Algorithms for Forest Pattern Matching

Kaizhong Zhang and Yunkun Zhu
Department of Computer Science
The University of Western Ontario, Canada

Outline

Introduction

Forest Edit Distance

Finding A Most Similar Closed Subforest

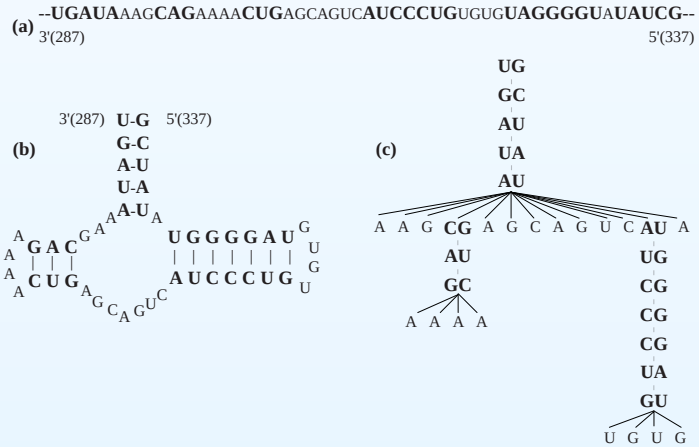
Finding A Most Similar Closed Substructure

Conclusion

Introduction

Motivation

- Trees and forests can represent many phenomena.
  - grammar parsing, image descriptions, structured texts, etc
  - RNA secondary structures

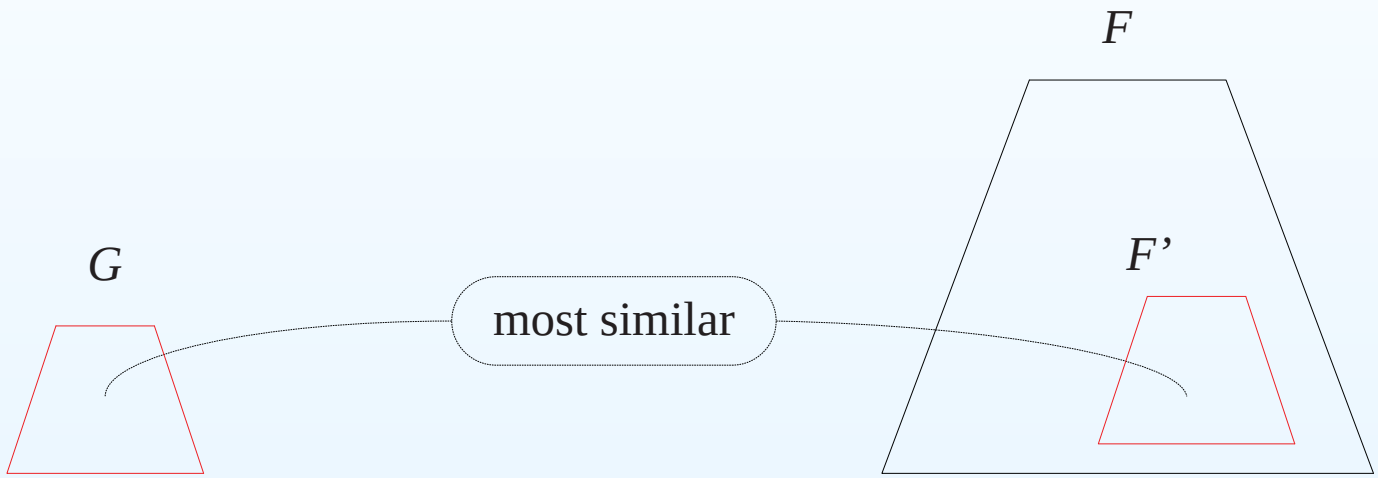


(a) a segment of the RNA GI: 2347024 primary structure
(b) its secondary structure (c) its forest representation

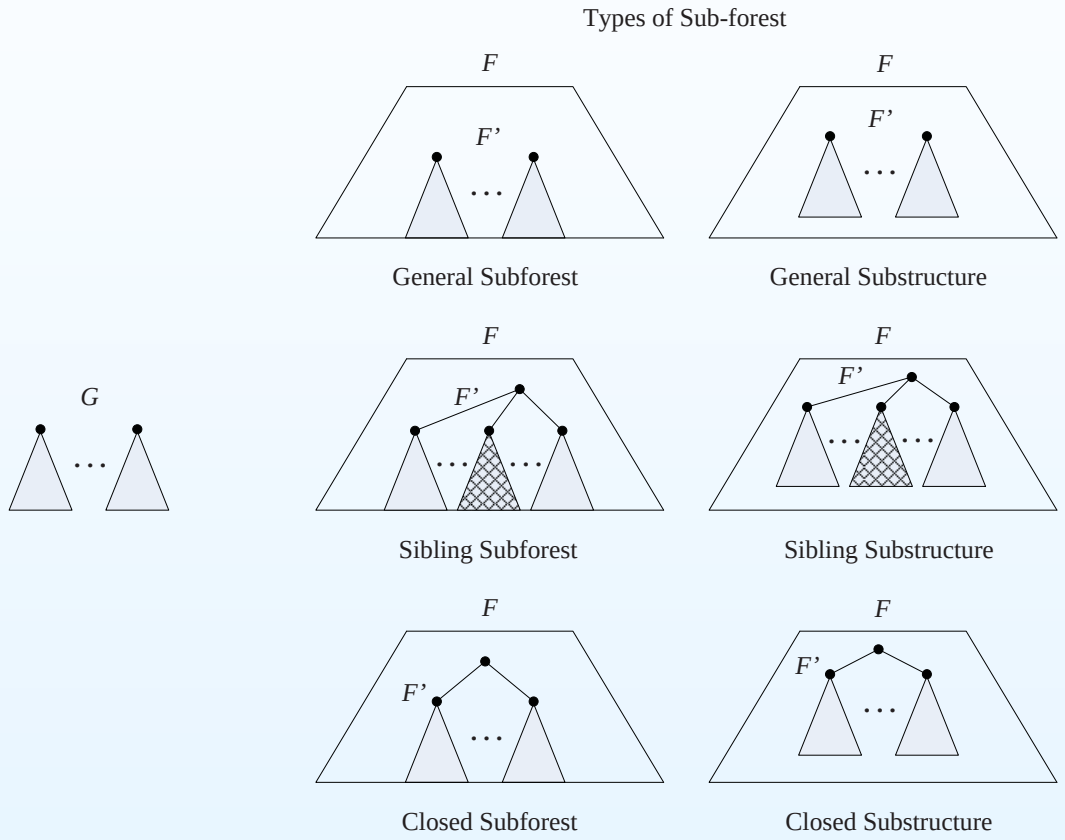
Locating structural or functional regions in RNA secondary structures makes the Forest Pattern Matching (FPM) problem become interesting and attract some attention.

