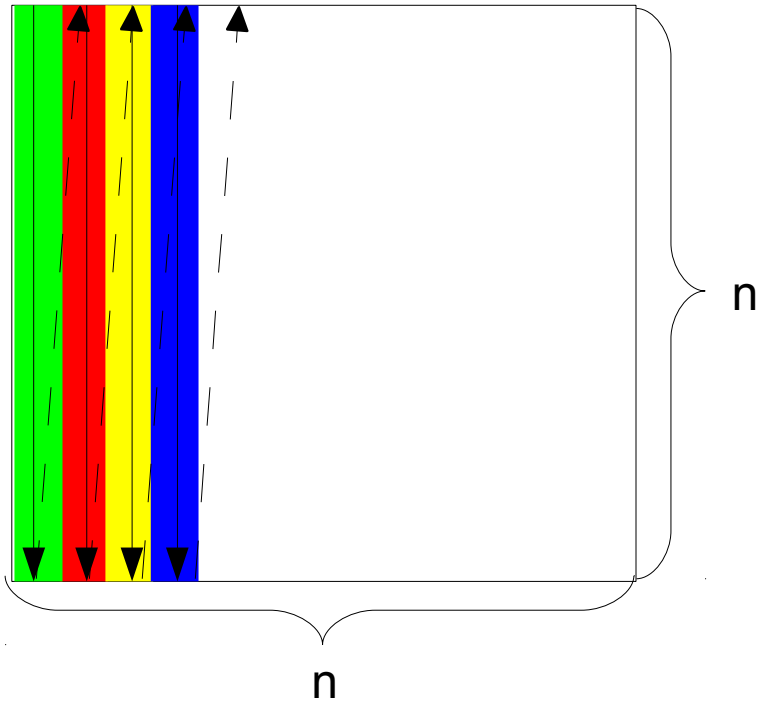
$A_{0,0}$	$A_{0,1}$	$A_{0,2}$
$A_{1,0}$	$A_{1,1}$	$A_{1,2}$
$A_{2,0}$	$A_{2,1}$	$A_{2,2}$

$A_{0,0}$	$A_{0,1}$	$A_{0,2}$	$A_{1,0}$	$A_{1,1}$	$A_{1,2}$	$A_{2,0}$	$A_{2,1}$	$A_{2,2}$

Dostęp do pamięci z dużymi odstępami.

```
for(int i=0;i<n;i++){
   b[i]=0.0;
   for(int j=0;j<n;j++)
      b[i]+=A[j][i];
}
```

- Zakładamy że macierzy nie ma w cache”u i jest za duża aby się tam zmieścić. Wektor B zmieści się w cache”u. Dostęp kolumnami:

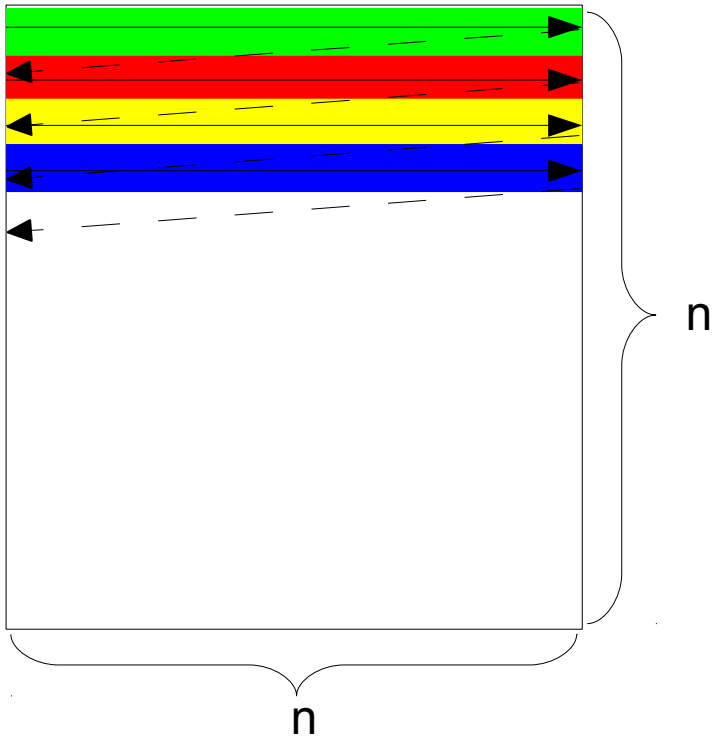


- Dostęp do elementów odległych w pamięci o n miejsc.
- Jeżeli: rozmiar liczby double 8 bajtów, rozmiar wiersza pamięci cache: 128 bajtów, to każdy dostęp generuje (a) chybienie cache-u (b) przeczytanie 15 dodatkowych liczb które są zbędne.

Ten sam algorytm z dostępem kolumnami.

```
for(int i=0;i<n;i++
   b[i]=0.0;
for(int i=0;i<n;i++){
   for(int j=0;j<n;j++)
      b[j]+=A[i][j];
}
```

- Dostęp kolumnami wierszami



- Dostęp do kolejnych elementów w pamięci.
- Jeżeli: rozmiar liczby double 8 bajtów, rozmiar wiersza pamięci cache: 128 bajtów, to co szesnasty dostęp generuje (a) chybienie cache-u (b) przeczytanie 15 dodatkowych liczb które zostaną wykorzystane w następnych iteracjach.
- Przy założeniu że czas dostępu do pamięci dominuje nad czasem pozostałych operacji ta metoda jest 15 razy szybsza !!!.
  - Obydwie mają złożoność O(nxn).

