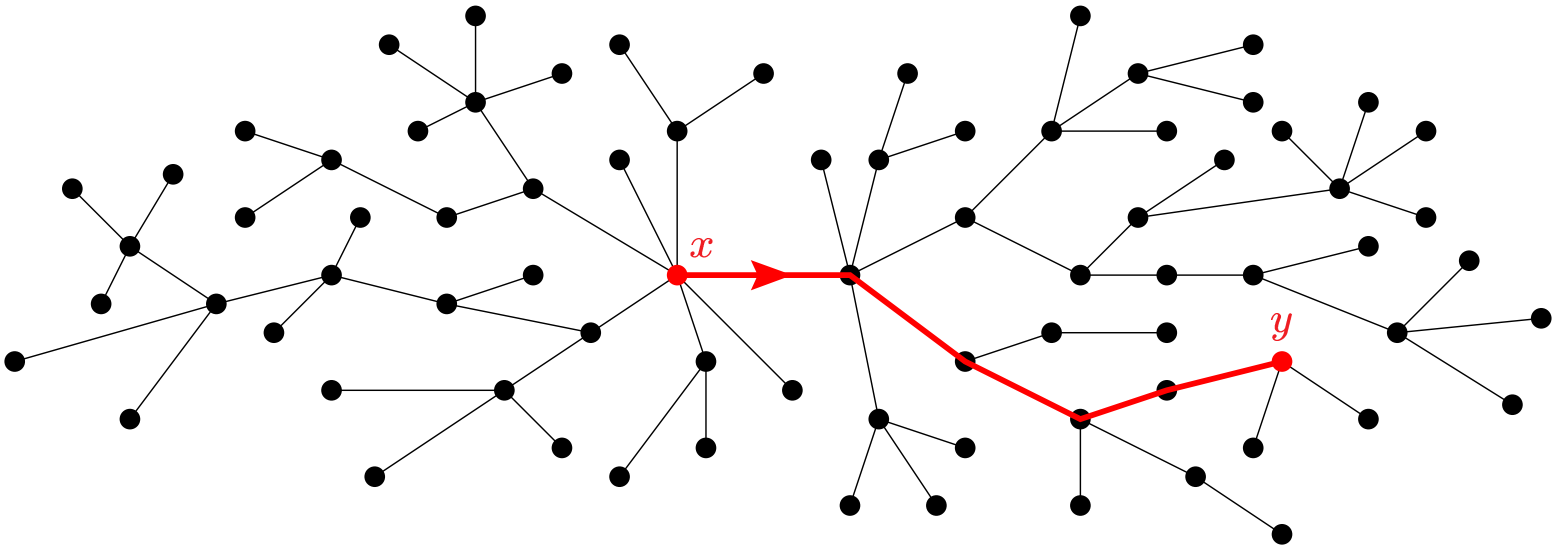
Comment router dans un arbre ?

