Algoritmos para Troca Completa de Mensagens de Diferentes Tamanhos

Fábio Massaaki Katayama

katayama@ime.usp.br

Orientador: Alfredo Goldman

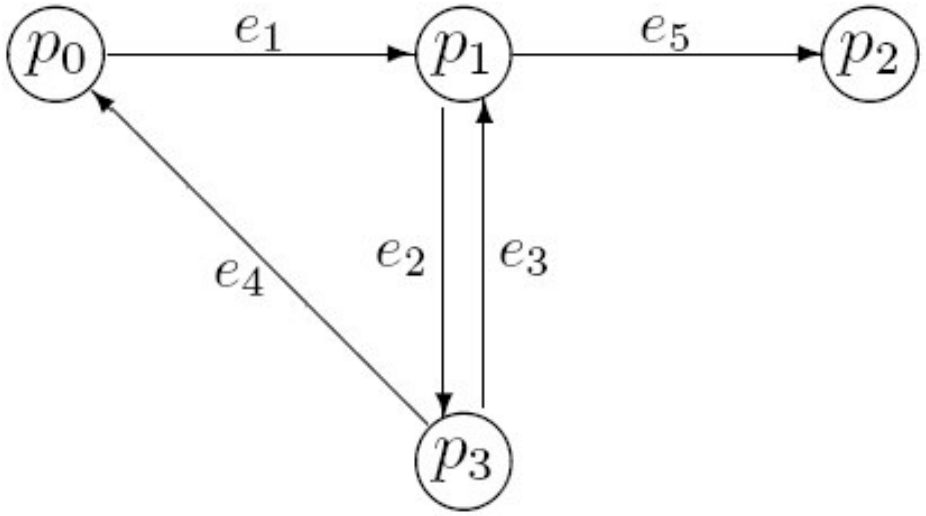
Motivação

- No modelo BSP, algoritmos alternam fases de computação e fases de comunicação.
- Em cada fase de comunicação, todos os processadores trocam dados entre si
- Uma boa implementação BSP necessita de um bom algoritmo de troca de dados

Descrição do Problema

- n processadores interligados
- Representações:
  - MED (*Message Exchange Digraph*)
  - grafo bipartido
  - Matriz
- Objetivo: minimizar o tempo para trocar todas as mensagens

MED – *Message Exchange Digraph*



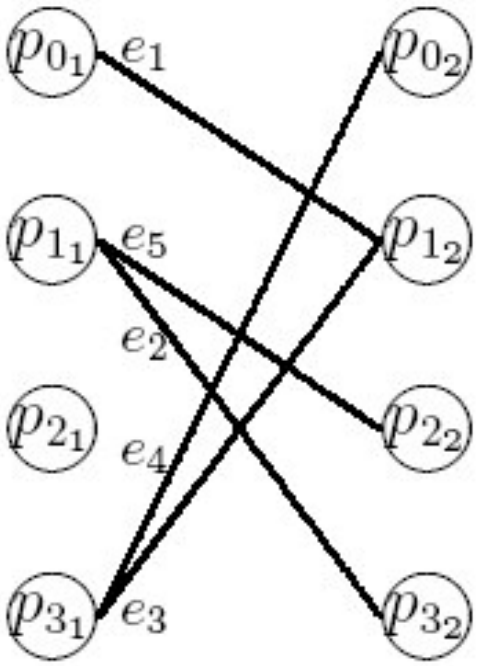
- $dG(V, E)$
- Grafo dirigido com peso
- $e_1 = (p_0, p_1)$: mensagens de p_0 para p_1
- $e_i \rightarrow w(e_i)$: tamanho da mensagem e_i

Matriz

$$\begin{pmatrix} 0 & w(e_1) & 0 & 0 \\ 0 & 0 & w(e_5) & w(e_2) \\ 0 & 0 & 0 & 0 \\ w(e_4) & w(e_3) & 0 & 0 \end{pmatrix}$$

- $W_{n,n}$: matriz de pesos
- $w(i, j)$: tamanho da mensagem a ser enviada de p_i para p_j

Grafo Bipartido



- $bG(V', E')$
- $V' = V_1 \; U \; V_2$
- $e' = (p_{i_1}, p_{j_2}), \forall e \in E$
- $w(e') = w(e)$

Características do Modelo

- Modelo de Comunicação:
  - *links* bidirecionais
  - *1-port full-duplex*
- Modelo de Transmissão: $\beta_{i,j} + w_{i,j}\tau_{i,j}$
  - β : *startup time*
  - $1/\tau$: largura da banda
- Modelo de Sincronização:
  - síncrono ou assíncrono

Tipos de Transmissão

- *message splitting*: possibilidade de quebrar as mensagens e enviá-las em várias transmissões.

