

A Gibbs Sampling Approach to Detection of Tree Motifs

Lidio Marx Carvalho Meireles
lidio@kuicr.kyoto-u.ac.jp

Tatsuya Akutsu
takutsu@kuicr.kyoto-u.ac.jp

Bioinformatics Center, Institute for Chemical Research, Kyoto University, Gokasho, Uji, Kyoto 611-0011, Japan.

Keywords: tree comparison, data mining, gibbs sampling, motif discovery

1 Introduction

The problem of finding common subtree structures arises in many different contexts in bioinformatics, such as the comparison of glycan structures, RNA secondary structures or simply the comparison of phylogenetic trees generated by many different methods.

Algorithms for aligning two trees have been reported in [2, 4]. In [1], a probabilistic model was used for mining subtree patterns in glycans. Algorithms for discovering frequent subtrees in a collection of trees appeared in [3, 7].

Here we present a new method for finding tree motifs. Given one database of rooted, labeled and unordered tree structures, our method finds in all (or most all) trees common substructures with good matches of the node labels. The sense of “good match” here is implemented according to a score function.

2 Method

Our method combines into one single program three previously reported methods and ideas: (i) efficient enumeration of rooted, unordered trees [3]; (ii) tree matching [6] and (iii) gibbs sampling similarly as applied to the multiple sequence alignment problem [5].

We fix one tree topology and match it against the whole database. If there are many possible matches in a given database tree, we randomly select one match. Then, we derive the common substructure profile, i.e., one fixed tree topology with the frequency of label occurrences attached in each node position. In the next step, we use this profile to score the matches of the fixed topology against the database. The scores are used as weights to bias the selection of the next tree match. If one tree match has high score, so it means it has high similarity with the tree profile, more likely it is to be selected. The profile is then updated with the new selected match. We repeat the biased sampling and profile updating steps until stabilization of the profile or a given number of interactions is reached. We perform the gibbs sampling procedure for a few times with different initial random matches and then we move to the next enumerated tree (fixed topology).

3 Results

We have implemented a prototype of the system using enumeration of ordered trees, instead of unordered trees, and we ran it against two types of databases of artificially created tree structures. Each database tree had 50 nodes and maximum degree of 5.

On the first database type, we guaranteed that all the trees had at least three 100% identical labeled subtrees with size 5, 8 and 10 nodes. The program could find not only these three common subtrees, but also others 100% identical subtrees of size 5, 6, 7, 8 and 9 embedded on the subtrees of size 8 and 10. On the second database type, we guaranteed that all the trees had at least three common subtrees with size 5, 8, and 10 nodes and with label identity of 80%, 75% and 60%, respectively. The program could find many common substructures with significant label identity. The experiments were performed on a laptop computer, processor Athlon XP 1.4 GHz, 480 MB RAM, operating system Windows XP. The programming language was the C programming language. In our implementation, we enumerated all ordered trees up to 10 nodes

(6917 trees) and we required that the common substructures were present in all trees of the database. In Table 1 we show running times for different database sizes. Running times for both database types were approximately the same. In Figure 1, we show an example of tree motif found by the program.

Table 1: Experiment running times.

Database size (#trees)	n=10	n=20	n=30	n=40	n=50
Running time (min)	6.0	8.6	11.5	14.2	16.2

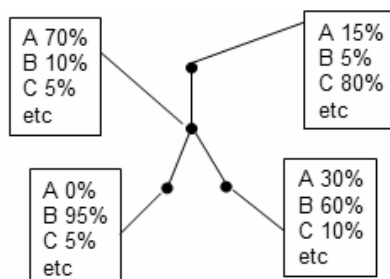


Figure 1: Example of tree motif.

4 Discussions

We have successfully introduced a new method that is able to find common tree substructures, along with their profiles, in a collection of trees within reasonable computational time. It is important to mention here that our experiment used enumeration of ordered trees, so many identical unordered trees were considered more than once. Thus the running time is expected to decrease significantly by implementing the enumeration of unordered trees.

References

- [1] Aoki, K. F., Ueda, N., Yamaguchi, A., Kanehisa, M., Akutsu, T., Mamitsuka, H., Application of a new probabilistic model for recognizing complex patterns in glycans. *Bioinformatics*, Suppl 1: I6-I14, 2004.
- [2] Aoki, K. F., Yamaguchi, A., Okuno, Y., Akutsu, T., Ueda, N., Kanehisa, M., Mamitsuka, H., Efficient Tree-Matching Methods for Accurate Carbohydrate Database Queries. *Genome Informatics*, 14: 134-143, 2003.
- [3] Asai, T., Abe, K., Kawasoe, S., Arimura, H., Sakamoto, H., Arikawa, S., Efficient Substructure Discovery from Large Semi-structured Data. *Proceedings of the 2nd Annual SIAM Symposium on Data Mining*, 158-174, 2002.
- [4] Jiang, T., Wang, L., Zhang, K., Alignment of trees – an alternative to tree edit. *Theoretical Computer Science*, 143: 137-148, 1995.
- [5] Lawrence C. E., Altschul S. F., Boguski M. S., Liu J. S., Newwald A. F., Wootton J. C., Detecting subtle sequence signals: A Gibbs sampling strategy for multiple alignment. *Science* 262:208-214, 1993.
- [6] Matula, D. W. Subtree isomorphism in $O(n^{2.5})$. In Alspach, B., Hell, P., Miller, D. J., eds: *Algorithmic Aspects of Combinatorics, Volume 2 of Ann. Discrete Math*, North-Holland, 91-106, 1978.
- [7] Zaki, M. J., Efficiently Mining Frequent Trees in a Forest. *Proceedings of the eight ACM SIGKDD international conference on Knowledge discovery and data mining*, ACM Press Session, Graphs and trees session: 71-80, 2002.