List Ranking: Solução probabilística

Baseado na dissertação de mestrado de Guilherme Pereira Vanni

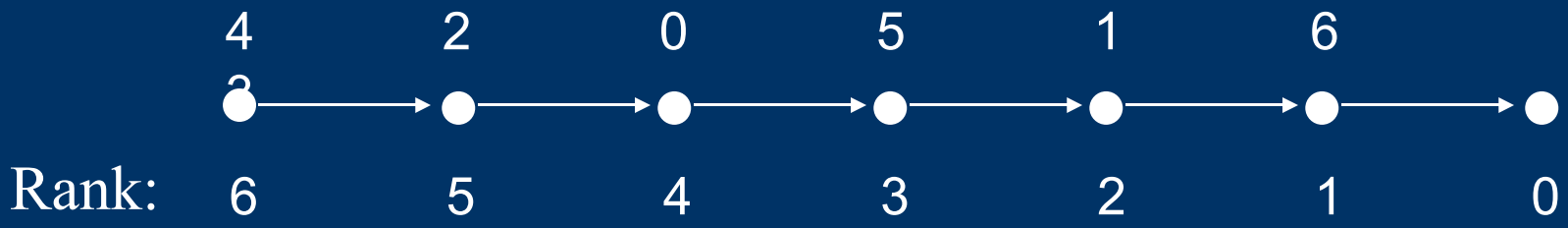
Orientador: Prof. Siang Wung Song

Introdução

- Definição

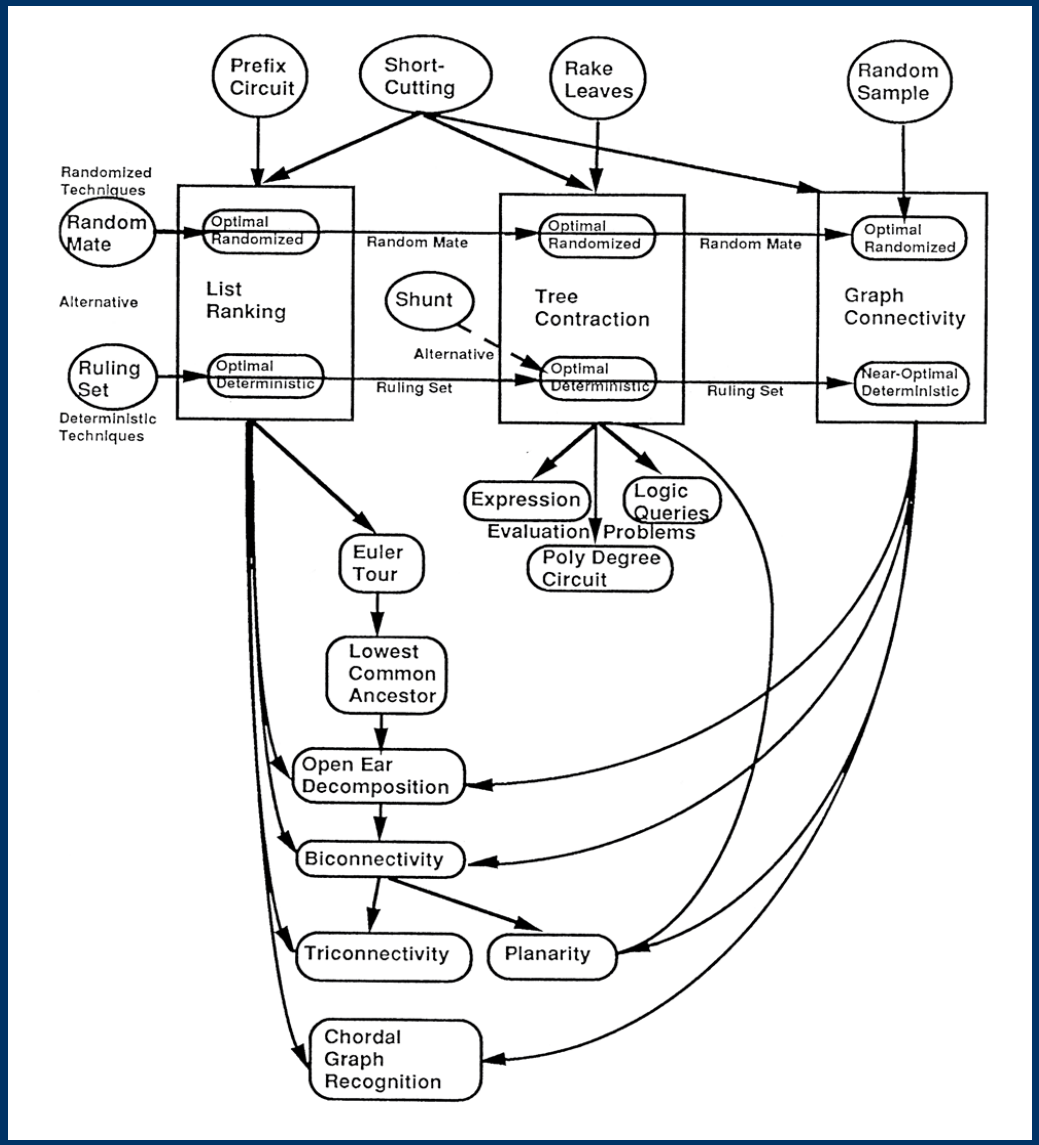
Lista ligada: uma seqüência de nós tal que cada nó aponta para outro nó, chamado seu sucessor, e não há ciclo em tal lista.

O problema de ***List Ranking*** consiste em determinar o *rank* para todos os nós, isto é, a distância para o último nó da lista.



Importância

[Reif 1993]



