A Tree Distance Function Based on Multi-sets
Protecting Open Source Software with Trees

A. Müller  K. Hirata  T. Shinohara

Department of Artificial Intelligence
Kyushu Institute of Technology (Iizuka, Japan)

ALSIP 2008 (Osaka, Japan)
Outline

1 Introduction
   - Motivation: Binary Program Matching

2 Proposed Tree Distance Function
   - Concepts
   - Related Works
   - Distance Definition
   - Examples
   - Characteristics

3 Case Study
   - Setup
   - Results

4 Conclusions
Introduction

Tree structured data, main contributions

- XML, RNA structures
  - Approximate Binary Program Matching (ABPM)
    - Detect program theft (OSS license violations)
    - Detect common low level functionality
  - Metrics are desirable:
    - $d(T_1, T_2) = 0 \iff T_1 = T_2$
    - Triangle inequality (MAM can be employed)
  - Formalized an $O(n^2)$ metric ($mtd$) proposed before. Müller, Shinohara (2006)
    - Other functions are not suitable for ABPM
    - Designed for ABPM
    - Experimentally studied properties
Introduction

Tree structured data, main contributions

- XML, RNA structures
- Approximate Binary Program Matching (ABPM)
  - Detect program theft (OSS license violations)
  - Detect common low level functionality
- Metrics are desirable:
  - \( d(T_1, T_2) = 0 \iff T_1 = T_2 \)
  - Triangle inequality (MAM can be employed)
- Formalized an \( O(n^2) \) metric (mtd) proposed before. Müller, Shinohara (2006)
  - Other functions are not suitable for ABPM
  - Designed for ABPM
  - Experimentally studied properties
Introduction

Tree structured data, main contributions

- XML, RNA structures
- Approximate Binary Program Matching (ABPM)
  - Detect program theft (OSS license violations)
  - Detect common low level functionality
- Metrics are desirable:
  - \( d(T_1, T_2) = 0 \iff T_1 = T_2 \)
  - Triangle inequality (MAM can be employed)
- Formalized an \( O(n^2) \) metric (mtd) proposed before. Müller, Shinohara (2006)
  - Other functions are not suitable for ABPM
  - Designed for ABPM
  - Experimentally studied properties
Introduction

Tree structured data, main contributions

- XML, RNA structures
- Approximate Binary Program Matching (ABPM)
  - Detect program theft (OSS license violations)
  - Detect common low level functionality
- Metrics are desirable:
  - \( d(T_1, T_2) = 0 \iff T_1 = T_2 \)
  - Triangle inequality (MAM can be employed)
- Formalized an \( O(n^2) \) metric (\( mtd \)) proposed before. Müller, Shinohara (2006)
  - Other functions are not suitable for ABPM
  - Designed for ABPM
  - Experimentally studied properties
Outline

1. Introduction
   - Motivation: Binary Program Matching

2. Proposed Tree Distance Function
   - Concepts
   - Related Works
   - Distance Definition
   - Examples
   - Characteristics

3. Case Study
   - Setup
   - Results

4. Conclusions
Pirate Candidate Program Fragments

Fragment Similarity (Stemming, "Normalization")

Ranking (Information Retrieval)
Motivation: Binary Program Matching

Expanded Fragment:
\[ \text{count}_1 = \Phi(\text{count}, \text{count}_2) \]
\[ \text{count}_1 = \Phi(1, \text{count}_1 + 1) \]

\begin{align*}
\text{i0} &= i \\
\text{res} &= 1 \\
\text{count} &= 1 \\
\text{block}_0: \\
\text{count}_1 &= \Phi(\text{count}, \text{count}_2) \\
\text{res}_1 &= \Phi(\text{res}, \text{res}_2) \\
\text{if} \text{count}_1 > \text{i0} \text{ goto block}_1 \\
\text{res}_2 &= \text{res}_1 \times \text{count}_1 \\
\text{count}_2 &= \text{count}_1 + 1 \\
\text{goto} \text{ block}_0 \\
\text{block}_1: \\
\text{return} \text{res}_1 \\
\text{count}_1 &= \Phi(\text{count}, \text{count}_2)
\end{align*}

