Propagation-Vectors for Trees(PVT):Concise yet Effective Summaries for Hierarchical Data and Trees

Venkata Snehith Cherukuri and K. Selçuk Candan

Comp. Sci. and Eng. Dept

Arizona State University

{vcheruku, candan}@asu.edu

Agenda

- Motivation
- PVT Summaries for trees.
  - Concept Propagation/Concept Vector(CP/CV)
  - Using CP/CV to construct tree summaries.
  - Use of PVT Summaries in Peer Search
- Experiments
- Conclusion

Motivation

- Data represented and exchanged in the form of tree-based XML documents is gaining popularity.
- Ability to do quick and dirty look-ups in large tree collections is a critical capability in a number of domains.
  - P2P Data Management Scenario
    - P2P systems are popular because of their decentralized and distributed nature resulting in high robustness, better use of resources, better scalability.
    - Nodes can be described by source descriptions(i.e. meta-data, such as taxonomies), which are tree-structured.
    - CHALLENGE: Finding peers with similar source descriptions in a quick fashion.

Motivation(Contd…)

- Example
  - *the Digital Archeological Record(tDAR)[1,2,3]* Project
    - **Aim**: Locate, search and query across distributed archeological data sets.
    - **Main task**: Locate relevant and comparable observations from numerous archeological data sets.
    - Locating involves identifying peers with similar taxonomies or most compatible data sets.
      - To find a compatible knowledge peer, a tDAR node searches for nodes that have similar tree structured source descriptions.

Motivation(Contd…)

- There has been lot of research on finding the edit-distance between tree based structures
  - Eg. Zhang-Sasha[4] and Klien[5]… but have high computational complexity of $O(n^4)$ and $O(n^3)$ respectively.
- A more scalable approach is to compare the summaries of tree-based sources.
  - CHALLENGE: Construct the summaries such that, they are concise, capture the structure accurately and are easy to compare .
- Thus, we focus on
  - …constructing accurate, concise and easy to compare summaries of tree-based sources.
  - …using these summaries to compare the tree-based sources.

Propagation Vectors for Trees(PVT) Approach

- Constructs very concise and accurate summaries of hierarchical data such as taxonomies or XML data trees.

- Relies on a structurally-informed label propagation scheme, and in the process, obtains tree summaries.

- Reduces the problem of comparing the similarity between the trees to the problem of computing similarities between the summary vectors.

Propagation Vectors for Trees(PVT)

Nodes represented as Vectors

- PVT summaries are based on the following observation:
  - A node in a given hierarchy clusters all its descendants and essentially acts as a context for the descendant nodes. Similarly, descendants of a given node may also act as a context for a node, differentiating the node from other nodes that are similarly labeled.

- Every node in the hierarchy is represented as a vector.
  - The vector represents the node's relationship with all the other nodes in the tree.

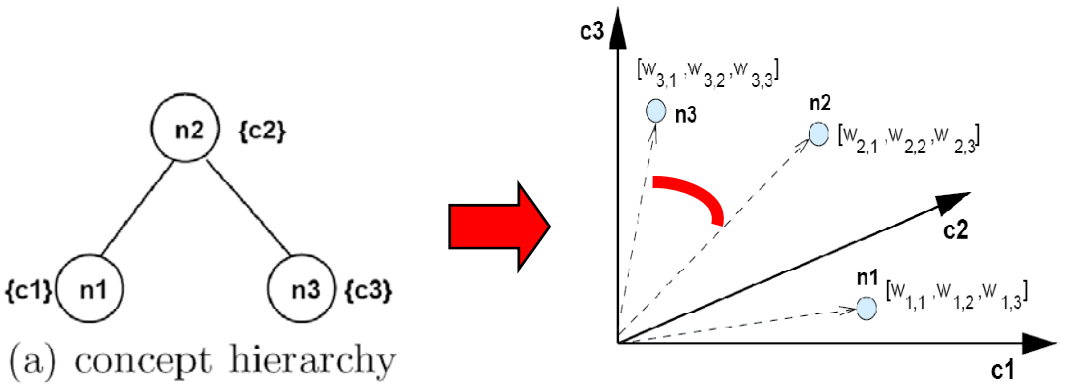
  - CP/CV[6] gives a way to represent every node in the hierarchy as a vector, representing the relationship of a node with the rest of the nodes.

Propagation Vectors for Trees(PVT)

Concept Propagation/Concept Vector(CP/CV)

- Originally developed to measure semantic similarities between terms/concepts in a taxonomy.