Objective

Forest Pattern Matching: Given a target forest F and a pattern forest G, find a sub-forest F' of F which is the most similar to G over all possible F'.



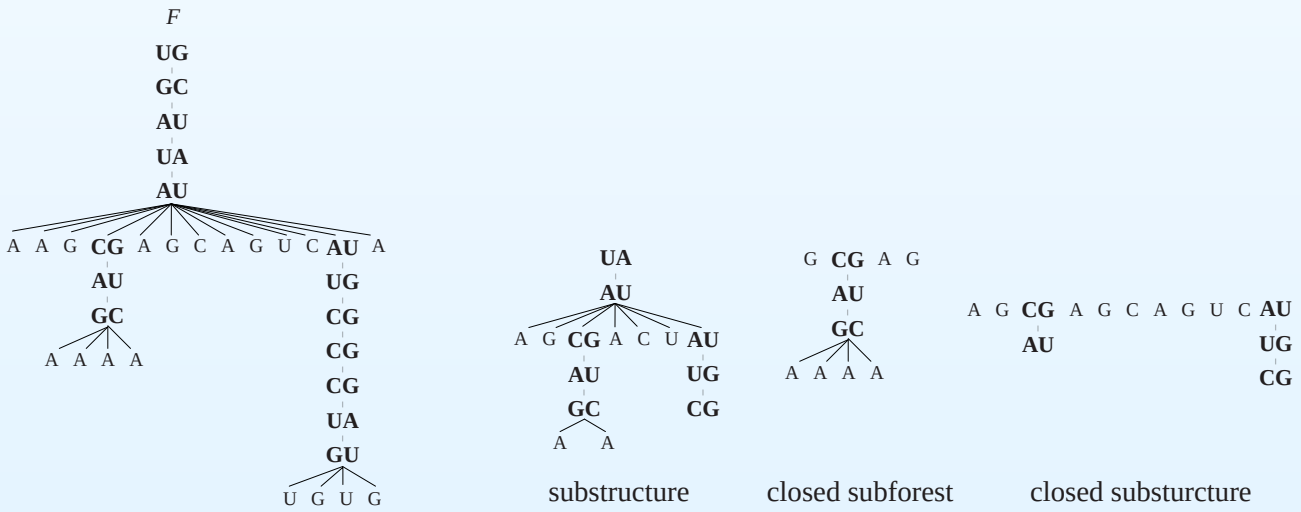
Types of Sub-forest



Types of Sub-forest in Our Paper

Closed Subforest: a sequence of subtrees of F such that their roots are consecutive siblings

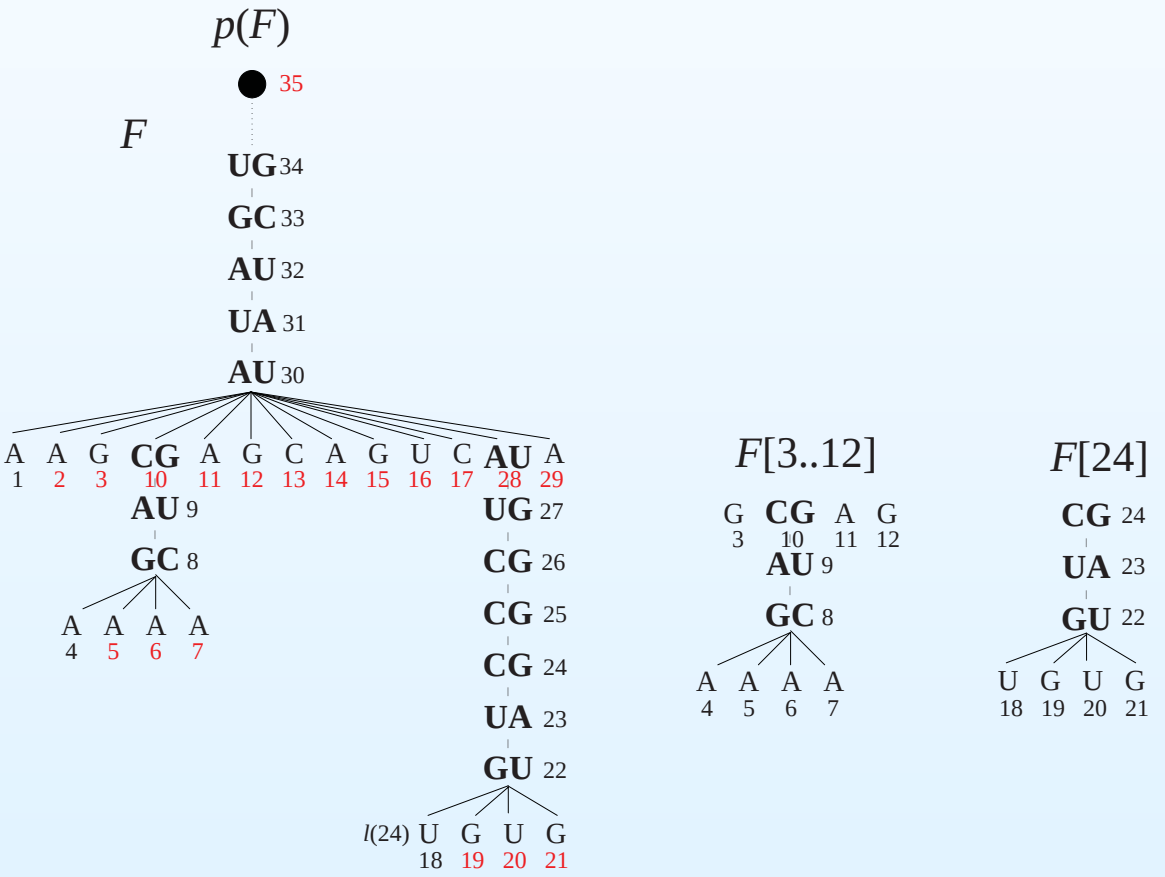
Closed Substructure: a sequence of substructures of F such that their roots are consecutive siblings



substructure closed subforest closed substurcture

Definitions and Notations

- left-to-right postorder numbering, $F[i..j]$, $F[i]$, $l(i)$, D_F, L_F, $p(F)$
- $K(F) = \{p(F)\} \cup \{i \mid i \in F \text{ and } i \text{ has a left sibling}\}$



Previous Work and Our Results

- Closed Subforest:

	Jansson and Peng (CPM 2006)
Time	$O(\|F\| \cdot \|G\| \cdot L_F \cdot \min\{D_G, L_G\})$
Space	$O(\|F\| \cdot \|G\| \cdot \min\{D_F, L_F\} \cdot \min\{D_G, L_G\})$

	Us
Time	$O(\|F\| \cdot \|G\| \cdot \min\{D_F, L_F\} \cdot \min\{D_G, L_G\})$ $O(\|F\| \cdot \|G\| \cdot (\|G\| \cdot (1 + \log \frac{\|F\|}{\|G\|}) + \min\{D_F, L_F\}))$
Space	$O(\|F\| \cdot \|G\|)$

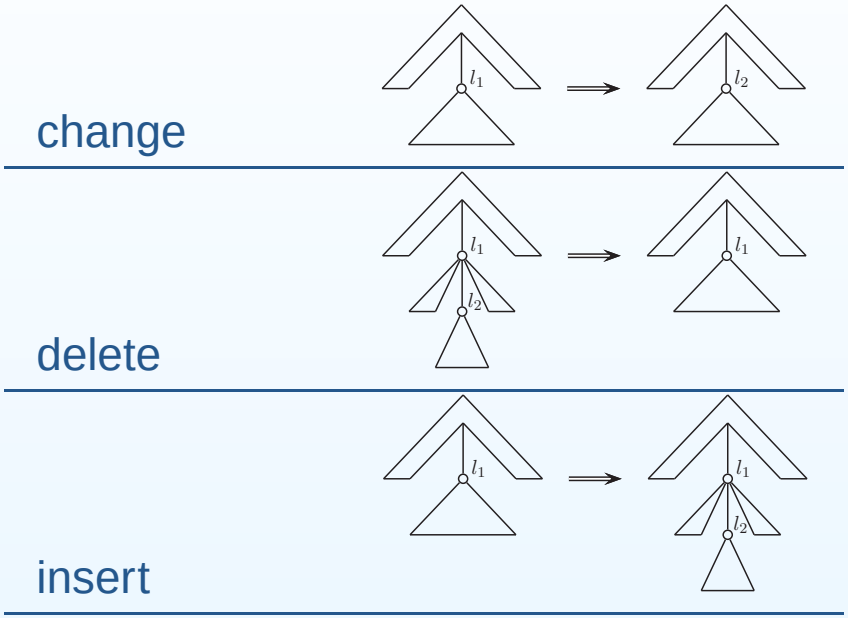
Previous Work and Our Results

- Closed Substructure:

	Us
Time	$\frac{O(\lvert F\rvert \cdot \lvert G\rvert \cdot \min\{D_F, L_F\} \cdot \min\{D_G, L_G\})}{O(\lvert F\rvert \cdot \lvert G\rvert \cdot (\lvert G\rvert \cdot (1 + \log \frac{\lvert F\rvert}{\lvert G\rvert}) + \min\{D_F, L_F\}))}$
Space	$O(\lvert F\rvert \cdot \lvert G\rvert)$

Forest Edit Distance

Edit Operations



distance function γ: (1) $\gamma(a,b) \geq 0$; (2) $\gamma(a,b) = \gamma(b,a)$;
(3) $\gamma(a,c) \leq \gamma(a,b) + \gamma(b,c)$; (4) $\gamma(a,b) = 0$ if and only if $a = b$

forest edit distance: minimum score of transforming F_1 into F_2 via a sequence of edit operations

$$\delta(F_1, F_2) = \min\{\gamma(S) \mid S \text{ is an edit operation sequence taking } F_1 \text{ to } F_2\}$$

Landmarks

1979: Tai

- introduced the forest (tree) edit distance
- gave the first algorithm

1989: Zhang and Shasha

- new approach

1998: Klein

- new recursion style of the Zhang-Shasha algorithm

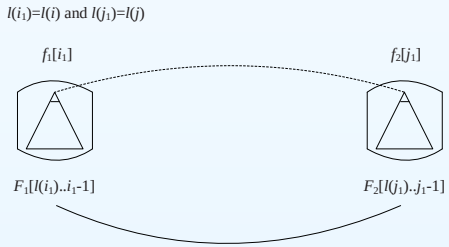
2006: Demaine, Mozes, Rosman and Weimann (Demaine *et al.*)

- subtle modification of the Klein algorithm

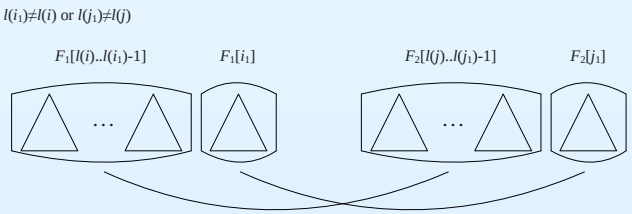
The Computation of the Forest Edit Distance

- consider the edit distance between $F_1[i]$ and $F_2[j]$

$$\delta(F_1[l(i)..i_1], F_2[l(j)..j_1]) = \min \begin{cases} \delta(F_1[l(i)..i_1-1], F_2[l(j)..j_1]) + \gamma(f_1[i_1], -), \\ \delta(F_1[l(i)..i_1], F_2[l(j)..j_1-1]) + \gamma(-, f_2[j_1]), \\ \delta(F_1[l(i)..i_1-1], F_2[l(j)..j_1-1]) \\ \quad + \gamma(f_1[i_1], f_2[j_1]). \end{cases}$$

