A linear-time algorithm for comparing similar ordered trees

Hélène Touzet

LIFL – University of Lille 1 – France

Comparison with k errors

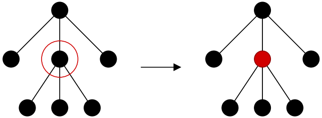
- **P**roblem :

Input	:	two ordered trees (that are assumed to be similar) a natural number k
Output	:	the best mapping $\mathcal{M}$ containing less than k errors, if it exists

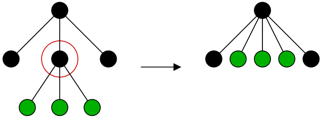
- **E**rror : insertion of a node, deletion of a node
- **E**dit operations : substitution, deletion, insertion
- **C**omparison model: edit distance *vs* alignment

How to compare trees: edit operations

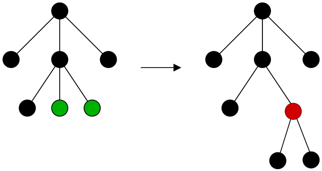
Substitution



Deletion

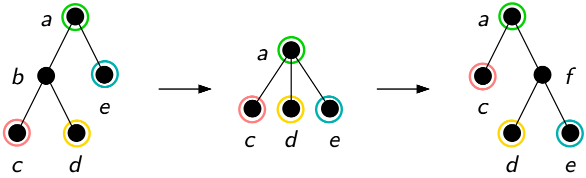


Insertion

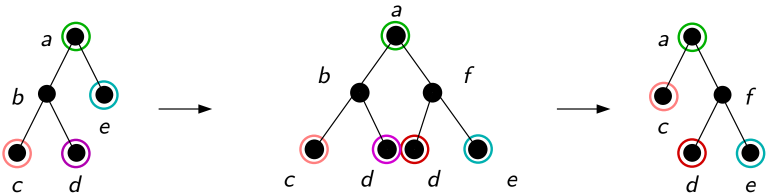


How to compare trees: comparison model

- **E**dit Distance [Tai 1979, Zhang-Shasha 1989, Klein 1998, Dulucq &Touzet 2003]
  - all mappings are valid
  - largest common subtree



- **A**lignment [Jiang *et al.* 1995]
  - insertions should precede deletions
  - smallest common supertree



Previous results

	Strings	Tree distance	Tree alignment
full mapping	$O(n^2)$	$O(n^4)$ Zhang-Shasha $O(n^3 \log(n))$ Klein	$O(n^2 d^2)$ Jiang *et al.*
k-errors	$O(kn)$		$O(n \log(n) d^3 k^2)$ Jansson-Lingas

n : size of the tree
d : maximal degree of the tree
k : bound on the number of errors - known in advance

Previous results

	Strings	Tree distance	Tree alignment
full mapping	$O(n^2)$	$O(n^4)$ Zhang-Shasha $O(n^3 \log(n))$ Klein	$O(n^2 d^2)$ Jiang *et al.*
k-errors	$O(kn)$	$O(k^3 n)$	$O(n \log(n) d^3 k^2)$ Jansson-Lingas

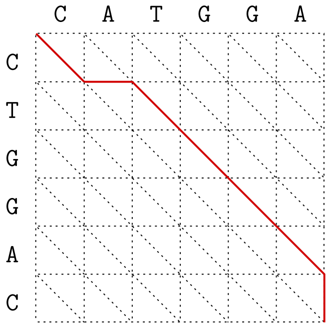
n : size of the tree

d : maximal degree of the tree

k : bound on the number of errors - known in advance

Edit graph for the string alignment problem

- **T**wo-dimensional grid
- **T**hree kinds of arcs: deletion, insertion and substitution

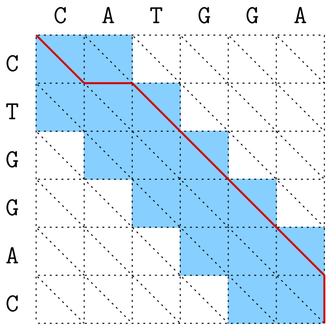



```
C  A  T  G  G  A  -
|     |  |  |  |
C  -  T  G  G  A  C
```

