Structural similarity based on MathML syntactic structure

[Extended Abstract] *

Michail Ponomarev†
Peter the Great St. Petersburg Polytechnic University
29 Polytechnicheskaya st.
195251 St. Petersburg Russia
ponmike92@gmail.com

Evgeny Pyshkin‡
Unversity of Aizu
Tsuruga, Ikki-machi, Aizu-Wakamatsu
Fukushima, Japan 965-8580
pyshe@u-aizu.ac.jp

ABSTRACT

In our days search engines in the Internet are an integral and necessary tools. Despite the fact that text information retrieval is well developed, searching for mathematical expressions is extremely limited. Current math retrieval systems limit themselves to exact matches ignoring structure of mathematical expression.

We propose a modification of well-known tree-overlapping algorithm which has been adoped to find stuctural similarity between to mathematical expressions represented in MathML.

General Terms

Keywords

1. INTRODUCTION

2. METHODS OF ESTIMATING SIMILARITY BETWEEN ORDERED TREES

2.1 Tree edit distance

Tree edit distance method defines similarity (distance) between two trees as weighted number of edit operations (insert, delete, and modify) to transform one tree to another.

2.2 Tree Kernel

Tree Kernel defines similarity between two trees as the number of shared subtrees. Subtree S of tree T is is a subgraph which consists of more than one node and except for the frontier nodes of itself and each node has the same daughter nodes as the corresponding node in T.

*
†
‡

ICAIT '16, Oct. 6 – 8, 2016, Aizu-Wakamatsu, Japan.

2.3 Subpath Set

Subpath Set similarity between two trees is defined as the number of subpaths shared by the tree. Given a tree, its subpaths is defined as a set of every path from the root node to leaves and their partial paths.

2.4 Tree Overlapping

3. DEFINITION OF MATHEMATICAL STRUCTUR SIMILARITY

4. MODIFICATION OF TREE OVERLAPPING ALGORITHM

4.1 Definition of similarity

4.2 Algorithm

When putting an arbitrary node n_1 of tree T_1 on node n_2 of tree T_2, there might be the same production rule ovelapping in T_1 and T_2. We define N_{TO} as the set of pairs of such overlapping production rules when n_1 overlaps n_2. In comparison with base tree overlapping algorithm, we also include terminal nodes as if they had same production rule.

$L(n_1, n_2)$ represents a set of pairs of nodes which overlap each other when outting n_1 on n_2. It is defined exactly the same as in base algorithm.

$N_{TO}(n_1, n_2)$ is defined by using $L(n_1, n_2)$ as follows:

$$N_{TO}(n_1, n_2) = \left\{ (m_1, m_2) \middle| \begin{array}{l} m_1 \in nodes(T_1) \\ \wedge\ m_2 \in nodes(T_2) \\ \wedge\ (m_1, m_2) \in L(n_1, n_2) \\ \wedge\ PR(m_1) = PR(m_2) \end{array} \right\}, \tag{1}$$

where $nodes(T)$ is a set of nodes in tree T, $PR(n)$ is a production rule rooted at node n.

For example in Figure 4.2, $N_{TO}(d_1, d_2) = (d^1, d^2), (f^1, f^2), (g^1, g^2)$

$P_{WPR}(n_1, n_2)$ is the set of nodes which is represented as path from (n_1, n_2) to the top last pair of nodes which have same number among their siblings. P_{WPR} is defined as follows. Here n_i and m_i are nodes of tree T_i, $ch(n, i)$ is the i'th child of node n.

1. $(n_1, n_2) \notin P_{WPR}$
2. If $PR(parent(n_1)) \neq PR(parent(n_2))$
   $\wedge\ ch(parent(n_1), i) = ch(parent(n_2), i)$

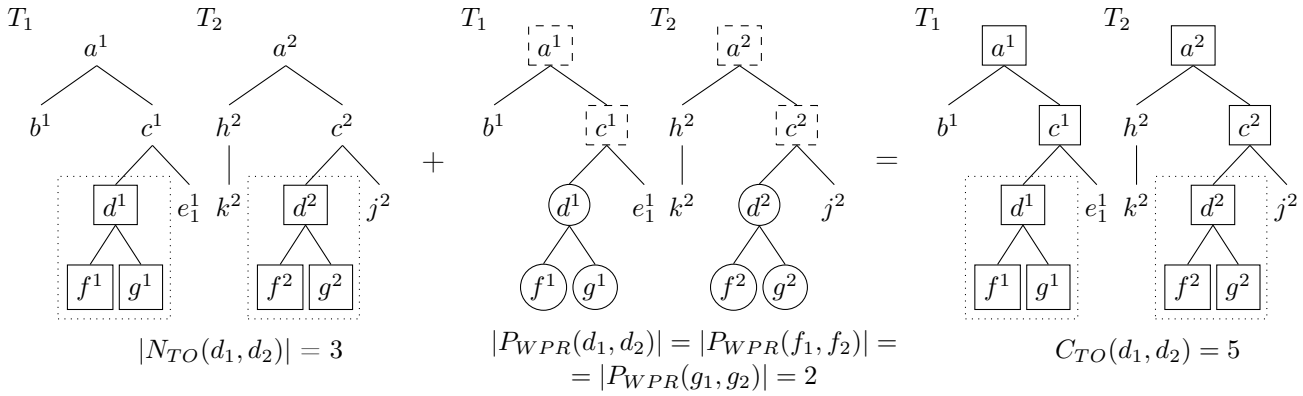

Figure 1: Example of Tree-Overlapping modification algorithm

$\wedge\ ch(parent(n_1), i) = n_1$
$\wedge\ h(parent(n_2), i) = n_2,$
$(parent(n_1), parent(n_2)) \in P_{WPR}$

3. $P_{WPR}(n_1, n_2)$ includes only pairs generated by applying 2. recursively.

$P_{TO}(n_1, n_2)$ is defined by using P_{WPR} as follows:

$$P_{TO}(n_1, n_2) = \left\{ (m_1, m_2) \;\middle|\; \begin{array}{l} (p_1, p_2) \in N_{TO}(n_1, n_2) \\ (m_1, m_2) \in P_{WPR}(p_1, p_2), \\ \qquad \text{if} \quad top(m_1, m_2) = (n_1, n_2) \end{array} \right\} \quad (2)$$

Tree overlapping similarity $C_{TO}(n_1, n_2)$ is defined as foolows by using $N_{TO}(n_1, n_2)$ and $P_{TO}(n_1, n_2)$.

$$C_{TO}(n_1, n_2) = |N_{TO}(n_1, n_2)| + |P_{TO}(n_1, n_2)| \quad (3)$$

Formula of tree overlapping similarity corresponds to basic formula by using $C_{TO}(n_1, n_2)$.

$$S'_{TO}(T_1, T_2) = \max_{n_1 \in nodes(T_1), n_2 \in nodes(T_2)} C''_{TO}(n_1, n_2) \quad (4)$$

$P_{WPR}(d_1, d_2) = P_{WPR}(f_1, f_2) = P_{WPR}(g_1, g_2) = 2\, C_{TO}(d_1, d_2) = |N_{TO}(d_1, d_2)| + |P_{WPR}(d_1, d_2)| = 5$

5. EXPERIMENTS

5.1 Data

5.2 Results

6. REFERENCES