- *message forwarding*: mensagens são encaminhadas através de processadores intermediários

Ambientes Homogêneos

- $\beta_{i,j} = \beta$ e $\tau_{i,j} = \tau$, para todo $0 \leq i, j < n$
- Então:

$$\beta + w_{i,j}\tau$$

Algoritmo Padrão Fixo

- Simples
- Não utiliza *message splitting*
- Não utiliza *message forwarding*

Padrão Fixo (2)

- for t = 1 to n-1 do

  do in parallel for all i ($0 \leq i \leq n-1$)

  p_i sends message addressed to $p_{(i+t)\%n}$

  p_i sends message addressed to $p_{(i-t)\%n}$

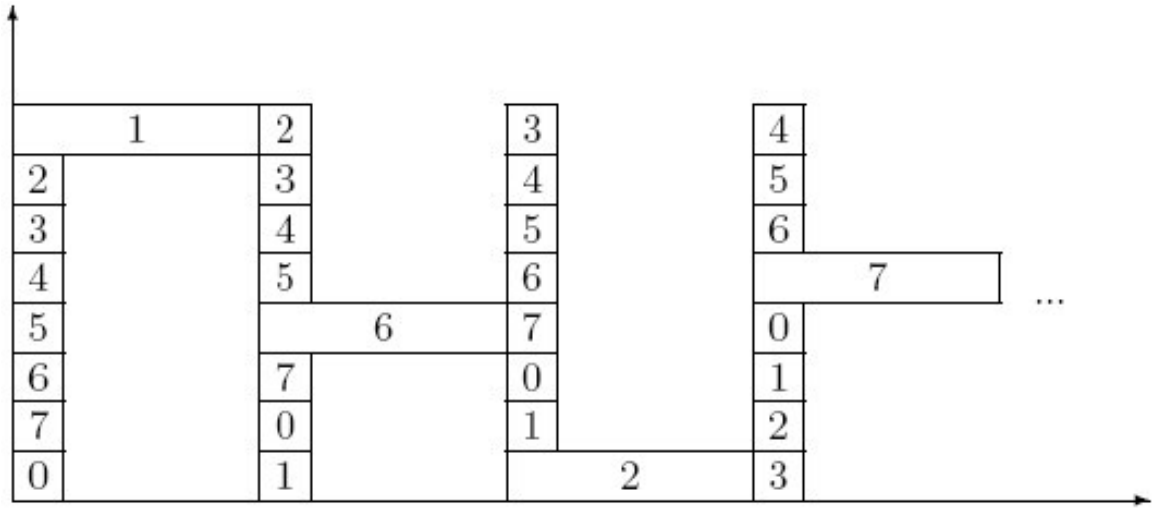
Padrão Fixo - Análise

- fase t:
  - mensagens trocadas: $w_{i,(i+t) \bmod n}$
  - tempo: $\beta + \max_i \{w_{i,(i+t) \bmod n}\}\tau$

- total:
  - *n-1* fases
  - $(n-1)\beta + (n-1)M\tau, \qquad M = \max_{i,j}\{w_{i,j}\}$

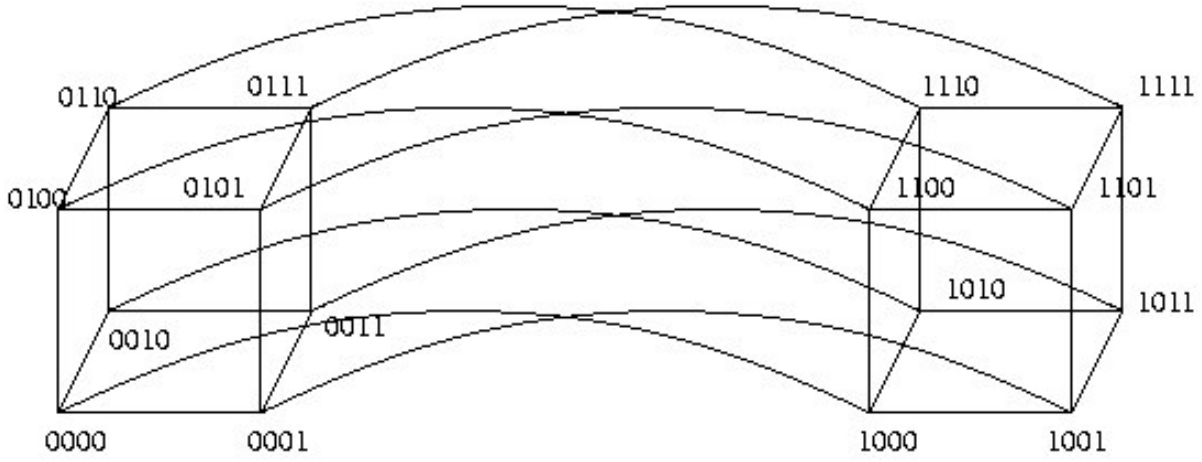
Padrão Fixo – Pior Caso

$$\begin{pmatrix} 0 & M & S & S & S & S & S & S \\ S & 0 & S & S & S & S & S & S \\ M & S & 0 & S & S & S & S & S \\ S & S & S & 0 & S & S & S & M \\ S & S & S & S & 0 & S & M & S \\ S & S & S & S & M & 0 & S & S \\ S & S & S & M & S & S & 0 & S \\ S & S & M & S & S & S & S & 0 \end{pmatrix}$$