Síntese de algoritmos paralelos para grafos

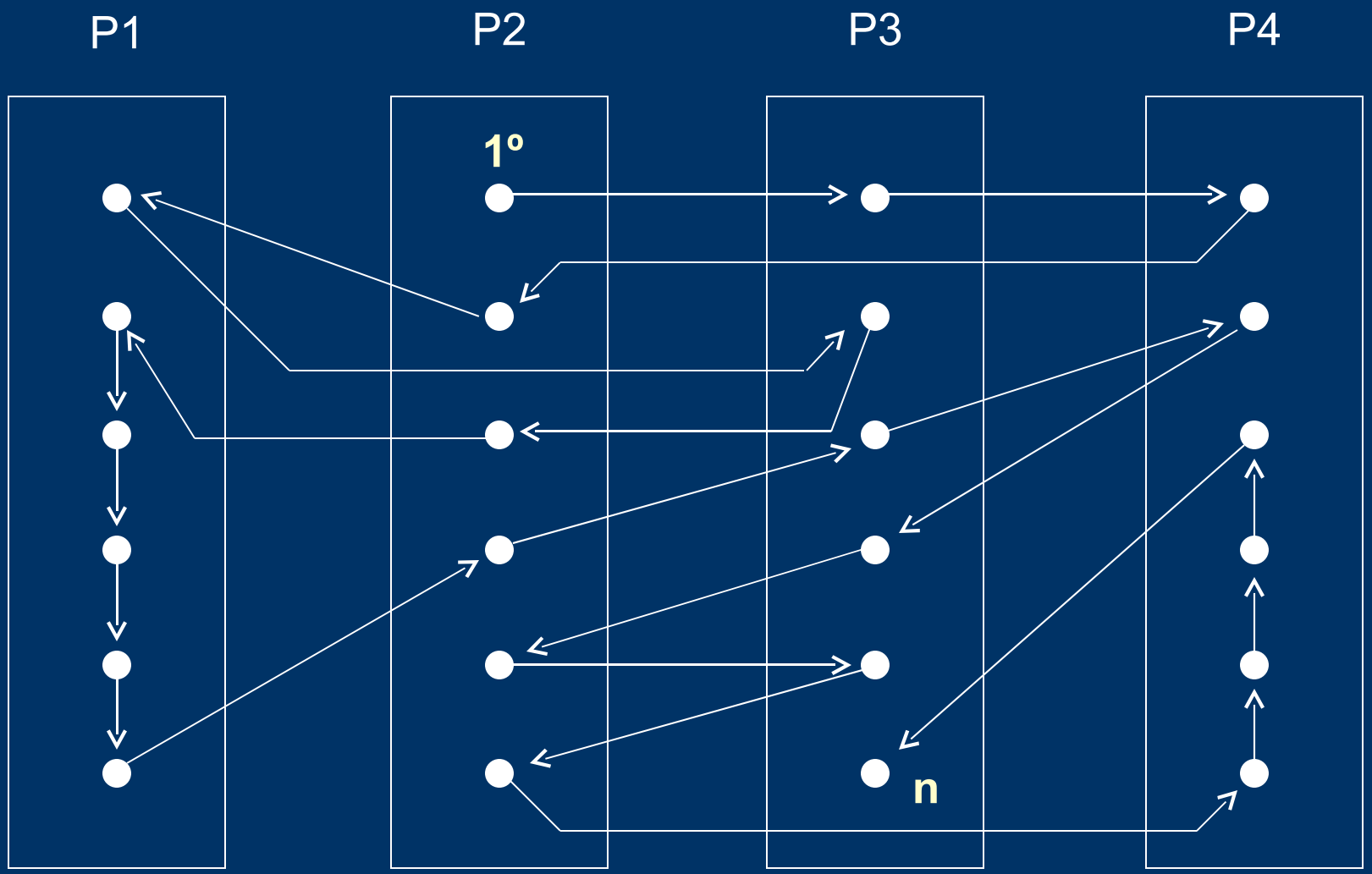
Modelo CGM

- Arquitetura: Memória distribuída.
- Problema de tamanho *n*
- *p* processadores cada um com memória local O(n/p).
- Rodada de computação + rodada de comunicação.
- Em cada rodada cada processador envia e recebe O(n/p) dados.
- Algoritmo eficiente ⇨ minimizar o nº rodadas de comunicação bem como o tempo de computação local total.
- parâmetros: n e p.

Algoritmos paralelos CGM para LR

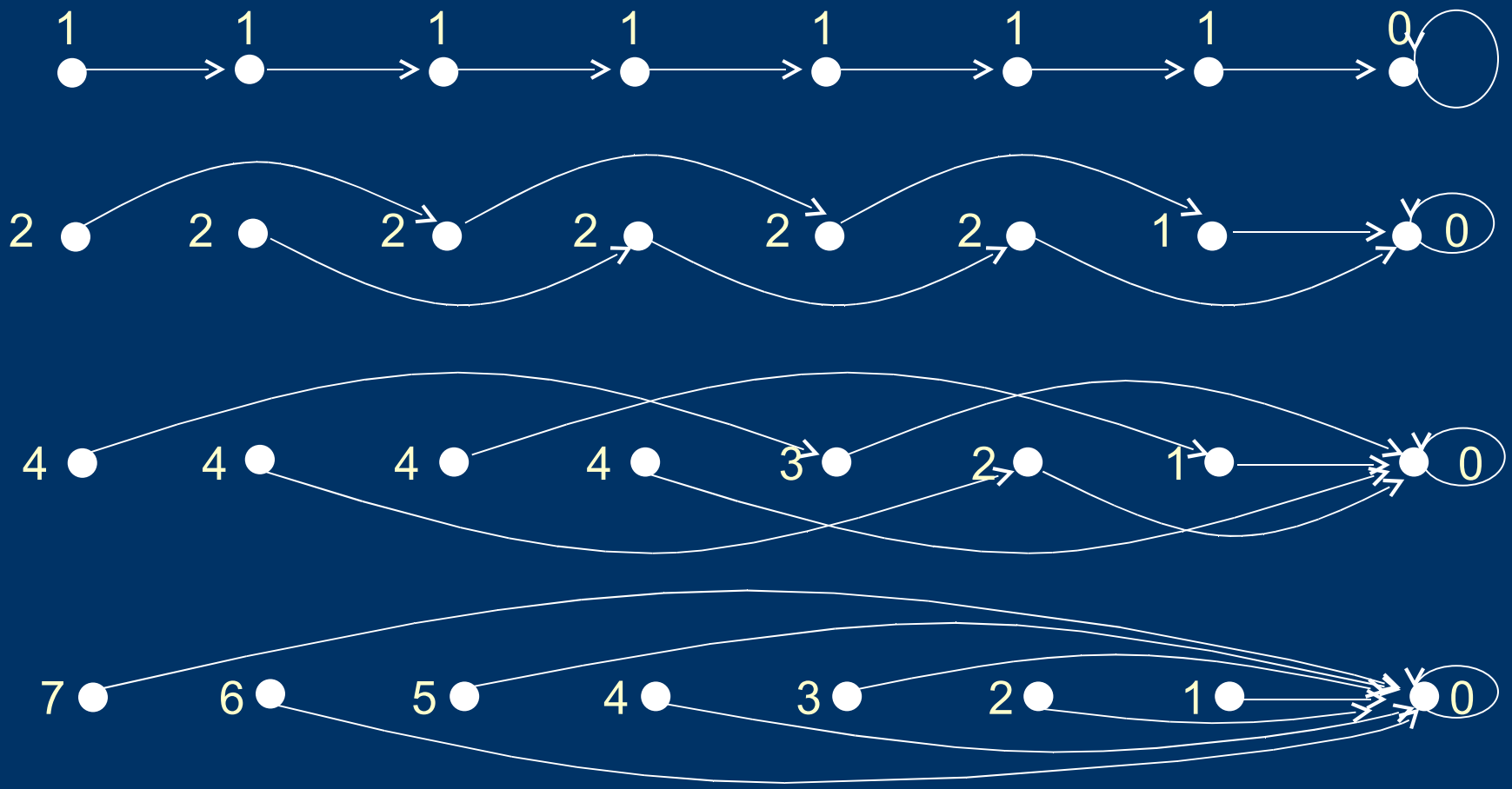
Autor	Número de Rodadas	Algoritmo
Denhe e Song	O(log p + log log n)	probabilístico
Denhe e Song	O(ln* n log p)	probabilístico
Sibeyn	O(p)	determinístico
Denhe et al.	O(log p)	determinístico

Lista ligada armazenada em 4 proc.



Algoritmo seqüencial O(n).

