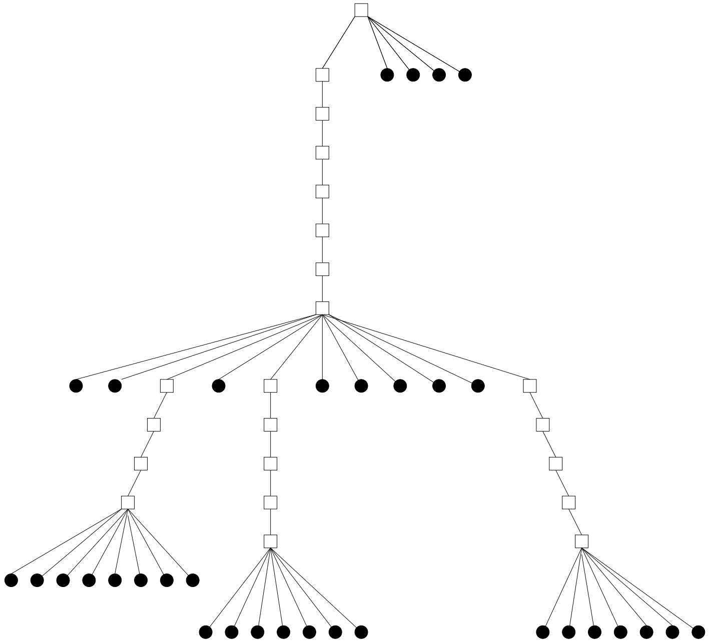
Tree edit distance analysis

Serge Dulucq (LABRI)

Hélène Touzet (LIFL)

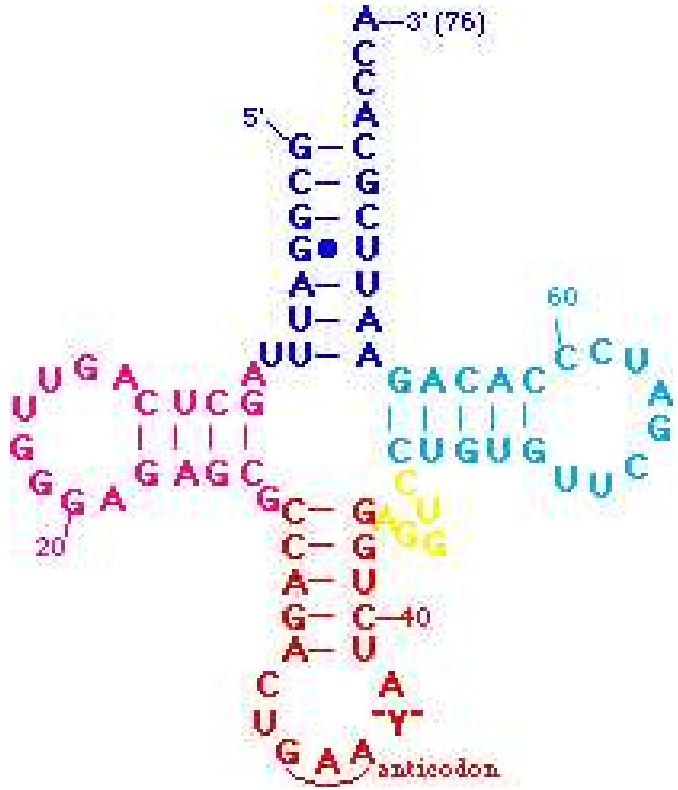
CPM 2003

Why comparing tree ?



□ : base pair (stem)

● : unpaired base (loop)



Tree representation for the secondary structure of tRNA

Tree Edit Distance Problem

- ▷ A pair of ordered rooted trees (A, B)
- ▷ Costs for edit operations
  - `sub`: substituting a node
  - `ins`: inserting a node
  - `del`: deleting a node
- ▷ **Distance: minimal cost to transform A into B**

String edit distance I

A L G O R I T H M

F L O W E R

substitution *insertion* *deletion*

A	LGORITHM	-	ALGORITHM	A	LGORITHM
F	LOWER	F	LOWER	-	FLOWER

Leftmost decomposition

Comparison of all pairs of suffixes

String edit distance II

A L G O R I T H M

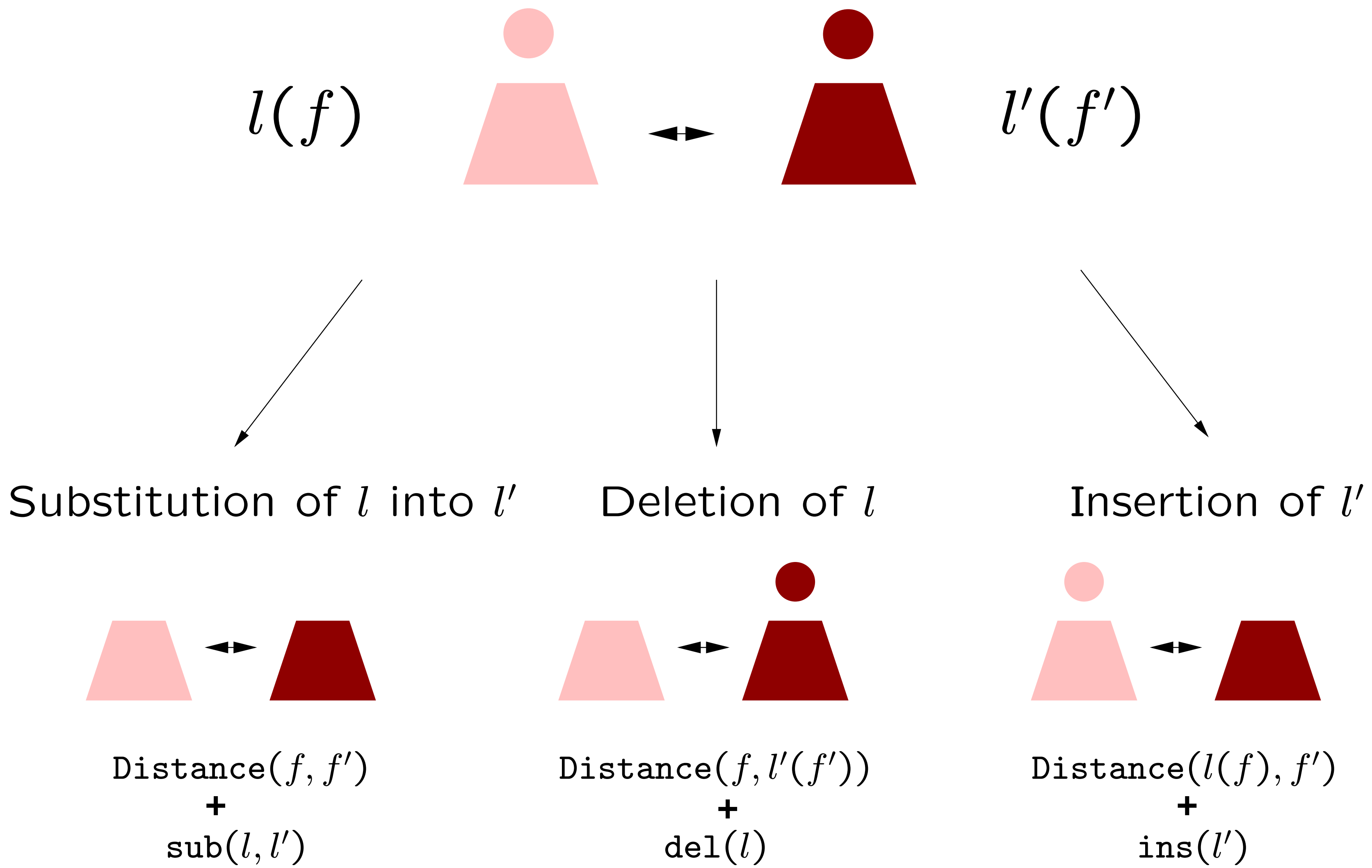
F L O W E R

substitution	*insertion*	*deletion*
ALGORITH M	ALGORITHM -	ALGORITH M
FLOWE R	FLOWE R	FLOWER -