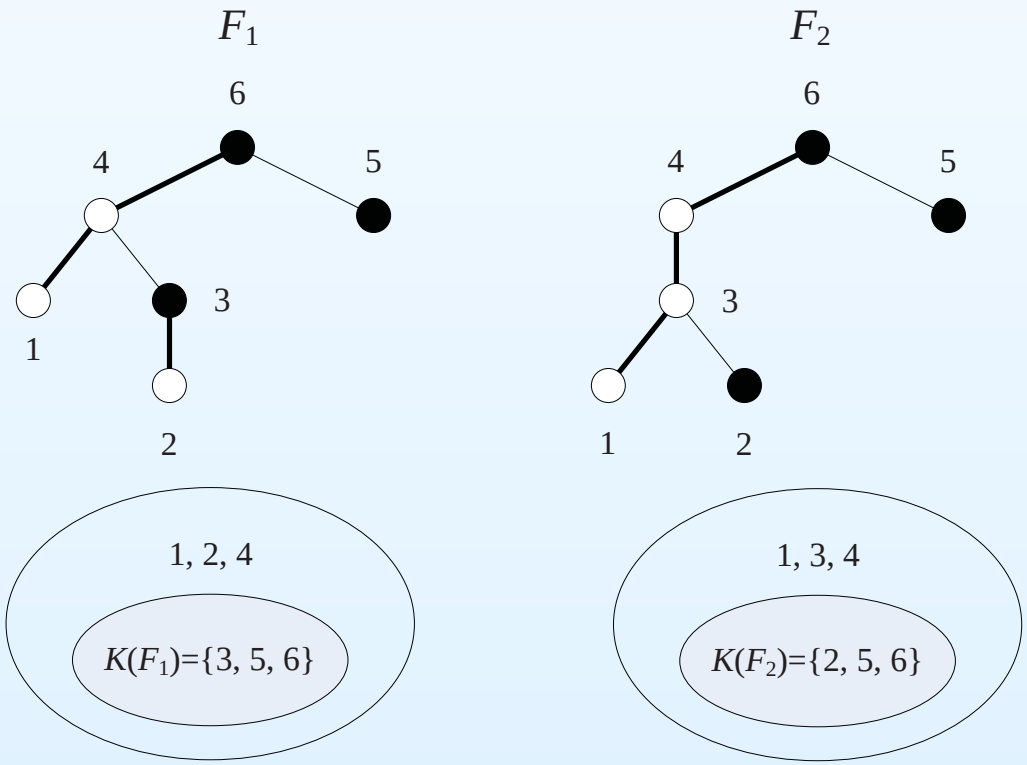


$$\delta(F_1[l(i)..i_1], F_2[l(j)..j_1]) = \min \begin{cases} \delta(F_1[l(i)..i_1-1], F_2[l(j)..j_1]) + \gamma(f_1[i_1], -), \\ \delta(F_1[l(i)..i_1], F_2[l(j)..j_1-1]) + \gamma(-, f_2[j_1]), \\ \delta(F_1[l(i)..l(i_1)-1], F_2[l(j)..l(j_1)-1]) \\ \quad + \delta(F_1[i_1], F_2[j_1]). \end{cases}$$



Basic Idea of the Zhang-Shasha Algorithm

- Key root: a subset of all subproblems
  - All tree-to-tree distances can be obtained by computing only this subset.
  - All the other tree-to-tree distances can be obtained as a byproduct.



Finding A Most Similar Closed Subforest

Goal

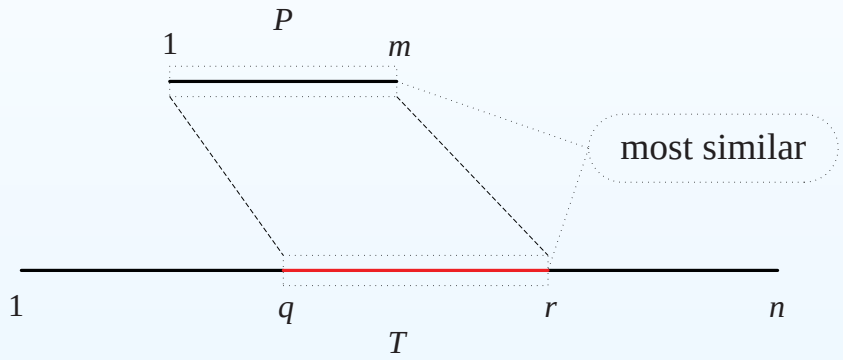
- Given a target forest F and a pattern forest G, find a closed subforest F' of F which minimizes the forest edit distance to G over all possible F'.

$$\min\{\delta(F[l(i_1)..i_2], G) \mid i_1 \text{ and } i_2 \text{ are siblings}\}$$

- $\Delta(,)$ denotes the edit distance for the problem of finding a most similar closed subforest.

Sequence Approximate Pattern Matching (SAPM)

- basis of the Forest Pattern Matching problem



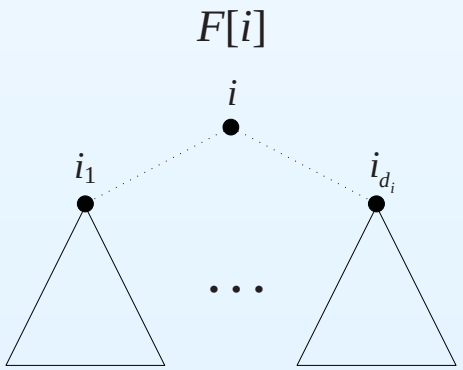
- In calculating the score for $T[1..i]$ and $P[1..j]$, any prefix of $T[1..i]$ could be deleted without any penalty.

$$score(T[1..i], P[1..j]) = \min\{\delta(T[i_1..i], P[1..j]) \mid 1 \leq i_1 \leq i+1\}$$

- dynamic programming

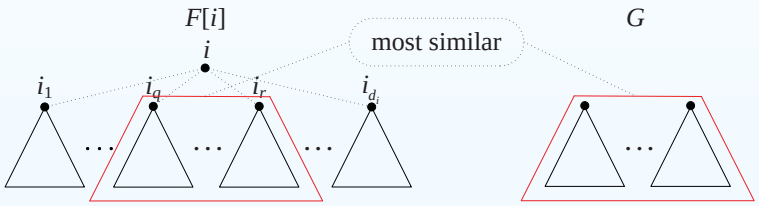
Finding A Most Similar Closed Subforest under One Node

- The Forest Pattern Matching problem is more difficult than the SAPM problem. We first consider finding a most similar closed subforest under one node i of the target forest F to the pattern forest G, then extend this technique to every node i of F to solve the Forest Pattern Matching problem.
- degree of i: d_i
- children of i: $i_1, i_2, \ldots, i_{d_i}$

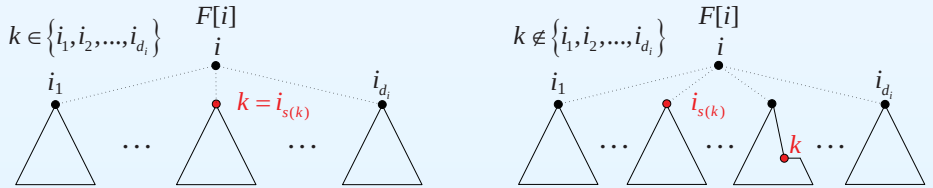


Finding A Most Similar Closed Subforest under One Node

- find q and r such that the closed subforest $F[l(i_q)..i_r]$ of $F[i]$ yields the minimum distance to G



- Let k be a node which satisfies $i_{s-1} < k \leq i_s$, we define $s(k)$: $s(k) = s$ if $k = i_s$; $s(k) = s - 1$ if $k < i_s$.