Output: machine instruction trees
- Complete Subtrees required
- Deeper change, greater semantic change

Müller, Hirata, Shinohara (KIT)
Motivation: Binary Program Matching

Expanded Fragment:
\[ \text{count}_1 = \text{Phi}(1, \text{count}_1 + 1) \]
\[ i_0 = i \]
\[ \text{res} = 1 \]
\[ \text{count} = 1 \]

block_0:
\[ \text{res}_1 = \text{Phi}(	ext{res}, \text{res}_2) \]
\[ \text{if} \ \text{count}_1 > i_0 \ \text{goto} \ \text{block}_1 \]
\[ \text{res}_2 = \text{res}_1 \times \text{count}_1 \]
\[ \text{count}_2 = \text{count}_1 + 1 \]
\[ \text{goto} \ \text{block}_0 \]

\[ \text{count}_1 = \text{Phi}(\text{count}, \text{count}_2) \]

\[ \text{block}_1: \]
\[ \text{return} \ \text{res}_1 \]

Output: machine instruction trees
- Complete Subtrees required
- Deeper change, greater semantic change
Outline

1. Introduction
   - Motivation: Binary Program Matching

2. Proposed Tree Distance Function
   - Concepts
   - Related Works
   - Distance Definition
   - Examples
   - Characteristics

3. Case Study
   - Setup
   - Results

4. Conclusions
Concepts

Concepts employed during the presentation

- Multi-set additive union: $\uplus$
  - $\{A, A, B\} \uplus \{A, B, C\} = \{A, A, A, B, B, C\}$

- Multi-set union: $\sqcup$
  - $\{A, A, B\} \sqcup \{A, B, C\} = \{A, A, B, C\}$

- Multi-set intersection: $\sqcap$
  - $\{A, A, B\} \sqcap \{A, B, C\} = \{A, B\}$

Definition

Let $T$ be a tree and $v$ a node in $T$. A complete subtree of $T$ at $v$ is a subtree of $T$ of which its root is $v$ and that contains all descendants of $v$. 

- Complete subtree preserves semantics
Concepts

Concepts employed during the presentation

- **Multi-set additive union**: \( \oplus \)
  
  \[ \{A, A, B\} \oplus \{A, B, C\} = \{A, A, A, B, B, C\} \]

- **Multi-set union**: \( \sqcup \)
  
  \[ \{A, A, B\} \sqcup \{A, B, C\} = \{A, A, B, C\} \]

- **Multi-set intersection**: \( \cap \)
  
  \[ \{A, A, B\} \cap \{A, B, C\} = \{A, B\} \]

**Definition**

Let \( T \) be a tree and \( v \) a node in \( T \). A *complete subtree* of \( T \) at \( v \) is a subtree of \( T \) of which its root is \( v \) and that contains all descendants of \( v \).
Proposed Tree Distance Function

Concepts

Concepts employed during the presentation

- **Multi-set additive union**: \(\uplus\)
  \[\{A, A, B\} \uplus \{A, B, C\} = \{A, A, A, B, B, C\}\]

- **Multi-set union**: \(\sqcup\)
  \[\{A, A, B\} \sqcup \{A, B, C\} = \{A, A, B, C\}\]

- **Multi-set intersection**: \(\sqcap\)
  \[\{A, A, B\} \sqcap \{A, B, C\} = \{A, B\}\]

**Definition**

Let \(T\) be a tree and \(v\) a node in \(T\). A *complete subtree* of \(T\) at \(v\) is a subtree of \(T\) of which its root is \(v\) and that contains all descendants of \(v\).
Concepts

Concepts employed during the presentation

- **Multi-set additive union**: $\uplus$
  
  $\{A, A, B\} \uplus \{A, B, C\} = \{A, A, A, B, B, C\}$

- **Multi-set union**: $\sqcup$
  
  $\{A, A, B\} \sqcup \{A, B, C\} = \{A, A, B, C\}$

- **Multi-set intersection**: $\sqcap$
  
  $\{A, A, B\} \sqcap \{A, B, C\} = \{A, B\}$

**Definition**

Let $T$ be a tree and $v$ a node in $T$. A *complete subtree* of $T$ at $v$ is a subtree of $T$ of which its root is $v$ and that contains all descendants of $v$.