Pierre Fraigniaud - LRI, CNRS Orsay
Cyril Gavoille - Université de Bordeaux

AlgoTel - 28 Mai 2001

Problème

Router un message de x à y dans un arbre, $\forall\ x, y$

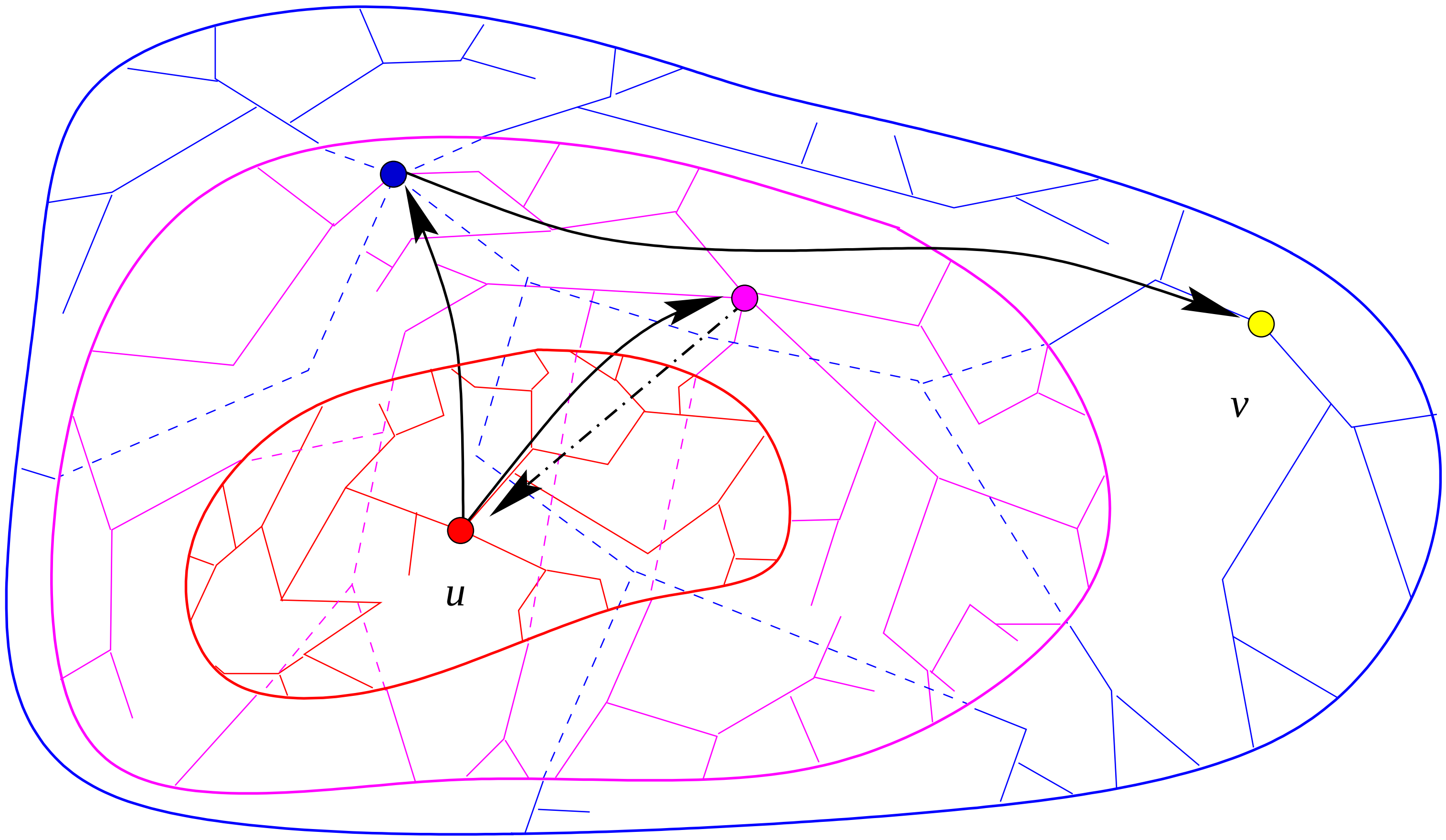


Contraintes :

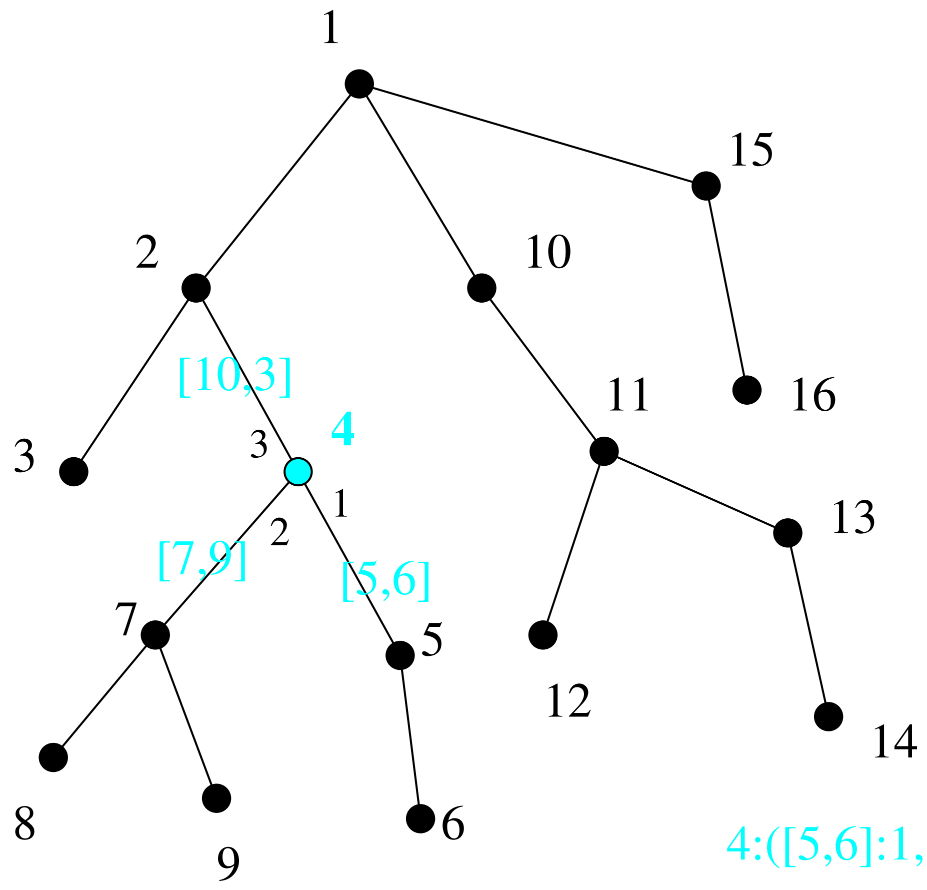
- plus courts chemins ;
- tables de routage <u>locales</u> et <u>compactes</u> ;
- décision de routage <u>rapide</u>.

