8 Software Birthmark Similarity

Comparing birthmarks is necessary to identify similarities between software. If two birthmarks are similar, then the software is similar. Birthmarks may be compared to show similarity, or an alternative to showing similarity is to show dissimilarity or distance. Similarity measures and metrics exist for the different types of data such as strings, vectors, trees, graphs, etc. This chapter examines the different similarity measures and metrics for the different classes of birthmarks.

Keywords: Software similarity, birthmark similarity, distance metrics, string similarity, vector similarity, set similarity, set of vectors similarity, tree similarity, graph similarity.

8.1 Distance Metrics

Definition 8.1. *A metric on a set X is a function (known as the distance function or distance):*

$$d : X \times D \rightarrow \mathbb{N}$$

For all x, y, z in X, this function is required to satisfy the following conditions:

1. $d(x,y) \geq 0$
2. $d(x,y) = 0$ *iff x=y*
3. $d(x,y) = d(y,x)$
4. $d(x,z) \leq d(x,y) + d(y,z)$ *(triangle inequality)*

If the distance function has the properties of a distance metric then indexing and searching a database can be performed more efficiently. Therefore it is beneficial to compare software using distance functions that are metric. Examples of metric access methods are in [1-3].

8.2 String Similarity

Strings can be compared using string metrics. The Levenshtein distance between two strings defines the number of edit operations that must be performed to transform one string to the other. An edit operation includes character insertion, deletion, and substitution. Other string metrics include the Smith-Waterman algorithm which is used to perform local string alignment, or using the longest common subsequence. Optimal solutions to edit distance and alignments are normally O(n.m) where n and m are the lengths of each respective string. The solutions are typically implemented using dynamic programming. The Levenshtein distance, Smith Waterman distance and Normalized Compression Distance are all metric.

8.2.1 Levenshtein Distance

Definition 8.2. *For two strings s and t, the Levenshtein distance is measured as follows:*

D(i,0)=0 $0 \le i \le len(s)$

D(0,j)=0 $0 \le j \le len(t)$

$$D(i,j) = \min \begin{cases} D(i-1, j-1) + d(si,tj), & \textit{substitution} \\ D(i-1,j)+1, & \textit{insertion} \\ D(i,j-1)+1 & \textit{deletion} \end{cases}$$

d(i,j) is a function whereby d(c,d)=0 if c=d, 1 else.

The Levenshtein distance is metric.

Definition 8.3. *A method of normalizing the edit distance to give a similarity in [0,1] is:*

$$sim(s,t) = 1 - \frac{ed(s,t)}{\max(len(s), len(t))}$$

8.2.2 Smith-Waterman Algorithm

Definition 8.4. For two strings s and t, the Smith-Waterman similarity score is measured as follows:

D(i,0)=0 $0 \le i \le len(s)$

D(0,j)=0 $0 \le j \le len(t)$

If $a_i=b_j$ $w(a_i,b_j)$=w(match) or $a_i \ne b_j$ $w(a_i,b_j)$=w(mismatch)

$$D(i,j) = \max\begin{cases} 0 & \\ H(i-1,j-1)+w(ai,bj) & \textit{match / mismatch} \\ H(i-1,j)+w(ai,-) & \textit{deletion} \\ H(i,j-1)+w(-,bj) & \textit{insertion} \end{cases}$$

The Smith-Waterman algorithm when constructed as a distance instead of a similarity is known to be metric. The similarity algorithm is known as an optimal local string alignment.

8.2.3 Longest Common Subsequence (LCS)

Definition 8.5. For two strings X and Y, the LCS is found as follows:

$$LCS(Xi,Yi) =$$
$$\begin{cases} 0 & \textit{if } i = 0 \textit{ or } j = 0 \\ (LCS(Xi-1,Yj-1),xi) & \textit{if } xi = yj \\ longest(LCS(Xi,Yj-1),LCS(Xi-1,Yj)) & \textit{if } xi \ne yj \end{cases}$$

The similarity between two strings X and Y is defined as $|LCS(X,Y)|$

8.2.3 Normalized Compression Distance

Definition 8.6. For two strings x and y where C(x) is the length of a compressed x, the normalized compression distance (NCD) [4] is:

$$NCD(x,y) = \frac{C(x,y) - \min(C(x),C(y))}{\max(C(x),C(y))}$$

The NCD is metric.