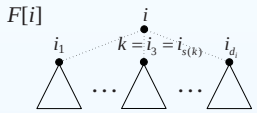


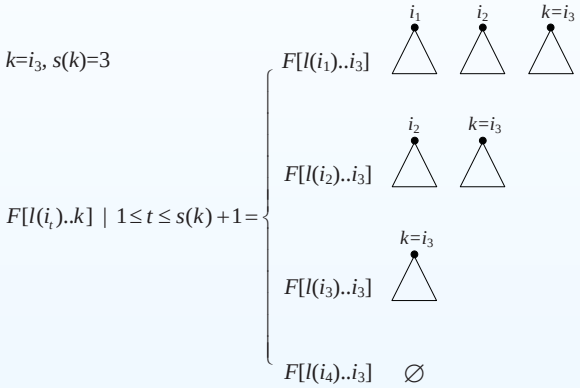
- We can extend the definition from sequences to forests:

$$\Delta(F[l(i_1)..k], G[1..j]) = \min\{\delta(F[l(i_t)..k], G[1..j]) \mid 1 \leq t \leq s(k) + 1\}$$

Different Cases of k

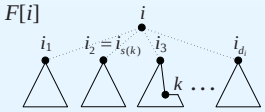
(1) $k \in \{i_1, i_2, \ldots, i_{d_i}\}$ $(k = i_{s(k)})$

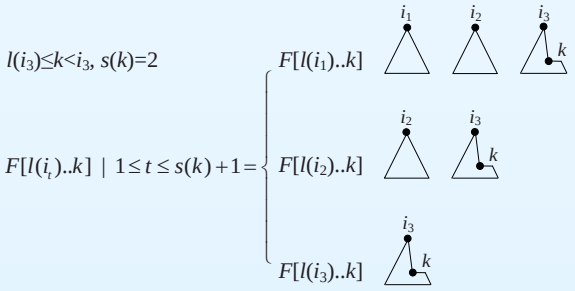




$$\Delta(F[l(i_1)..k], G[1..j]) = \min\{\delta(F[l(i_t)..i_{s(k)}], G[1..j]) \mid 1 \le t \le s(k)+1\}$$

(2) $k \notin \{i_1, i_2, \ldots, i_{d_i}\}$





$$\Delta(F[l(i_1)..k], G[1..j]) = \min\{\delta(F[l(i_t)..k], G[1..j]) \mid 1 \le t \le s(k)+1\}$$

The Computation of $\Delta(F[l(i_1)..k], G[1..j])$

(1) $i_1 \leq k \leq i_{d_i}$ and $1 \leq j \leq |G|$

$$\Delta(F[l(i_1)..k], \emptyset) = \begin{cases} 0 & \text{if } k \in \{i_1, i_2, \ldots, i_{d_i}\} \\ \Delta(F[l(i_1)..k-1], \emptyset) + \gamma(f[k], -). & otherwise \end{cases}$$

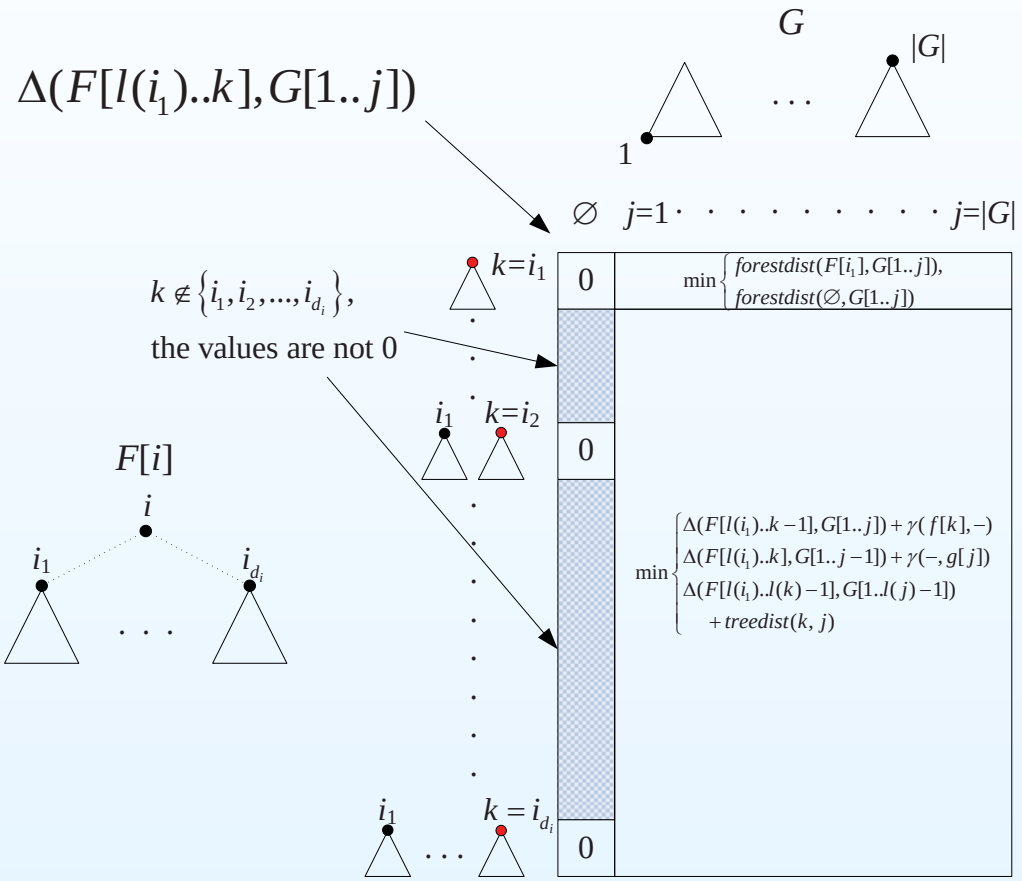
(2) $k = i_1$ and $1 \leq j \leq |G|$

$$\Delta(F[l(i_1)..k], G[1..j]) = \min \begin{cases} forestdist(F[i_1], G[1..j]), \\ forestdist(\emptyset, G[1..j]). \end{cases}$$

(3) $i_1 < k \leq i_{d_i}$ and $1 \leq j \leq |G|$

$$\Delta(F[l(i_1)..k], G[1..j]) = \min \begin{cases} \Delta(F[l(i_1)..k-1], G[1..j]) + \gamma(f[k], -), \\ \Delta(F[l(i_1)..k], G[1..j-1]) + \gamma(-, g[j]), \\ \Delta(F[l(i_1)..l(k)-1], G[1..l(j)-1]) \\ \quad + treedist(k, j). \end{cases}$$