- Complete subtree preserves semantics
Outline

1. Introduction
   - Motivation: Binary Program Matching

2. Proposed Tree Distance Function
   - Concepts
   - Related Works
   - Distance Definition
   - Examples
   - Characteristics

3. Case Study
   - Setup
   - Results

4. Conclusions
Overview

Distance Functions

- TED Tai (1979)
  - Insert, delete, rename operations
  - Minimum edit script
  - Fastest Algorithm: $O(n^3)$. Demaine et al. (2007)
  - Changes are treated equally, not good for ABPM

- Extension of edit operations. Chawathe et al. (1997)
  - Operations on subtrees: move, copy, glue (inverse of copy)
  - NP-Complete, Heuristic $O(n^3)$
  - Not a metric
Overview

Distance Functions

- **TED Tai (1979)**
  - Insert, delete, rename operations
  - Minimum edit script
  - Fastest Algorithm: $O(n^3)$. Demaine et al. (2007)
  - Changes are treated equally, not good for ABPM

- **Extension of edit operations. Chawathe et al. (1997)**
  - Operations on subtrees: move, copy, glue (inverse of copy)
  - NP-Complete, Heuristic $O(n^3)$
  - Not a metric
Overview (2)

Distance Functions

- **Constrained edit distance**: Wang, Zhang (2005)
  - Like ted but equal subtrees are matched first
  - $O(n^2)$, complete subtrees not preserved

- **N-Gram Based**: Yang et al. (2005), Ohkura et al. (2004)
  - Partition trees, match those parts
  - Semantics are lost
  - $O(n)$

- No suitable distance functions available, we implemented *mtd*
Overview (2)

Distance Functions

  - Like ted but equal subtrees are matched first
  - $O(n^2)$, complete subtrees not preserved
- N-Gram Based: Yang et al. (2005), Ohkura et al. (2004)
  - Partition trees, match those parts
  - Semantics are lost
  - $O(n)$

- No suitable distance functions available, we implemented \textit{mtd}
Constrained edit distance: Wang, Zhang (2005)

- Like ted but equal subtrees are matched first
- $O(n^2)$, complete subtrees not preserved

N-Gram Based: Yang et al. (2005), Ohkura et al. (2004)

- Partition trees, match those parts
- Semantics are lost
- $O(n)$

No suitable distance functions available, we implemented $mtd$
Outline

1. Introduction
   - Motivation: Binary Program Matching

2. Proposed Tree Distance Function
   - Concepts
   - Related Works
   - Distance Definition
   - Examples
   - Characteristics

3. Case Study
   - Setup
   - Results

4. Conclusions
Proposed Tree Distance Function

**Distance Definition**

$mtd$

Tree distance function based on multi-sets

- $s(T)$: multi-set of all complete subtrees of $T$
- $n(T)$: multi-set of all the nodes of $T$

\[
\delta(A, B) = |A \cup B| - |A \cap B|, \quad (1)
\]

\[
d_s(T_1, T_2) = \delta(s(T_1), s(T_2)), \quad (2)
\]

\[
d_n(T_1, T_2) = \delta(n(T_1), n(T_2)), \quad (3)
\]

\[
mtd(T_1, T_2) = \frac{d_s(T_1, T_2) + d_n(T_1, T_2)}{2} \quad (4)
\]
Outline

1. Introduction
   - Motivation: Binary Program Matching

2. Proposed Tree Distance Function
   - Concepts
   - Related Works
   - Distance Definition
   - Examples
   - Characteristics