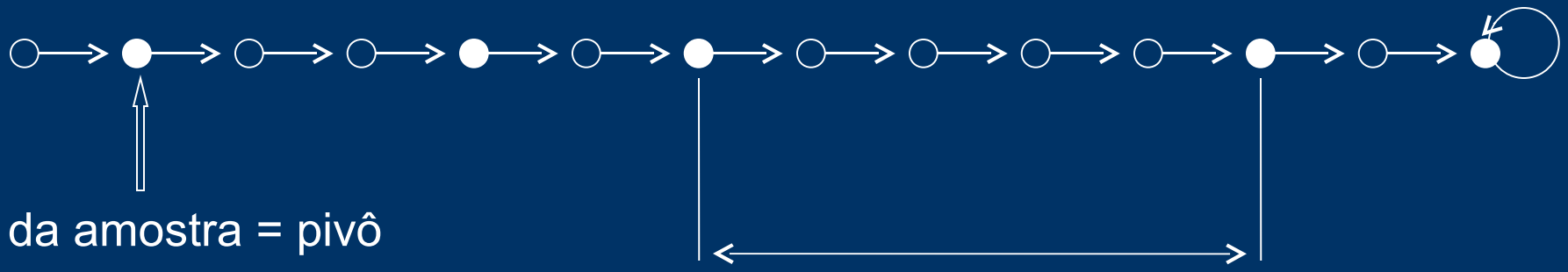
Algoritmo paralelo usando “pointer jumping” O(log n).



Cada sucessor pode estar em outro proc. ⟹ O(log n) rodadas de comunicação.

[Dehne e Song 1997]

Amostra: cada nó é escolhido com probabilidade de 1/p.

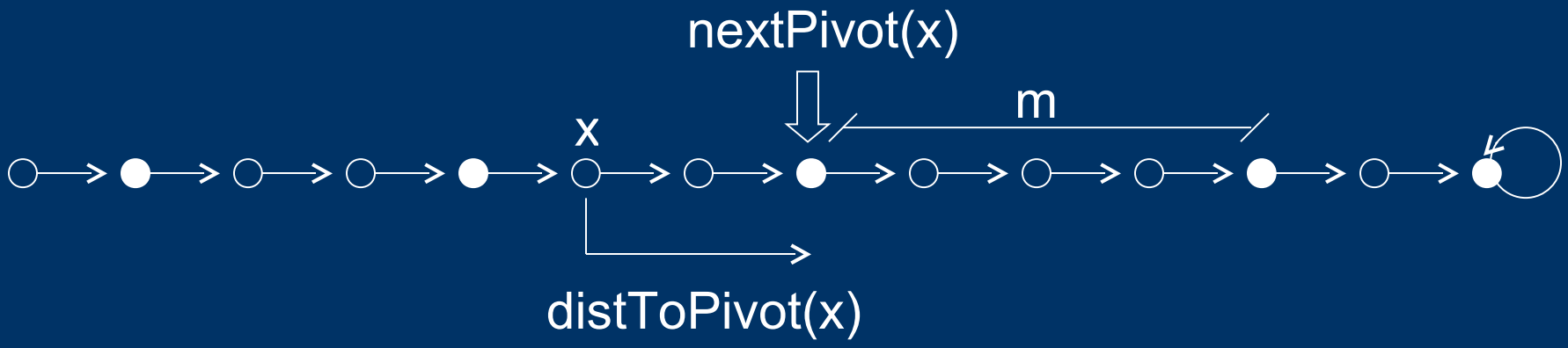


m = distância máxima entre 2 pivôs

m <= 3 p ln n com alta probabilidade

Prob. {m > c(3 p ln n)} <= 1/ n^c, c >2

Algoritmo Probabilístico - nomenclatura



nextPivot(x) = pivô mais próximo a direita

distToPivot(x) = distância entre x e nextPivot(x)

List ranking modificado

Determinar para cada x:

nextPivot(x) e distToPivot(x)

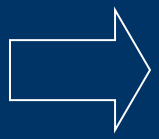
Algoritmo Probabilístico 1

n – nós, *p* – processadores, *O(n/p)* – nós por processador

1. Cada processador seleciona nós como pivô com probabilidade 1/p.
2. Todos processadores resolvem o problema do *list ranking Modificado*.
3. Os valores de nextPivot(x) e distToPivot(x) de todo pivô são enviados de cada processador a todos os outros.
4. Cada processador resolve seqüencialmente o problema do *list ranking* para seus nós.

Número de rodadas do Algoritmo 1

passo	nº rodadas
1	–
2	log 3 p + log ln n
3	1
4	–
total	1 + log 3 p + log ln n



O(log p + log log n) rodadas de comunicação com alta prob.

Algoritmo Probabilístico 2

1. Selecione nós como pivô com probabilidade 1/p. Pinte os pivôs de pretos e os demais de branco
2. Para i=1,..,k faça
    1. Para cada nó preto x pinte de vermelho nós a distância pelo menos (2/3)p a direita e a esquerda de x
    2. Para cada nó branco x selecione x como pivô com probabilidade 1/p
    3. Pinte de branco todos os nós vermelhos
3. Seja S' o conjunto de nós selecionados. Execute os passos 2 a 4 do Algoritmo Probabilístico 1 usando S'