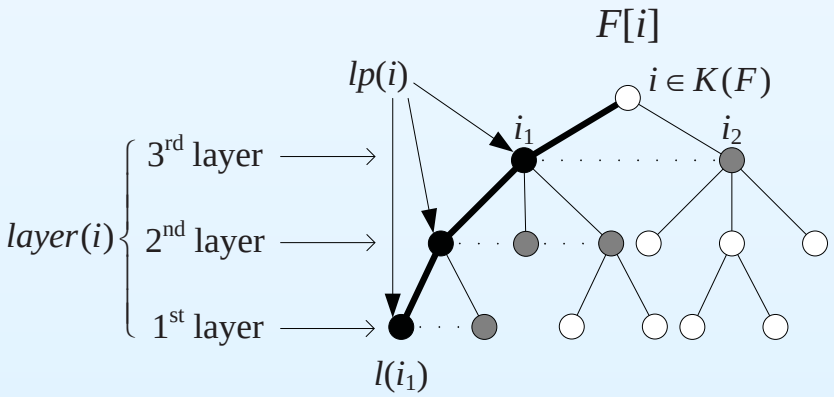
Finding A Most Similar Closed Subforest under One Node



- We can calculate $\min\{\Delta(F[l(i_1)..i_t], G[1..|G|]) \mid 1 \leq t \leq d_i\}$ using dynamic programming.

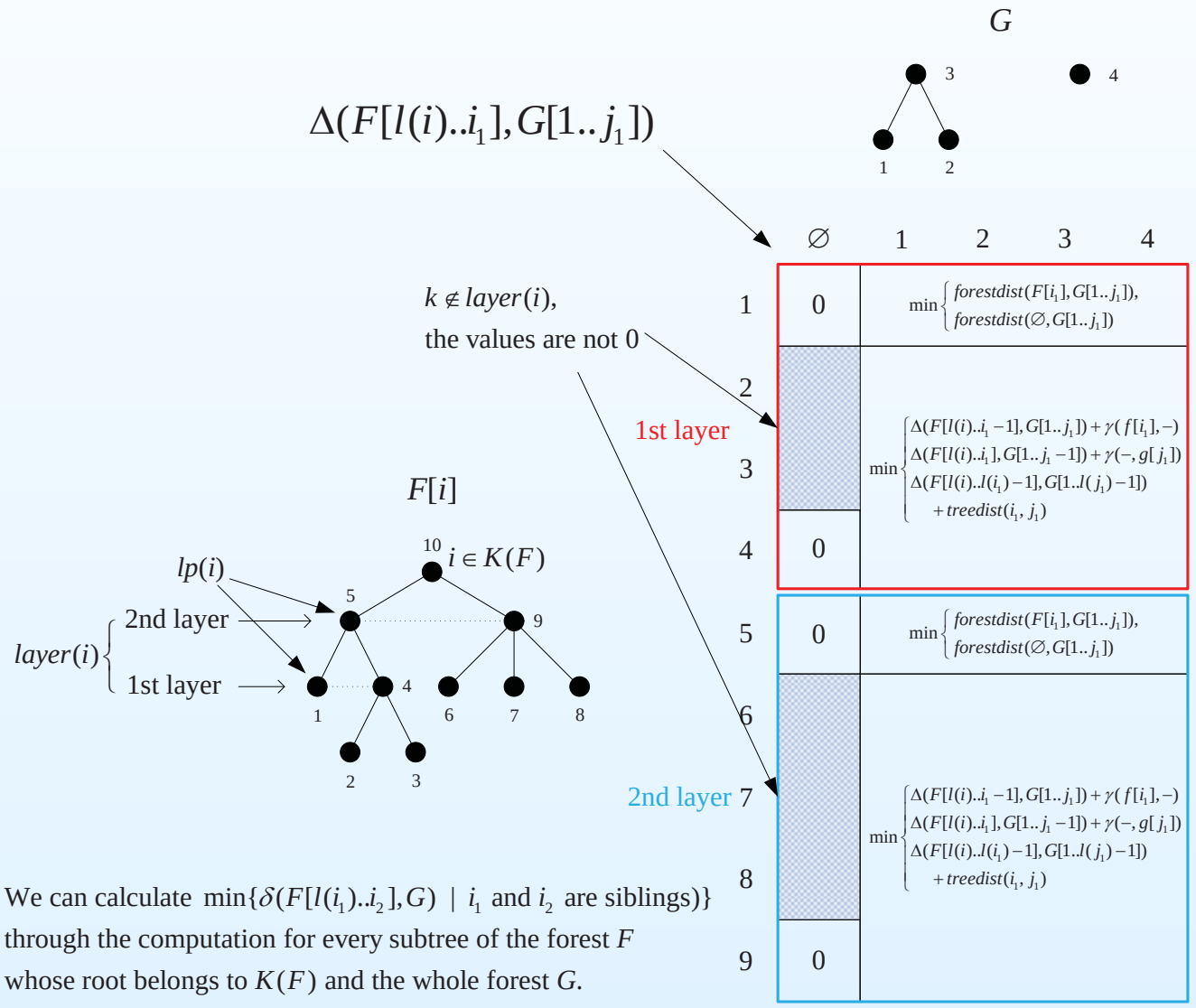
Concepts of $lp(i)$ and $layer(i)$

- We have to do the calculation for every node of F.
- Because for finding a most similar closed subforest under node i of F the calculation starts at i_1 instead of $l(i_1)$, we do the calculations for all the nodes on the path from a leaf to its nearest ancestor key root together.
- We do the computation layer by layer.
  $lp(i)$: a set which contains the nodes on the leftmost path of $F[i]$ except the root i
  $layer(i)$: a set which contains all of the sibling nodes of nodes in $lp(i)$ including $lp(i)$



Finding A Most Similar Closed Subforest of A Key Root Subtree

- $Delta(F[i], |G|)$



We can calculate $\min\{\delta(F[l(i_1)..i_2], G) \mid i_1$ and i_2 are siblings$\}$ through the computation for every subtree of the forest F whose root belongs to $K(F)$ and the whole forest G.