3. Case Study
   - Setup
   - Results

4. Conclusions
Proposed Tree Distance Function

Examples

\[ s(T_1) = \{ A(B, C(E, F(G)), D), B, C(E, F(G)), E, F(G), G, D \} \]
\[ s(T_2) = \{ H(B, D, C(E, F(G))), B, C(E, F(G)), E, F(G), G, D \} \]
\[ n(T_1) = \{ A, B, C, E, F, G, D \} \]
\[ n(T_2) = \{ H, B, C, E, F, G, D \} \]

\[ |s(T_1) \cap s(T_2)| = 6 \quad mtd(T_1, T_2) = \frac{(8-6)+(8-6)}{2} = 2 \]
\[ |n(T_1) \cap n(T_2)| = 6 \quad ted(T_1, T_2) = 3 \]
Proposed Tree Distance Function

Examples

\[ s(T_3) = \{ A(B(F, C(D, E))), B(F, C(D, E)), F, C(D, E), D, E \} \]

\[ s(T_4) = \{ A(B(F, C(G, E))), B(F, C(G, E)), F, C(G, E), G, E \} \]

\[ n(T_3) = \{ A, B, F, C, D, E \} \]

\[ n(T_4) = \{ A, B, F, C, G, E \} \]

\[ |s(T_3) \cap s(T_4)| = 2 \quad mtd(T_3, T_4) = \frac{(10-2)+(7-5)}{2} = 5 \]

\[ |n(T_3) \cap n(T_4)| = 5 \quad ted(T_3, T_4) = 1 \]

- Complete subtrees are necessary
Outline

1. Introduction
   - Motivation: Binary Program Matching

2. Proposed Tree Distance Function
   - Concepts
   - Related Works
   - Distance Definition
   - Examples
   - Characteristics

3. Case Study
   - Setup
   - Results

4. Conclusions
Properties

Characteristics of $mtd$

$mtd$ is an $O(n^2)$ metric

Result is always in $\mathbb{N}$

- $d_n$: same root nodes, different children
- $d_s$: preserves semantic chunks of expressions
- $d_s$: very sensitive to changes
- $d_n$: trees become close to each other

Is the average between $d_s$ and $d_n$ useful?

- Approximate program matching: only $d_s$ is enough
- Minimum and maximum values become smaller
Properties

Characteristics of \( mtd \)

- \( mtd \) is an \( O(n^2) \) metric
- Result is always in \( \mathbb{N} \)
- \( d_n \) same root nodes, different children
- \( d_s \) preserves semantic chunks of expressions
- \( d_s \): very sensitive to changes
- \( d_n \): trees become close to each other
- Is the average between \( d_s \) and \( d_n \) useful?
  - Approximate program matching: only \( d_s \) is enough
  - Minimum and maximum values become smaller
**Properties**

**Characteristics of \( mtd \)**

- \( mtd \) is an \( O(n^2) \) metric
- Result is always in \( \mathbb{N} \)
- \( d_n \) same root nodes, different children
- \( d_s \) preserves semantic chunks of expressions
- \( d_s \): very sensitive to changes
- \( d_n \): trees become close to each other

Is the average between \( d_s \) and \( d_n \) useful?
- Approximate program matching: only \( d_s \) is enough
- Minimum and maximum values become smaller
Properties

Characteristics of \( mtd \)

- \( mtd \) is an \( O(n^2) \) metric
- Result is always in \( \mathbb{N} \)
- \( d_n \): same root nodes, different children
- \( d_s \): preserves semantic chunks of expressions
- \( d_s \): very sensitive to changes
- \( d_n \): trees become close to each other
- Is the average between \( d_s \) and \( d_n \) useful?
  - Approximate program matching: only \( d_s \) is enough
  - Minimum and maximum values become smaller
Outline

1. Introduction
   - Motivation: Binary Program Matching

2. Proposed Tree Distance Function
   - Concepts
   - Related Works
   - Distance Definition
   - Examples
   - Characteristics

3. Case Study
   - Setup
   - Results

4. Conclusions
Setup

Experiment Definition

- 244668 trees
- Average depth: 4.75
- Average number of nodes: 11.11
- Randomly selected 1000 trees (queries)
  - Compare them against the dataset
- Distance functions:
  - \( \text{ted} \, O(n^3) \) Demaine (2007), \( \text{BDist} \, O(n) \) Yang (2005), \( \text{mtd} \, O(n^2) \)
  - Intel(R) Xeon(R) CPU 2.66 G-Hz with 4 processors
Distribution between data and queries

