Order Theoretical Semantic Recommendation

Cliff Joslyn[1], Emilie Hogan[1], Patrick Paulson[1],
Elena Peterson[1], Eric Stephan[1], and Dennis Thomas[1]

Pacific Northwest National Laboratory, *firstname.lastname*`@pnnl.gov`

Abstract. Mathematical concepts of order and ordering relations play multiple roles in semantic technologies. Discrete totally ordered data characterize both input streams and top-k rank-ordered recommendations and query output, while temporal attributes establish numerical total orders, either over time points or in the more complex case of start-end temporal intervals. But also of note are the fully partially ordered data, including both lattices and non-lattices, which actually dominate the semantic strcuture of ontological systems. Scalar semantic similarities over partially-ordered semantic data are traditionally used to return rank-ordered recommendations, but these require complementation with true metrics available over partially ordered sets. In this paper we report on our work in the foundations of order measurement in ontologies, with application to top-k semantic recommendation in workflows. We conclude that true ordered set metrics are strongly preferable to traditional semantic similarities.

1 Introduction

Order and sequence occur frequently in the semantic web and semantic technologies, but in different contexts and manners. First, data themselves can be ordered, for example in a time sequence or order of arrival to a data stream. When time stamps are available, a mathematical "total" (or "linear") order arises. If events can occur simultaneously, then a "weak order" is needed so as to allow "ties". And if event occurences can extend over time, then "interval orders" are needed to reflect the interactions between start and stop times of possibly overlapping events with different durations.

But data can be ordered from other ways and sources. In fact, the heart of semantic technology may be considered to be class structures, which sit at the cores of ontologies as taxonomic semantic categorization hierarchies. Such structures are mathematically "partial orders", that is tree-like hierarchies supporting multiple inheritance, including lattices. Note that this is the most general case, as all total orders are weak orders, which are all interval orders, which are all partial orders. Also, all lattices are partial orders.

Whether input data or semantic categories are totally, weakly, intervally, or partially ordered, in any semantic selection task there is a need to rank-order the resulting calculations, in order to recommend particular choices to the user. And the algorithmic methods used to make such recommendations must measure the data in a way which acccommodates and respects its underlying mathematically ordered nature, perhaps up to and including being fully partially ordered.

Our interest is 1) the proper use of metrics in partially-ordered data in order to 2) make top-k rank-ordered recommendations in semantic systems. We report on the performance of order metrics in a test analytic semantic recommendation task within multi-domain scientific workflows as part of the Signature Discovery Initiative[1] (SDI) [1] hosted by the Pacific Northwest National Laboratory (PNNL). Comparing true order metrics against traditional, non-metric semantic similarities, we conclude that within our test knowledgebase, the rank-order recommendations provided by partial order metrics are strongly preferable.

2 Workflows Using the Sequence Analysis Knowledgebase

We now introduce our test knowledgebase and use case within which we make recommendations. Our use case concerns identifying signatures within generic sequential data objects from multiple domains. While alignment of biosequences has been in use for decades, it has only been recently recognized as a significant method for analyzing non-biological sequences [8]. The overall workflow is shown in Fig. 1. In the initial "alphabet construction" phase, members of an arbitrary data set "input corpus" are translated into a linear sequence of exchange primitives here called "symbols". Each primitive is then assigned a unique character called a "token" drawn from a limited alphabet using a "map file", a data structure mapping symbols to tokens via a lossy encoding. This encoding of data to a limited character set allows linear structure to be maintained for analysis while reducing the vocabulary of characters to a computationally manageable number. Sequences encoded as these token strings are represented in the FASTA file format[2] as used in bio-informatics. We call encoding outputs "pseudo-proteins", as they are represented in FASTA files, similarly to real proteins, even if they are representing any arbitrary input. Once encoded, pseudo-proteins are BLASTed against themselves, then hierarchically clustered. Additional stages involve multiply aligning pseudo-proteins to generate sequence features for downstream detection, but these do not concern the current effort being reported on here.