8.3 Vector Similarity

Vector distance can be performed using metrics such as the Euclidean distance or Manhattan distance. Non metric similarity measures can include the cosine similarity which is often used in text mining.

8.3.1 Euclidean Distance

Definition 8.7. The Euclidean distance between vectors p and q is:

$$d(p,q) = \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2}$$

The Euclidean distance is metric.

8.3.2 Manhattan Distance

Definition 8.8. The Manhattan distance between vectors p and q is:

$$d(p,q) = \sum_{i=1}^{n} \left| q_i - p_i \right|$$

The Manhattan distance is metric.

8.3.3 Cosine Similarity

Definition 8.9. The cosine similarity between vectors A and B is:

$$similarity = \cos(\phi) = \frac{A \cdot B}{\|A\|\|B\|}$$

The cosine similarity is not metric.

8.4 Set Similarity

Two sets can be compared using a variety of measures. The Dice coefficient and Jaccard Index are two such measures. The Jaccard Index is not metric, but its parallel the Jaccard Distance is, which allows for efficient indexing and searching. Containment and the Tversky index are examples of asymmetric similarity measures. Because they are asymmetric, they do not qualify as metric distance functions.

8.4.1 Dice Coefficient

Definition 8.10. The Dice coefficient betweens sets A and B is:

$$s = \frac{2|A \cap B|}{|A| + |B|}$$

The Dice coefficient is not metric.

8.4.2 Jaccard Index

Definition 8.11. The Jaccard Index between sets A and B is:

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|}$$

The Jaccard Index is not metric, however the Jaccard distance is.

8.4.3 Jaccard Distance

Definition 8.12. The Jaccard distance between sets A and B is:

$$J_d(A,B) = 1 - J(A,B)$$

The Jaccard distance is metric.

8.4.4 Containment

Definition 8.13. The Containment of set B in A is:

$$C(A,B) = \frac{|A \cap B|}{|A|}$$

Containment is an asymmetric measure and therefore not metric.

8.4.5 Overlap Coefficient

Definition 8.14. The overlap coefficient between sets A and B.

$$overlap(X,Y) = \frac{|A \cap B|}{\min(|X|,|Y|)}$$

The overlap coefficient is not metric.

8.4.6 Tversky Index

Definition 8.15. The Tversky Index of sets X and Y is:

$$S(X,Y) = \frac{|X \cap Y|}{|A \cap B| + \alpha|X - Y| + \beta|Y - X|}$$

The Tversky index is an asymmetric measure and therefore not metric.

8.5 Set of Vectors Similarity

A set of vectors can be compared using the minimum matching distance [5], which constructs a minimum weight matching between pairs of vectors in each set. This distance is metric and can be evaluated in polynomial time.

8.6 Tree Similarity

Trees can be compared for equality using tree isomorphism. Ordered trees are trees such that the children of each node are in a specific sequence. Ordered trees are significantly more efficient to process than unordered trees. Approximate matching and similarity between trees can also be found using the tree edit distance [6]. The tree edit distance is metric. Alternatives to the tree edit distance include using the largest common subtree as an indicator of similarity. These are similar to the graph based version of the problem and are shown in the next section.

Definition 8.16. *The tree edit distance between two graphs* $d : T_1 \times T_2 \rightarrow \mathbb{N}$ *is the minimum number of edge or vertex insertions, deletions, and substitutions to transform one tree to the other.*

8.7 Graph Similarity

8.7.1 Graph Isomorphism

Graphs can be tested for structural equality by graph isomorphism testing. Graph isomorphism has not been demonstrated to belong to the complexity class P but it has not been proven to be in NP either.