Motivation : problème de base

Schémas de routage hiérarchiques, applicables à tous les réseaux



Technique classique



- @ = numérotation DFS
- routage par intervalles
- $O(n \log n)$ bits en tout !

4:([5,6]:1,[7,9]:2,[10,3]:3)

4:([5,6]:1,[7,9]:2,[10,3]:3)

Recherche dichotomique $\Rightarrow$ routage en temps $O(\log n)$

Espace $\Rightarrow$ $O(d \log n)$ bits, d = degrés

Problème : Espace $\Omega(n)$ bits lorsque $d \geqslant n/\log n$

Autre solution : 4:

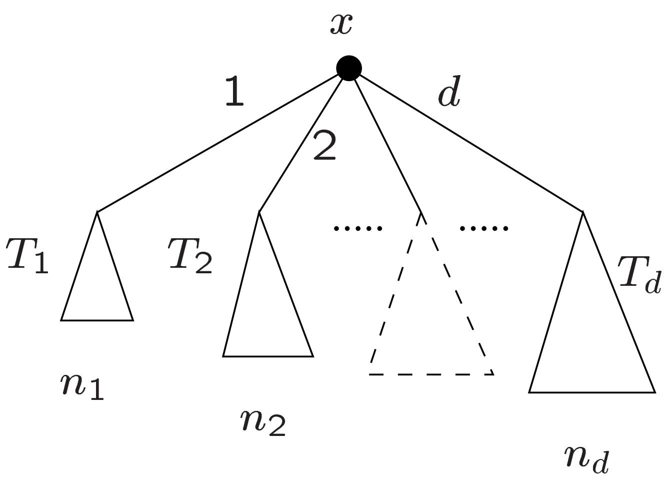
								y ↓							
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
0	**0**	**0**	**0**	**1**	**0**	**1**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**0**
				#1			#2			#3					

Routage : calculer le nombre de “1” (mod d) avant la position y.

Espace n bits, mais $\Omega(n)$ en temps ! (En fait, on peut résoudre ce problème en $n + o(n)$ bits et en temps $O(1)$.)

Peut-on faire mieux ?

Limite inférieure : $c\sqrt{n}$ bits au moins



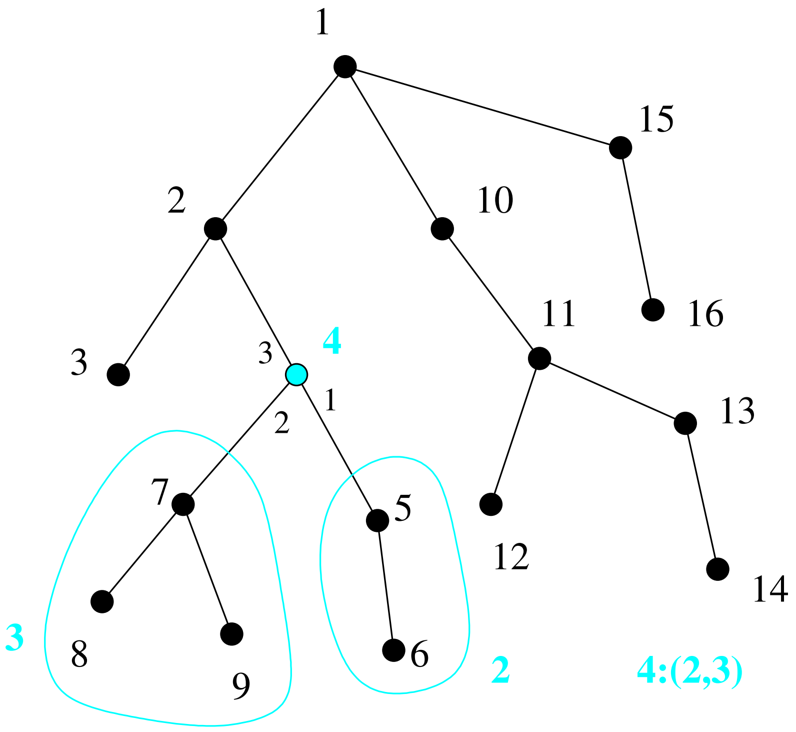
- tester en x sa table de routage vers toutes les destinations $y \in \{1, \ldots, n\} \setminus \{x\}$
- trier les réponses par fréquence $n_i = |T_i|$, $n_1 \leqslant \cdots \leqslant n_d$