Número de rodadas do Algoritmo 2

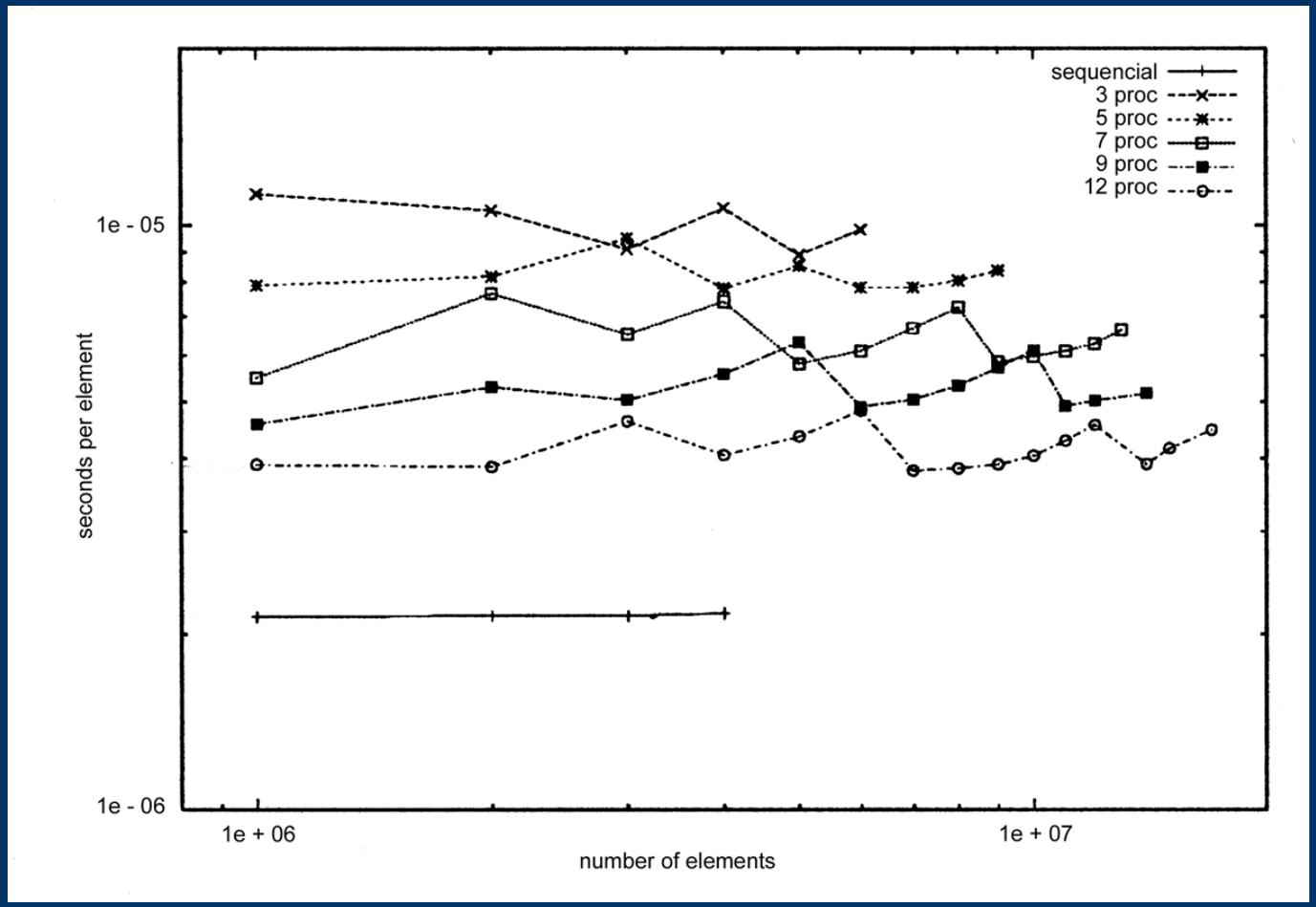
passo	nº rodadas
1	–
2	O(k log p)
3	1 + log p
total	1 + log p + log p ln* n

O(log p log* n) rodadas de comunicação com alta prob.

Resultados Experimentais

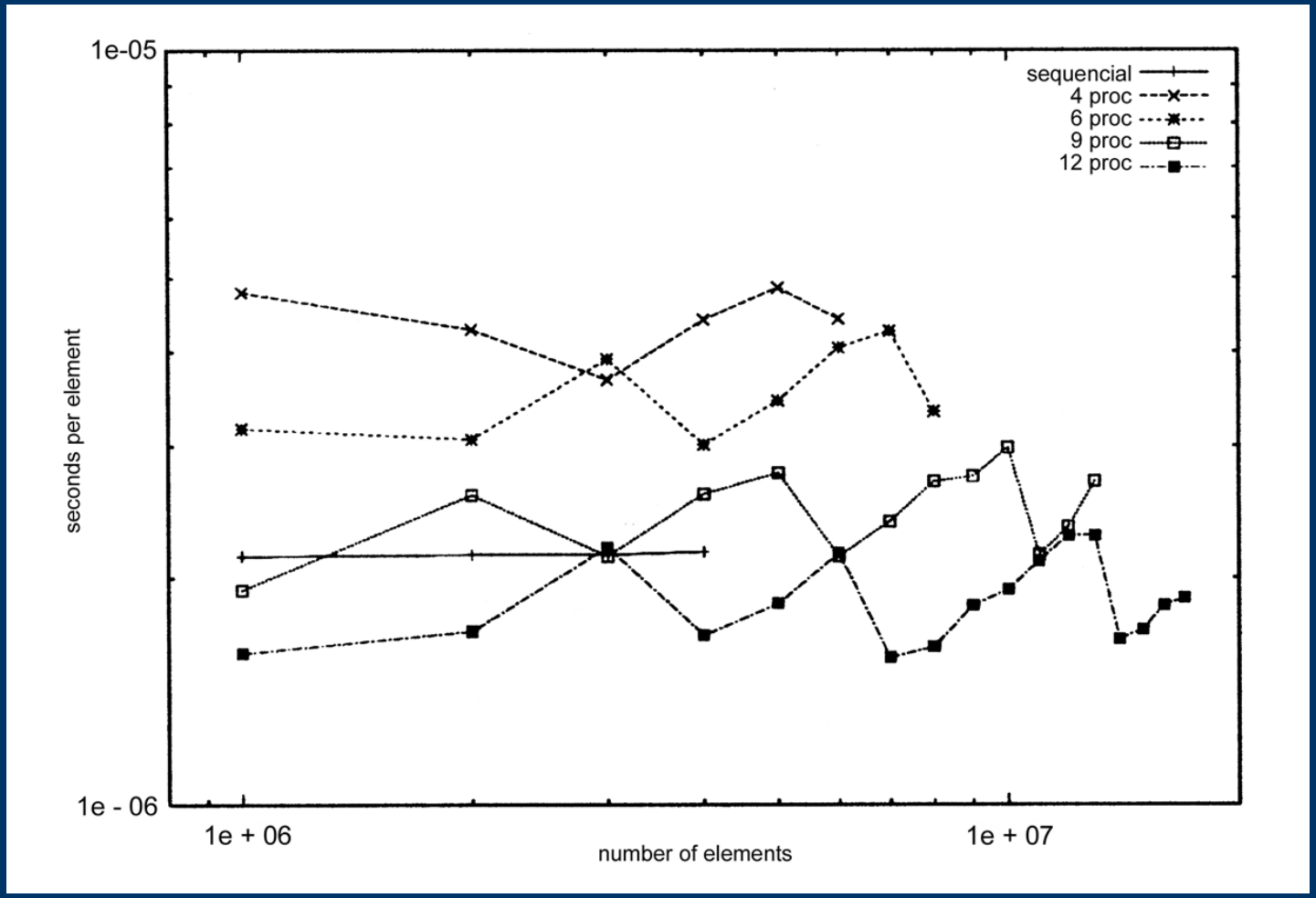
- Reid-Miller algoritmos PRAM com resultados satisfatórios para o Cray C-90, mas específicos para a máquina.
- Gustedt apresenta dois algoritmos CGM (PC-cluster): probabilístico e um determinístico.
- Sibeyn algoritmo CGM com O(p) rodadas de comunicação. Para p = 36 e n = 600 mil aceleração de aproximadamente 5.
- Chan e Dehne algoritmo com O(log p) rodadas de comunicação. Resultados conhecidos mais eficientes.