While this workflow targets any sequential data object which can be translated and encoded into a pseudo-protein, the obvious cases are from computational biology. Here inputs can be either gene sequences, with a map file implementing the genetic code of codons to amino acids; or protein sequences directly (with a null map file). But as shown in Table 1, we also include cyber objects, in particular disassembled executable files with actual processor opcodes as symbols; and system log files, with log events (e.g. "login" or "logoff") as symbols.

The goal for recommendation is to rank order available choices of data objects and component connections in terms of semantic appropriateness, e.g. not recommending a cyber objet for a genetics task. We developed a Sequence Analysis Ontology (SAO) to model our use case, including the types of data objects passed in to and out of the workflow components, and the operations performed by the workflow components. The SAO, and our methodology, uses object relations to

[1] http://signatures.pnnl.gov
[2] http://zhanglab.ccmb.med.umich.edu/FASTA/

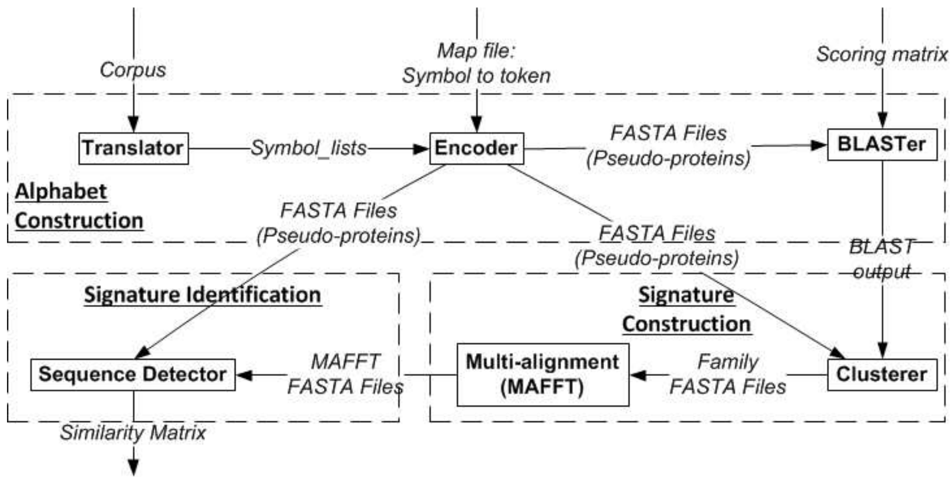


Fig. 1. Sequence analysis workflow.

represent semantic relations of interest. For analytical and testing purposes, we augment the SAO with OWL individuals to create a Sequence Analysis KnowledgeBase (SA-KB) as a static representation of one time state of the overall workflow environment. Each OWL individual thus represents a specific data object or component port which may be available at any one time.

Domain	**Subdomain**	**Symbols**
Biology	Genomics	Codons
	Proteomics	Amino acids
Cyber	Executables	Opcodes
	Log files	Log events

Table 1. Domains and symbols used in sample workflows.

SA-KB statistics are shown in Table 2. Note that we regard the class hierarchy, the core of the ontology, as a (partially) ordered data object of 157 classes organized into nine levels. But it's not a tree, with an average of 1.68 parents per class. Thus methods in order theory are necessary to properly measure it.

Fig. 2 show an illustrative portion of the SA-KB around the individual `FASTAs_Linux`, which is a KB entry for a particular FASTA file produced from encoding a Linux executable file. `FASTAs_Linux` is a member of the class `FASTA_File_Set` (classes shown in ovals, dashed lines show class membership). Below we will denote $C(x)$ for the classes (potentially multiple) of an individual x, so here we have $C(\texttt{FASTAs\_Linux}) = \{\texttt{FASTA\_File\_Set}\}$.

`FASTAs_Linux` is connected to other individuals (shown in boxes), representing data objects on which it depends semantically, by object properties. For

Property	Value
# Classes (asserted)	157
# Classes (inferred)	168
Average # children/class	2.02
Average # parents/class	1.68
Class hierarchy height	9
# Object properties	31
# Individuals	116

Table 2. SA-KB statistics.