Algoritmo para Hipercubos

- sem *message splitting*
- *utiliza message forwarding*

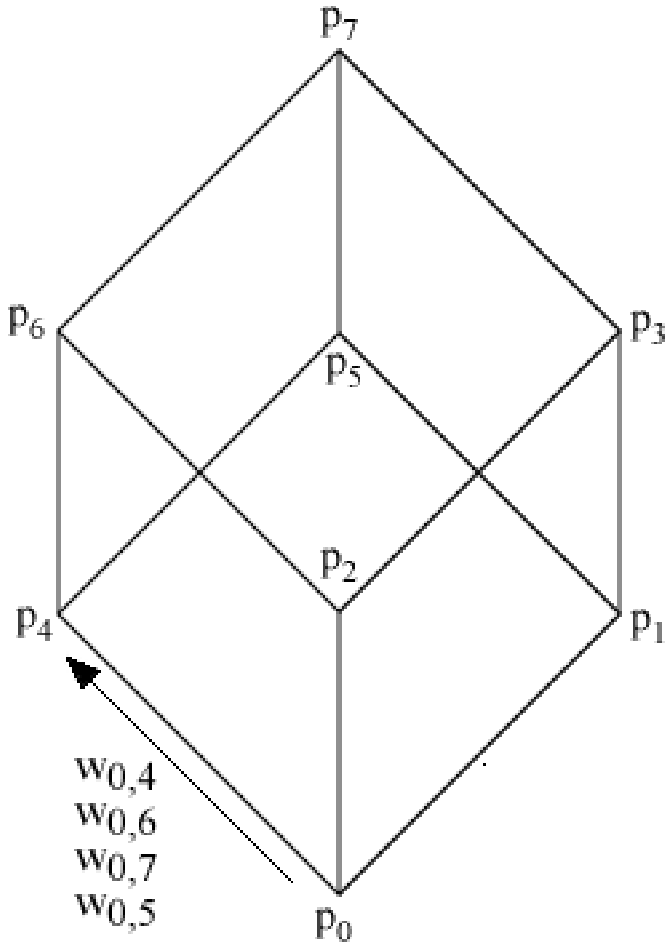


Hipercubo

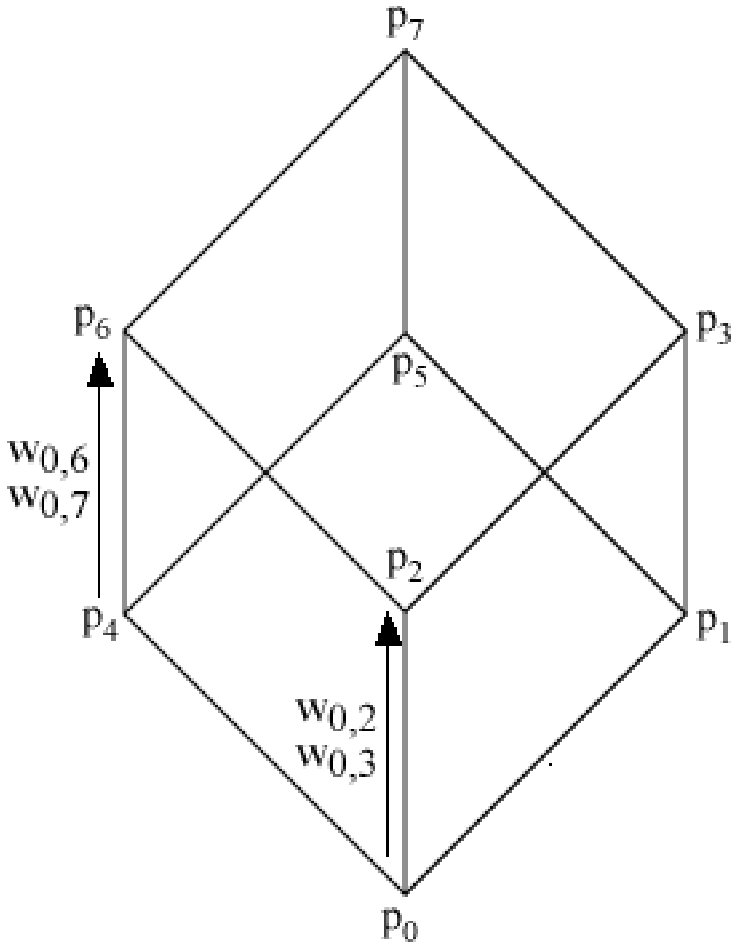
- p_i^t representa o processador que difere de p_i no t-ésimo bit.
- Portanto: p_i e p_i^t são adjacentes

para $t = 1$ até $log(n)$ faça
    faça em paralelo para todo $0 \leq i < n$
        p_i troca todas mensagens necessárias com p_i^t