Gustedt - desempenho do programa determinístico



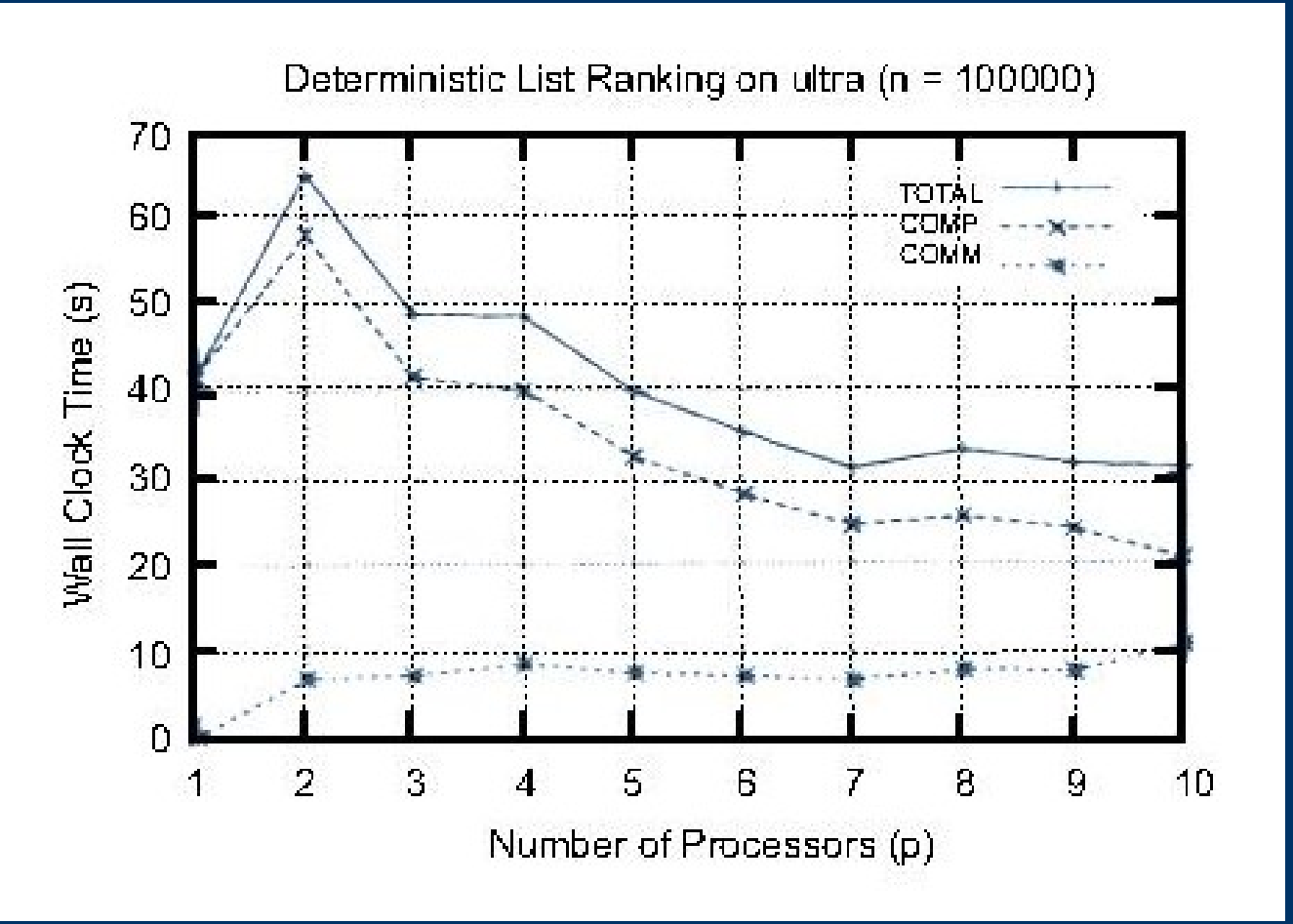
Tempo de execução por elemento para o programa determinístico

Gustedt - desempenho do programa probabilístico



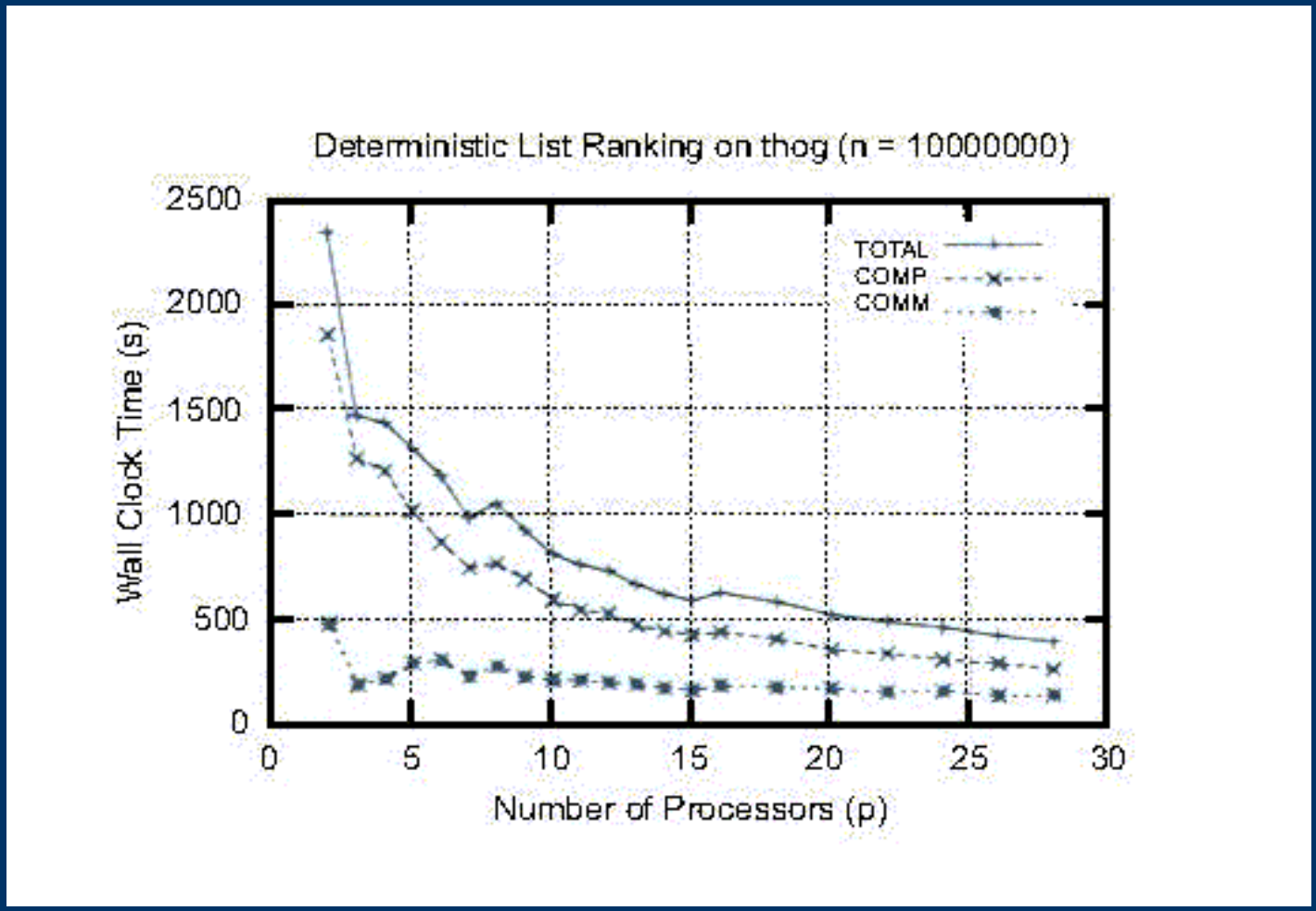
Tempo de execução por elemento para o programa probabilístico