Rightmost decomposition

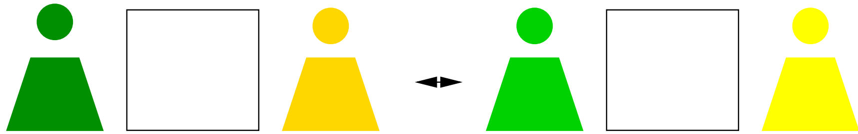
Comparison of all pairs of prefixes

Tree edit distance



Forest edit distance I

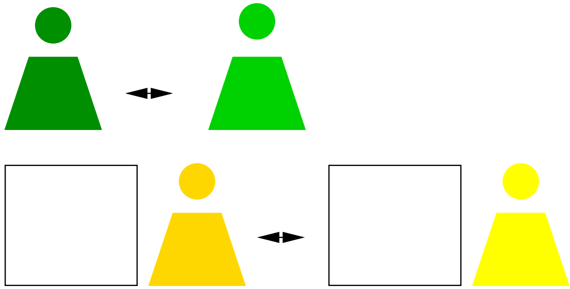
$l(f) \circ t$



$l'(f') \circ t'$

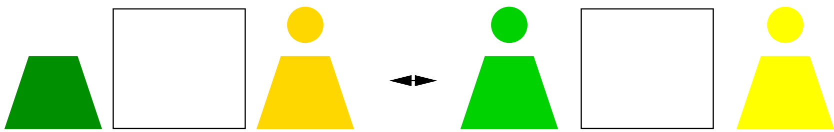
Leftmost decomposition

Substitution of l into l'



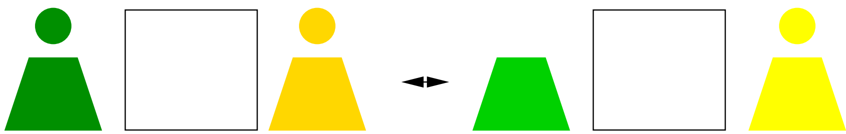
$$\texttt{Distance}(l(f), l'(f')) + \texttt{Distance}(t, t')$$

Deletion of l



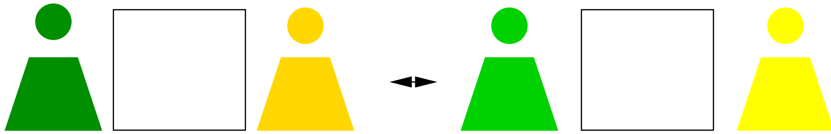
$$\texttt{Distance}(f \circ t, l'(f') \circ t') + \texttt{del}(l)$$

Insertion of l'



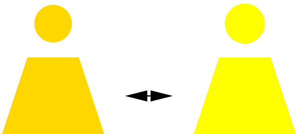
$$\texttt{Distance}(l(f) \circ t, f' \circ t') + \texttt{ins}(l')$$

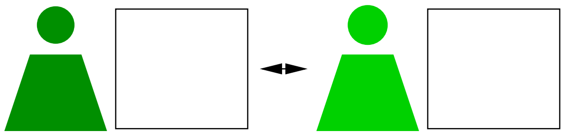
Forest edit distance II

$t \circ l(f)$  $t' \circ l'(f')$

Rightmost decomposition

Substitution of l into l'



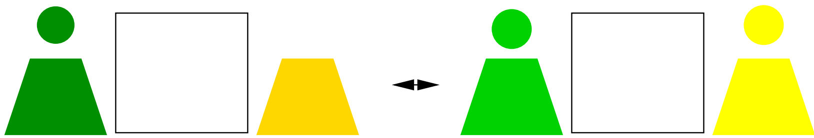


$\texttt{Distance}(l(f), l'(f'))$

$+$

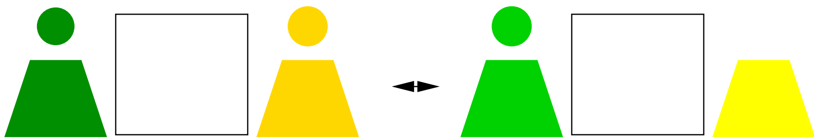
$\texttt{Distance}(t, t')$

Deletion of l



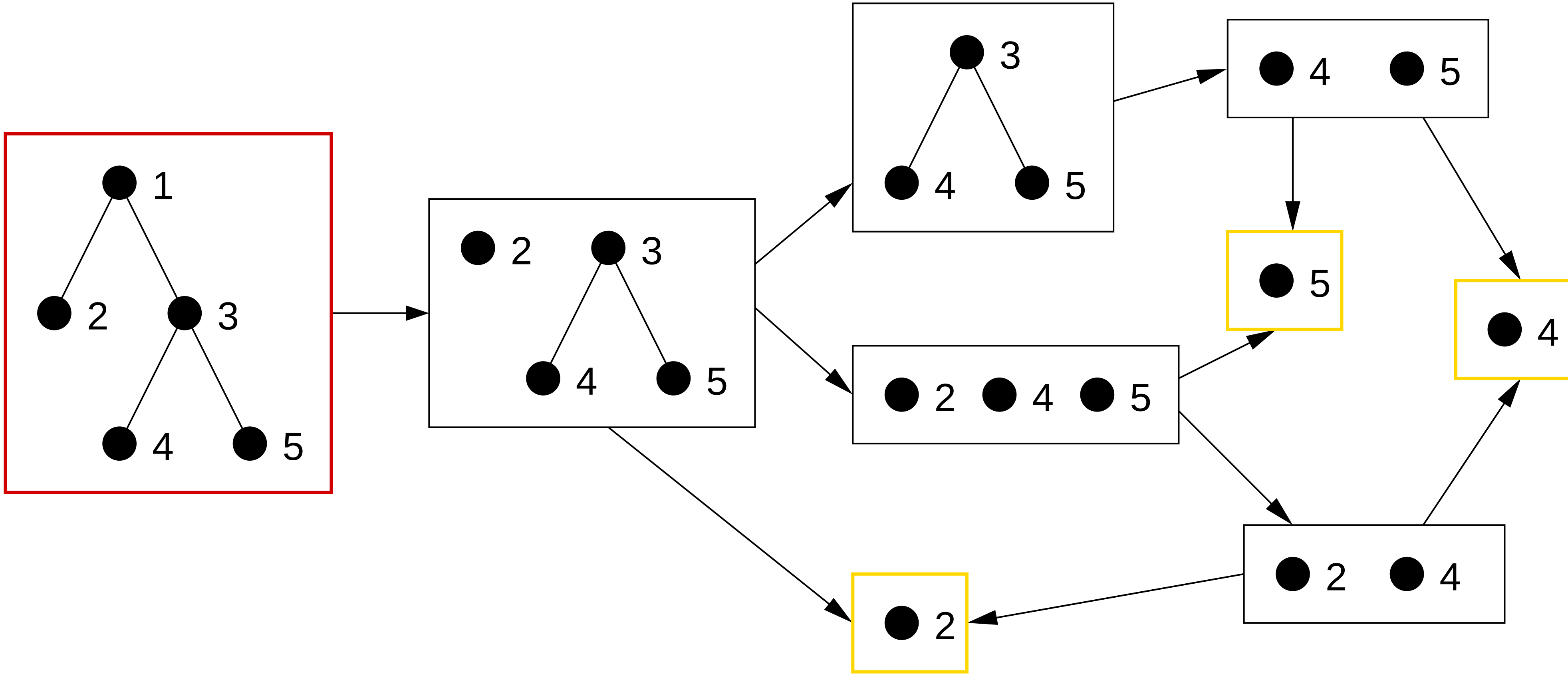
$\texttt{Distance}(t \circ f, t' \circ l'(f')) + \texttt{del}(l)$

Insertion of l'