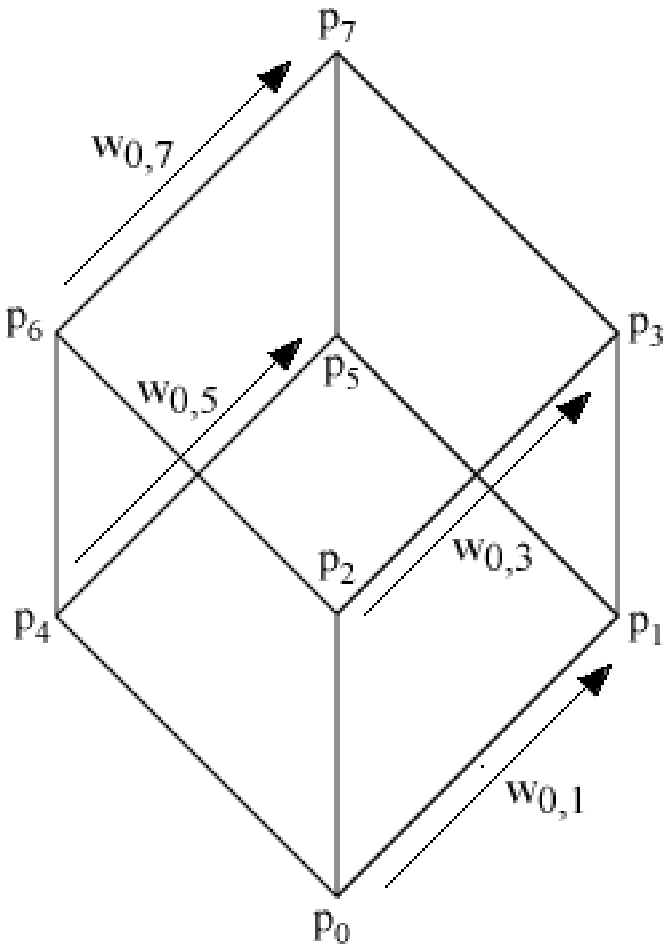
Hipercubo – Fase 1



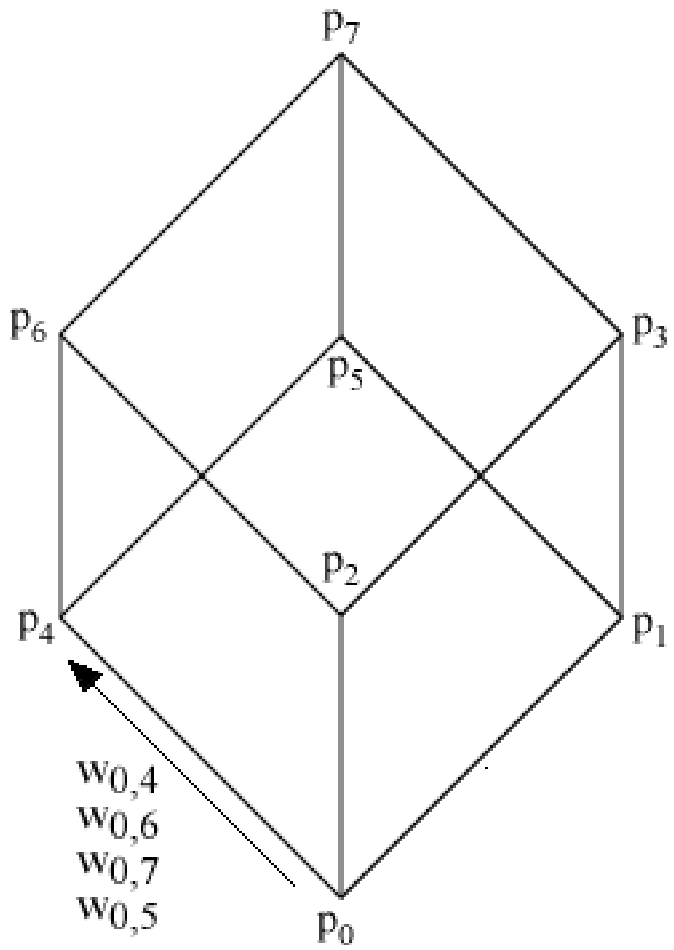
Hipercubo – Fase 2



Hipercubo – Fase 3

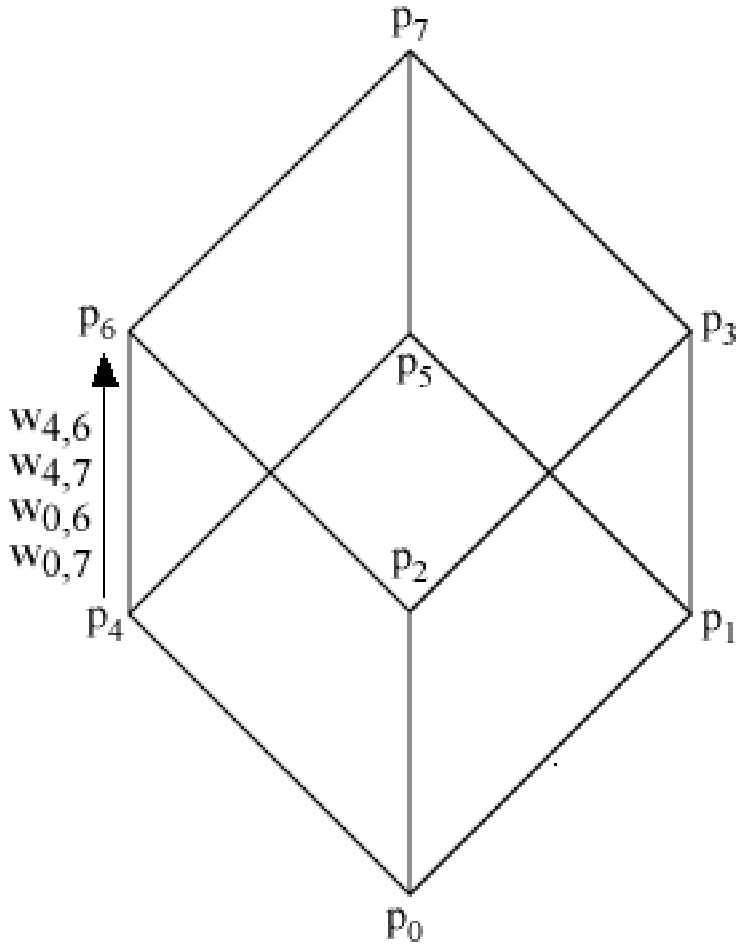


Hipercubo - Análise



- Na fase 1, cada processador envia:
- n/2 = 4 mensagens de suas mensagens
- 0 mensagens encaminhadas
- Total: n/2 mensagens