Chan e Dehne - desempenho do programa determinístico



Curva dos tempos observados na plataforma ultra (switch de 100Mb)

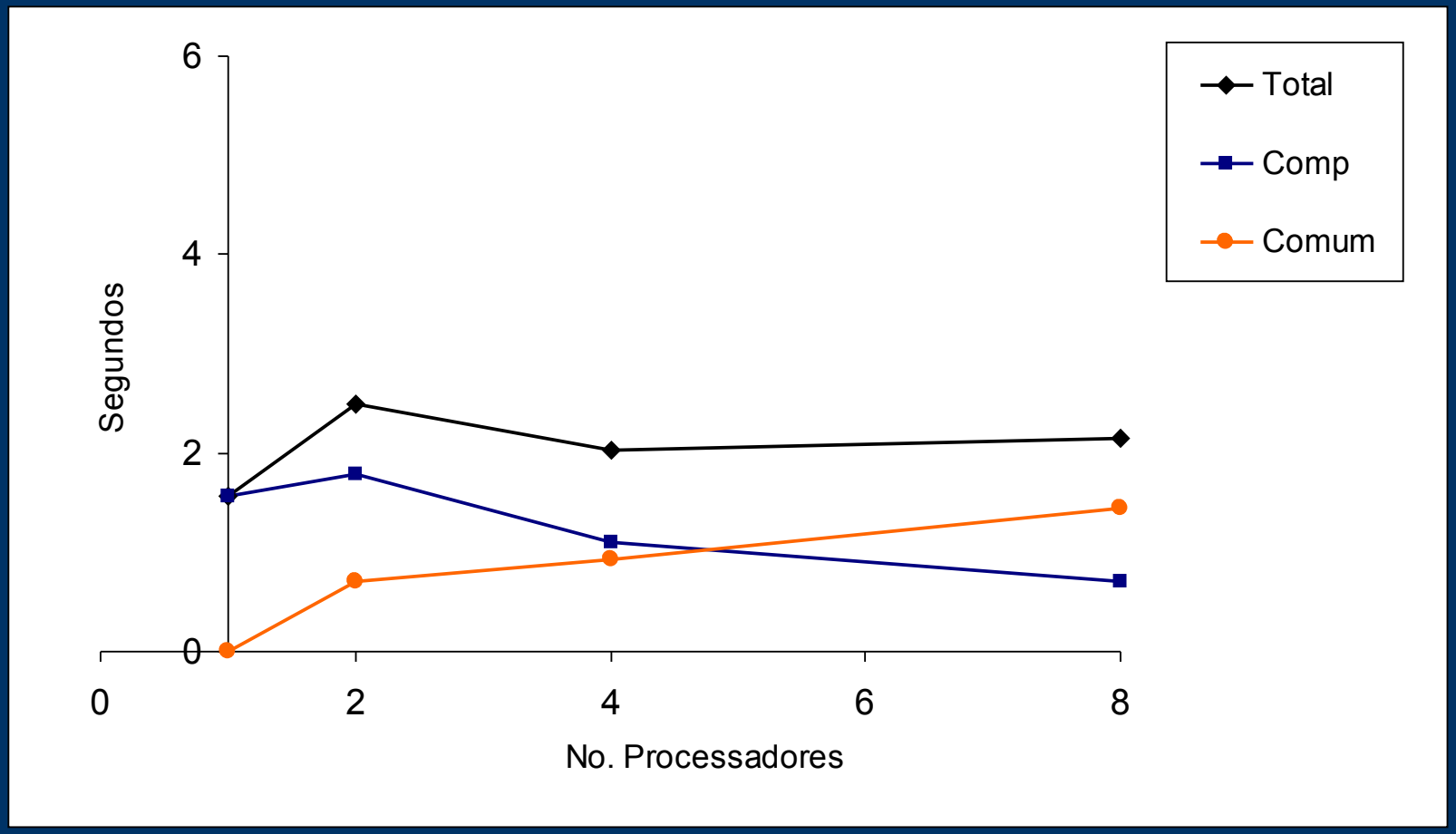
Chan e Dehne - desempenho do programa determinístico



Curva dos tempos observados na plataforma thoga (switch de 1Gb)