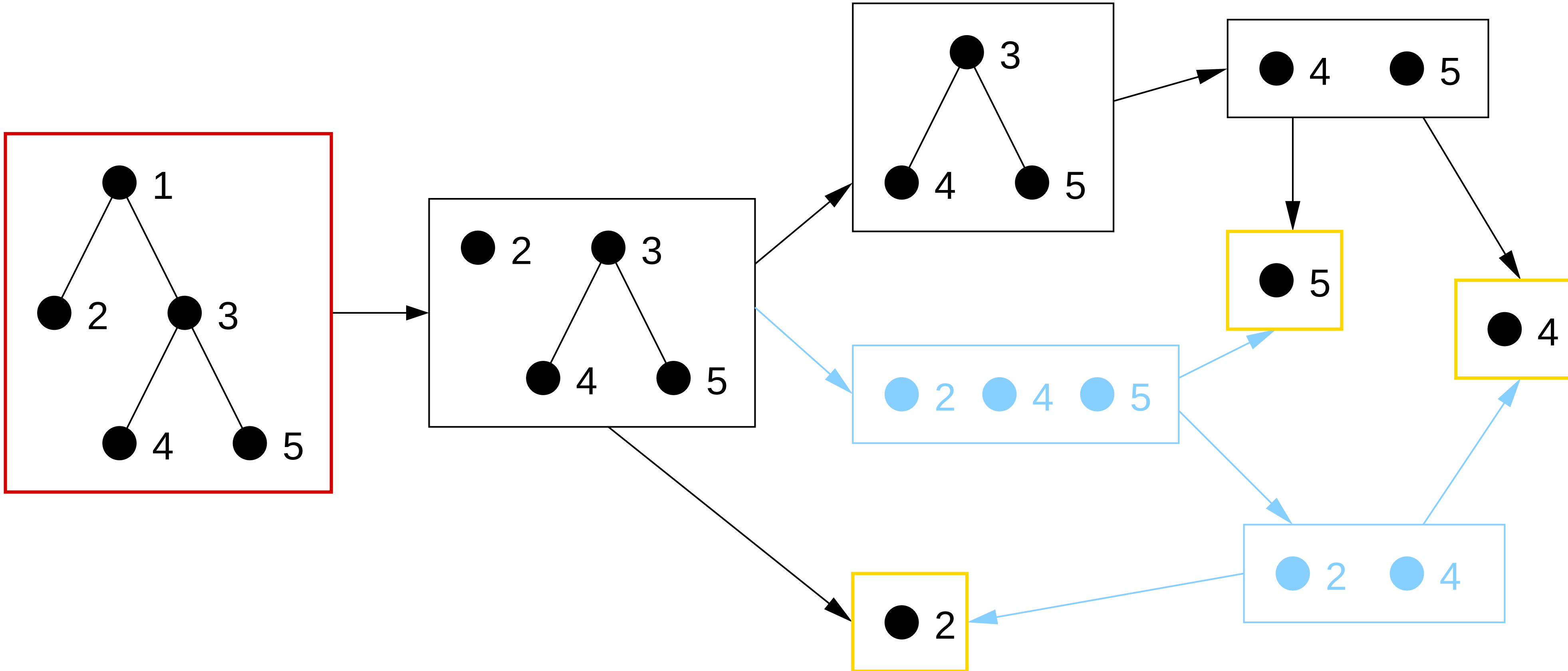
$\texttt{Distance}(t \circ l(f), t' \circ f') + \texttt{ins}(l')$

Left/right for trees ?



rightmost decomposition

Left/right for trees ?



leftmost decomposition

Decomposition strategies

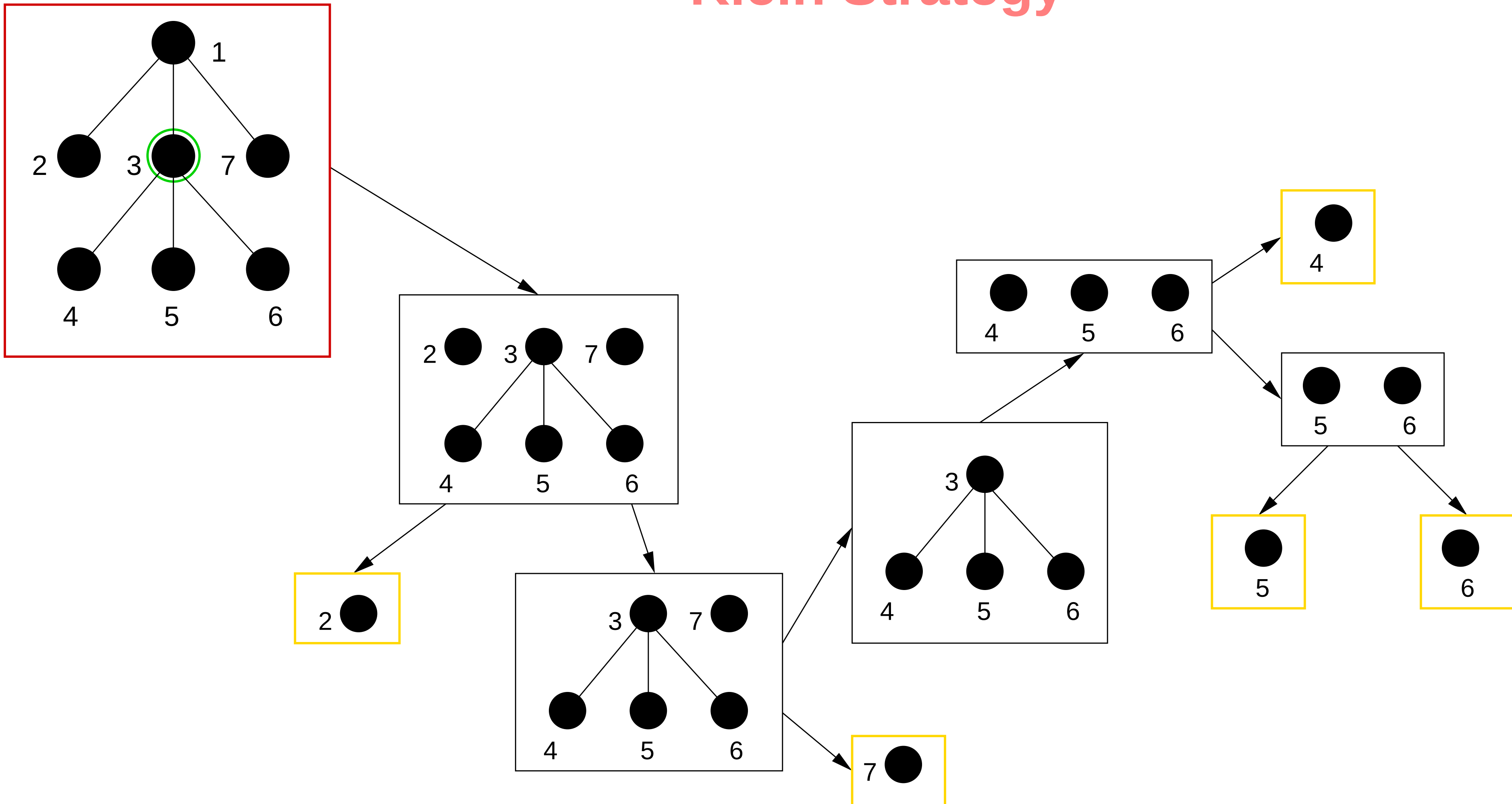
- ▷ Succession of choices **left** or **right**
- ▷ $\mathcal{S}$: *forest* $\times$ *forest* $\rightarrow \{\text{left}, \text{right}\}$
- ▷ **Zhang & Shasha** (1989) :