Hipercubo - Análise



- Na fase 2, cada processador envia:
- n/4 = 2 mensagens de suas mensagens
- n/4 = 2 mensagens encaminhadas
- Total: n/2 mensagens

Hipercubo - Análise

Em geral, na fase t cada processador envia:

- $n/2^t$ mensagens próprias
- $\sum_{j=2}^{t} \frac{n}{2^j} = \frac{n}{2} - \frac{n}{2^t}$ mensagens encaminhadas

Hipercubo - Análise

- Em cada fase, o tempo de comunicação é limitada por:

$$\beta + \frac{n}{2} M\tau, \; onde \;\; M = \max_{i,j}\{w_{i,j}\}$$

- Tempo Total:

$$\log_2 n\beta + \log_2 n \frac{n}{2} M\tau$$

Algoritmo Padrão Central

- Precisa conhecer os tamanhos das mensagens para determinar o escalonamento.

  All-to-all no tamanho das mensagens

  Determina globalmente o escalonamento

  Executa o escalonamento

Padrão Central – Estágio 1

- Pré-processamento para troca dos tamanhos das mensagens
- MEP com mensagens de mesmo tamanho: *sizeof(int)(n-1)*

Padrão Central – Estágio 1

Padrão Fixo	$(n-1)\beta + sizeof(\text{int})(n-1)^2\tau$
Hipercubo	$\log_2 n\beta + \log_2 n\frac{n}{2} sizeof(\text{int})(n-1)\tau$

Padrão Central

- Determinar emparelhamentos em *bG(V, E)*
- *Cada emparelhamento representará uma fase de comunicação*

- *Algoritmos diferem na estratégia de encontrar emparelhamentos no grafo*

Padrão Central

- MM: Max-Min Algorithm
  - Maximiza o tamanho da menor mensagem do emparelhamento
- MS: Max-Sum Algorithm
  - Maximiza a soma do tamanho das mensagens do emparelhamento

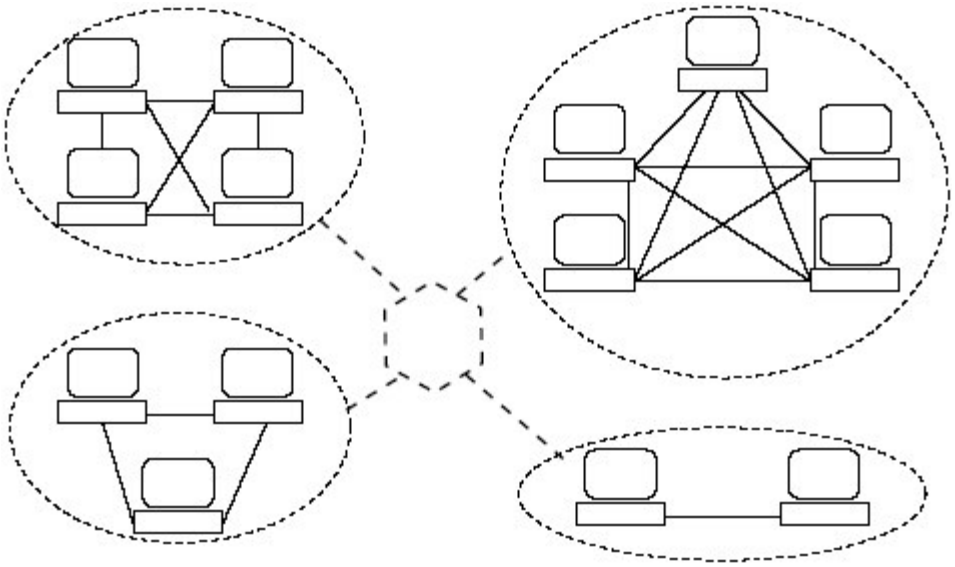
Padrão Central

- Unif: Uniform Algorithm
  - Obter fases de comunicação onde todas as mensagens tem o mesmo tamanho
  - Utiliza *message splitting*
  - Gera matriz *Q* com linhas e colunas somando α
  - Aplica *Max-Min* utilizando a matriz *Q*

