

Approximation: Theory and Algorithms

Pruning with Reference Sets

Nikolaus Augsten

Free University of Bozen-Bolzano
Faculty of Computer Science
DIS

Unit 10 – May 15, 2009

Outline

- 1 Reference Sets
 - Definition
 - Upper and Lower Bound
 - Example: Reference Sets
 - Combined Approach

- 2 Conclusion

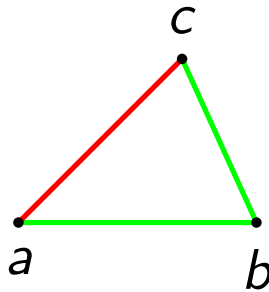
Reference Sets [GJK⁺02]

- Reference sets take advantage of the **triangle inequality** for metrics.
- An **extended version** of the triangle inequality is:

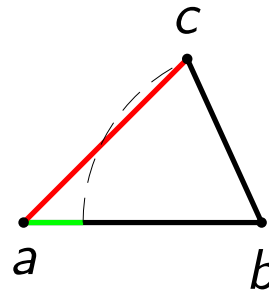
$$|\delta(a, b) - \delta(b, c)| \leq \delta(a, c) \leq \delta(a, b) + \delta(b, c),$$

where $\delta()$ is a metric distance function computed between elements a , b , and c .

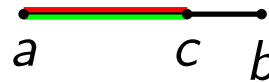
Illustration: Triangle Inequality



$$\delta(a, c) < \delta(a, b) + \delta(b, c) \quad |\delta(a, b) - \delta(b, c)| < \delta(a, c)$$



$$\delta(a, c) = \delta(a, b) + \delta(b, c) \quad |\delta(a, b) - \delta(b, c)| = \delta(a, c)$$



Notation

- S_1 and S_2 are sets of trees (unordered forests)
- $K \subseteq S_1 \cup S_2$ is called the reference set
- $T_i \in S_1$ and $T_j \in S_2$ are trees
- $k_l \in K$, $1 \leq l \leq |K|$, is a tree of the reference set

Reference Vector

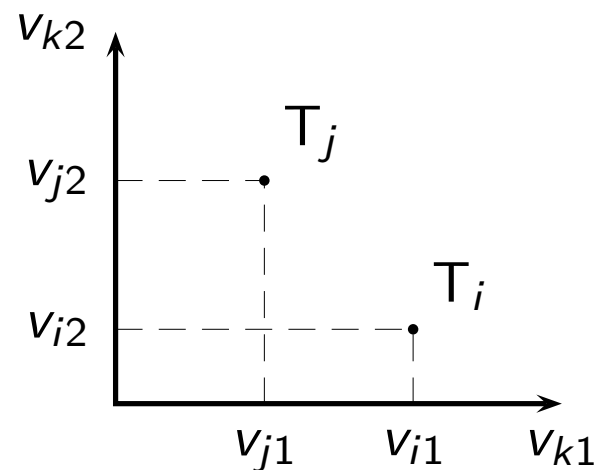
- v_i is a vector
 - of size $|K|$
 - that stores the distance from $T_i \in S_1$ to each tree $k_l \in K$
 - the l -th element of v_i stores the edit distance between T_i and k_l :

$$v_{il} = \delta_t(T_i, k_l)$$

- v_j is the respective vector for $T_j \in S_2$

Metric Space

- **Metric space** of the reference set:
 - the elements of the reference set define the basis of a metric space
 - the vector v_k represents tree T_k as a point in this space
 - v_{kl} is the coordinate of the point on the l -th axis
- **Example:** Two trees T_i and T_j in the metric space defined by the reference set $K = \{k_1, k_2\}$.