Resultados do Algoritmo Probabilístico

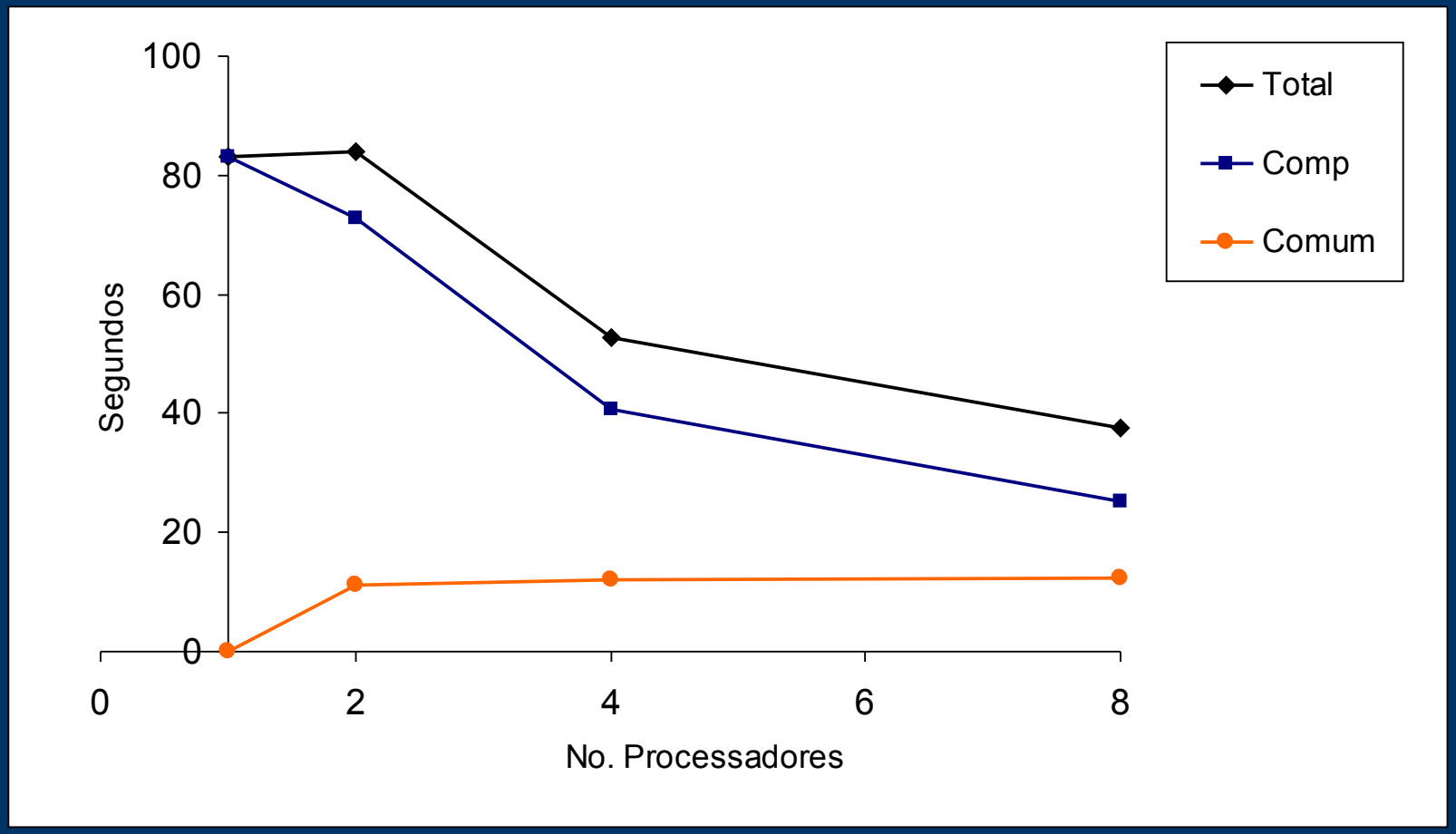
Programa Probabilístico com n = 1M



Curva dos tempos observados para o algoritmo probabilístico com entrada n = 1M.

Resultados do Algoritmo Probabilístico

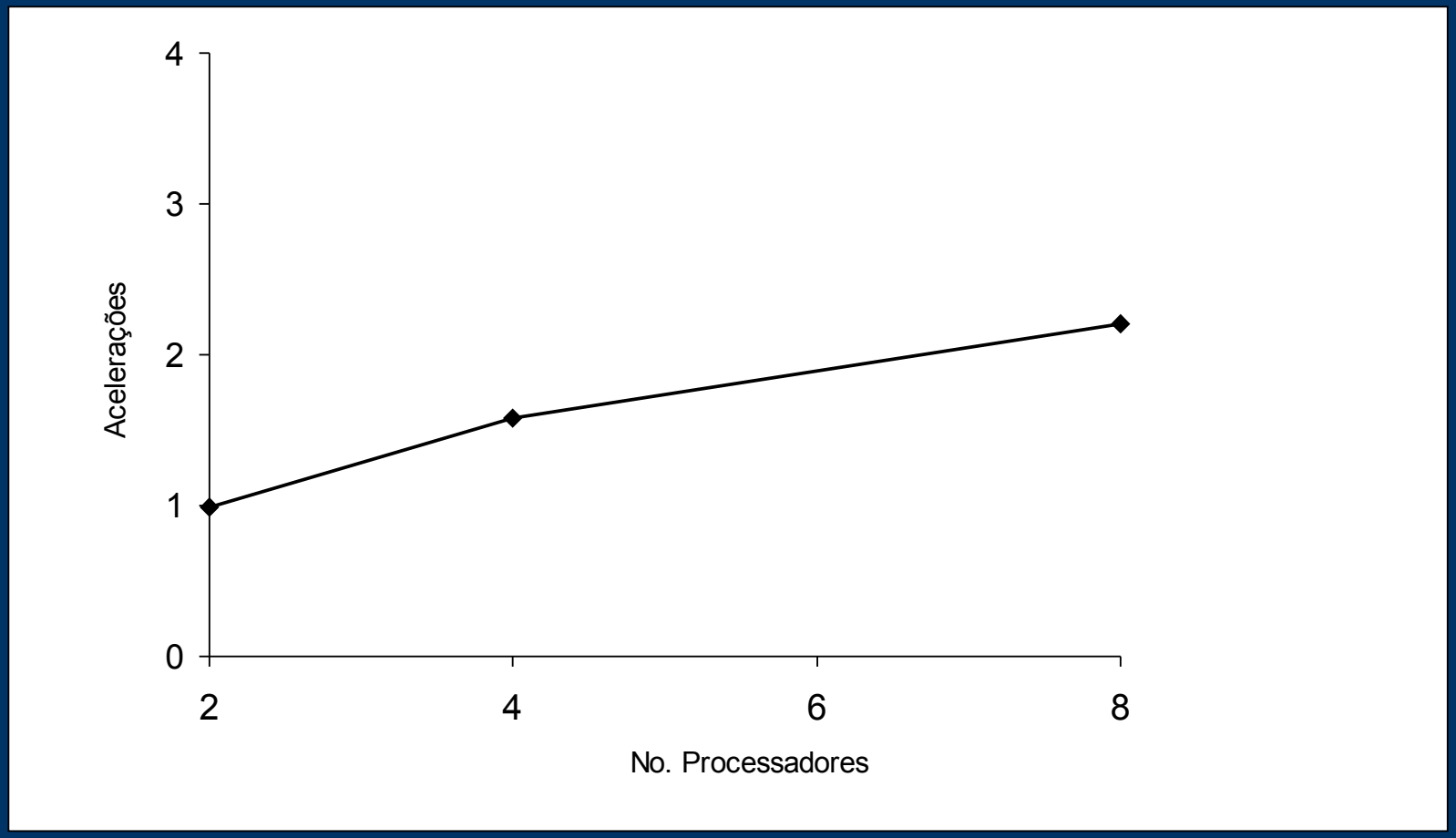
Programa Probabilístico com n = 32M



Curva dos tempos observados para o algoritmo probabilístico com entrada n = 32M.