$$(f, g) \rightarrow \text{left}$$

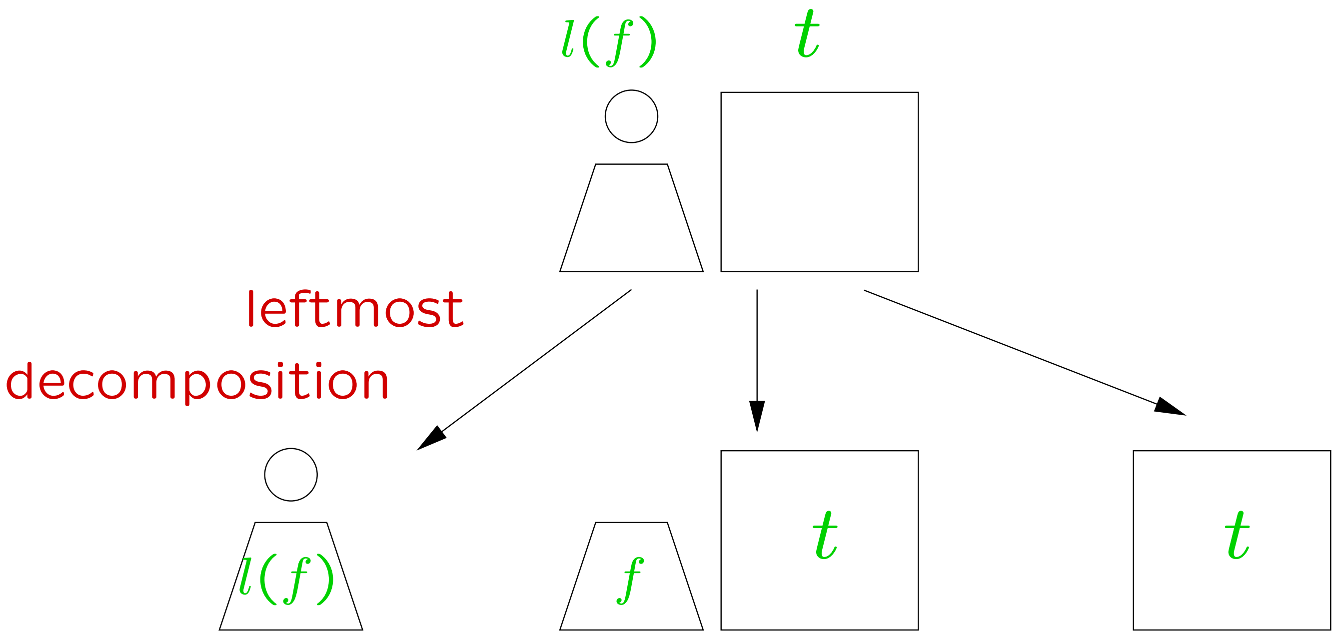
- ▷ **Klein** (1998) :

$$(f, g) \rightarrow \begin{cases} \textbf{right}, \text{ when the first node of } f \text{ is the heaviest child} \\ \textbf{left}, \text{ otherwise} \end{cases}$$

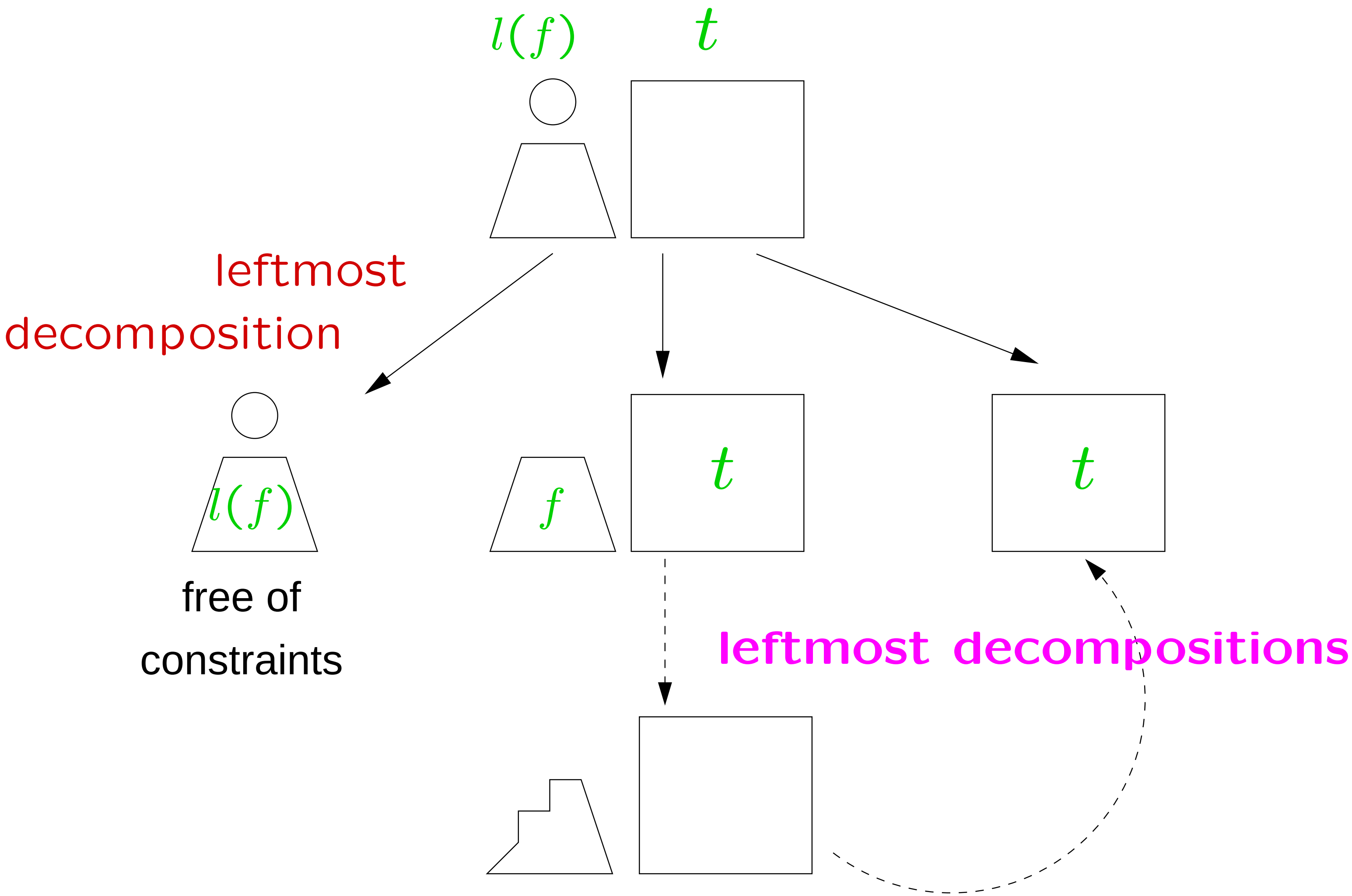
Klein Strategy



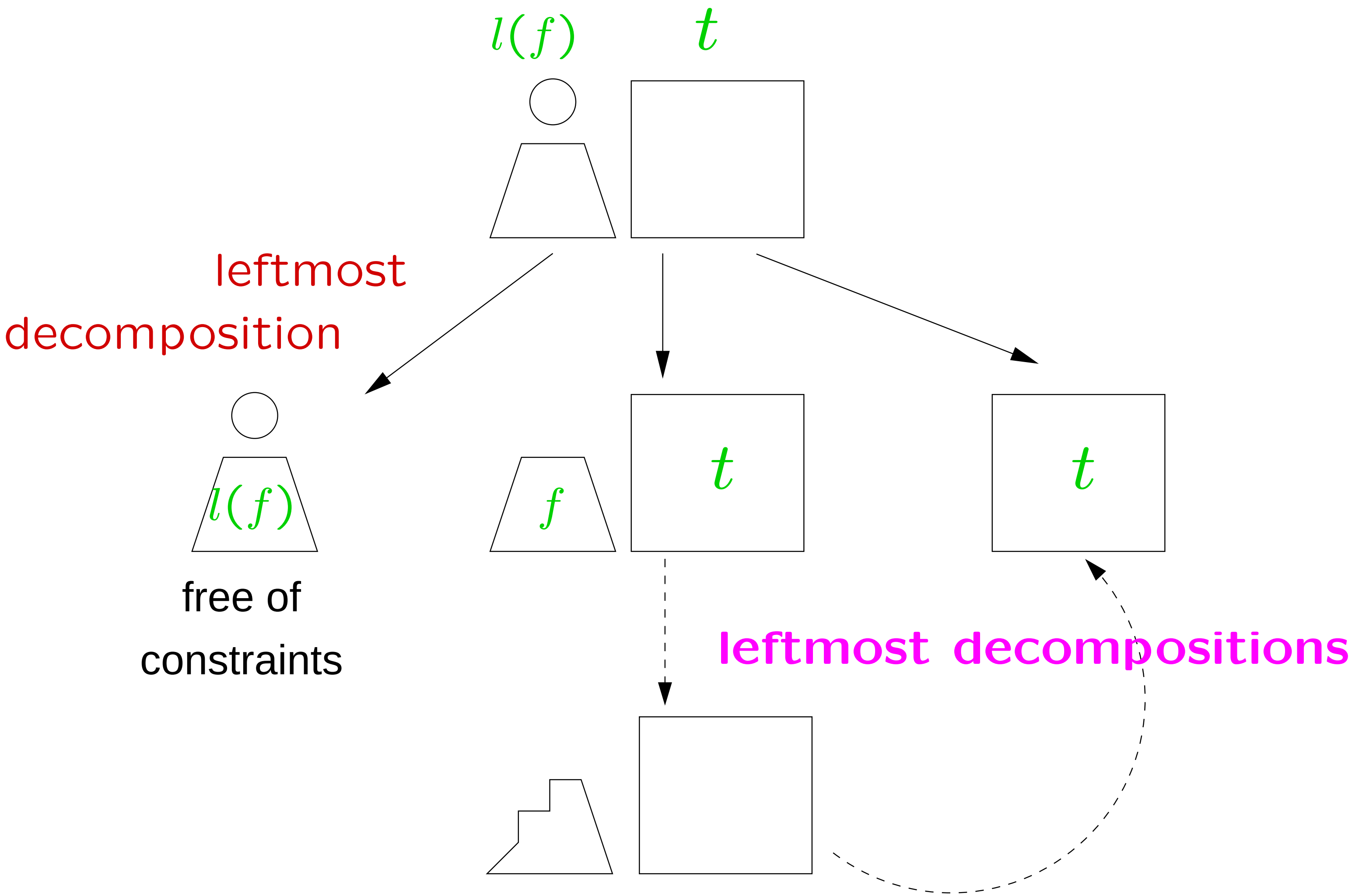
How to built up an economical strategy ?