Time complexity: $O(n^2)$

Edit graph for the string alignment problem

- **T**wo-dimensional grid
- **T**hree kinds of arcs: deletion, insertion and substitution



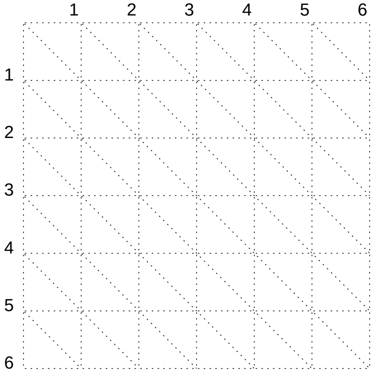

```
C   A   T   G   G   A   -
|       |   |   |   |
C   -   T   G   G   A   C
```

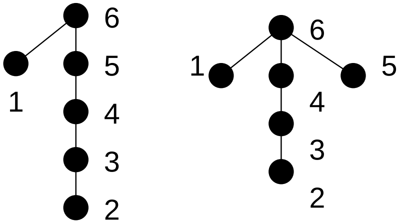
Time complexity: $O(n^2)$

With k-errors : $O(kn)$

Tree edit graph

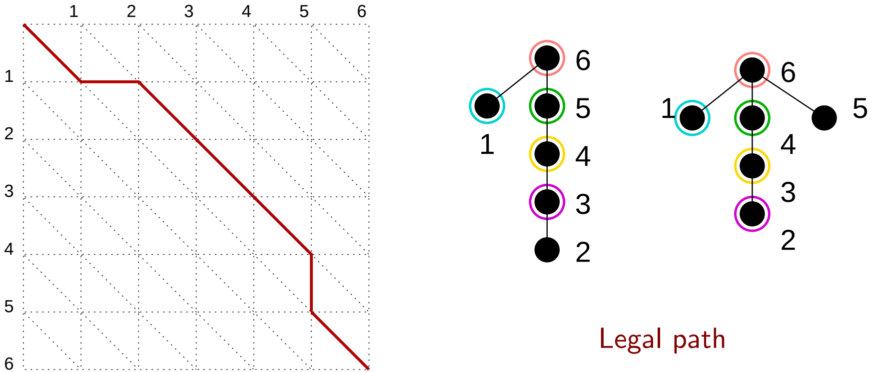
- **T**rees as strings : enumerate the nodes in postorder traversal
- **S**upplementary constraints imposed by the tree structure





Tree edit graph

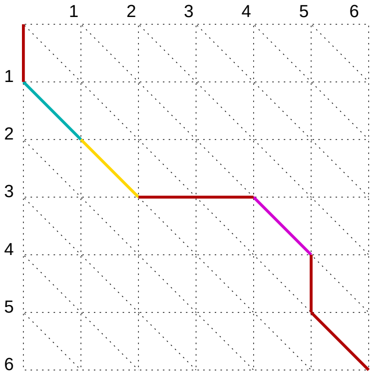
- **T**rees as strings : enumerate the nodes in postorder traversal
- **S**upplementary constraints imposed by the tree structure

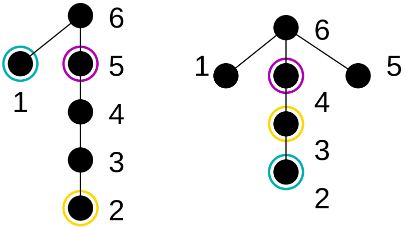


Legal path

Tree edit graph

- **T**rees as strings : enumerate the nodes in postorder traversal
- **S**upplementary constraints imposed by the tree structure

