List Ranking

Um algoritmo CGM/BSP probabilístico

F. Dehne e S. W. Song

Um algoritmo paralelo escalável para "list ranking"

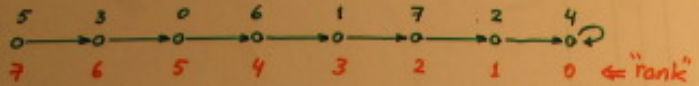
Frank Dehne
School of Computer Science
Carleton University

Siang W. Song
Depto. Ciência da Comp.
IME / USP

15 nov. 1995

"List ranking"

Lista ligada de n elem.



Dada lista de pares $(x, next(x))$

$0 \rightarrow 6$
$1 \rightarrow 7$
$2 \rightarrow 4$
$3 \rightarrow 0$
$4 \rightarrow 4$
$5 \rightarrow 3$
$6 \rightarrow 1$
$7 \rightarrow 2$

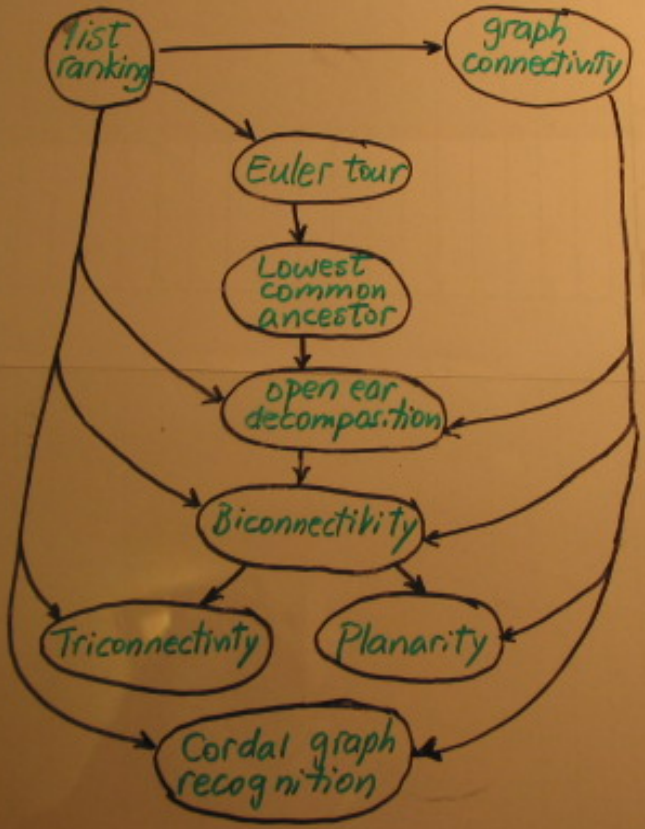
Aplicações: problemas em árvores

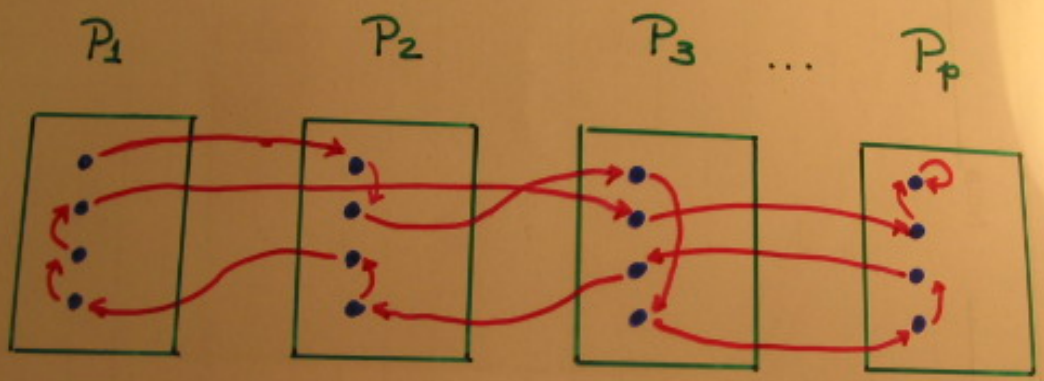
- atribuição de raízes
- pré-ordem e pós-ordem
- obtenção do nível de cada nó
- obtenção do nº descendentes de cada nó

problemas em grafos: nº de componentes conexos

Family of parallel algs.