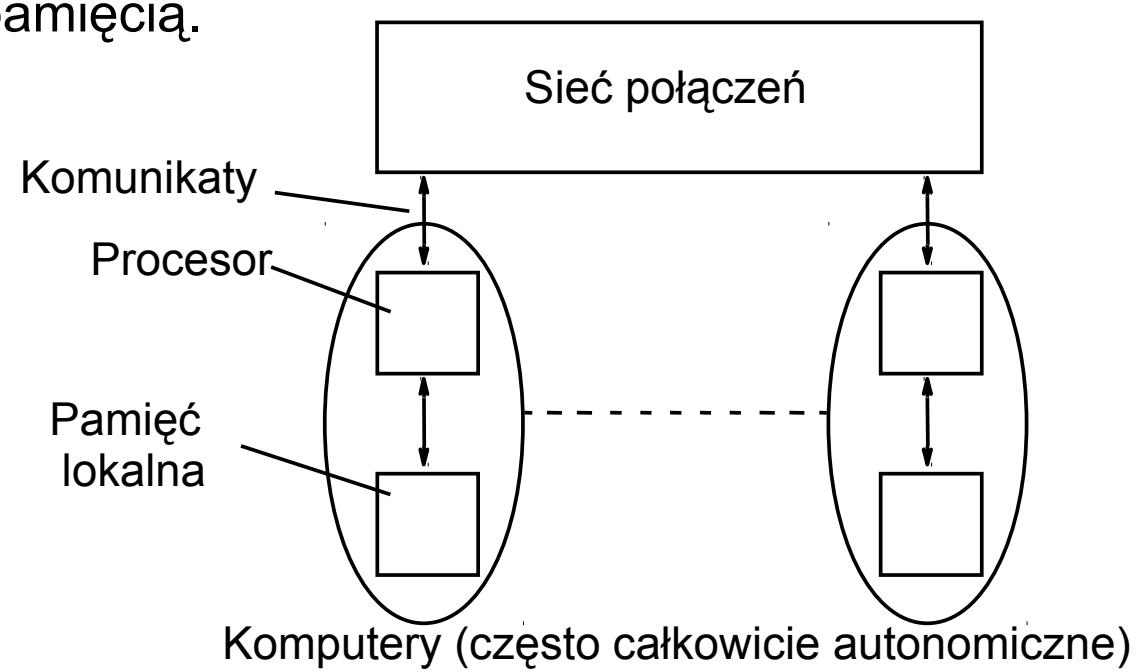
Jaki jest morał

- Asymptotyczna złożoność to nie wszystko (choć jest ważna)

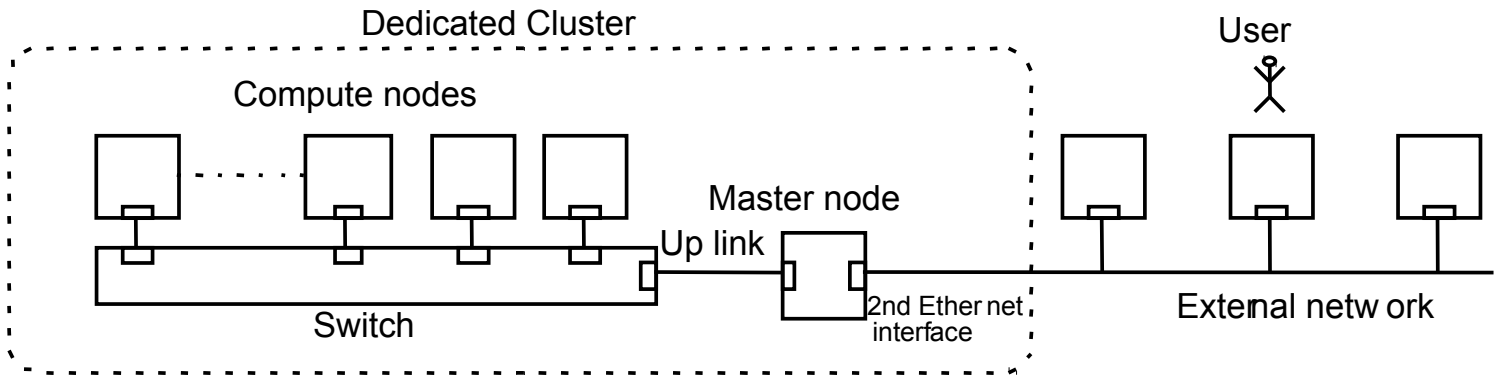
- Przed rozpoczęciem zrównoleglania sprawdź czy optymalnie zaimplementowałeś algorytm szeregowy.

System z przesyłaniem komunikatów (ang. message passing)

- Również zwany systemem ***z przesyłaniem komunikatów*** (ang. message passing).
- Często klastry (ang. cluster) niezależnych systemów komputerowych.
- Każdy procesor ma dostęp wyłącznie do swojej lokalnej pamięci.
- Współpraca z innymi procesorami odbywa się poprzez wymianę komunikatów
- Sieć połączeń: dziś co najmniej Gigabit Ethernet (1Gb/s), często Infiniband (10 Gb/s lub 20 Gb/s) lub wyspecjalizowane rozwiązania.
- Możliwe rozwiązanie hybrydowe: każdy komputer jest systemem wieloprocesorowym ze wspólną pamięcią.