$\sum_{i=1}^{d} n_i = n - 1$ et $1 \leqslant n_1 \leqslant \cdots \leqslant n_d$.

x "calcule" $(n_1, \ldots, n_d)$ sachant n et x.

Nombre de partitions de n is $\sim 2^{c\sqrt{n}}$ $\Rightarrow$ $c\sqrt{n}$ bits pour x.

Peut-on faire mieux que n bits ? (suite) : OUI !



- @ = parcours DFS par rapport au nombre de descendants
- port #1 = fils de + léger
  port #2= 2e fils le + léger
  ...

4 : $(2,3)$ $\Rightarrow$ théoriquement représentable en $O(\sqrt{n})$ bits.

$y \in]4, 4+2]$? $y \in]6, 6+3]$? sinon router sur 3.

Problème : Temps de routage ? Que se passe-t-il si les addresses sont codées sur plus de $\log_2 n$ bits ? Numérotation des #ports ?

Résultats (pour les arbres à n sommets)

Permut. ports ?	Taille des adresses	Taille des tables locales	Temps de routage	Pré-calcule des tables
oui oui	$\log n$ $\log n$	$\geqslant 3.7\sqrt{n}$ $\leqslant 3.7\sqrt{n}$	$2^{O(\sqrt{n})}$	$2^{O(\sqrt{n})}$
non non	$\log n$ $\log n$	$\geqslant n - o(n)$ $\leqslant n + o(n)$	$O(1)$	$O(n^2)$
non	$O\left(\frac{\log^2 n}{\log\log n}\right)$	$O\left(\frac{\log^2 n}{\log\log n}\right)$	$O(1)$	$O(n \log n)$
oui	$\leqslant 5 \log n$	$\leqslant 3 \log n$	$O(1)$	$O(n \log n)$

Principe : définitions

r = racine

id(x) = identifiant de x suivant un parcourts DFS

poids(x) = nombre de descendants de x

port$(x, \text{fils}(x)) = p + 1$ pour le p-ème fils le + lourds de x

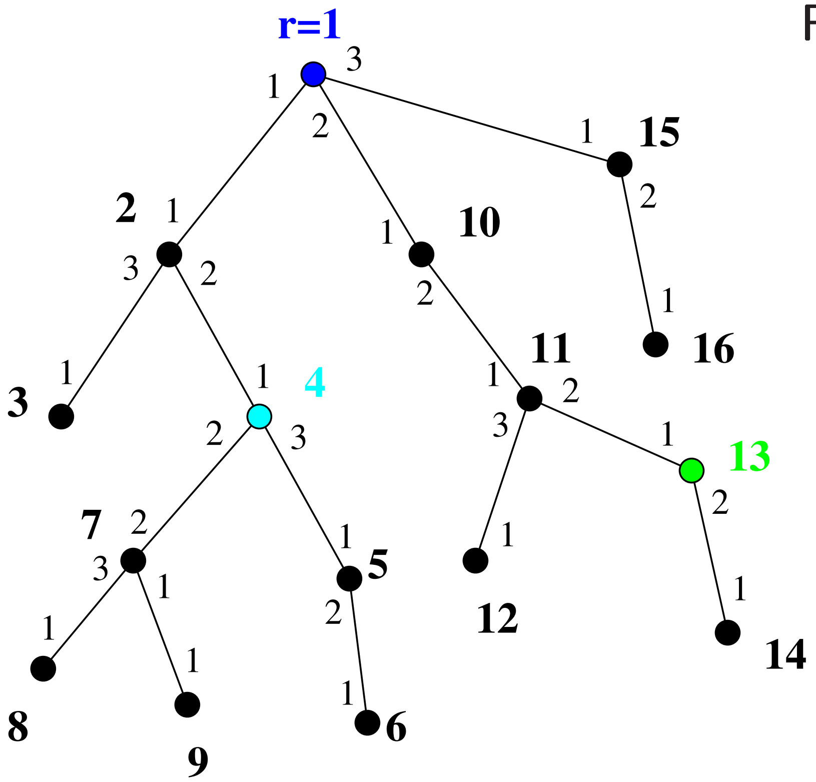
léger(x) = liste des ports "légers" allant de r à x