Desempenho

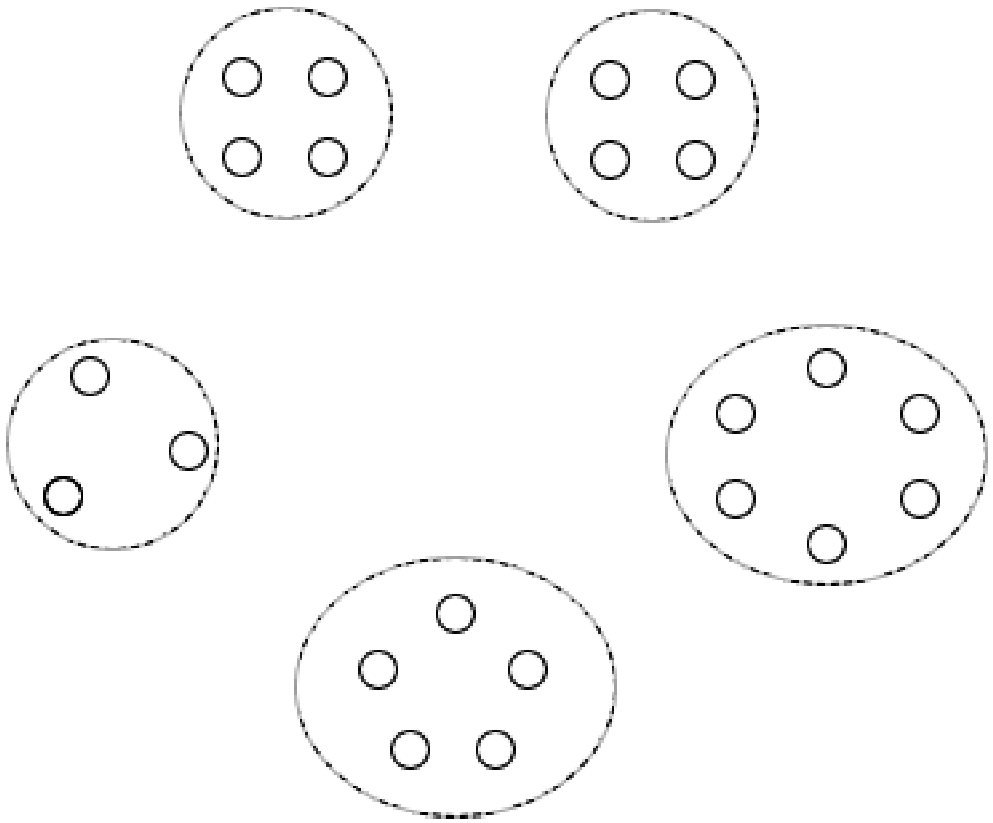
- Hipercubo minimiza o número de fases de comunicação
- Uniforme minimiza o fator *bandwidth*
- *MM, MS e FP dependem dos tamanhos das mensagens*

Multi-Clusters

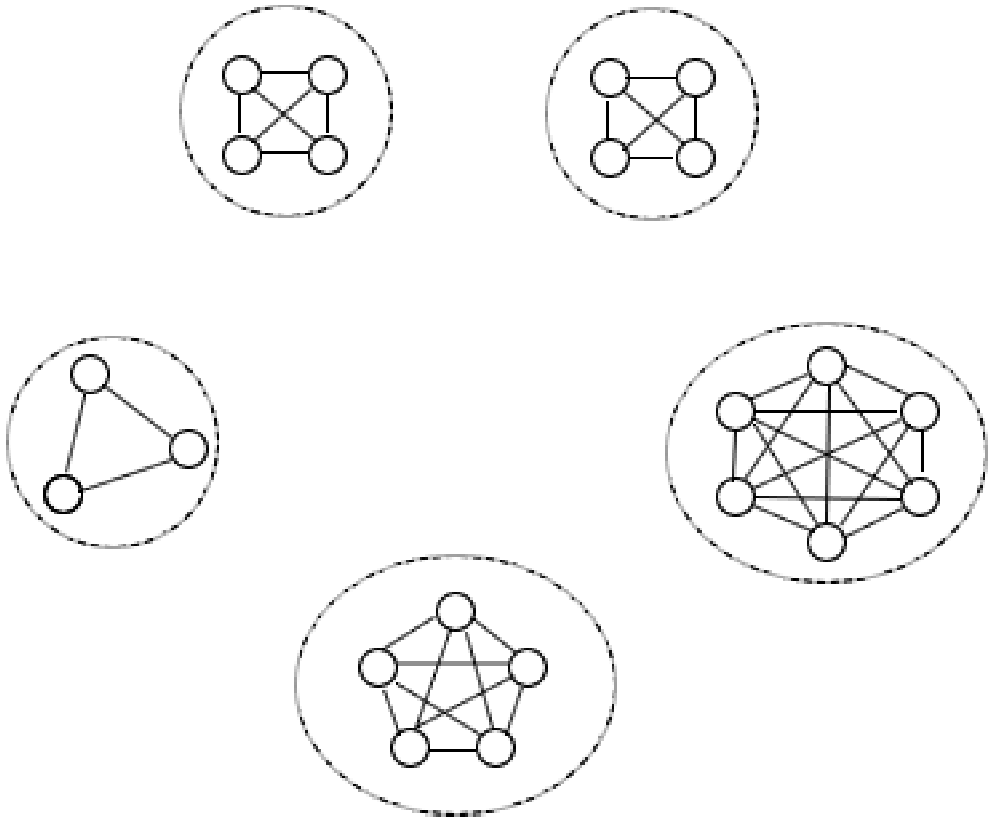


- Trocar mensagens dentro do clusters
- Trocar mensagens entre os clusters
- Distribuir mensagens dentro de cada clusters