How to built up an economical strategy ?



How to built up an economical strategy ?



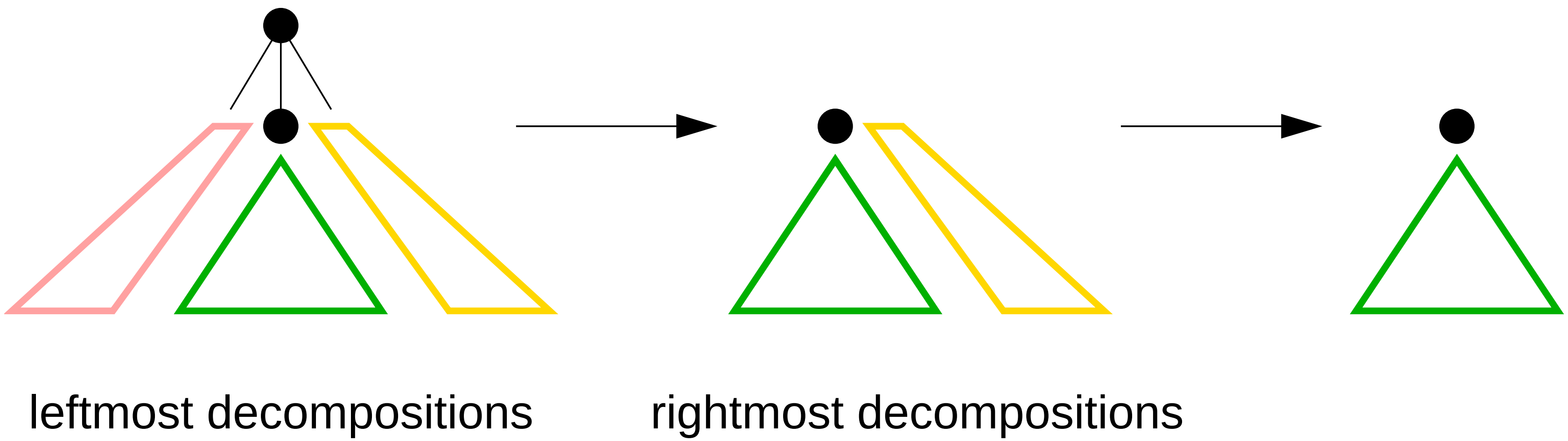
$$\#\text{subforest}(l(f) \circ t) = \#\text{subforest}(l(f)) + |l(f)| + \#\text{subforest}(t)$$

Cover strategies are economical strategies

For a tree A, define a **cover** ϕ for A as

▷ $\phi(i) \in \{left, right\}$ if the degree of i is 0 or 1 : **direction**

▷ $\phi(i)$ is a child of i: **favorite** child



Zhang & Shasha and Klein are cover strategies.

Number of subforests for one tree

$A = l(A_1 \circ \ldots \circ A_n)$

▷ <u>Lower bound</u> (no assumption on the strategy)

$\texttt{\#subforest}(A) \geq |A| - |A_j| + \texttt{\#subforest}(A_1) + \cdots + \texttt{\#subforest}(A_n)$ $\quad O(n \log(n))$

A_j is the heaviest child

▷ <u>Upper bound</u> (no assumption on the strategy)

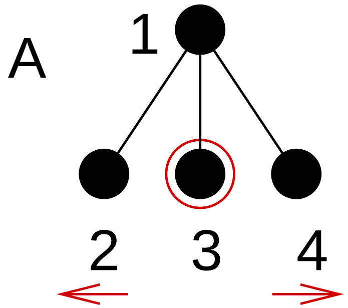
$\texttt{\#subforest}(A) \leq \frac{n(n+3)}{2} - \sum_{i \in A} |A(i)|$ $\quad \frac{1}{2}\, n^2 + \frac{\sqrt{\pi}}{2}\, n^{\frac{3}{2}} + O(n)$ in average

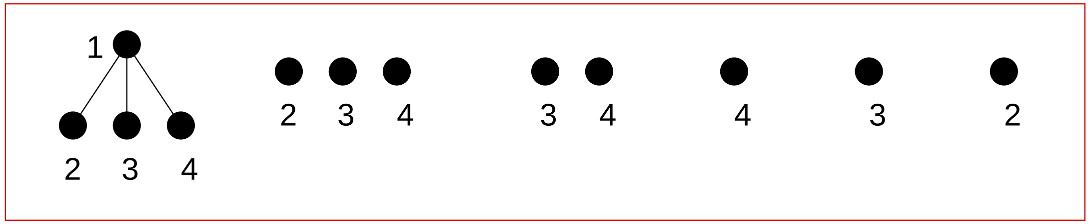
▷ <u>Exact number for a cover strategy</u>

$\texttt{\#subforest}(A) = |A| - |A_j| + \texttt{\#subforest}(A_1) + \cdots + \texttt{\#subforest}(A_n)$

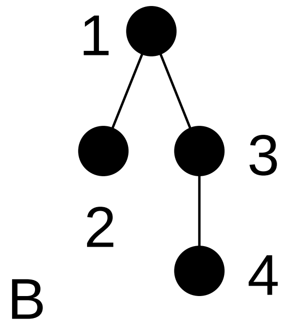
A_j is the favorite child

Example





$4 - 1 + 1 + 1 + 1 = 6$ subforests for A



What happens to the other tree B?