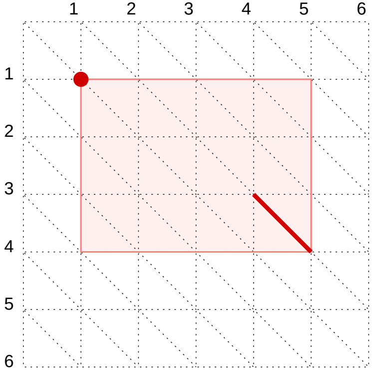


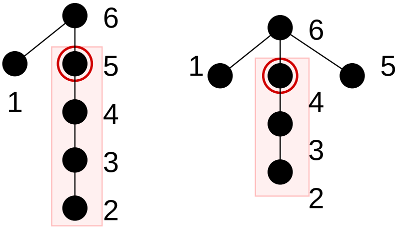


Illegal path

Tree edit graph

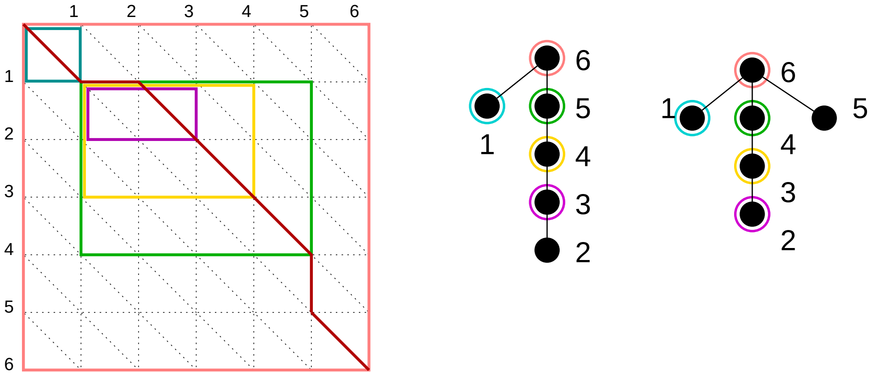
- **T**rees as strings : enumerate the nodes in postorder traversal
- **S**upplementary constraints imposed by the tree structure





Tree edit graph

- **T**rees as strings : enumerate the nodes in postorder traversal
- **S**upplementary constraints imposed by the tree structure



Edit graph for trees

- **D**eletion arcs (horizontal arcs):

  $$(x, y) \rightsquigarrow (x-1, y) \text{ labeled by } \texttt{del}$$

- **I**nsertion arcs (vertical arcs):

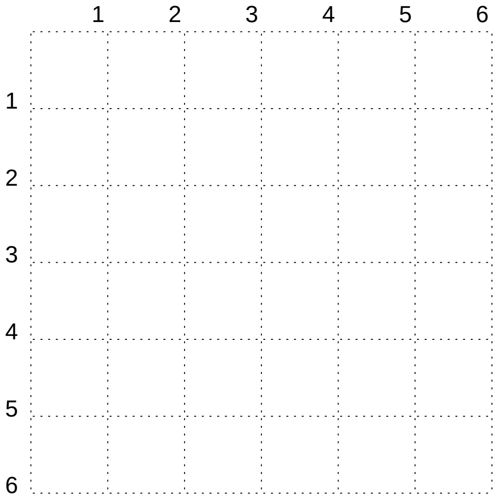
  $$(x, y) \rightsquigarrow (x, y-1) \text{ labeled by } \texttt{ins}$$

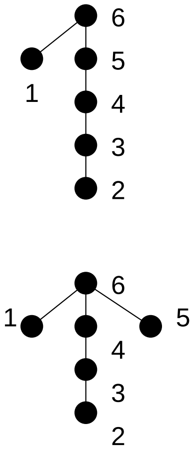
- **S**ubstitution arcs :