Distance Distribution

% of calculations

Distance

ted
mtd
BDist

Müller, Hirata, Shinohara (KIT)
Tree Distance Function Based on Multi-sets
ALSIP 2008
Outline

1. Introduction
   - Motivation: Binary Program Matching

2. Proposed Tree Distance Function
   - Concepts
   - Related Works
   - Distance Definition
   - Examples
   - Characteristics

3. Case Study
   - Setup
   - Results

4. Conclusions
Variation of $mtd$ and $BDist$

On average similar, but $mtd$ is a metric

Variation of $mtd$ when $ted$ is $x$

Variation of $Bdist$ when $ted$ is $x$
Variation of $d_s$ and $d_n$

On average $d_n$ is closer to $ted$!

Variation of $d_s$ when $ted$ is $x$

Variation of $d_n$ when $ted$ is $x$
Comparing $d_n$, $d_s$ and $mtd$
Benchmarks

Execution time results

- Queries: 1000
- DB: 244668

<table>
<thead>
<tr>
<th>Function</th>
<th>Total</th>
<th>Per Function Call</th>
<th>Improvement over \textit{ted}</th>
</tr>
</thead>
<tbody>
<tr>
<td>\textit{mtd}</td>
<td>7.4 min.</td>
<td>0.001 millisecond.</td>
<td>689x</td>
</tr>
<tr>
<td>\textit{BDist}</td>
<td>8 min.</td>
<td>0.001 millisecond.</td>
<td>637x</td>
</tr>
<tr>
<td>\textit{ted}</td>
<td>85 hr.</td>
<td>1 millisecond.</td>
<td>1x</td>
</tr>
</tbody>
</table>

- SMAP + Spatial Index / High Dimensional Index
## Benchmarks

### Execution time results

- **Queries:** 1000
- **DB:** 244668

<table>
<thead>
<tr>
<th>Function</th>
<th>Total</th>
<th>Per Function Call</th>
<th>Improvement over \textit{ted}</th>
</tr>
</thead>
<tbody>
<tr>
<td>\textit{mtd}</td>
<td>7.4 min.</td>
<td>0.001 millisec.</td>
<td>689x</td>
</tr>
<tr>
<td>\textit{BDist}</td>
<td>8 min.</td>
<td>0.001 millisec.</td>
<td>637x</td>
</tr>
<tr>
<td>\textit{ted}</td>
<td>85 hr.</td>
<td>1 millisec.</td>
<td>1x</td>
</tr>
</tbody>
</table>

- **SMAP + Spatial Index / High Dimensional Index**
OBSearch!

Open Source Metric Access Method (MAM)

- Nearest neighbor
- S-Map and P+Tree
- GPL 2.0
- http://obsearch.net
Conclusions

$mtd$ is fast, and very sensitive to changes.

- Introduced an $O(n^2)$ metric, $mtd$
  - Tuned to perform ABPM
- Our implementation is as fast as $BDist$
- Suffix trees can make $mtd$ $O(n)$, Vishwanathan (2002)
- Future work:
  - Analyze other distance functions (ABPM)
  - Comparison with other bottom-up distances
Conclusions

$mtd$ is fast, and very sensitive to changes.

- Introduced an $O(n^2)$ metric, $mtd$
  - Tuned to perform ABPM
- Our implementation is as fast as $BDist$
- Suffix trees can make $mtd$ $O(n)$, Vishwanathan (2002)

Future work:
- Analyze other distance functions (ABPM)
- Comparison with other bottom-up distances
Conclusions

*mtd* is fast, and very sensitive to changes.

- Introduced an $O(n^2)$ metric, *mtd*
  - Tuned to perform ABPM
- Our implementation is as fast as *BDist*
- Suffix trees can make *mtd* $O(n)$, Vishwanathan (2002)
- Future work:
  - Analyze other distance functions (ABPM)
  - Comparison with other bottom-up distances