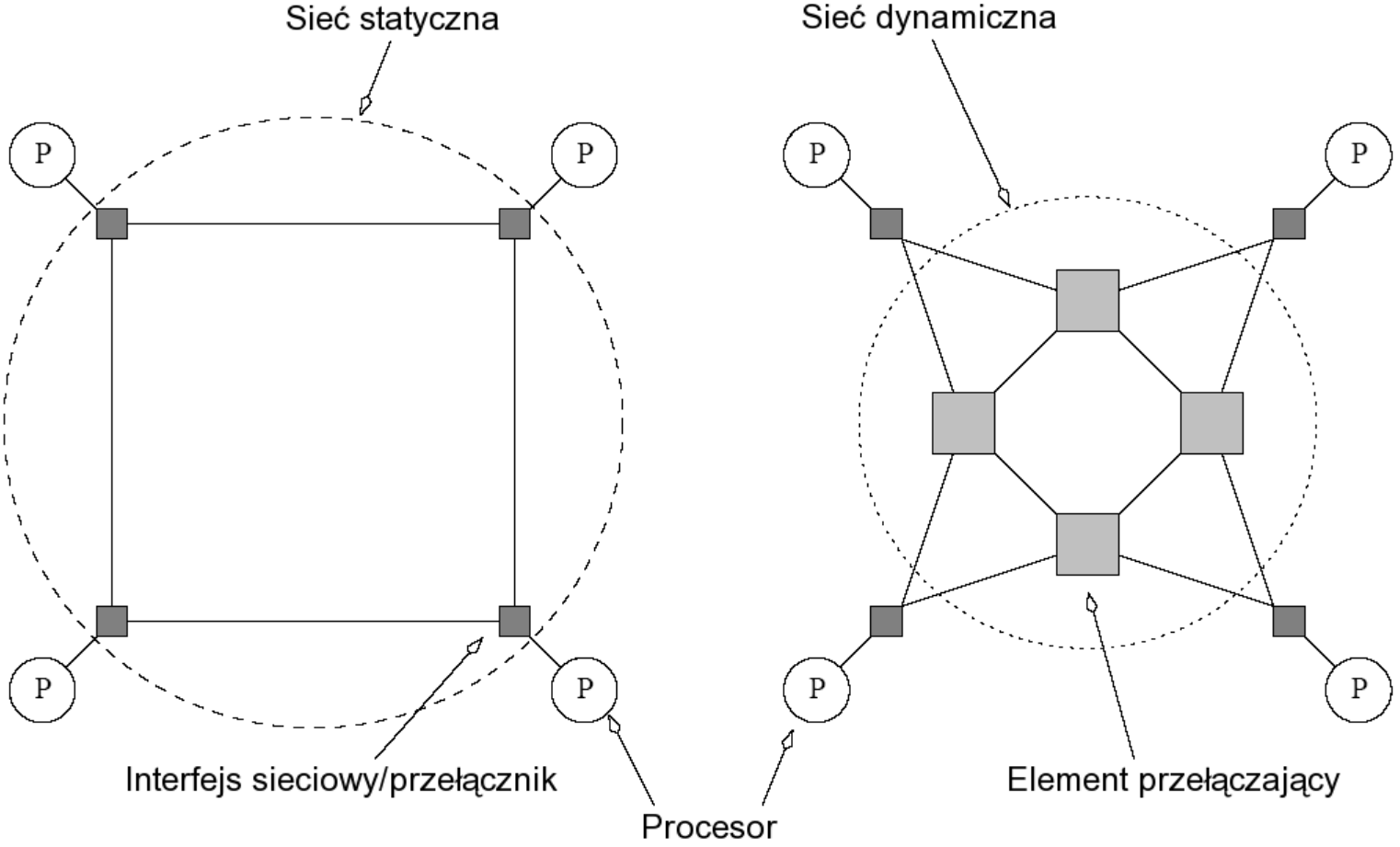


Klaster niezależnych maszyn (rys. Wilkinson i Allen)



- Węzełeł zarządzający (master node) pełni funkcję firewalla izolując węzły obliczeniowe od internetu.
- Węzły obliczeniowe to komputery zdolne do samodzielnej pracy (W naszym laboratorium maszyny typu desktop).
- Użytkownicy logują się z publicznego internetu na węzeł zarządzający.
- Każdy węzeł obliczeniowy jest komputerem zdolnym do autonomicznej (samodzielnej) pracy z własnym systemem operacyjnym.

Dynamiczne i statyczne sieci połączeń



- Sieć statyczna – połączenia punktowe pomiędzy procesorami
- Sieć dynamiczna – zbudowana przy pomocy połączeń i elementów przełączających.
  - Szyna i przełącznik krzyżowy to sieci dynamiczne

Parametry połączenia komunikacyjnego

- Czas t_{comm} przesyłania komunikatu możemy wstępnie przybliżyć równaniem:

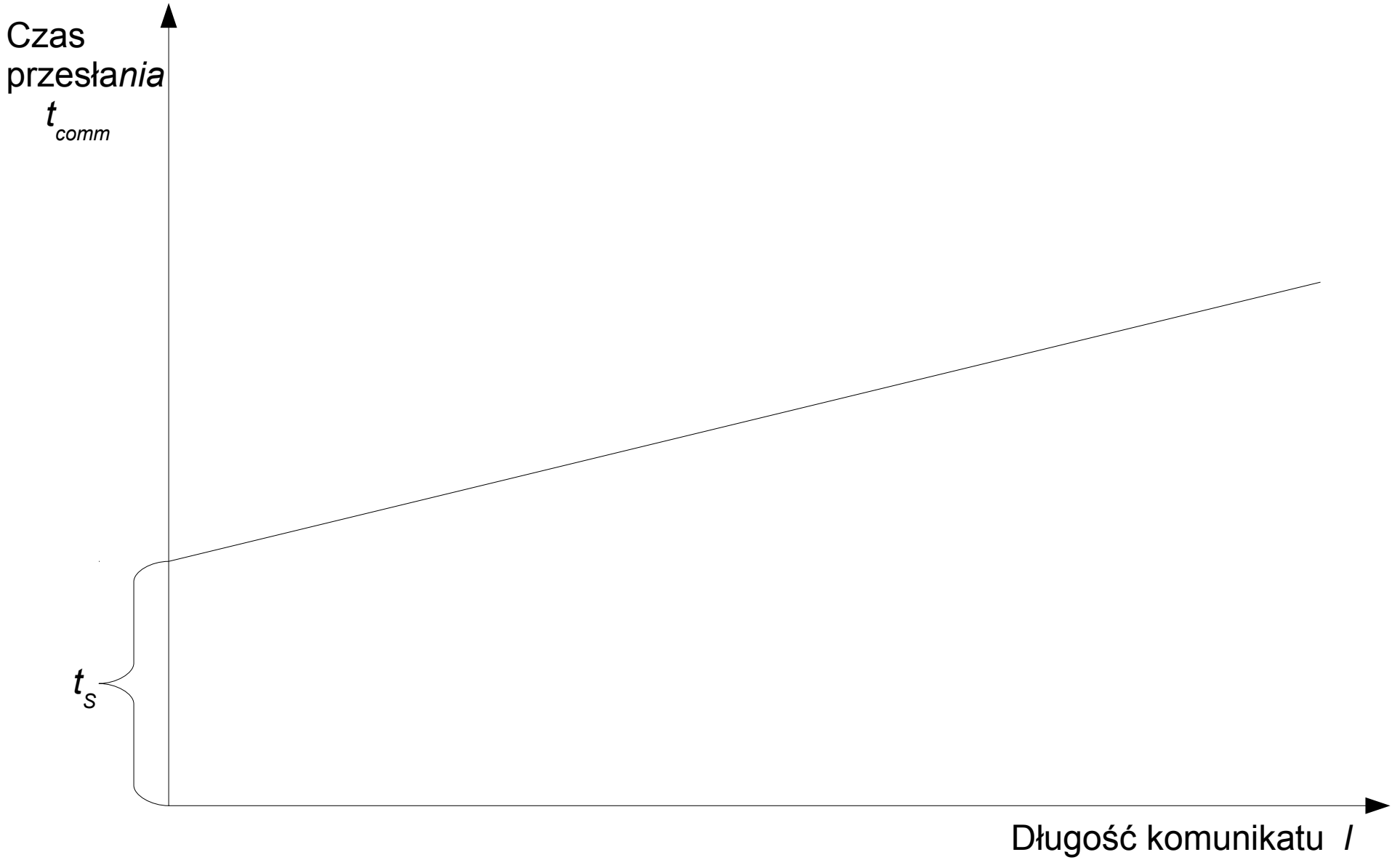
$$t_{comm} = t_S + t_B * l$$

gdzie

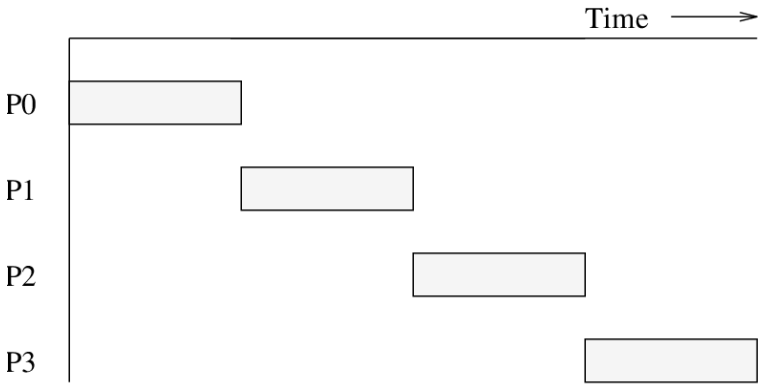
  - t_S (startup time, latency - opóźnienie) jest czasem potrzebnym na obsługę komunikatu u nadawcy i odbiorcy. Zalicza się do niego czas opóźnienia programowego, przekazania komunikatu do interfejsu sieciowego, skonfigurowania łącza, wykonania algorytmu routującego.
  - t_B jest czasem transmisji jednego bajtu, włączając narzuty sprzętowe (sieć) i programowe. $1/t_B$ jest przepustowością łącza w bajtach na sekundę.
  - l jest długością komunikatu w bajtach.

- To przybliżenie pomija wiele parametrów, między innymi:
  - natężenie ruchu w sieci połączeń.
  - odległość pomiędzy nadawcą i odbiorcą (w sensie liczby procesorów w sieci statycznej albo elementów przełączających w sieci dynamicznej).

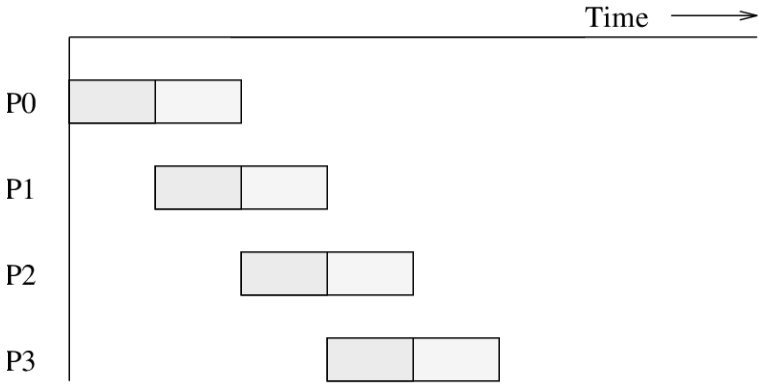
Czas transmisji komunikatu



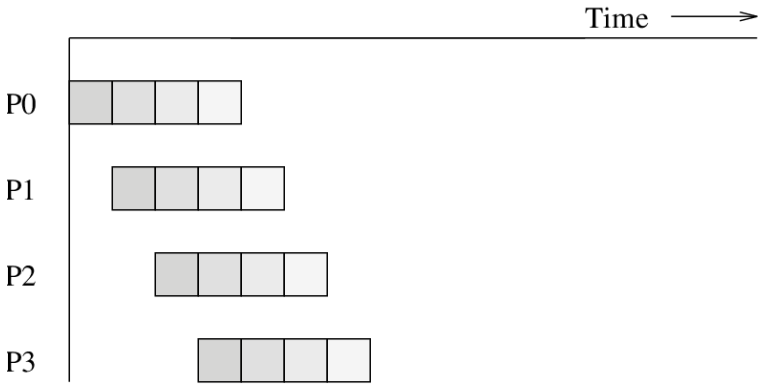
Techniki wyznaczania trasy (ang. routing) z uwzględnieniem liczby hopów (Grama i wsp. 2003)



(a) A single message sent over a store-and-forward network



(b) The same message broken into two parts and sent over the network.



(c) The same message broken into four parts and sent over the network.

Komunikaty zdominowane przez opóźnienie i przepustowość

- Komunikat o długości l_g, dla której $t_S=t_B*l_g$ (stąd $l_g=t_S/t_B$) jest komunikatem granicznym.
- Komunikaty o długości krótszej l_g od są zdominowane przez opóźnienie.
- Komunikaty dłuższe l_g niż są zdominowane przez przepustowość.

- Na przykład dla klastra Mordor 2 l_g=3.8us/(1/1400MB/s)= około 5500 bajtów.

Przegląd niektórych sieci połączeń

- Sieci statyczne
  - Sieć kompletna i gwiazda
  - Hipersześcian
  - Sieci typu mesh
- Sieci stosowane przy budowie klastrów

Sieci typu mesh (Grama i wsp.)