$$@(x) = \langle \text{id}(x), \text{léger}(x) \rangle$$

$$\text{table}(x) = \langle \text{id}(x), \text{poids}(x), \text{poids}(\text{fils}_1(x)), |\text{léger}(x)| \rangle$$

Principe : routage

$$@(x) = \langle \mathrm{id}(x), \mathrm{léger}(x) \rangle$$
$$\mathrm{table}(x) = \langle \mathrm{id}(x), \mathrm{poids}(x), \mathrm{poids}(\mathrm{fils}_1(x)), |\mathrm{léger}(x)| \rangle$$



Routage de x vers y :

- 0 si $\mathrm{id}(y) = \mathrm{id}(x)$
- 1 si $\mathrm{id}(y) \notin [\mathrm{id}(x), \mathrm{id}(x) + \mathrm{poids}(x)[$
- 2 si $\mathrm{id}(y) \in [\mathrm{id}(x), \mathrm{id}(x) + \mathrm{poids}(\mathrm{fils}_1(x))[$
- $p+1$ sinon, p-ième élément de $\mathrm{léger}(y)$

<13,2,1,1> $\mathrm{léger}(13) = (2, \cancel{2}, \cancel{2})$
<4,6,2,1> $\mathrm{léger}(4) = (\cancel{1}, 3)$

Analyse

rang(x, y) = rang du poids du fils y de x ($\approx$ port$(x, y) - 1$: $1, 2, \ldots$)

Observations :

- Si y fils de x, rang$(x, y) \leqslant$ poids(x)/poids(y)
- Si léger$(x) = (p_1, \ldots, p_k)$, $\prod p_i \leqslant n$ et $k \leqslant \log n$

Codage de léger(x)

Ex: léger$(x) = (3, 9, 2, 5, 2, 4, 6) \longrightarrow (A, B)$

léger(x)	$=$	3	9	2	5	2	4	6
écriture binaire : A	$=$	11	1001	10	101	10	100	110
délimitations : B	$=$	10	1000	10	100	10	100	100

$$|B| = |A| \quad \text{et} \quad |A| \leqslant \sum_{i=1}^{k} \lceil \log p_i \rceil \leqslant \log \left(\prod p_i \right) + k \leqslant 2 \log n$$

$\Rightarrow$ léger$(x) = (A, B)$ sur $4 \log n$ bits

$\Rightarrow$ @(x) sur $5 \log n$ bits

$\Rightarrow$ table(x) sur $3 \log n + O(\log \log n)$ bits

Extraction du p-ième élément

$\text{sélecte}_B(t)$ = position du t-ième "1" dans B.

Si B fixée, $\text{sélecte}_B(\cdot)$ implémentable avec espace $|B| + o(|B|)$ bits et temps constant si multiplication entière (sur $\log n$ bits) en temps constant.

Ex: $\text{léger}(x) = (3, 9, 2, \underline{5}, 2, 4, 6)$, $p = 4$.

$A = \overline{11100110}101\overline{10100110}$

$B = 10100010\underline{1}00\underline{1}0100100$

$a := \text{sélecte}_B(p)$,

$b := \text{sélecte}_B(p+1)$,

$s := (A \ll (a-1)) \gg (|A| + a - b) =$ "101" $\to$ 5.

Encore plus compact ? OUI !

Lemme (Kalmár 1930). Soit $Z(n)$ le nombre de suites d'entiers $p_1, p_2, \ldots$ telles que $p_i \geqslant 2$ et $\prod p_i \leqslant n$. Alors, $Z(n) \sim n^{1.7286\ldots}$ (solution réelle de $\zeta^{-1}(2) = 1.7286\ldots$, $\zeta(x) = \sum_{i \geqslant 1} i^{-x}$).

D'où @(x) sur $2.7286 \log n$ bits et table(x) sur $3 \log n$ bits.

[Thorup-Zwick, SPAA/STOC '01]:
@(x) sur $(1 + o(1)) \log n$ bits et
table(x) sur $(1 + o(1)) \log n$ bits.