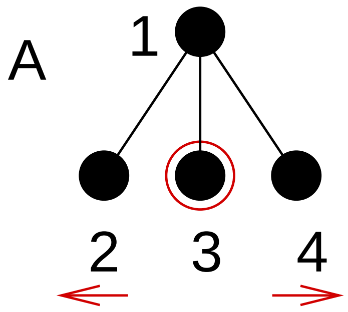
There are only three possibilities for the subforests of the cover tree A

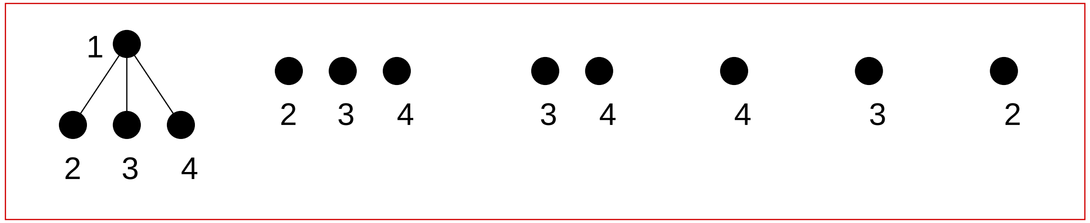
- $\triangleright$ being compared with all leftmost forests of B
- $\triangleright$ being compared with all rightmost forests of B
- $\triangleright$ being compared with all forests of B

$$\begin{aligned} \texttt{\#left}(B) &: \text{ number of leftmost subforests of } B \\ \texttt{\#right}(B) &: \text{ number of rightmost subforests of } B \\ \texttt{\#special}(B) &: \text{ number of subforests of } B \end{aligned}$$

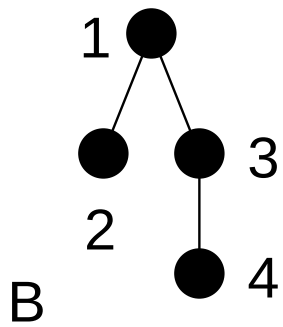
$\texttt{\#left}(B)$, $\texttt{\#right}(B)$, $\texttt{\#special}(B)$ are known

Example (continued)

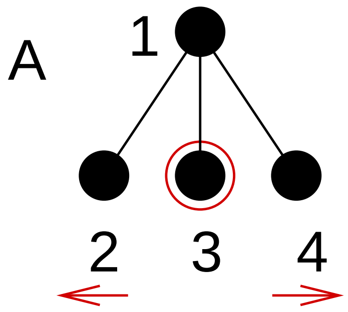


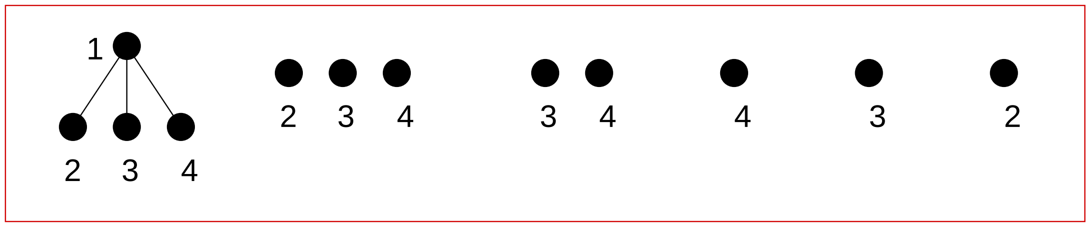


$4 - 1 + 1 + 1 + 1 = 6$ subforests for A

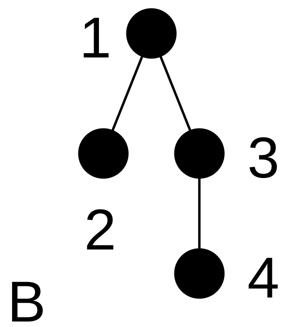


Example (continued)





$4 - 1 + 1 + 1 + 1 = 6$ subforests for A



$\#\mathtt{right}(B) = 4 - 2 + 1 + 2 = 5$
$\#\mathtt{left}(B) = 4 - 1 + 1 + 2 = 6$
$\#\mathtt{special}(B) = 4 * 7/2 - 4 - 1 - 2 - 1 = 6$

The favorite children inherit subforests from their parent.

4 kinds of nodes :

- ▷ **Free:** nodes that do not receive anything
- ▷ **Left :** nodes that inherit leftmost forests of B
- ▷ **Right :** nodes that inherit rightmost forests of B
- ▷ **All:** nodes that inherit all subforests of B

The status of a node depends of the direction and of the heritage.

1. A is reduced to a node with direction right

$$\begin{aligned} \mathsf{Free}(A) &= \mathsf{Left}(A) &= \texttt{\#left}(B) \\ \mathsf{All}(A) &= \mathsf{Right}(A) &= \texttt{\#special}(B) \end{aligned}$$

2. A is reduced to a node with direction left

$$\begin{aligned} \mathsf{Free}(A) &= \mathsf{Right}(A) &= \texttt{\#right}(B) \\ \mathsf{All}(A) &= \mathsf{Left}(A) &= \texttt{\#special}(B) \end{aligned}$$

3. $A = l(A')$ and the direction of l is right

$$\begin{aligned} \mathsf{Free}(A) &= \mathsf{Left}(A) &= \texttt{\#left}(B) + \mathsf{Right}(A') \\ \mathsf{All}(A) &= \mathsf{Right}(A) &= \texttt{\#special}(B) + \mathsf{All}(A') \end{aligned}$$

4. $A = l(A')$ and the direction of l is left

$$\begin{aligned} \mathsf{Free}(A) &= \mathsf{Right}(A) &= \texttt{\#right}(B) + \mathsf{Left}(A') \\ \mathsf{All}(A) &= \mathsf{Left}(A) &= \texttt{\#special}(B) + \mathsf{All}(A') \end{aligned}$$

5. $A = l(A_1 \circ \cdots \circ A_n)$ and the favorite child is A_1?

$$\begin{array}{rcccl} \mathsf{Free}(A) & = & \mathsf{Left}(A) & = & \sum_{i>1} \mathsf{Free}(A_i) + \mathsf{Left}(A_1) + \mathtt{\#left}(B)(|A| - |A_1|) \\ \mathsf{All}(A) & = & \mathsf{Right}(A) & = & \sum_{i>1} \mathsf{Free}(A_i) + \mathsf{All}(A_1) + \mathtt{\#special}(B)(|A| - |A_1| \end{array}$$

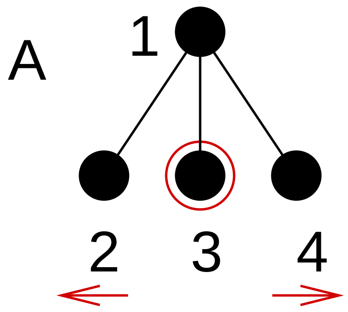
6. $A = l(A_1 \circ \cdots \circ A_n)$ and the favorite child is A_n?

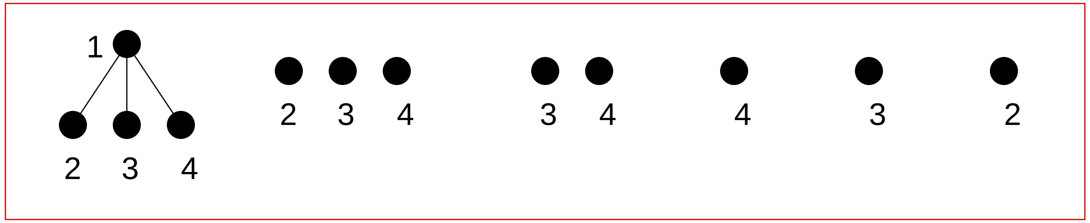
$$\begin{array}{rcccl} \mathsf{Free}(A) & = & \mathsf{Right}(A) & = & \sum_{i<n} \mathsf{Free}(A_i) + \mathsf{Right}(A_n) + \mathtt{\#right}(B)(|A| - |A_n \\ \mathsf{All}(A) & = & \mathsf{Left}(A) & = & \sum_{i<n} \mathsf{Free}(A_i) + \mathsf{All}(A_n) + \mathtt{\#special}(B)(|A| - |A_n| \end{array}$$

7. otherwise: let A_j $(1 < j < n)$ be the favorite child

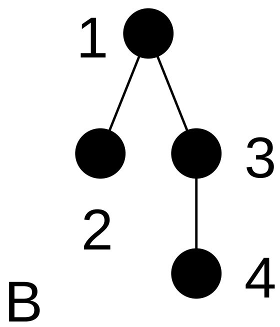
$$\begin{array}{rl} \mathsf{Free}(A) = & \sum_{i \neq j} \mathsf{Free}(A_i) + \mathsf{All}(A_j) + \mathtt{\#right}(B)(1 + |A_1 \circ \cdots \circ A_{j-1}|) \\ & + \mathtt{\#special}(B)|A_j \circ \cdots \circ A_n| \\ \mathsf{Right}(A) = & \mathsf{Free}(A) \\ \mathsf{All}(A) = & \mathsf{Left}(A) = \sum_{i \neq j} \mathsf{Free}(A_i) + \mathsf{All}(A_j) + \mathtt{\#special}(B)(|A| - |A_j|) \end{array}$$

Example (end)