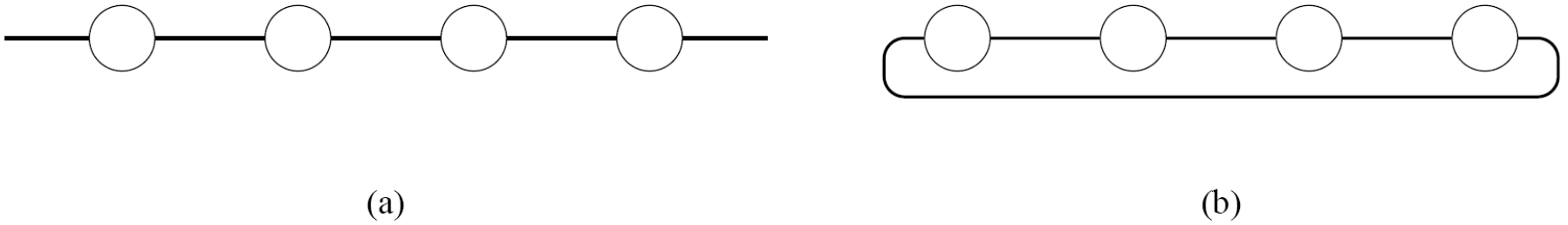


Figure 2.15 Linear arrays: (a) with no wraparound links; (b) with wraparound link.

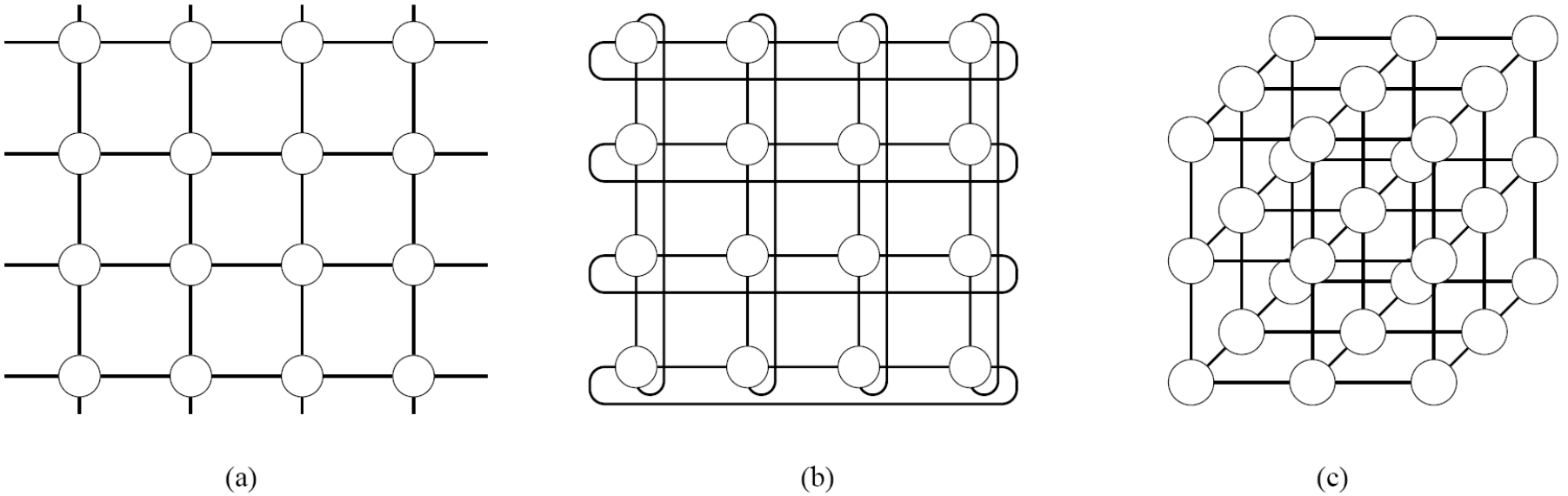
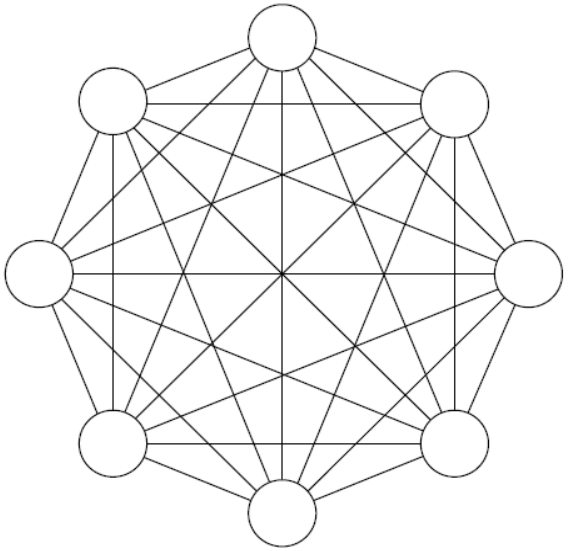
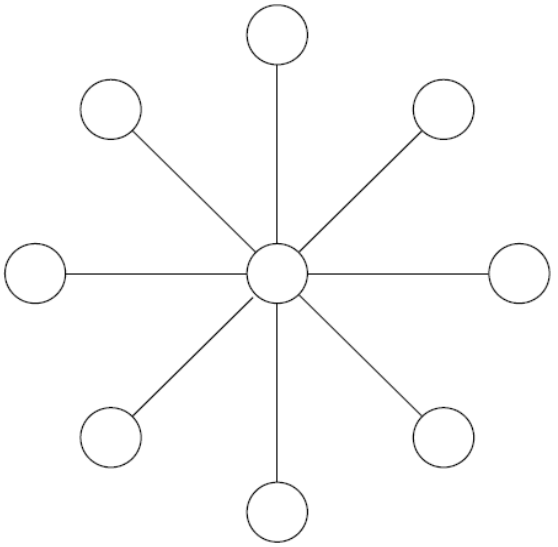


Figure 2.16 Two and three dimensional meshes: (a) 2-D mesh with no wraparound; (b) 2-D mesh with wraparound link (2-D torus); and (c) a 3-D mesh with no wraparound.