Upper and Lower Bound

- From the **triangle inequality** it follows that for all $1 \leq l \leq |K|$

$$|v_{il} - v_{jl}| \leq \delta_t(T_i, T_j) \leq v_{il} + v_{jl}$$

- **Upper bound:**

$$\delta_t(T_i, T_j) \leq \min_{l, 1 \leq l \leq |K|} v_{il} + v_{jl} = u_t(T_i, T_j)$$

- **Lower bound:**

$$\delta_t(T_i, T_j) \geq \max_{l, 1 \leq l \leq |K|} |v_{il} - v_{jl}| = l_t(T_i, T_j)$$

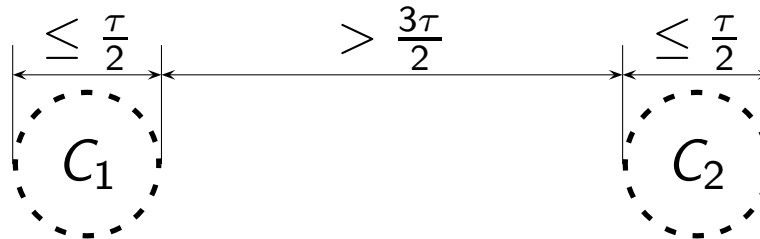
- **Approximate join:** match all trees with $\delta_t(T_1, T_2) \leq \tau$
 - **Upper bound:** If $u_t(T_i, T_j) \leq \tau$, then T_i and T_j match.
 - **Lower bound:** If $l_t(T_i, T_j) > \tau$, then T_i and T_j do *not* match.

Reference Set — Tradeoff

- **Small reference set:** efficient reference vector computation
 - we have to compute $|S|$ distances for each additional tree in the reference set to construct the vectors
 - for small reference sets the construction of the vectors is cheaper
- **Large reference set:** effective filters
 - large reference sets make u_t and l_t more effective
 - thus, once the reference vectors are computed, less distance computations are needed
- Where is the **optimum?**

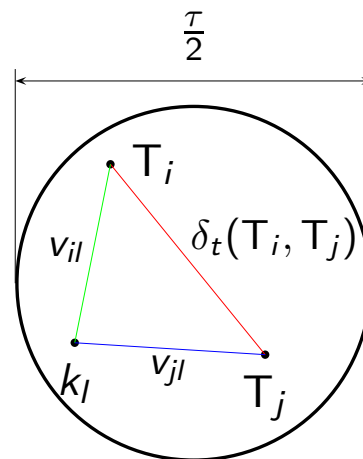
Well Separated Clusters

- We cluster the set $S = S_1 \cup S_2$.
- The clusters are **well separated** for threshold τ if
 - trees within a cluster have small distance (within $\frac{\tau}{2}$)
 - trees from different cluster have large distance (more than $\frac{3\tau}{2}$)
- **Example:** Two well separated clusters C_1 and C_2 .



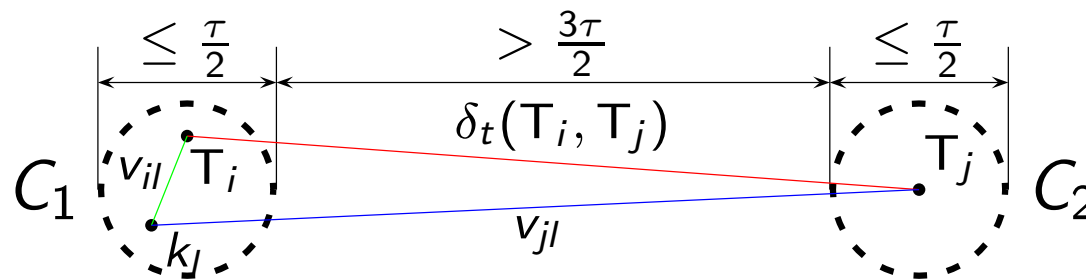
Trees within the Same Cluster — Upper Bound Applies

- Upper bound: $\delta_t(T_i, T_j) \leq v_{il} + v_{jl} = \delta_t(T_i, k_l) + \delta_t(k_l, T_j)$
- Assumption: $T_i \in S_1$ and $T_j \in S_2$ are in the same cluster C .
 - The clusters are well separated.
 - The cluster C contains a reference tree $k_l \in K$.
- In this case $v_{il} \leq \frac{\tau}{2}$ and $v_{jl} \leq \frac{\tau}{2} \Rightarrow \delta_t(T_i, T_j) \leq \tau$.
- Result: If T_i , T_j , and k_l are in the same cluster
 - from v_{il} and v_{jl} we conclude that T_1 and T_2 match
 - thus we need not to compute $\delta_t(T_i, T_j)$