Definition 8.17. Let $g_1 = (V_1, \alpha_1, \beta_1)$ and $g_2 = (V_2, \alpha_2, \beta_2)$ be two graphs. A graph isomorphism between g_1 and g_2 is a bijective mapping $f : V_1 \to V_2$ such that

$$\alpha_1(x) = \alpha_2(f(x)) \forall x \in V_1$$
$$\beta_1((x, y)) = \beta_2((f(x), f(y))) \forall (x, y) \in V_1 \times V_1$$

If $V_1 = V_2 = 0$ then f is called the empty graph isomorphism

8.2.2 Graph Edit Distance

A harder problem is calculating the approximate similarity or distance between two graphs. The two main approaches are the graph edit distance and using the maximum common subgraph. The graph edit distance is metric. These problems are proven not to belong to P. However, polynomial time approximate solutions exist to the graph edit distance.

Definition 8.18. The graph edit distance $d : G_1 \times G_2 \to \mathbb{N}$ between two graphs is the minimum sum cost of basic edit operations to transform one graph to another.

8.2.3 Maximum Common Subgraph

Definition 8.19. Let $g_1 = (V_1, \alpha_1, \beta_1)$ *and* $g_2 = (V_2, \alpha_2, \beta_2)$ *be two graphs and* $g_1' \subseteq g_1, g_2' \subseteq g_2$. *If there exists a graph isomorphism between* g_1' *and* g_2'*, then both* g_1' *and* g_2' *are called a common subgraph of* g_1 *and* g_2.

Definition 8.20. Let g_1 *and* g_2 *be two graphs. A graph g is called the maximum common subgraph of* g_1 *and* g_2 *if g is a common subgraph of* g_1 *and* g_2 *and there exists no other common subgraph of* g_1 *and* g_2 *that has more nodes than g.*

Definition 8.21. The distance between graphs g_1 *and* g_2 *is:*

$$d(g_1, g_2) = \frac{|MCS(g_1, g_2)|}{|g1|} \quad \textit{where } |g| = |V| + |E|$$

Definition 8.22. The distance between graphs g_1 *and* g_2 *is:*

$$d(g_1, g_2) = \frac{|MCS(g_1, g_2)|}{\max(|g_1|, |g_2|)} \quad \textit{where } |g| = |V| + |E|$$

An approximate or inexact maximum common subgraph is also possible.

Definition 8.23. The graph edit distance between two graphs $d : G_1 \times G_2 \to \mathbb{N}$ *is the minimum number of edge or vertex insertions, deletions, and substitutions to transform one graph to the other.*

Distances based on the maximum common subgraph are not metric.

References

1. Peter NY Data structures and algorithms for nearest neighbor search in general metric spaces. In: Proceedings of the fourth annual ACM-SIAM Symposium on Discrete algorithms, Austin, Texas, United States, 1993. Society for Industrial and Applied Mathematics, pp 311-321

2. Vieira MR, Chino FJT, Traina C, Jr., Traina AJM DBM-Tree: A Dynamic Metric Access Method Sensitive to Local Density Data. In: Brazilian Symposium on Databases, Brazil, 2004. pp 163-177
3. Paolo C, Marco P, Pavel Z (1997) M-tree: An Efficient Access Method for Similarity Search in Metric Spaces. Paper presented at the Proceedings of the 23rd International Conference on Very Large Data Bases,
4. Cilibrasi R, Vitányi PMB (2005) Clustering by compression. Information Theory, IEEE Transactions on 51 (4):1523-1545
5. Brecheisen S (2007) Efficient and Effective Similarity Search on Complex Objects. Ludwig-Maximilians-Universität München,
6. Bille P (2005) A survey on tree edit distance and related problems. Theoretical Computer Science 337 (1-3):217-239