If each concept-node in a given hierarchy could be represented as a vector, then these vectors could be compared to compute concept similarity values

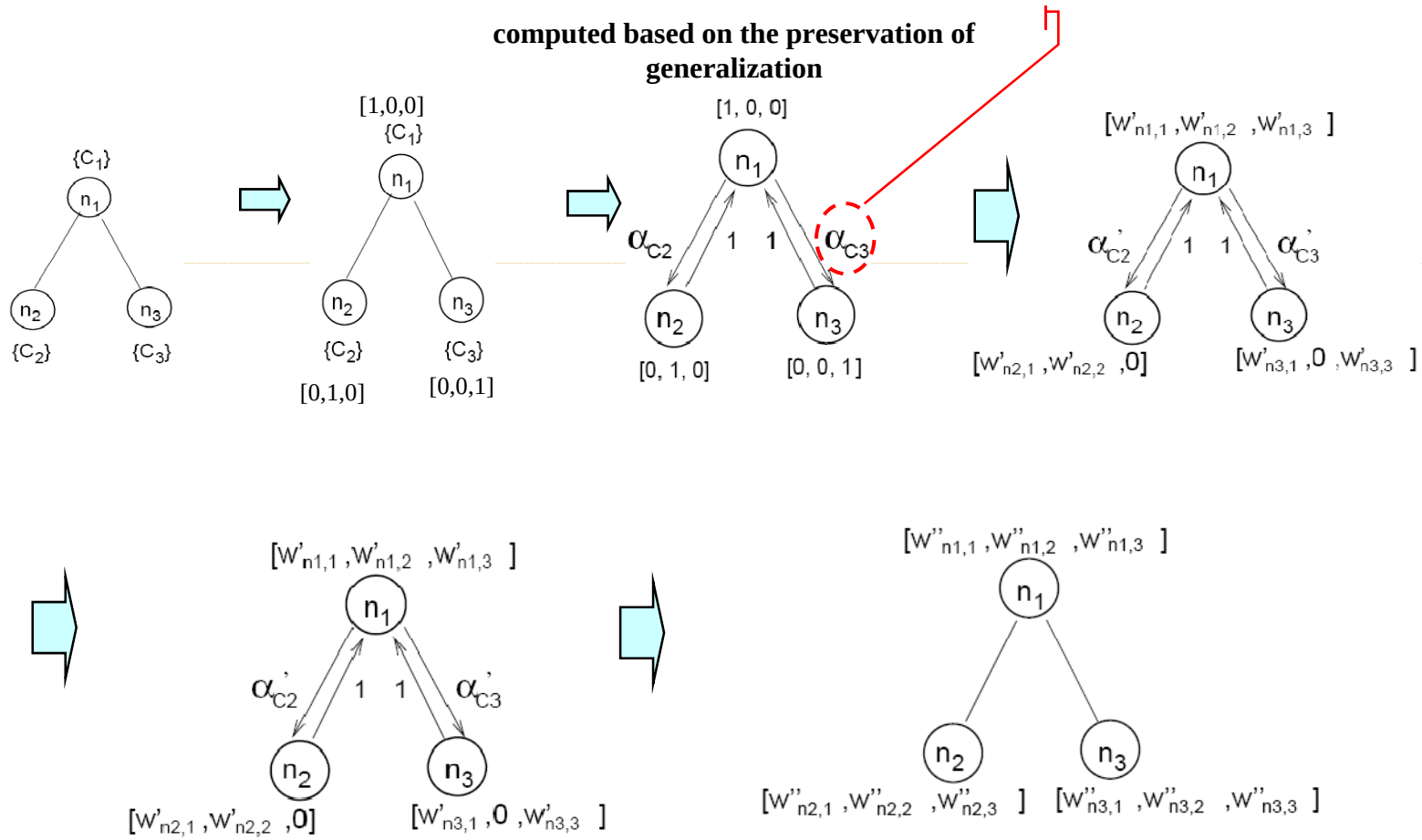


(a) concept hierarchy

(b) concept space

Propagation Vectors for Trees(PVT)

Concept Propagation/Concept Vector(CP/CV)