Sieć kompletna i gwiazda



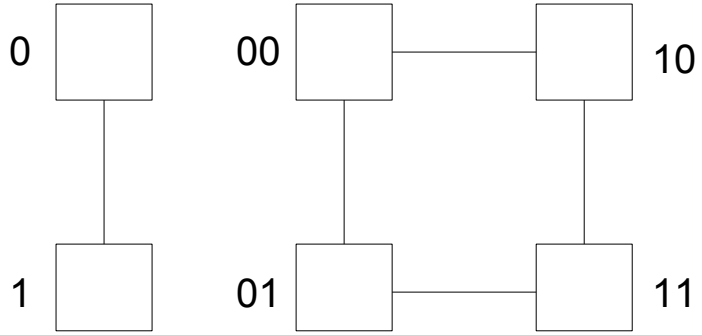
Sieć kompletna

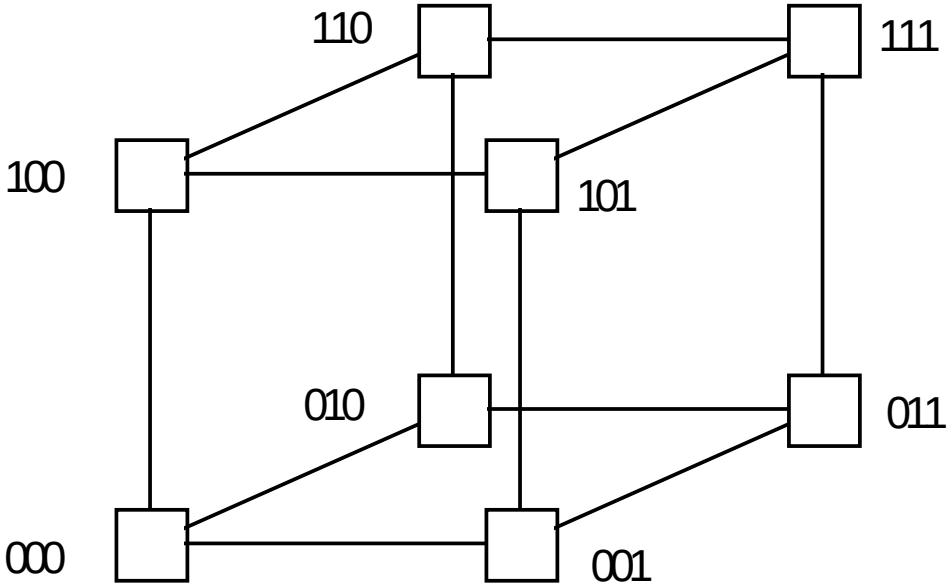


Gwiazda

- Sieć kompletna – statyczny odpowiednik przełącznika krzyżowego.
- Gwiazda – statyczny odpowiednik szyny.