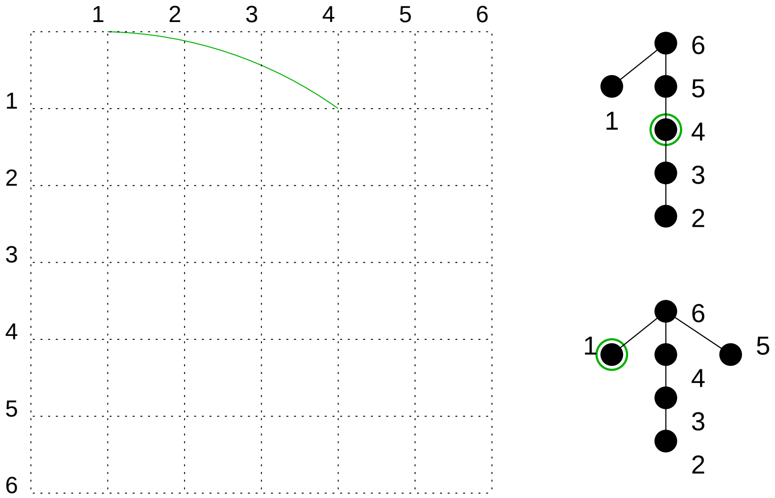
  $$(x, y) \rightsquigarrow (x - \texttt{size}(x), y - \texttt{size}(y))$$

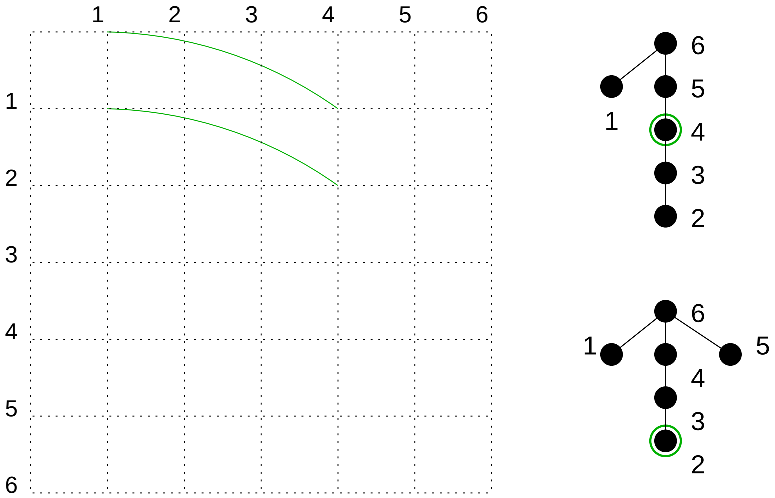
  labeled by the distance between $A(x)$ and $B(y)$