Trees in Different Clusters — Lower Bound Applies

- Lower bound: $\delta_t(T_i, T_j) \geq |v_{il} - v_{jl}| = |\delta_t(T_i, k_l) - \delta_t(k_l, T_j)|$
- **Assumption:** $T_i \in S_1$ is in cluster C_1 , $T_j \in S_2$ is in cluster C_2 .
 - The clusters are well separated.
 - The cluster C_1 contains a reference tree $k_l \in K$.
- In this case $v_{il} \leq \frac{\tau}{2}$ and $v_{jl} > \frac{3\tau}{2} \Rightarrow \delta_t(T_i, T_j) > \tau$.
- **Result:** If T_i and k_l are in the same cluster, but T_j is in a different cluster
 - from v_{il} and v_{jl} we conclude that T_1 and T_2 do *not* match
 - thus we need not to compute $\delta_t(T_i, T_j)$



Bounds for Well Separated Clusters

Scenario:

- Assume that $S = S_1 = S_2$ is **well separated** into clusters.
- Consider a cluster of size n' that contains a tree $k_l \in K$ of the reference set.

Cost:

- We need to compute $|S| - 1$ distances to the tree $k_l \in K$

Benefit:

- From the **upper bound** we conclude that
 - all trees within a cluster match ($u_t \leq \tau$)
 - we save $n'(n' - 1)/2$ edit distance computations
- From the **lower bound** we conclude that
 - trees of different clusters do not match ($l_t > \tau$)
 - we save $n'(|S| - n')$ edit distance computations

Optimum — Well Separated Clusters

- Optimum:
 - clusters are well separated
 - the reference set contains one tree per cluster
- Guha et al. [GJK⁺02] find **clusters by sampling** and estimate a reference set size.

Example

- $S_1 = \{T_1, T_2, T_3\}, S_2 = \{T_4, T_5, T_6\}$
- Reference set $K = \{T_1\}, 1 \leq l \leq |K| = 1$
- Distances:

	T_1	T_2	T_3	T_4	T_5	T_6
T_1	0	4	1	1	4	1
T_2		0	4	4	1	4
T_3			0	1	4	1
T_4				0	4	1
T_5					0	4
T_6						0

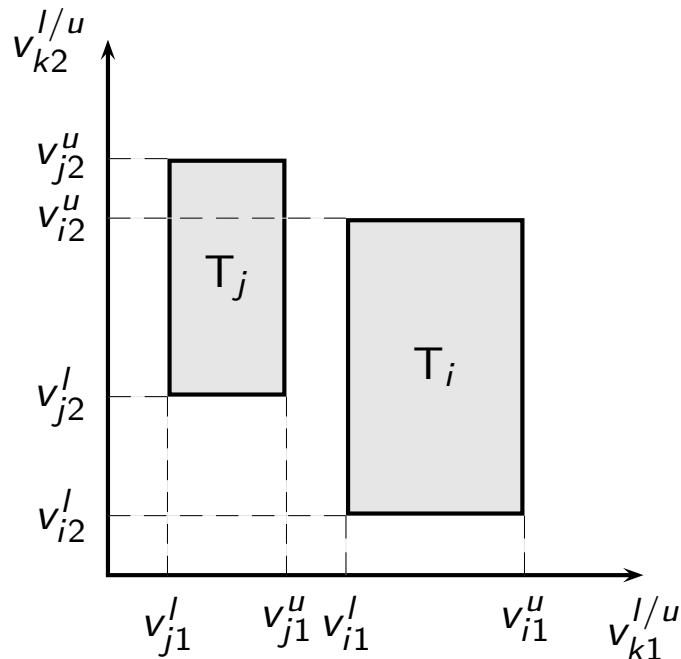
- Reference Vectors:
 $v_{1,1} = (0), v_{2,1} = (4), v_{3,1} = (1), v_{4,1} = (1), v_{5,1} = (4), v_{6,1} = (1)$
- The clusters $C_1 = \{T_1, T_3, T_4, T_6\}$ and $C_2 = \{T_2, T_5\}$ are well separated.

Combined Approach

- We combine the previous approaches:
 - lower bound with traversal strings
 - upper bound with constrained edit distance
 - reference sets
- Instead of one vector v_i we compute **two vectors** for each tree T_i :
 - **lower bound**: v_i^l contains the traversal string distance
 - **upper bound**: v_i^u contains the constrained edit distance

Metric Space

- **Metric space** of the reference set:
 - the elements of the reference set define the basis of a metric space
 - each tree T_k is represented by a rectangle in this metric space
 - v_k^l and v_k^u are two opposite corners of this rectangle
- **Example:** Two trees T_i and T_j in the metric space defined by the reference set $K = \{k_1, k_2\}$.