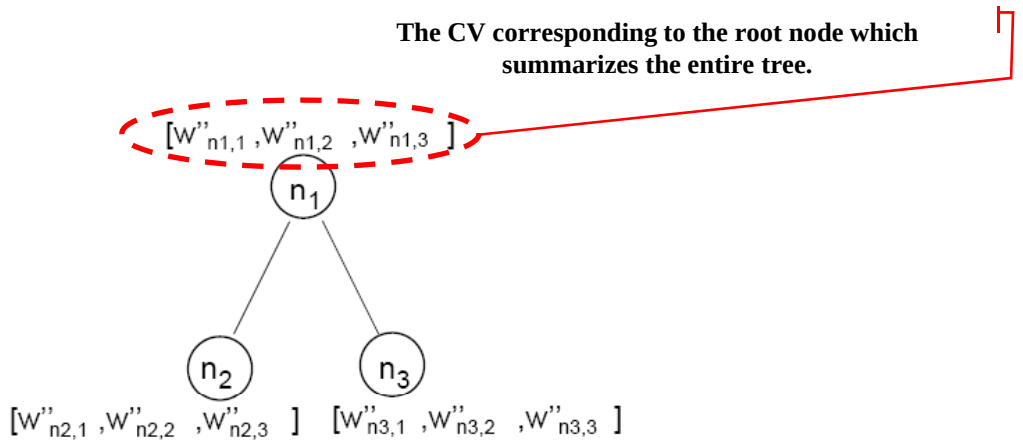


STOP: All nodes received all relevant concepts in the structure

→ Spreading activation with semantic preservation[9,10]

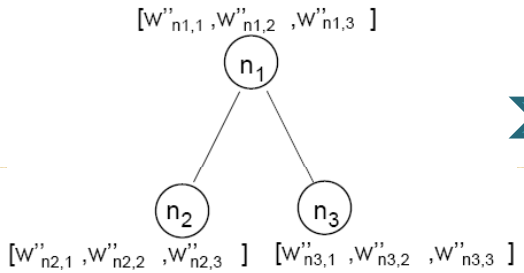
PVT: Root Vectors as Tree Summaries

- Idea:
  - Vector corresponding to the root node represents the context provided to it through all its descendants(i.e., the entire hierarchy)
  - Thus, the vector representation of the root node could be considered as a structural summary of the entire tree.



PVT: Root Vectors as Tree Summaries

- Challenge



What if ($n_1 = n_2$)?

  - In many hierarchies(such as XML data), there can be nodes with identical labels.

  - PVT overcomes this challenge by representing a repeating label as a combined entity, whose magnitude is computed as the square root of the sum of the squares of the individual components.

PVT: Root Vectors as Tree Summaries

- Advantage
  - The summary corresponding to a tree consists of unique labels.
  - Given a tree with ‘n’ nodes and ‘u’ unique node labels, space complexity for our approach is O(u).
  - The summarization approaches discussed in Helmer[7] require O(n) space to store the summaries.
  - For large data trees with n >> u, the space overhead incurred by the PVT summaries would be low compared to Helmer[7].

Use of PVT Summaries in Peer Search

- Each peer computes the PVT summary of its meta data off-line.
- Peer discovery involves picking peers with the most similar summary vectors.
- Summary vectors can be compared using different similarity measures:
  - Cosine Similarity
  - KL-Divergence
  - Intersection Similarity
- Experimental results show that KL-Divergence provides the best results for search and classification.

Experiments

- Experimental Setup
  - Quality measure
    - Nierman[8] and Helmer[7] use Hierarchical Agglomerative Clustering(HAC), to cluster XML documents.

    - The performance of the various summarization techniques is compared using amount of "mis-clustering".

    - *Mis-Clustering:* Minimum number of documents that have to be moved from one cluster to another so that all the documents belonging to a particular DTD are in the same cluster.

Experiments

- **Mis-clusterings for DTD's from Helmer[7].**
  - Helmer[7] presents comparisons of various summarization algorithms.
  - We used this data set so as to be able compare PVT with the other algorithms reported in the literature.

Data sets with highly similar elements

Num. of docs	SIGMOD 57	INEX 60	Music 34(*)	DFT1 140	DFT2 140	Elem 80	Freq 80	Pos 80	Depth 80	Overall
Tree-edit	1	0	13	26	0	0	0	5	0	6.0%
DFT										
direct ML	0	3	9	52	1	27	9	32	33	22.1%
pairwise ML	0	4	7	39	3	10	0	42	22	18.1%
Path Shingles										
tags	0	0	0	42	0	0	0	14	48	13.8%
pairwise	0	0	1	6	0	0	0	6	39	6.9%
full path	0	0	0	6	0	0	0	0	30	4.8%
gzip										
simple	1	2	0	15	19	3	29	15	44	17.0%
tags	0	0	0	7	20	0	16	0	6	6.5%
pairwise	0	0	0	7	38	0	20	0	26	12.1%
full path	0	0	1	2	38	0	24	0	3	9.1%
family order	0	0	0	23	34	0	13	8	0	10.4%
Ziv-Merhav										
tags	0	2	0	1	0	0	0	0	0	0.4%
pairwise	0	2	0	6	0	0	7	0	20	4.7%
full path	0	2	0	6	0	0	7	0	0	2.0%
family order	0	0	0	1	0	0	0	2	0	0.4%
PVT										
cosine sim.	0	0	0	10	0	0	12	5	4	4.13%
intersection similarity	0	3	0	6	0	0	12	11	2	4.53%
KL distance	0	0	0	6	0	0	0	0	3	1.19%