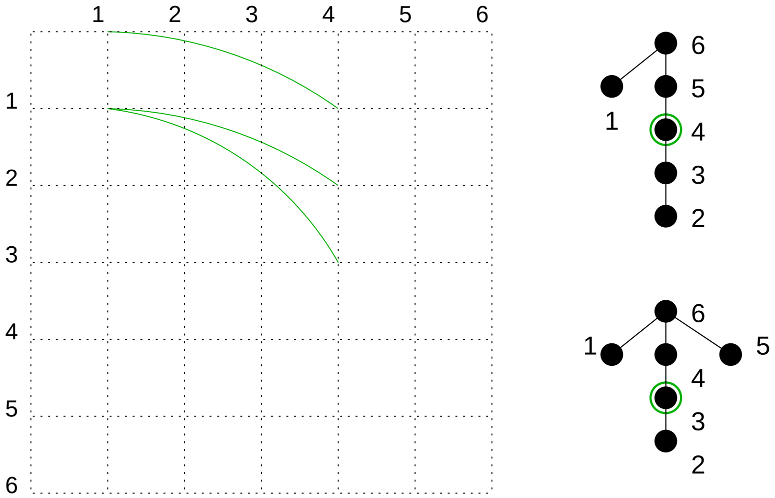
- **S**ize of the graph :$O(mn)$


1 2 3 4 5 6
1
2
3
4
5
6

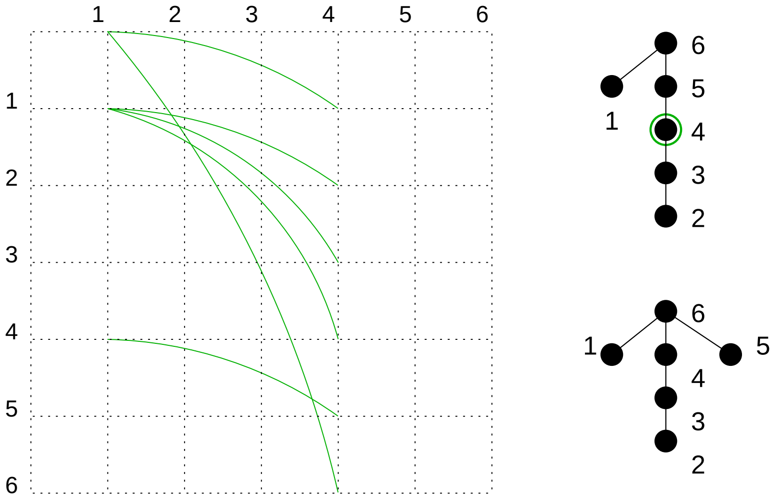

6
5
1
4
3
2
6
1
5
4
3
2

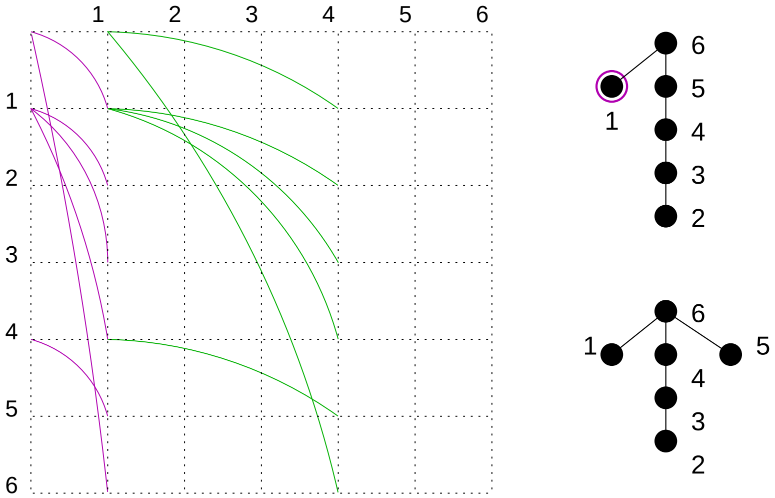

1
2
3
4
5
6
1
2
3
4
5
6
6
5
1
4
3
2
6
1
5
4
3
2

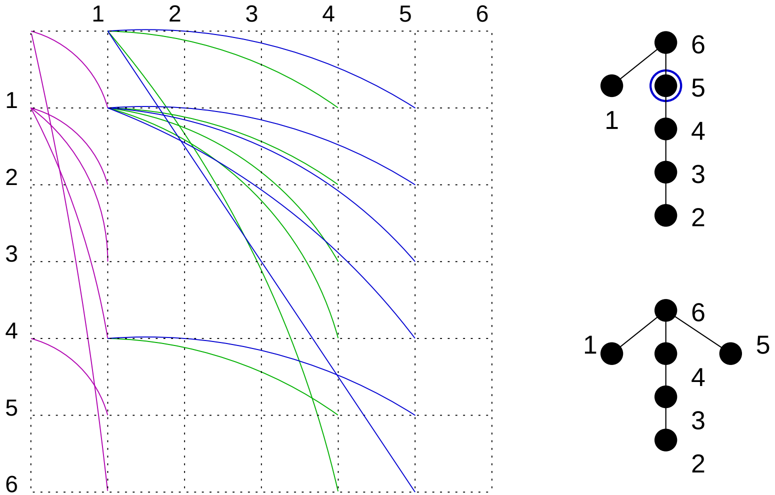

1 2 3 4 5 6
1 2 3 4 5 6
6 5 4 3 2 1
6 1 5 4 3 2



1
2
3
4
5
6
1
2
3
4
5
6
6
5
1
4
3
2
6
1
5
4
3
2


1 2 3 4 5 6
1 2 3 4 5 6
6 5 4 3 2 1
6 1 5 4 3 2


1
2
3
4
5
6
1
2
3
4
5
6
6
5
1
4
3
2
6
1
5
4
3
2



and so on . . .

Usage of the tree edit graph

How to compute the valuations of the arcs ?