(Reif 1993)

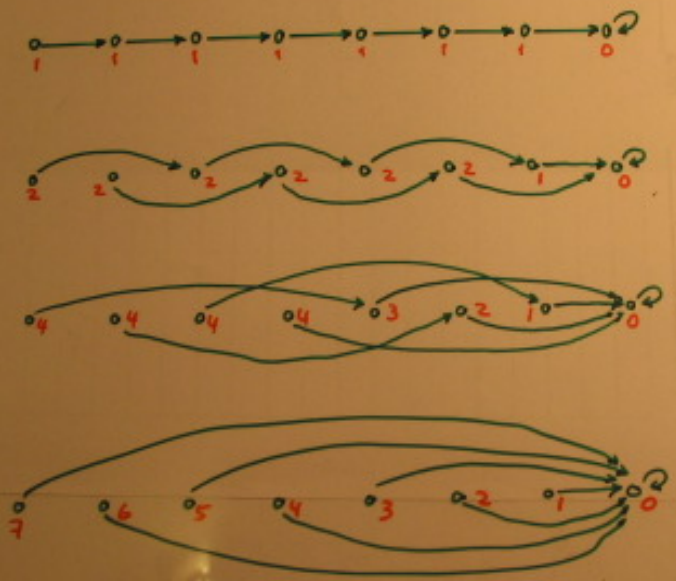



P_1
P_2
P_3
...
P_p

Sequential Alg. $O(n)$

Parallel Alg. using "pointer jumping"

$O(\log n)$



Each sucessor can be in another proc.

$O(\log n)$ communication rounds

Algoritmo 1:

$1 + \log(3p) + \log \ln(n)$ rodadas comun.

com alta probabilidade.

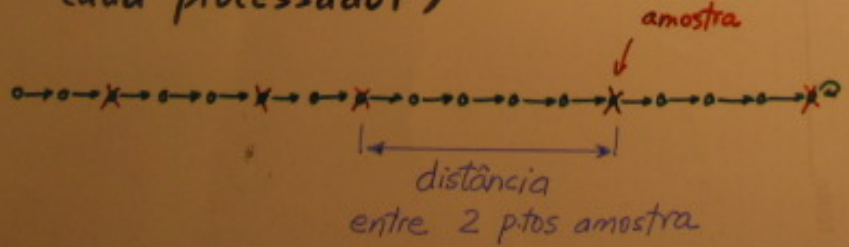
$$\text{Prob.}\left\{ \#\text{ rodadas comun.} \geq c\left(1 + \log(3p) + \log \ln(n)\right)\right\} \leq \frac{1}{n^c}$$

$c > 2$

Amostra de tamanho $\frac{n}{p}$

armazenada em cada processador

(além dos $\frac{n}{p}$ dados próprios de cada processador)



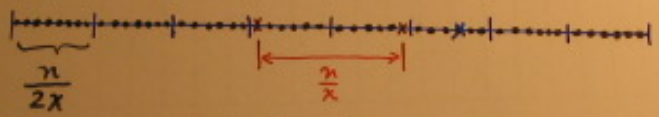
Lema 1: (Belloch et al. 1991)

$x.k \leq n$ elem. escolhidos aleat. particiona lista S em sublistas S_i t.q. o tamanho da maior sublista

$$m = \max_i |S_i| \leq \frac{n}{x}$$

com prob. pelo menos $1 - 2x\left(1 - \frac{1}{2x}\right)^{x.k}$

Prova: Considere $2x$ segmentos



Se cada segm. contém ≥ 1 pto escolhido:

$$m \leq \frac{n}{x}$$

Considere um segm. e um p.to escolhido

Prob. { o p.to não cair no segm. } $= 1 - \frac{1}{2x}$

Prob. { $x.k$ p.tos não caírem no segm } $= \left(1 - \frac{1}{2x}\right)^{x.k}$

São $2x$ segm.:

Prob. existir um segm. sem p.to escolhido

$$\leq 2x\left(1 - \frac{1}{2x}\right)^{x.k}$$

Corolário 1:

Existe uma sublista com tamanho

$$\geq c\frac{n}{x}$$

com prob. no máx.

$$\frac{2x}{c}\left(1-\frac{c}{2x}\right)^{x\cdot k} \leq \frac{2x}{c}e^{-\frac{1}{2}ck}$$

na prova anterior basta considerar

$\frac{2x}{c}$ segmentos (menos segmentos)

Lema 2:

x. k elem. escolhidos aleat.

se $k \geq \ln x + 2 \ln n$

então $\text{Prob.}\{ m \geq c\frac{n}{x} \} \leq \frac{1}{n^c}$, $c > 2$

Prova:

Corolário 1: $\text{Prob.}\{ m > c\frac{n}{x} \} \leq \frac{2x}{c} e^{-\frac{1}{2}ck}$

p/ $c > 2$

$$\ln x + 2 \ln n \leq k$$

$$\Rightarrow \frac{2}{c} \ln\left(\frac{2x}{c}\right) + 2 \ln n \leq k$$

$$\Rightarrow \ln\left(\frac{2x}{c}\right) + c \ln n \leq \frac{ck}{2}$$

$$\Rightarrow \frac{2x}{c} n^c \leq e^{\frac{ck}{2}}$$

$$\text{Prob.}\left\{ m > c\frac{n}{x} \right\} \leq \frac{1}{n^c}$$

Teorema 1:

$\frac{n}{p}$ elem. escolhidos aleat. particiona S em sublistas S_i com $m = \max_i |S_i|$

t.q.

$$\text{Prob.}\{m \geq c\, 3p \ln(n)\} \leq \frac{1}{n^c}, \quad c > 2$$

Prova:

Fazer

$$x = \frac{n}{3p \ln(n)}$$

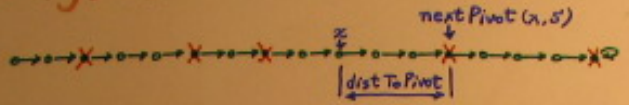
$$k = \ln(x) + 2\ln(n) = 3\ln(n) - \ln(3p \ln n)$$

Então

$$x \cdot k = \frac{n}{p} \frac{3\ln(n) - \ln(3p \ln n)}{3\ln(n)} \leq \frac{n}{p}$$

o teorema 1 segue do Lema 2.

Algoritmo 1



S lista dos n pontos

S' p.tos escolhidos chamados Pivots

Para cada $x \in S$ ($\frac{n}{p}$ pivots)

nextPivot (x, S') é o pivot mais próx. seguindo x na lista S.

distToPivot (x, S') é a distância entre x e nextPivot (x, S').

"Modified pointer jumping":

Determ. p/ cada $x \in S$

{ nextPivot (x, S')
{ distToPivot (x, S')

Usamos ainda:

$$m(S, S') = \max_{x \in S} \text{distToPivot}(x, S')$$

Algoritmo 1:

1. Seleciona $S' \subset S$: cada proc. escolhe x como pivot com prob. $\frac{1}{p}$.

2. Todos os proc. resolvem "Modified Pointer Jumping"

3. Os valores $nextPivot(x, S')$
   $distToPivot(x, S')$
   $x \in S'$ são enviados a todos os proc.

4. Usando dados do passo 3 cada proc. P_i faz "pointer jumping" p/ os p.tos de P_i sequencialmente.

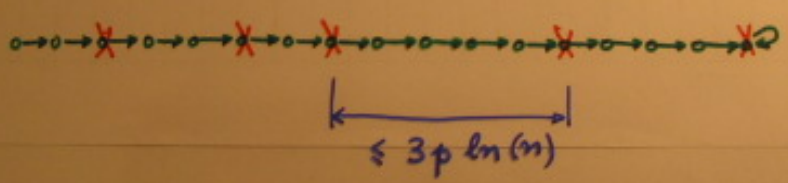
Teorema 2:

Algorithm 1 requer, com alta prob. no máx.

$$1 + \log(3p) + \log \ln(n)$$

rodadas de comunicação e tempo de computação seq. $O(\frac{n}{p})$.

Prova:



Passo 2: #rodadas comun.

$$\log 3p + \log \ln(n)$$

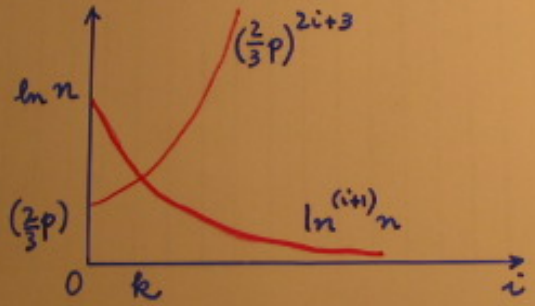
passo 3: 1 rodada de comun.

Algoritmo 2:

$r \leq (4k+6)\log(\frac{2}{3}p) + 8$ rodadas de comunicação, com alta prob.

$k = \min\{i \geq 0 \mid \ln^{(i+1)} n \leq (\frac{2}{3}p)^{2i+3}\}$

($k < \ln^* n$ é um nº muito pequeno.)