Experiments

- **Mis-clusterings for Distinct DTD data Set**
  - Data sets used in Helmer[7] are very specific.
  - Thus we created more general data sets to compare the performance of PVT with Helmer[7].
  - Started with a data set generated from DTD's which have completely different elements.

Num. of docs	DS1 700	DS2 700	DS3 700	DS4 700	Overall
gzip					
tags	34	5	32	79	5.35%
pair wise	6	0	36	23	2.32%
full path	11	0	25	27	2.25%
family order	5	0	25	3	1.17%
Ziv-Merhav					
tags	34	25	26	25	3.92%
pair wise	181	169	190	156	24.85%
full path	178	174	190	161	25.10%
family order	149	116	126	116	18.10%
PVT					
Cosine	0	0	0	0	0%
Intersection	0	0	0	0	0%
KL Distance	0	0	0	0	0%

Experiments

- **Mis-clusterings for Hybrid DTD dataset.**
  - Distinguishing documents from DTD's with different element sets is not very challenging.
  - Thus, we combined the elements of the various DTD's to generate a set of hybrid DTD's.

Num. of docs	fHyb1 300	fHyb2 300	fHyb3 300	fHyb4 300	Overall
gzip					
tags	48	28	32	0	9%
pair wise	42	10	84	0	11.33%
full path	63	17	38	0	9.83%
family order	29	10	33	0	6%
Ziv-Merhav					
tags	119	0	57	43	18.25%
pair wise	139	0	45	52	16.41%
full path	139	0	50	0	15.75%
family order	94	0	41	52	15.58%
PVT					
cosine	40	0	48	0	7.33%
Intersection	98	0	43	0	11.75%
KL Distance	40	0	34	0	6.16%

Experiments

- Comparison of run time for computing all-pairs similarities for 700 documents.
  - Helmer[7] indicated that implementation not optimized for efficiency.
  - Results should be seen as ball-park figures.

	Time (seconds)
gzip	
tags	3359
pairwise	3439
full path	3466
family order	3424
Ziv-Merhav	
tags	529
pairwise	478
full path	795
family order	485
PVT	228

Conclusion

- We present a tree summarization method to enable nodes in an informational network to find peers, both quickly and accurately.

- We present the PVT approach which:
  - …maps each node in the tree to a multi-dimensional vector, capturing the node's relationship with the rest of the nodes.
  - …uses the vector representation of the root node as the summary.
  - …computes the similarities between the trees by comparing the root vectors.

- Experimental results have shown that the PVT summaries are highly accurate and the similarity comparisons are faster than the existing approaches.

References

[1] K. Kintigh. The promise and challenge of archaeological data integration. *American Antiquity, 71(3):567–578, 2006.*

[2] Y. Qi, K. S. Candan, and M. L. Sapino. Ficsr: feedback-based inconsistency resolution and query processing on misaligned data sources. SIGMOD 2007.

[3] Y. Qi, K. S. Candan, M. L. Sapino, and K. W. Kintigh. Integrating and querying taxonomies with quest in the presence of conflicts. SIGMOD 2007.

[4] K. Zhang and D. Shasha. Simple fast algorithms for the editing distance between trees and related problems. *SIAM J.Comput., 18(6):1245–1262, 1989.*

[5] P. Klein. Computing the edit-distance between unrooted ordered trees. *Proceedings of the 6th Annual European Symposium, number 1461, 1998.*

[6] J. W. Kim and K. S. Candan. CP/CV: Concept similarity mining without frequency information from domain describing taxonomies. CIKM 2006.

[7] S. Helmer. Measuring the structural similarity of semistructured documents using entropy. VLDB 2007.

[8] A. Nierman and H. V. Jagadish. Evaluating structural similarity in XML documents. WebDB 2002.

[9] F. Crestani. Application of Spreading Activation Techniques in Information Retrieval. Artificial Intelligence Review, 11(6), 97.

[10] F. Gelgi et al. Improving Web Data Annotations with Spreading Activation. In *WISE, 2005.*

Questions???