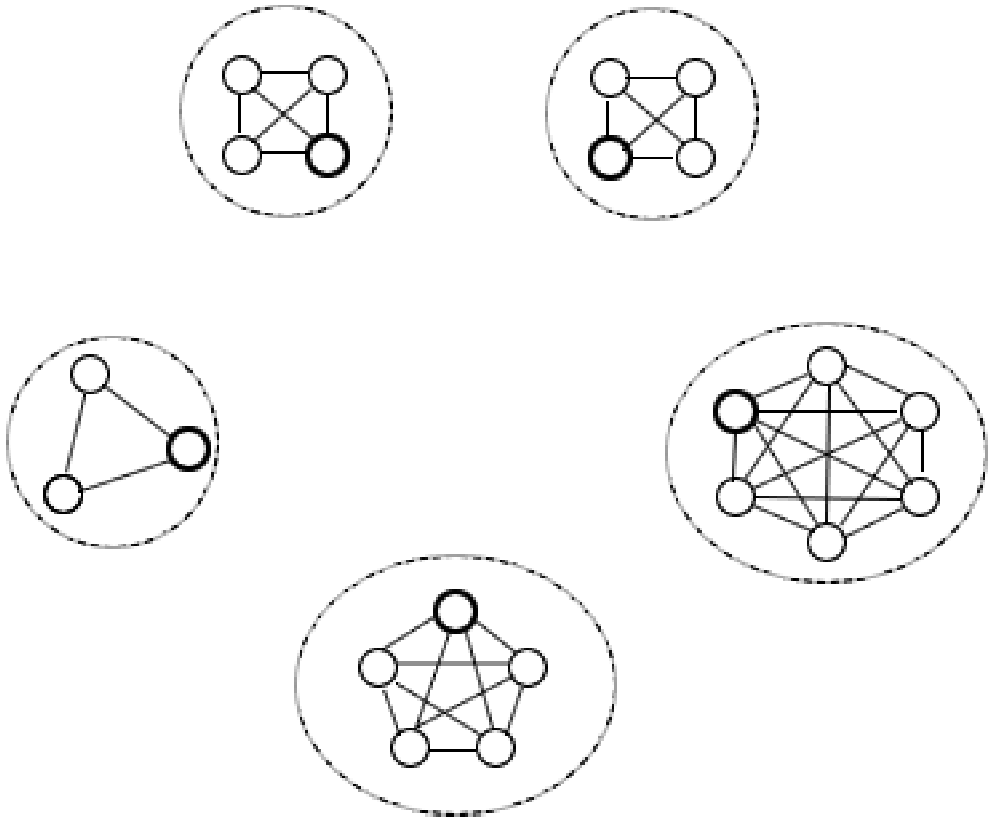
Multi-Clusters



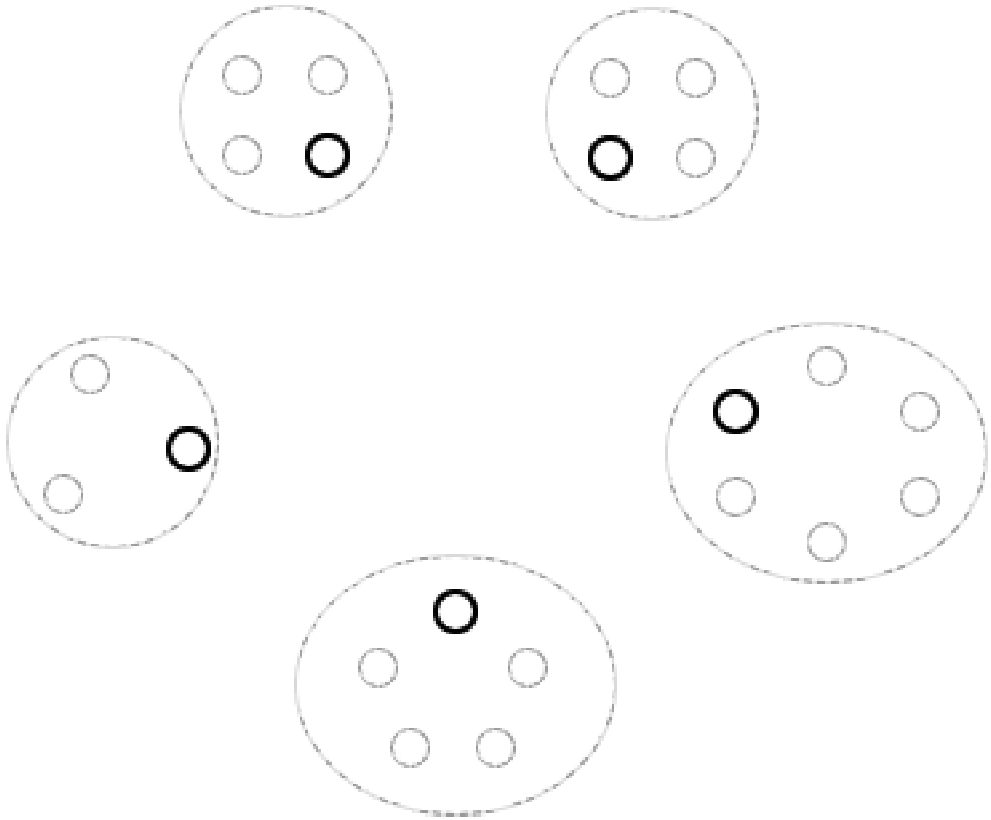
Multi-Clusters



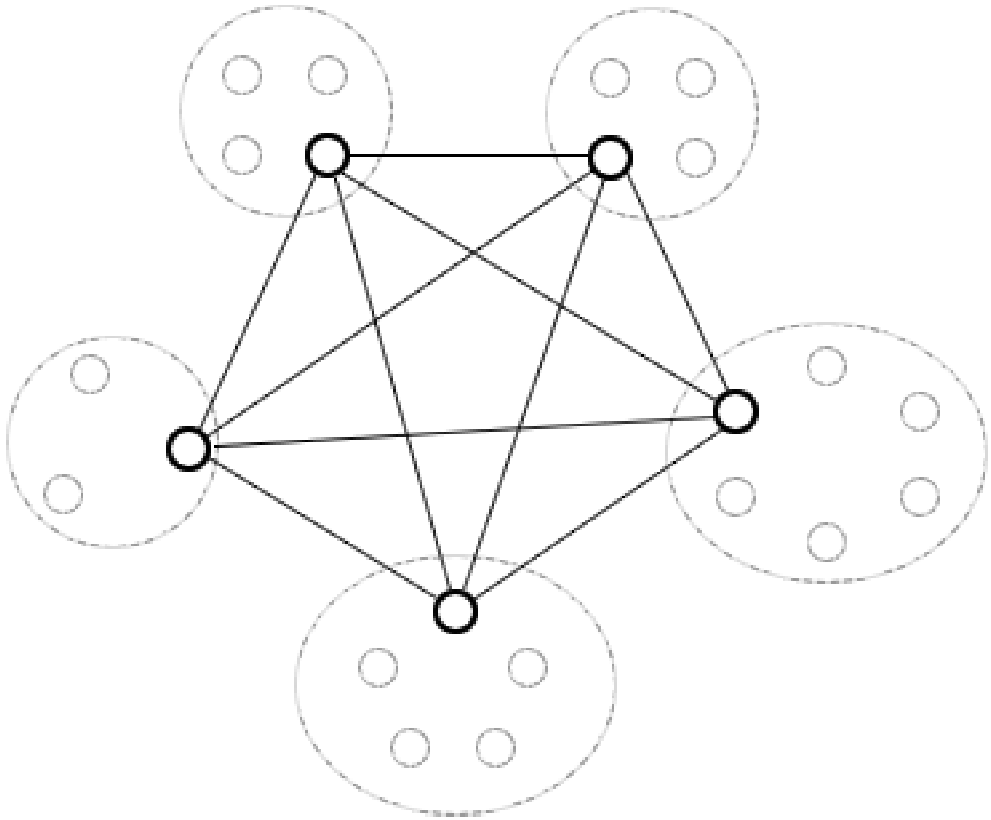
Multi-Clusters



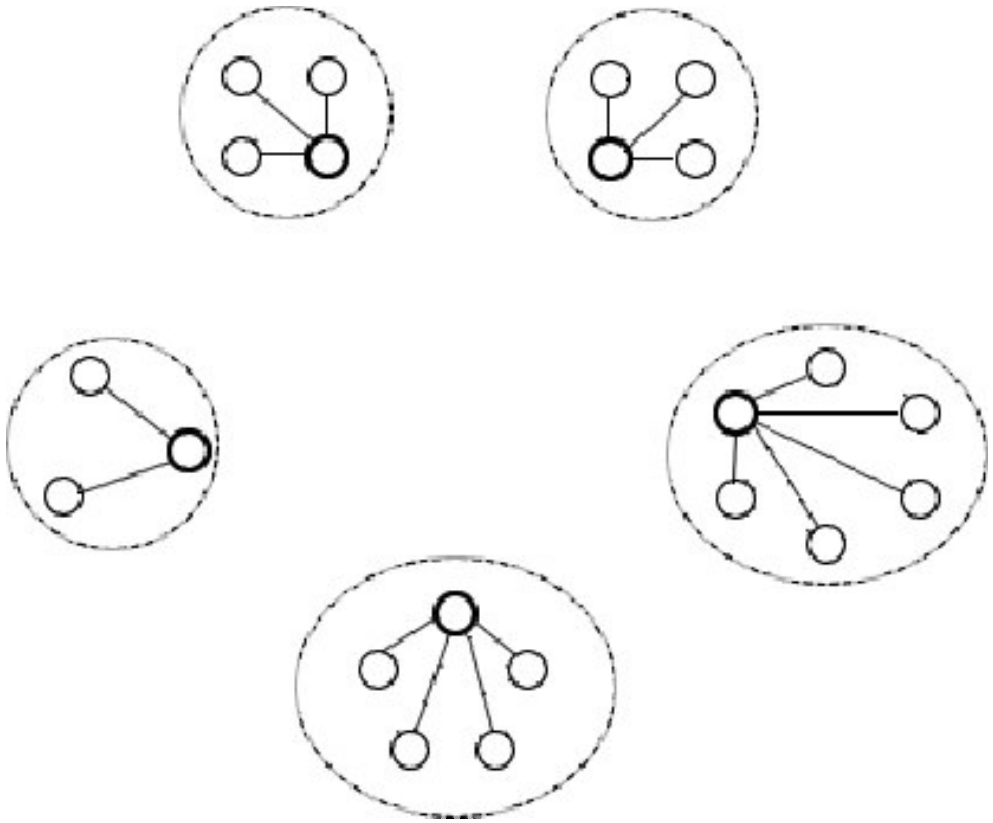
Multi-Clusters



Multi-Clusters



Multi-Clusters



Próximas Tarefas

- Generalizar o algoritmo para mais de dois tipos de comunicações. Com diferentes valores de β e τ para cada cluster

- Fim!