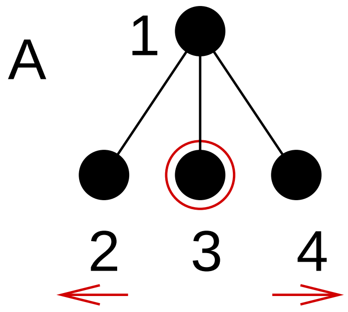


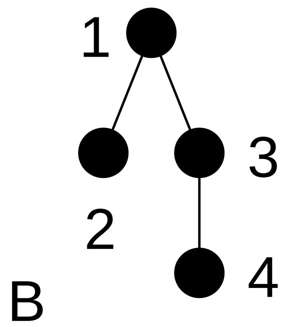
$4 - 1 + 1 + 1 + 1 = 6$ subforests for A



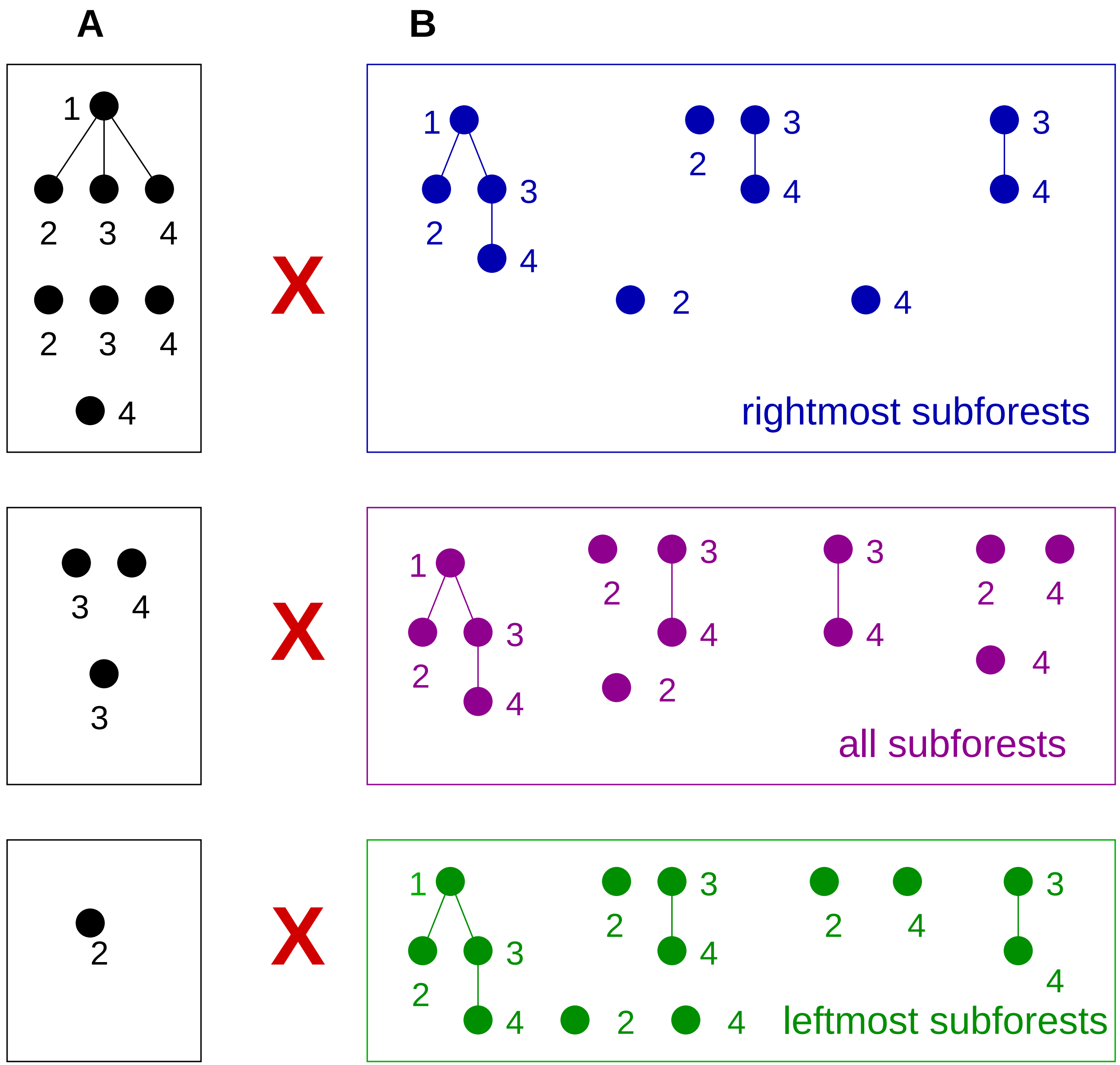
$$\texttt{\#right}(B) = 4 - 2 + 1 + 2 = 5$$
$$\texttt{\#left}(B) = 4 - 1 + 1 + 2 = 6$$
$$\texttt{\#special}(B) = 4 * 7/2 - 4 - 1 - 2 - 1 = 6$$

Example (end)





32 pairs of subforests

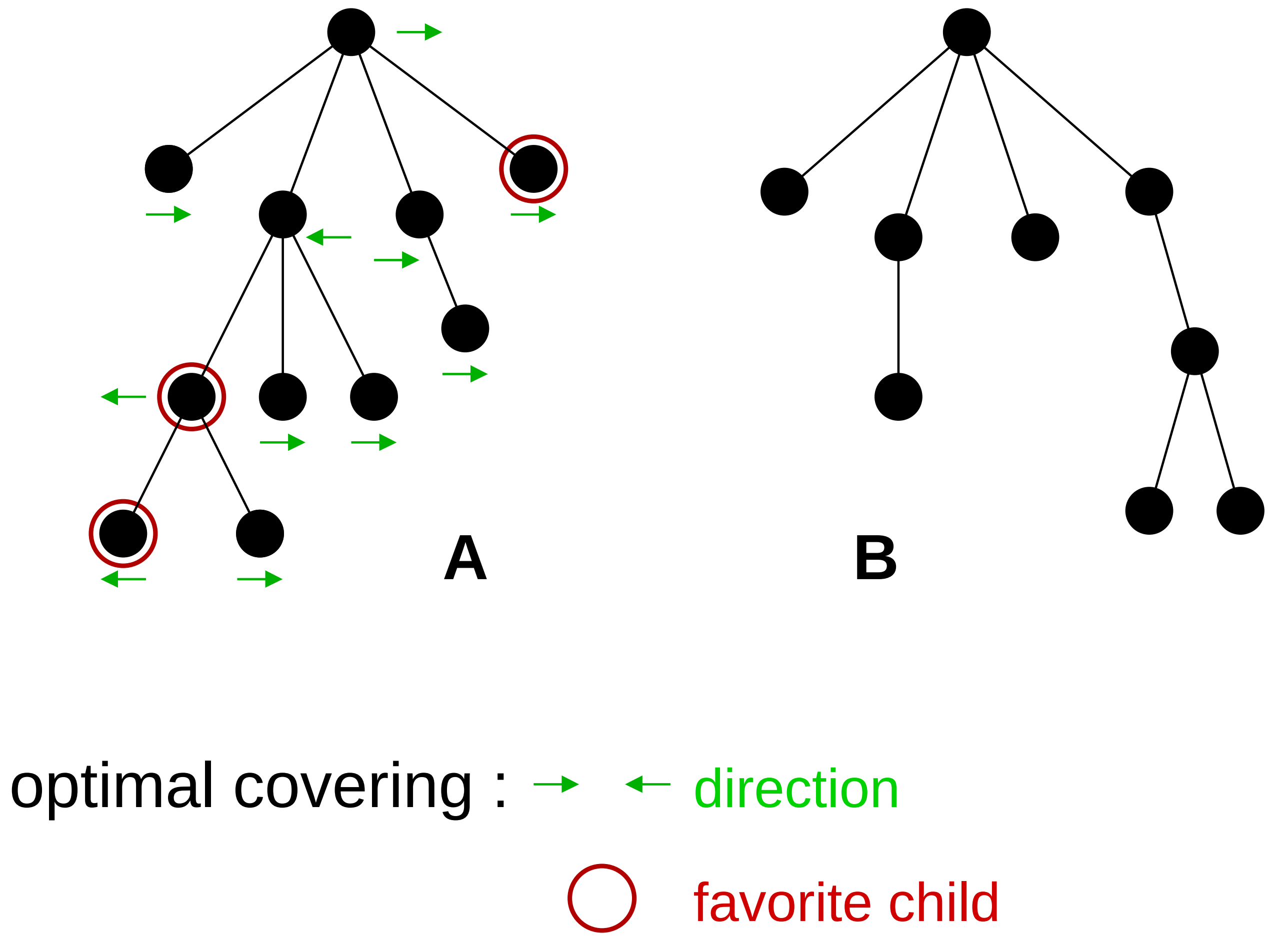


How to construct an optimal cover ?