Resultados do Algoritmo Probabilístico

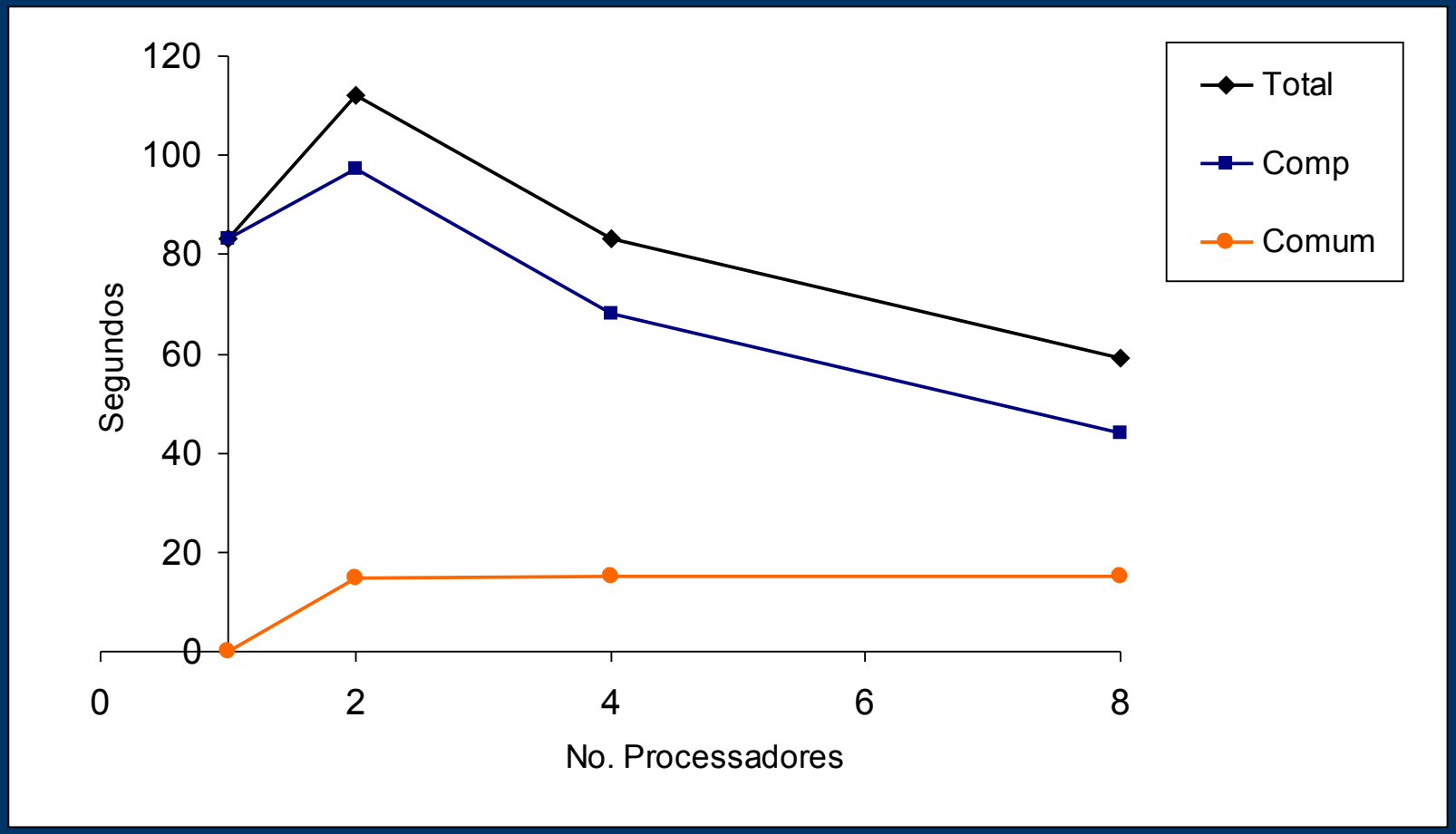
Programa Probabilístico com n = 32M



Acelerações (speedups) obtidas para o algoritmo probabilístico para n = 32M.

Resultados do Algoritmo Probabilístico

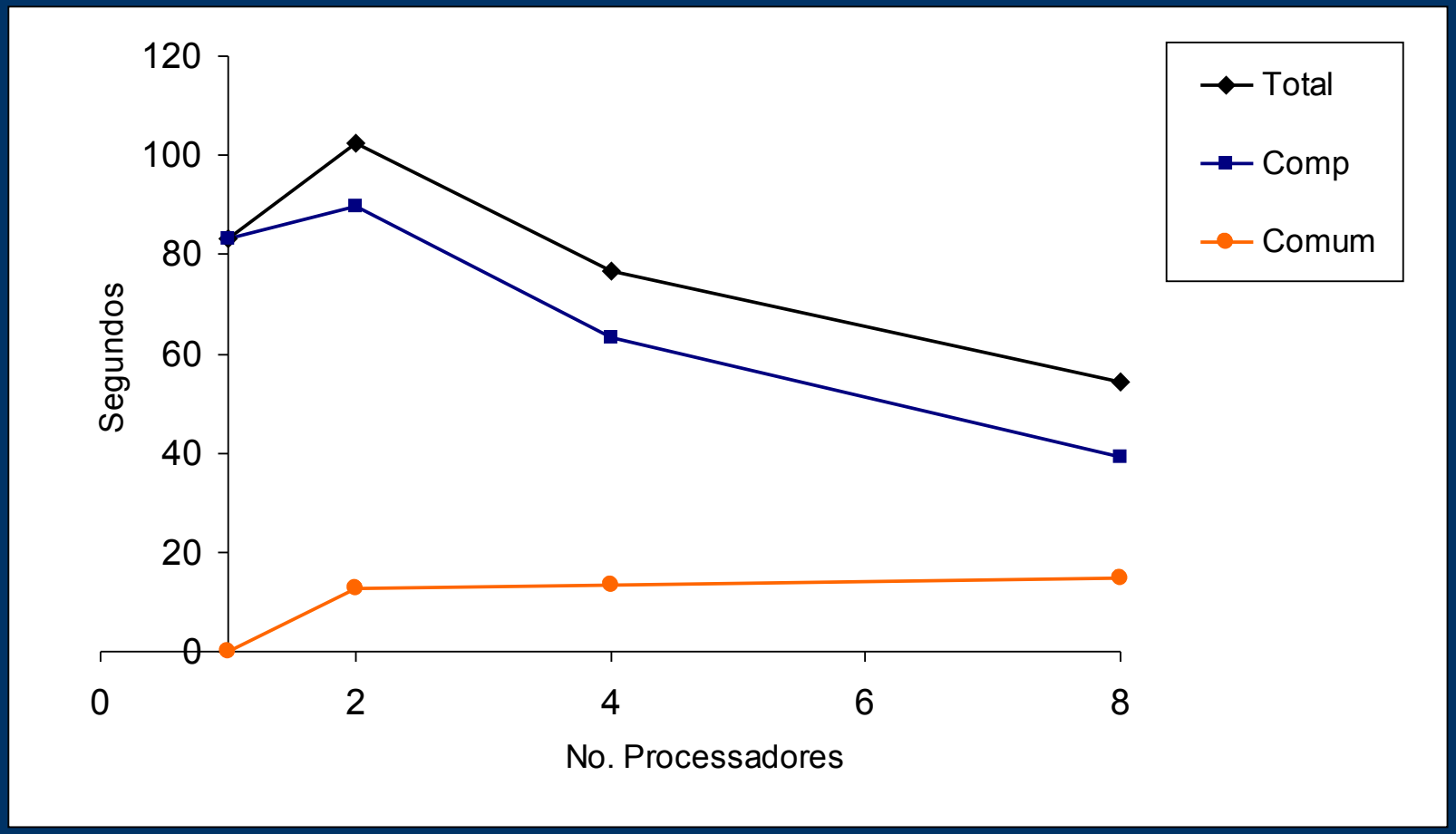
Programa Determinístico com n = 32M



Curva dos tempos observados para o algoritmo determinístico com entrada n = 32M.

Resultados do Algoritmo Probabilístico

Programa Determinístico Modificado com n = 32M



Curva dos tempos observados para o algoritmo probabilístico com entrada n = 32M.

Comentários

- Os dados experimentais obtidos mostram um desempenho satisfatório dos programas.
- Programas mais eficientes para maiores valores da entrada (n).
- O tempo de execução diminui com o aumento do número de processadores, exceto para n = 1M, onde o tempo de comunicação não é compensado pela quantidade de dados. Para p = 8 e n = 16M todos os programas paralelos são mais rápidos que o programa seqüencial.
- Programa determinístico modificado é um pouco mais rápido que o programa determinístico.
- A implementação do algoritmo probabilístico (que possui na teoria maior número de rodadas de comunicação) obteve melhor desempenho que a do algoritmo determinístico.