Algorithm

Algorithm 1: $Closed - Subforest(F, G)$

Input: A target forest F and a pattern forest G.
Output: $\min\{\Delta(F[l(x_1)..x_2], G) \mid x_1$ is x_2's leftmost sibling$\}$.

```
1 TreeDistance(F, G) —- Zhang-Shasha or Demaine et al. (treedist(,))
2 for i' := 1 to |K_F| do
3 |   i := K_F[i']
4 |   ForestDistance(F[i], |G|) —- Zhang-Shasha (forestdist(,))
5 |   Delta(F[i], |G|)
6 end
```

- The time complexity of our algorithm:
  $O(|F| \cdot |G| \cdot \min\{D_F, L_F\} \cdot \min\{D_G, L_G\})$
  - Stage one
    $TreeDistance(F, G)$: $O(|F| \cdot |G| \cdot \min\{D_F, L_F\} \cdot \min\{D_G, L_G\})$ or $O(|F| \cdot |G|^2 \cdot (1 + \log \frac{|F|}{|G|}))$
  - Stage two
    Both $ForestDistance(F[i], |G|)$ and $Delta(F[i], |G|)$:
    $O(|F[i]| \cdot |G|)$
    The total time for all the key roots of F: $O(|F| \cdot |G| \cdot \min\{D_F, L_F\})$
- The space complexity of our algorithm: $O(|F| \cdot |G|)$

Finding A Most Similar Closed Substructure

Goal

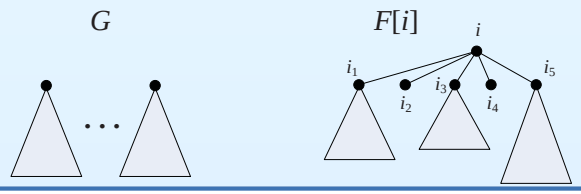
- Given a target forest F and a pattern forest G, find a closed substructure F' of F which minimizes the forest edit distance to G over all possible F'.
- Forest removing distance (Zhang-Shasha):

$$\delta_r(F, G) = \min_{f \in subf(F)} \{\delta(F \setminus f, G)\}$$

- Another forest removing distance:

$$\delta_R(F[l(i_1)..k], G[1..j]) = \min_{f \in subf(F[l(i_1)..k], \{i_1, \ldots i_{d_i}\})} \delta(F[l(i_1)..k] \setminus f, G[1..j])$$

$subf(F, node\_set)$: the set of subforests of F such that nodes in $node\_set$ are not in any of the subforests; k: $i_1 \leq k < i$;
d_i: degree of i; $i_1, i_2, \ldots, i_{d_i}$: children of i



Finding A Most Similar Closed Substructure under One Node

- From the definition of $\delta_R(,)$ and the Zhang-Shasha algorithm, we have the following formula for $\delta_R(F[l(i_t)..k], G[1..j])$, where $1 \leq t \leq s$.

$$\delta_R(F[l(i_t)..k], G[1..j]) =$$
$$\min \begin{cases} \delta_R(F[l(i_t)..l(k)-1], G[1..j]), & \text{if } k \notin \{i_1, i_2, \ldots, i_{d_i}\} \\ \delta_R(F[l(i_t)..k-1], G[1..j]) + \gamma(f[k], -), \\ \delta_R(F[l(i_t)..k], G[1..j-1]) + \gamma(-, g[j]), \\ \delta_R(F[l(i_t)..l(k)-1], G[1..l(j)-1]) + \gamma(f[k], g[j]) \\ \quad + \delta_r(F[l(k)..k-1], G[l(j)..j-1]). \end{cases}$$

- $\Psi(,)$ denotes the edit distance for the problem of finding a most similar closed substructure.

$$\Psi(F[l(i_1)..k], G[1..j]) = \min\{\delta_R(F[l(i_t)..k], G[1..j]) \mid 1 \leq t \leq s(k)+1\}$$

The Computation of $\Psi(F[l(i_1)..k], G[1..j])$

(1) $i_1 \leq k \leq i_{d_i}$ and $1 \leq j \leq |G|$

$$\Psi(F[l(i_1)..k], \emptyset) = 0.$$

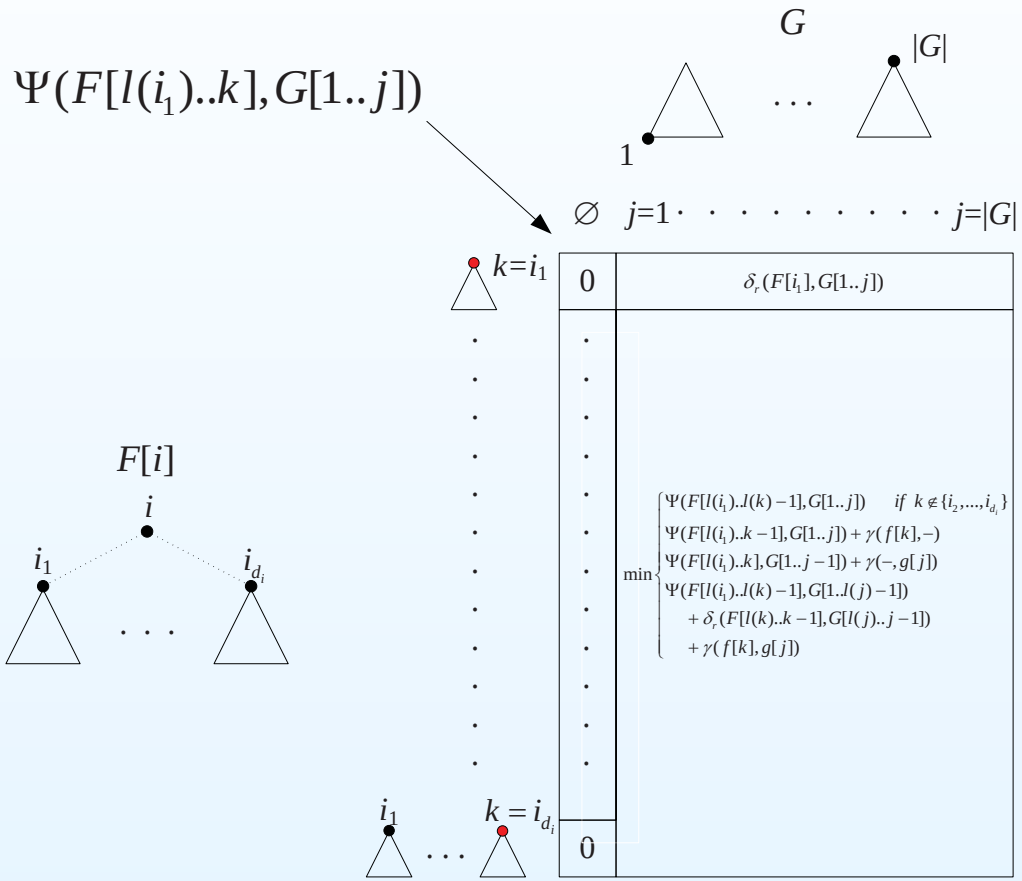
(2) $k = i_1$ and $1 \leq j \leq |G|$

$$\Psi(F[l(i_1)..k], G[1..j]) = \min \left\{ \begin{array}{l} \delta_R(F[i_1], G[1..j]), \\ \delta_R(\emptyset, G[1..j]) \end{array} \right. = \delta_r(F[i_1], G[1..j]).$$