Combined Approach: Triangle Inequality

- The triangle equations changes as follows:

(a) For all $1 \leq l \leq |K|$

$$\delta_t(T_i, T_j) \leq v_{il}^u + v_{jl}^u$$

(b) For all $1 \leq l \leq |K|$

$$\delta_t(T_i, T_j) \geq \begin{cases} v_{jl}^l - v_{il}^u & \text{if } v_{jl}^l > v_{il}^u \\ v_{il}^l - v_{jl}^u & \text{if } v_{il}^l > v_{jl}^u \\ 0 & \text{otherwise} \end{cases}$$

- Note:

- If $v_{jl}^l > v_{il}^u$ or $v_{il}^l > v_{jl}^u$ then $[v_{il}^l, v_{il}^u]$ and $[v_{jl}^l, v_{jl}^u]$ are disjoint intervals.
- In all other cases we can not give a lower bound (other than 0).

Combined Approach: Upper and Lower Bounds

- Upper bound:

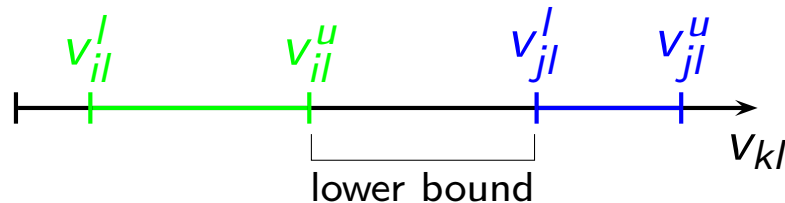
$$u_t(T_i, T_j) = \min_{l, 1 \leq l \leq |K|} v_{il}^u + v_{jl}^u$$

- Lower bound:

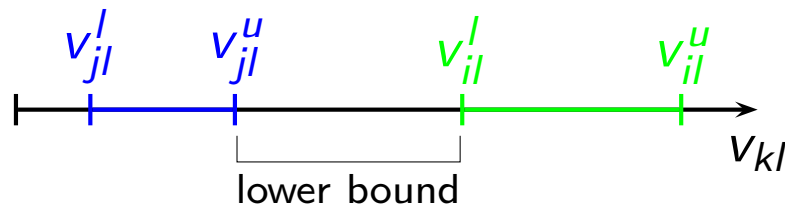
$$l_t(T_i, T_j) = \max_{l, 1 \leq l \leq |K|} \begin{cases} v_{jl}^l - v_{il}^u & \text{if } v_{jl}^l > v_{il}^u \\ v_{il}^l - v_{jl}^u & \text{if } v_{il}^l > v_{jl}^u \\ 0 & \text{otherwise} \end{cases}$$

Illustration: Cases for the Lower Bound

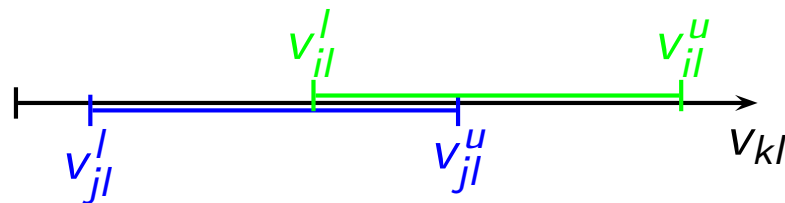
- Case $v_{jl}^l > v_{il}^u$: lower bound is $v_{jl}^l - v_{il}^u$



- Case $v_{il}^l > v_{jl}^u$: lower bound is $v_{il}^l - v_{jl}^u$



- All other cases ($[v_{il}^l, v_{il}^u]$ and $[v_{jl}^l, v_{jl}^u]$ overlap): no lower bound



Summary

- Reference Sets: Upper and Lower Bound
- Combination of reference sets with other bounds

What's Next?

- Binary Branch Distance
 - lower bound proof
 - complexity



Sudipto Guha, H. V. Jagadish, Nick Koudas, Divesh Srivastava, and Ting Yu.

Approximate XML joins.

In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 287–298, Madison, Wisconsin, 2002. ACM Press.