- The label of the substitution arc starting from (x, y) is the weight of an optimal path in the subgraph delimited by $A(x) \times B(y)$

Time complexity : $O(n^4)$

Space complexity : $O(n^2)$

How to recover the mapping from the tree edit graph ?

Multi-level tracing back :

- Construction of an optimal path for $A \times B$
- Iteration for subgraphs induced by matching pairs of nodes

Time complexity : $O(n^3)$

Space complexity : $O(n^2)$

- **O**ptimal paths for $td(x, y)$

  $h = x - \mathtt{size}(x),\ l = y - \mathtt{size}(y)$

$$
\begin{array}{rcl}
fd(h, l, h, l) & = & 0 \\
fd(i, l, h, l) & = & fd(i-1, l, h, l) + \mathtt{del} \\
fd(h, j, h, l) & = & fd(h, j-1, h, l) + \mathtt{ins} \\
fd(i, j, h, l) & = & \min \left\{ \begin{array}{l} fd(i-1, j, h, l) + \mathtt{del} \\ fd(i, j-1, h, l) + \mathtt{ins} \\ fd(i - \mathtt{size}(i), j - \mathtt{size}(j), h, l) + td(i, j) \end{array} \right.
\end{array}
$$

- **F**or the subtrees

  if $fd(x-1, y-1, h, l) + \mathtt{sub}(x, y) <$
  $\min\{fd(x-1, y, h, l) + \mathtt{del}, fd(x, y-1, h, l) + \mathtt{ins}\}$
  then $td(x, y) \leftarrow fd(x-1, y-1, h, l) + \mathtt{sub}(x, y)$
  else $td(x, y) \leftarrow +\infty$

- **T**his is Zhang&Shasha algorithm
- **K**lein and Dulucq&Touzet algorithms build the same edit graph, but they use alternative strategies to compute the valuations of the arcs.

Edit distance with k errors

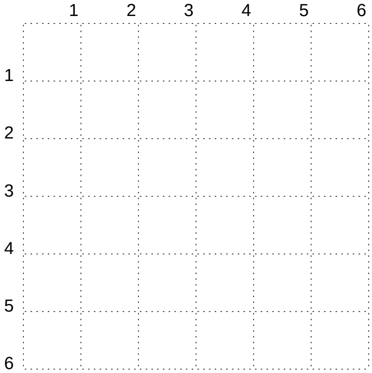
- **E**rror : insertion of a node, deletion of a node
- **P**roblem :

Input	:	two ordered trees, a natural number k
Output	:	the best mapping containing less than k errors, (if it exists)

- **M**ethod : pruning the tree edit graph

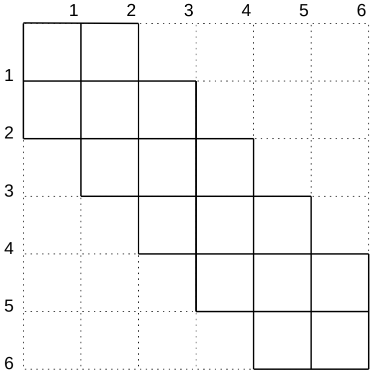
Edit distance with k errors

Idea 1 : the best mappings have their path near the main diagonal



Edit distance with k errors

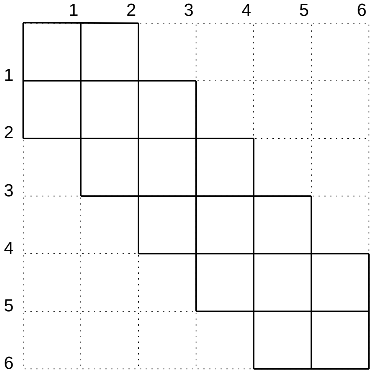
Idea 1 : the best mappings have their path near the main diagonal