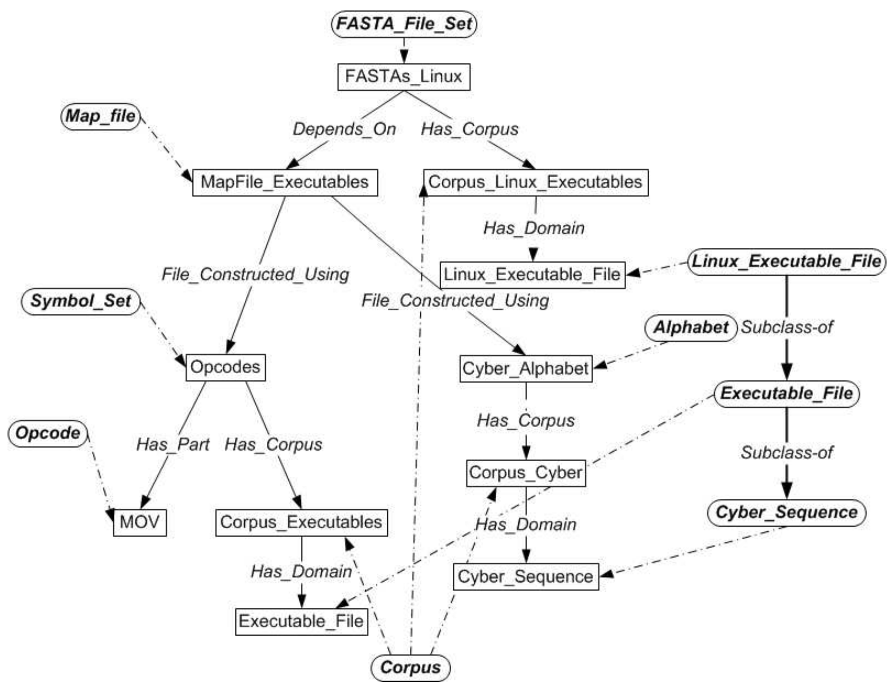


Fig. 2. A portion of the SA-KB around the `FASTAs_Linux` individual. Individuals are in boxes, classes are in ovals. Dashed lines indicate class membership, all others are object properties or class hierarchy.

example, FASTAs_Linux was encoded with a particular mapfile represented by MapFile_Executables, of class Map_File, and is thus linked to it by Depends_On. MapFile_Executables effects a mapping from opcode symbols to tokens, and so is itself related by File_Constructed_Using to two individuals: Opcodes, of class Symbol_Set, representing the symbols of opcodes; and Cyber_Alphabet of class Alphabet, representing the token set used by the cyber domain.

Recommendation proceeds when, given a fixed target individual (e.g. the FASTA file FASTAs_Linux) at a particular point in the workflow (e.g. as input to the BLASTer), calling for an object of a given type (e.g. scoring matrix), which of many candidates are recommended? We wish to see recommendations rank ordered by being most semantically similar, for example the matrix SM_Executable used for executable files being highly ranked, while SM_Genetics used for genetic sequences loweest ranked.

3 Ontology Metric Annotation Methodology

We now describe our semantic annotation metric methodology. Given a class hierarchy, we have a number of metrics to measure pairwise distances between classes. We collect the sets of classes associated with individuals, and can then measure the overall extent of those sets of classes within the ontology, and using a Hausdorff set distance can measure the overall metric similarity between them. This supports the rank-ordered recommendation of the most closely related objects of the same data type otherwise.

3.1 Distances in Ordered Semantic Structures

Real-world semantic hierarchies such as subsumption and composition are usually cast as Directed Acyclic Graphs (DAGs) on a finite set of classes P. They are typically bounded above with a top element $\top \in P$, with $\forall a \in P, a \le \top$; can have (a moderate amount of) multiple inheritance; and branch downward very strongly. These are best represented mathematically as partially ordered sets (posets) $\mathcal{P} = \langle P, \le \rangle$ [3], where $\le \subseteq P^2$ is a reflexive, anti-symmetric, and transitive binary relation. Each DAG as described above uniquely determines a bounded poset $\mathcal{P}$ by taking its transitive closure and including a bottom bound $\perp \in P$ such that $\forall a \in P, \perp \le a$.