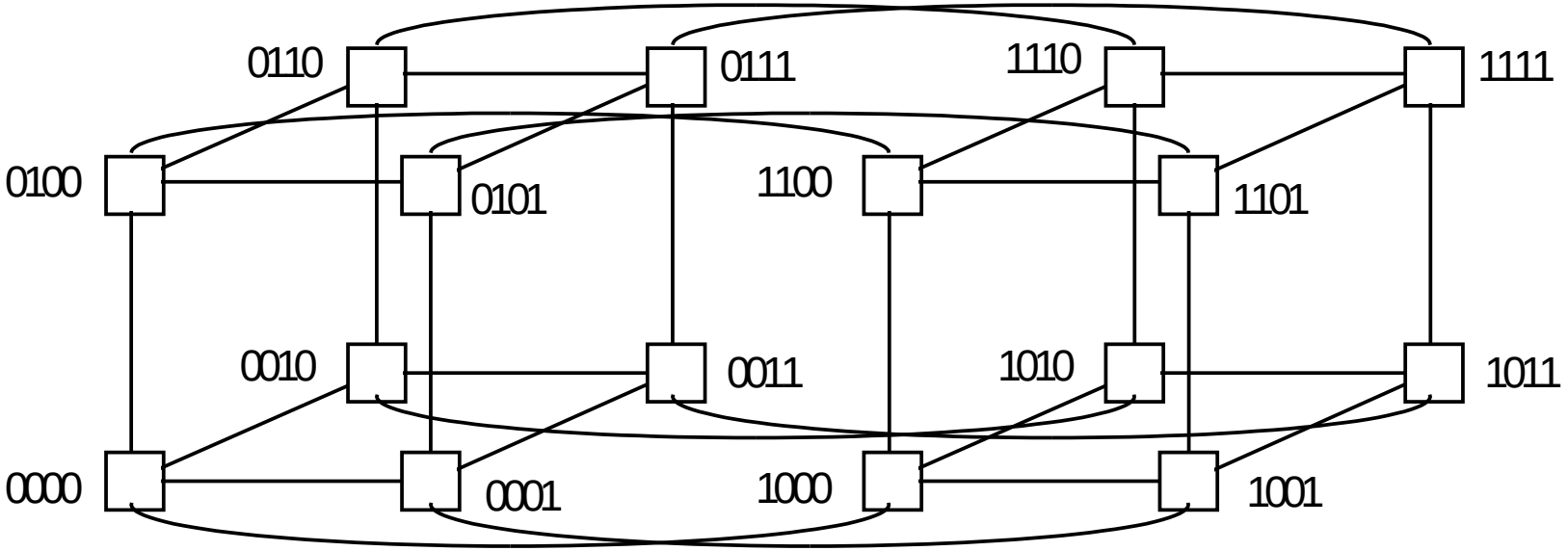
1,2,3-wymiarowy hipersześcian





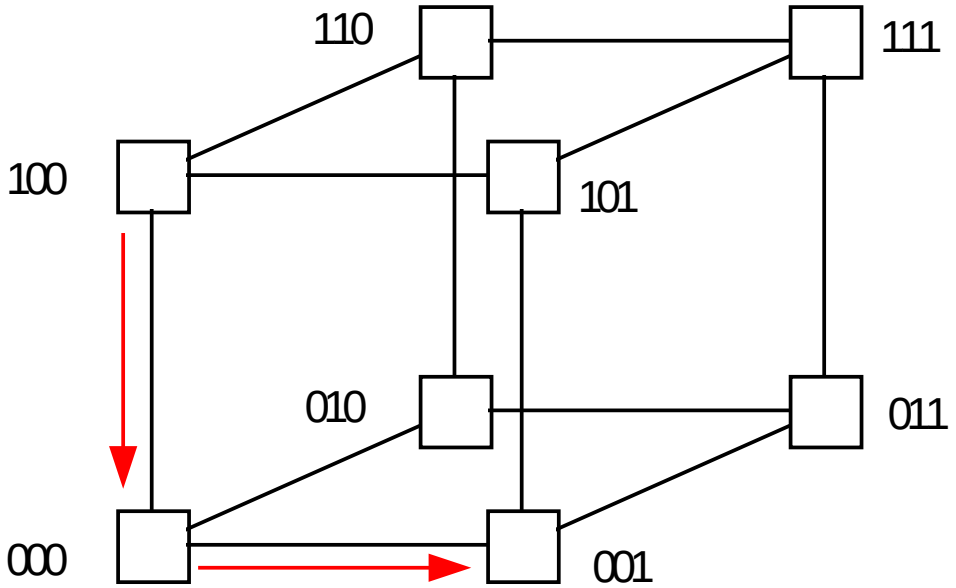
- k-wymiarowy hipersześcian ma k-połączeń wychodzących z każdego z wierzchołków.
- Algorytm routujący komunikaty jest stosunkowo prosty.
- Kiedyś była to bardzo popularna sieć połączeń.

4-wymiarowy hipersześcian



- Aby skonstruować k+1-wymiarowy hipersześcian
  - zestawiamy dwa k-wymiarowe hipersześciany, łączymy węzły o tych samych numerach.
  - dodajemy nową cyfrę binarną w jednym hipersześcianie równą jeden a w drugim zero.

Algorytm wyznaczania trasy (E-cube routing)

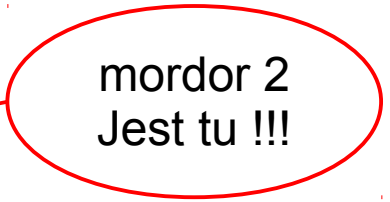


- P_s (100) – wierzchołek źródłowy, P_d (001) – wierzchołek docelowy.
- Krok 1: P_s XOR P_d = 101 (1 oznacza, że pakiet musi „podróżować przez wymiar”)
  - Zmieniamy wymiar począwszy od najstarszej pozycji ustawionej na 1
  - 100->000
- Krok 2: P_s =000, P_d=001, P_s XOR P_d=001
  - 000->001

Sieci połączeń stosowane przy budowie klastrów

- Obecnie minimum to Gigabit Ethernet .
  - Przepustowość około 1 Gb/s , opóźnienie to 50 µs.
  - W sali 225 włączone są ramki jumbo (9000 bajtów)
- Infiniband: sieć przełączana o b. małym opóźnieniu (rzędu pojedynczych µs) i o b. dużej przepustowości.

Krotność	SDR	DDR	QDR	FDR	EDR
1X	2	4	8	13.64	25
4X	8	16	32	54.54	100
12X	24	48	96	116.36	300



Efektywna teoretyczna przepustowość w Gb/s

- Na klastrze mordor2 zmierzona przepustowość w teście ping-pong to nieco ponad 1400 MB/s (najtańsze karty HCA ? stary chipset ?) a opóźnienie to 3.8 µs.

Sieci połączeń stosowane przy budowie klastrów

- Obecnie minimum to Gigabit Ethernet .
  - Przepustowość około 1 Gb/s , opóźnienie to 50 µs.
  - W sali 225 włączone są ramki jumbo (9000 bajtów)
- Infiniband: sieć przełączana o b. małym opóźnieniu (rzędu pojedynczych µs) i o b. dużej przepustowości.

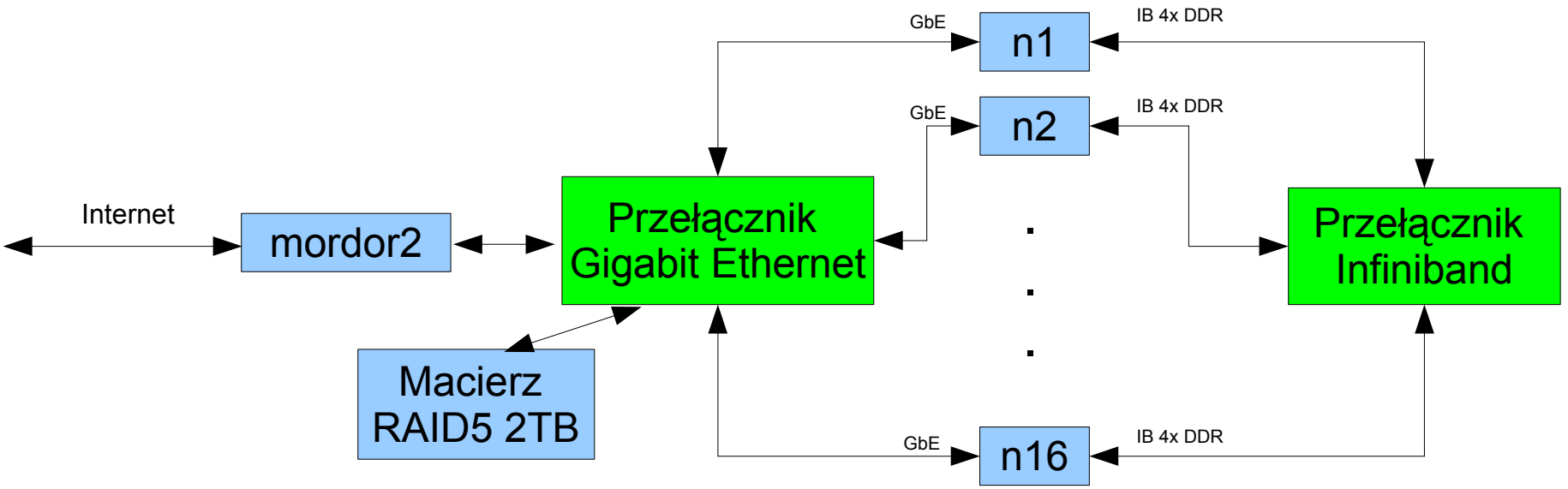
Krotność	SDR	DDR	QDR	FDR	EDR
1X	2	4	8	13.64	25
4X	8	16	32	54.54	100
12X	24	48	96	116.36	300

mordor 2
Jest tu !!!