k-strip$=\{(x, y);\ \ |x-y| \leq k\}$

Edit distance with k errors

Idea 1 : the best mappings have their path near the main diagonal



k-strip$=\{(x,y);\ \ |x-y|\leq k\}$

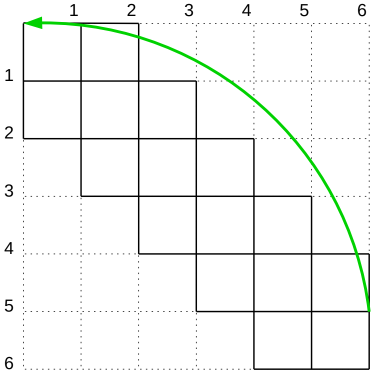
Size of the graph :$O(nk)$

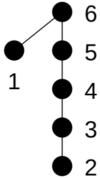
Computation time for each node: $O(\texttt{size}(A,x)k)$

$$\boxed{O(k^2\sum\texttt{size}(A,x))}$$

Edit distance with k errors

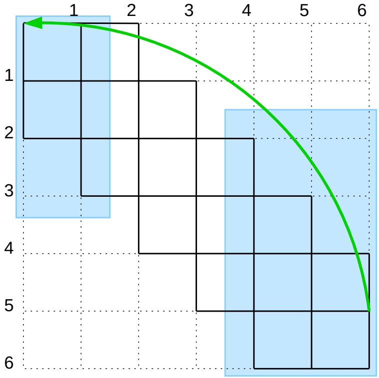
Idea 2 : when inspecting the subtree rooted at x, there is no need to visit the nodes of depth $> k+1$

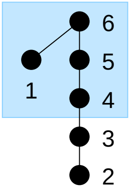




Edit distance with k errors

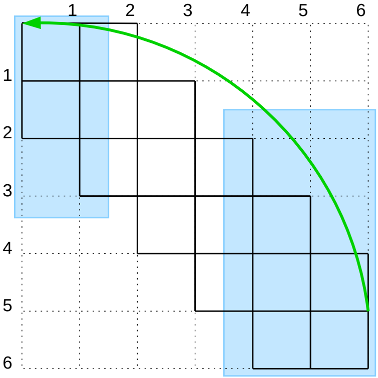
Idea 2 : when inspecting the subtree rooted at x, there is no need to visit the nodes of depth $> k + 1$





Edit distance with k errors

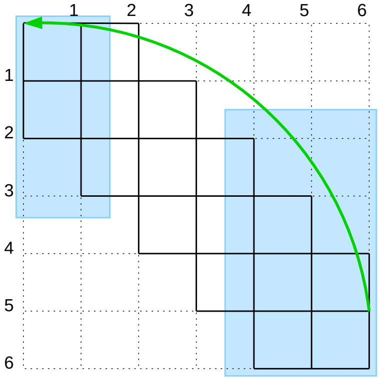
Idea 2 : when inspecting the subtree rooted at x, there is no need to visit the nodes of depth $> k+1$



$$A(x,k) = \{i \in A(x);$$
$$\texttt{depth}(i) - \texttt{depth}(x) \leq k+1\}$$

Edit distance with k errors

Idea 2 : when inspecting the subtree rooted at x, there is no need to visit the nodes of depth $> k+1$



$A(x,k) = \{i \in A(x);$
$\texttt{depth}(i) - \texttt{depth}(x) \leq k+1\}$

Size of the graph: $O(nk)$

Computation time for each node: $O(\texttt{size}(A,x,k)k)$

$$\boxed{O(k^2 \sum \texttt{size}(A,x,k)) = O(k^3 n)}$$

- **T**ree edit graph for k errors : $O(k^3 n)$

  Input: two trees A and B, positive integer k
  Output: tree edit graph

  for $(x, y) \in k\text{-}strip(A, B)$ **do** $O(k^2 \sum \mathtt{size}(A, x, k)) = O(k^3 n)$
      if not k-relevant(x, y)
          then $td(x, y) \leftarrow +\infty$
          else **for** $i \in A(x, k)$ **do** $O(k\mathtt{size}(A, x, k))$
                  for $j \in B$ such that $(i, j) \in k\text{-}strip(A, B)$ **do** $O(k)$
                      compute $fd(i, j)$ $O(1)$
                      end do
                  end do
              compute $td(x, y)$ $O(1)$
      end if
  end do

- **R**ecovering the optimal mapping : $O(k^3 n)$