- Dynamic programming
- Four tables : *Free*, *All*, *Left*, *Right*

$$
\begin{array}{rcl}
\mathit{Free}(A) & = & \sum_{i \geq 1} \mathit{Free}(A_i) \\
 & & + \min \left\{ \begin{array}{l} \mathit{Left}(A_1) - \mathit{Free}(A_1) + \texttt{\#left}(B) * (|A| - |A_1|) \\ \mathit{All}(A_j) - \mathit{Free}(A_j) + \texttt{\#special}(B) |A_j \circ \cdots \circ A_n| \\ + \texttt{\#right}(B)(1 + |A_1 \circ \cdots \circ A_{j-1}|), \quad 1 < j < n \\ \mathit{Right}(A_n) - \mathit{Free}(A_n) + \texttt{\#right}(B) * (|A| - |A_n|) \end{array} \right.
\end{array}
$$

- Preprocessing : $O(\sum_i \text{degree(A(i))}) + O(|B|) = O(|A|) + O(|B|)$


A
B
optimal covering :
direction
favorite child

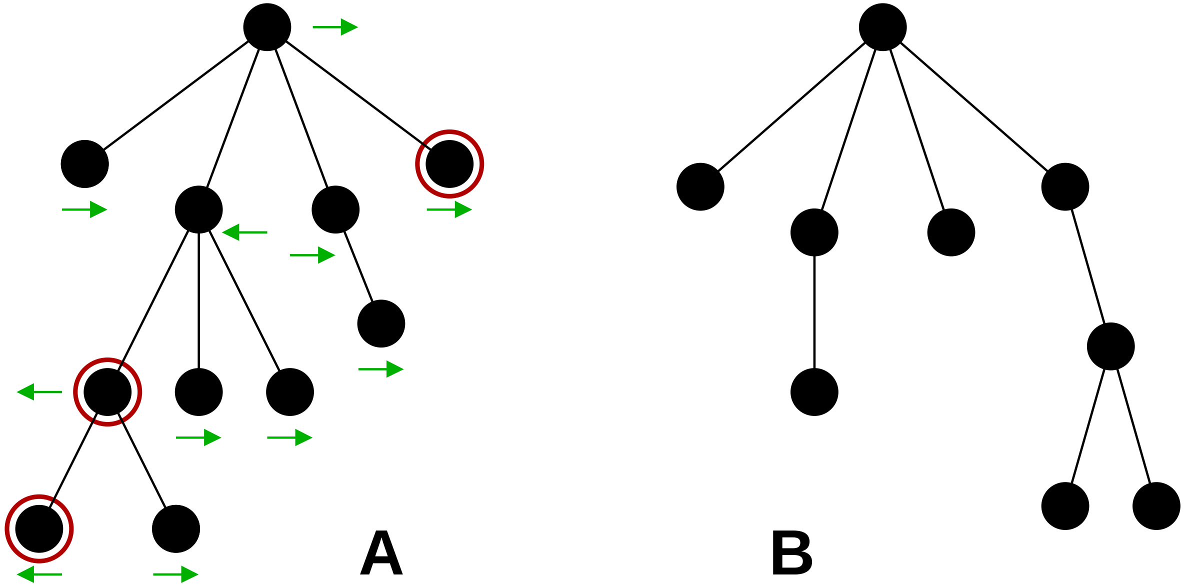
Number of pairs
of subforests

optimal : 340

right : 405

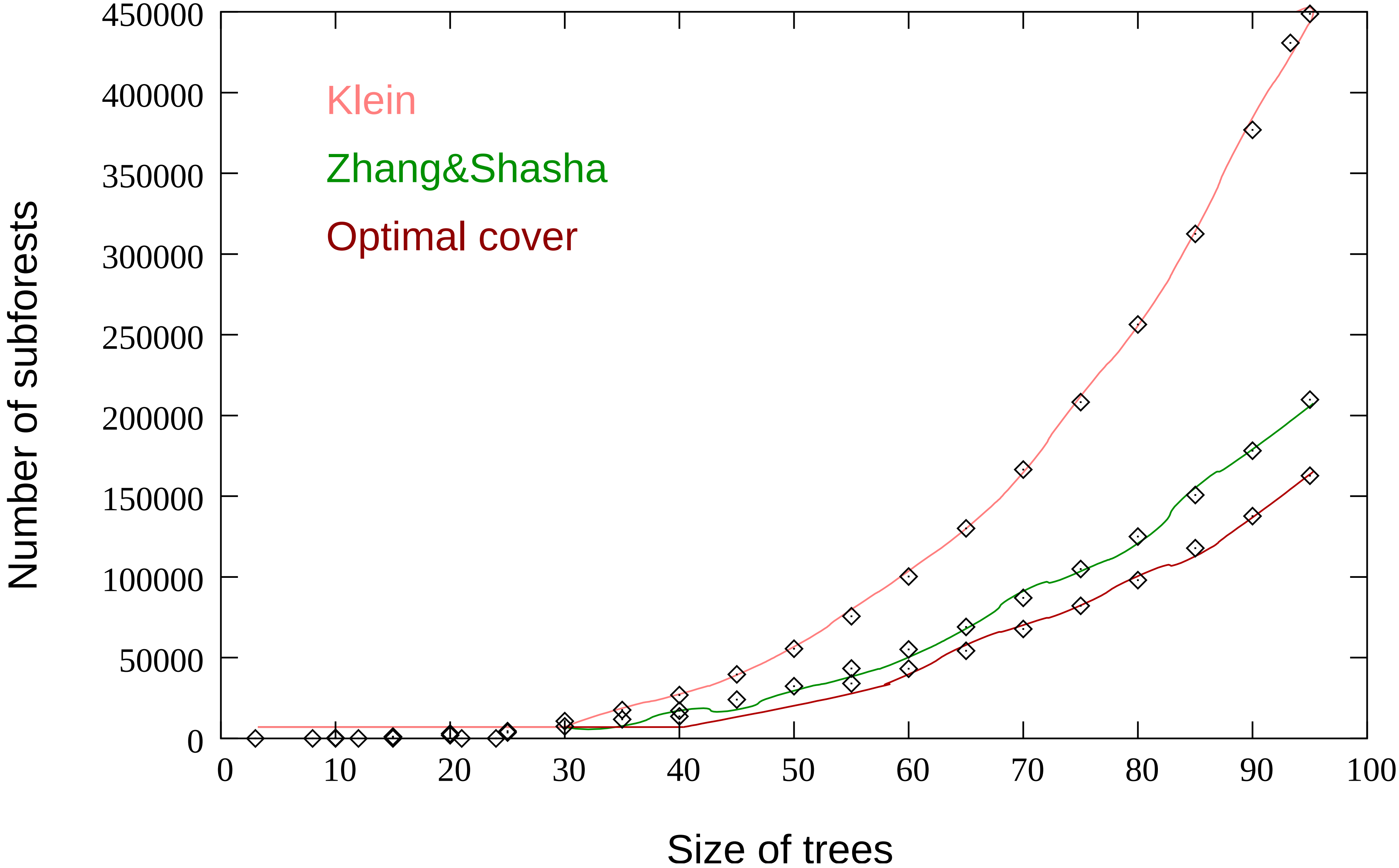
left : 350

Klein : 391



optimal covering : → ← direction

favorite child



Klein
Zhang&Shasha
Optimal cover
Number of subforests
Size of trees
0
50000
100000
150000
200000
250000
300000
350000
400000
450000
0
10
20
30
40
50
60
70
80
90
100