(3) $i_1 < k \leq i_{d_i}$ and $1 \leq j \leq |G|$

$$\Psi(F[l(i_1)..k], G[1..j]) =$$
$$\min \left\{ \begin{array}{l} \Psi(F[l(i_t)..l(k)-1], G[1..j]), \quad \text{if } k \notin \{i_2, \ldots, i_{d_i}\} \\ \Psi(F[l(i_1)..k-1], G[1..j]) + \gamma(f[k], -), \\ \Psi(F[l(i_1)..k], G[1..j-1]) + \gamma(-, g[j]), \\ \Psi(F[l(i_1)..l(k)-1], G[1..l(j)-1]) + \gamma(f[k], g[j]) \\ \quad + \delta_r(F[l(k)..k-1], G[l(j)..j-1]). \end{array} \right.$$

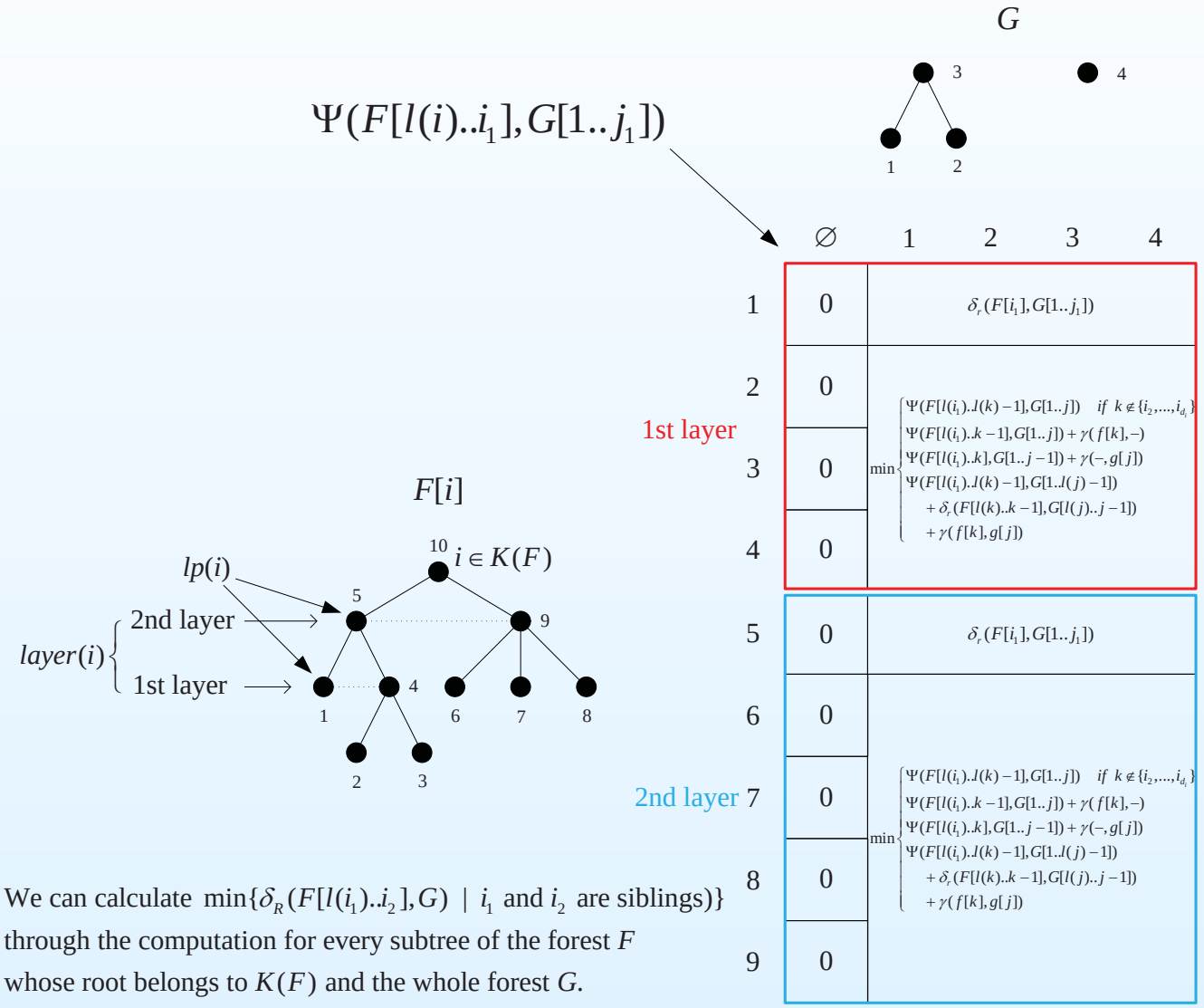
Finding A Most Similar Closed Substructure under One Node



- We can calculate $\min\{\Psi(F[l(i_1)..i_t], G[1..|G|]) \mid 1 \leq t \leq d_i\}$ using dynamic programming.

Finding A Most Similar Closed Substructure of A Key Root Subtree

- $Psi(F[i], |G|)$



We can calculate $\min\{\delta_R(F[l(i_1)..i_2], G) \mid i_1 \text{ and } i_2 \text{ are siblings}\}$ through the computation for every subtree of the forest F whose root belongs to $K(F)$ and the whole forest G.

Algorithm

Algorithm 2: $Closed - Substructure(F, G)$

Input: A target forest F and a pattern forest G.
Output: $\min\{\Psi(F[l(x_1)..x_2], G) \mid x_1$ is x_2's leftmost sibling$\}$.

```
1 Tree_RemoveDistance(F, G) —- Zhang-Shasha
2 for i' := 1 to |K_F| do
3 |   i := K_F[i']
4 |   Forest_RemoveDistance(F[i], |G|) —- Zhang-Shasha
5 |   Psi(F[i], |G|)
6 end
```

- The time complexity of our algorithm:
  $O(|F| \cdot |G| \cdot \min\{D_F, L_F\} \cdot \min\{D_G, L_G\})$
  - Stage one
    $Tree\_RemoveDistance(F, G)$:
    $O(|F| \cdot |G| \cdot \min\{D_F, L_F\} \cdot \min\{D_G, L_G\})$
  - Stage two
    Both $Forest\_RemoveDistance(F[i], |G|)$ and $Psi(F[i], |G|)$:
    $O(|F[i]| \cdot |G|)$
    The total time for all the key roots of F: $O(|F| \cdot |G| \cdot \min\{D_F, L_F\})$
- The space complexity of our algorithm: $O(|F| \cdot |G|)$

Conclusion

Conclusion

Our Contributions:

- Search for local forest patterns
- FPM-Closed Subforest: improve the time and space complexities
- FPM-Closed Substructure: reduces to sequence approximate pattern matching algorithm when the input are two linear trees

Thank you