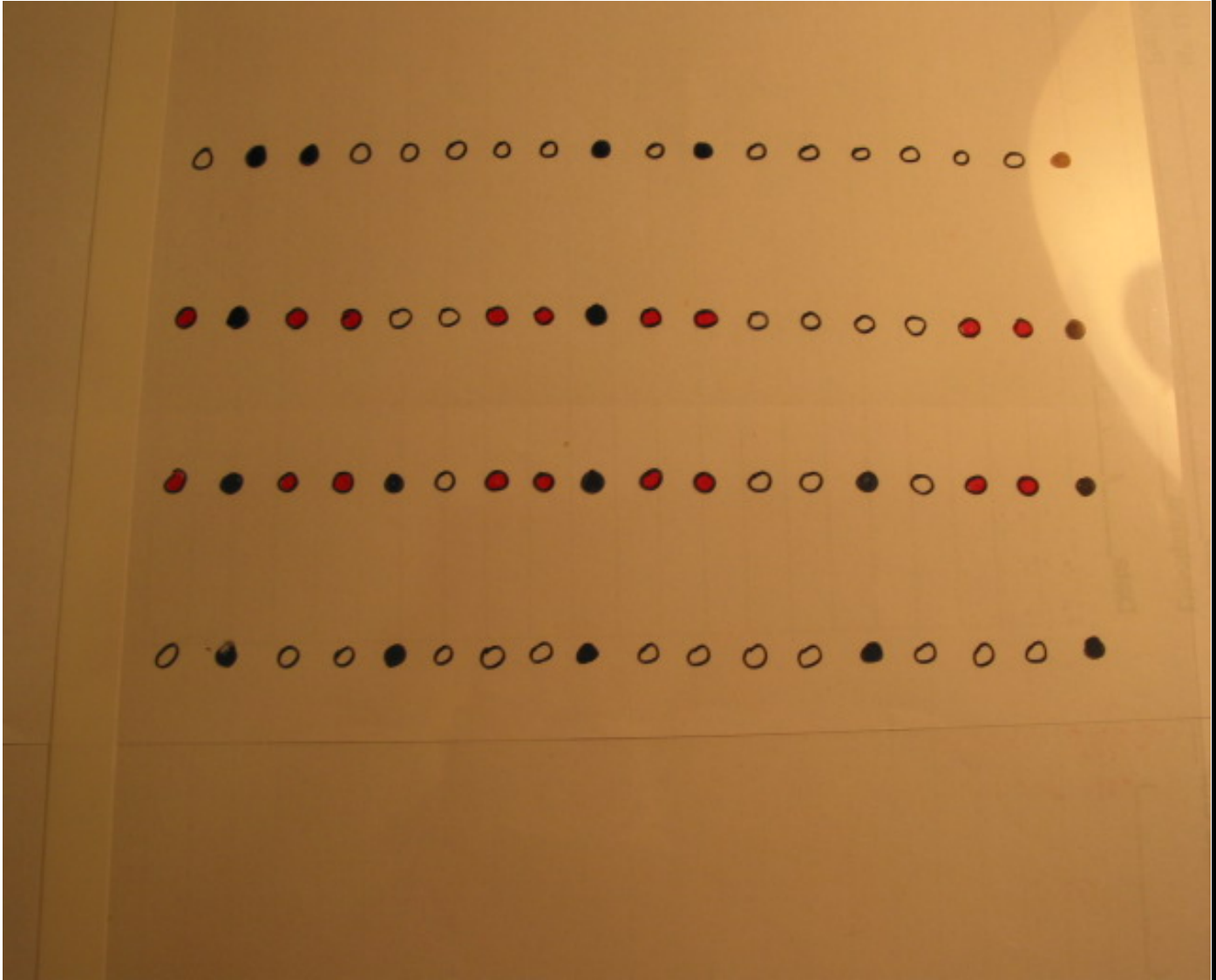


p =	8		32		512		2048	
n	k	r	k	r	k	r	k	r
10^{10}	1	18	0	38	0	62	0	74
$10^{(10^9)}$	1	18	1	58	1	98	0	74
$10^{(10^{20})}$	2	22	1	58	1	98	1	118
$10^{(10^{100})}$	2	22	1	58	1	98	1	118

Idéia do alg. 2

- 2 pontos escolhidos ("Pivots") não devem muito "próximos" ($\leq O(p)$) pois criam "buracos" em outros lugares.
- Usaremos cores p/ distinguir os p.tos :
  - ● pretos : "Pivots"
  - ● vermelhos : um nó próx. a Pivot
  - ○ Brancos : todos os outros

Seja $k = \min\left\{ i \geq 0 \;\middle|\; \ln^{(i+1)} n \leq \left(\frac{2}{3}p\right)^{2i+1} \right\}$



Algoritmo 2:

(1) Exec. passo 1 do Alg 1.
Marque todos os Pivots pretos,
e outros brancos.

(2) For $i = 1, \ldots, k$ do

(2a) Para cada p.to preto x, colore vermelho todo p.to à dist. até $\frac{2}{3}p$ p/ a direita de x.

(2b) idem p/ esquerda.

(2c) P/ cada p.to branco x em proc. P_i, P_i escolhe x como Pivot c/ Prob. $\frac{1}{p}$

(2d) Cada proc. P_i marca como branco todo p.to vermelho.

(3) Seja $S' \subseteq S$ o subconj. de p.tos pretos assim obtidos.
Continue com passos 2 a 4 do Algoritmo 1.