Efektywna teoretyczna przepustowość w Gb/s

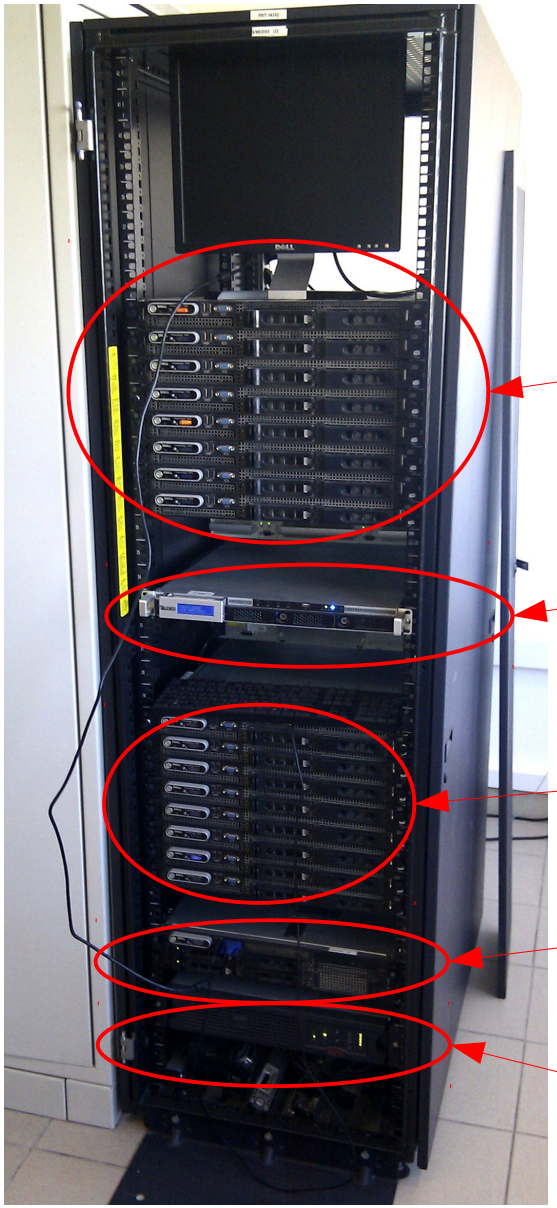
- Na klastrze mordor2 zmierzona przepustowość w teście ping-pong to nieco ponad 1400 MB/s (najtańsze karty HCA ? stary chipset ?) a opóźnienie to 3.8 µs.

Klaster Obliczeniowy mordor2



- Węzeł zarządzający (mordor 2)
  - Dell PowerEdge 2950 (16GB RAM, 2xXeon 5150 @ 2.66 GHz, cztery rdzenie łącznie).
  - Sieciowy system kont, sieciowy system plików, firewall + NAT dla węzłów obliczeniowych, system kolejkowy slurm, fully automated install (FAI), kompilator icc, MPI over Infiniband.
  - Login przez ssh, brak GUI
- Węzły obliczeniowe (n1-n16)
  - Dell PowerEdge 1950 (16GB RAM 2xXeon X5355 @ 2.66GHz, 8 rdzeni łącznie)

Mordor 2



8 węzłów obliczniowych
(Obudowa rack 19” 1U)

Sieciowa pamięć dyskowa
(Obudowa rack 19” 1U)

8 węzłów obliczniowych
(Obudowa rack 19” 1U)

Węzeł zarządzający
Obudowa rack 19” 2U

UPS dla
węzła zarządzającego

Parametry sieci

- Średnica - jest to maksymalna odległość pomiedzy dwiema węzłami w sieci.
- Liczba połączeń wychodzących z wierzchołka.
- Koszt sieci - całkowita liczba połączeń.
- Szerekość bisekcji (ang. bisection width). Minimalna liczba połączeń, które należy usunąć, aby podzielić sieć na dwie równe połowy.

Parametry różnych sieci połączeń (Grama i wsp.)

Nazwa Sieci	Średnica	Szerokość bisekcji	Liczba połączeń wierzchołka	Całkowita liczba połączeń
Kompletna	1	$p^2/4$	p-1	p(p-1)/2
Gwiazda	2	1	1	p-1
Linia	p-1	1	1	p-1
Drzewo binarne	2*log((p+1)/2)	1	1	p-1
2D-mesh	2*(sqrt(p)-1)	sqrt(p)	2	2(p-sqrt(p))
2D-torus	sqrt(p)	2*sqrt(p)	4	2p
Hipersześcian	log(p)	p/2	log(p)	p*log(p)/2
Przełącznik krzyżowy	1	p	1	p^2

- p jest liczbą węzłów (procesorów).

Systemy ze wspólną pamięcią a systemy z przesyłaniem komunikatów

- Budowa systemu z przesyłaniem komunikatów jest łatwiejsza (nie jest wymagany specjalizowany sprzęt, oprócz sieci).

- W systemach z przesyłaniem komunikatów koszty komunikacji są jawne, w systemach ze wspólną pamięcią ukryte (np. w postaci protokołu spójności).

- System ze wspólną pamięcią może łatwo emulować system z przesyłaniem komunikatów odwrotność jest bardzo trudna w implementacji (wydajnie).

- Systemy z przesyłaniem komunikatów potencjalnie się lepiej skalują, ale ze względu na ograniczenie kosztów często buduje się systemy hybrydowe (np. mordor2 klaster systemów ze wspólną pamięcią połączonych siecią).

- Które systemy łatwiej programować ? Zdania są podzielone.