The first thought towards a metric in a graphical structure such as a semantic hierarchy is as the length of the minimum undirected path

$$d_{MP}(a,b) = \min_{p \in \text{undirected paths}(a \leftrightarrow b)} |p|.$$

But such a distance ignores both the directional nature and the level structure of the semantic hierarchy. Instead, the knowledge systems literature has focused on **semantic similarities** [2, 10] to perform a similar function. These are available when $\mathcal{P}$ is equipped with a probabilistic weighting function $p \colon P \to [0,1]$, with $\sum_{a \in P} p(a) = 1$. While p can be derived, for example, from the frequency with which terms appear in documents [4], or which genes are annotated to bio-ontology nodes [5], the simplest approach is to uniformaly weight each $a \in P$ proportionally, so that $\forall a \in P, p(a) \equiv \pi$, where $\pi = \frac{1}{|P|}$.

This then allows the definition of the **information content** of a node as

$$IC(a) = -\log_2\left(\pi \sum_{b \leq a} p(b)\right) = -\log_2\left(\pi |\downarrow a|\right),$$

where $\downarrow a := \{b \leq a\}$ is the **downset of** a, then set of its descendants, so that $|\downarrow a|$ is the number of such descendants. Then the **information content of the most informative common ancestor** of $a, b \in P$ is

$$M(a,b) := \max_{c \in \uparrow a \cap \uparrow b} IC(c).$$

where $\uparrow a$ is defined analogously to $\downarrow a$ as the **upset of** a.

This then allows us to define the first of three standard semantic similarity-based (SS-based) distances, the **Jiang-Conrath** distance [10], as

$$d_{JC}(a,b) = IC(a) + IC(b) - 2M(a,b).$$

The other two follow from the **Resnik** and **Lin similarity measures** [10], after normalization, to dervie corresponding distances as:

$$d_R(x,y) = 1 + \frac{M(a,b)}{\log_2(\pi)}, \qquad d_L(x,y) = 1 - \frac{2M(a,b)}{IC(a) + IC(b)}.$$

While all of these, including d_{MP}, are somewhat standard metrics, our purpose is more general, since we may not have such a weighting function available, and semantic similarities are not required to be metrics satisfying the triangle inequality. In seeking out the proper mathematical grounding, we turn to **order metrics** [6, 9] which can use, but do not require, a quantitative weighting. For details about order metrics built from isotone and antitone lower and upper semimodular functions on ordered sets, see [9]. In this work, we use the **cardinality-based distances**. First we have the **upper** and **lower** distances:

$$d_u(a,b) = |\uparrow a| + |\uparrow b| - 2|\uparrow a \cap \uparrow b|, \qquad d_l(a,b) = |\downarrow a| + |\downarrow b| - 2|\downarrow a \cap \downarrow b|,$$

Having two distances, upper and lower, may appear arbitrary or unfortunate, but they behave differently.[3] It may at first appear to be more natural to use upper distance, since we're then "looking upwards" towards the top bound $\top \in P$. But when $\mathcal{P}$ is top-bounded, strongly down-branching, and with multiple inheritance (as in our cases), then it might be argued that it is preferable to use lower distance, since up-sets are typically very small and narrow, frequently single chains; where down-sets are large, branching structures. Additionally, this allows siblings deep in the hierarchy to be closer together than siblings high in the

[3] Note that it can be the case that the upper distance $d_u(a,b)$ is the same as $d_{MP}(a,b)$, but this is only required to be true when $\mathcal{P}$ is an upper-bounded tree: path length and all these other metrics are generally unrelated.

hierarchy (this will be demonstrated below). This is considered valuable, for example, where e.g. "mammal" and "reptile" are considered farther apart than "horse" and "goat". In practice, these can be combined. Defining the **hourglass** of a node $\Xi a = \uparrow a \cup \downarrow a$, we then have the hourglass-based measure:

$$d_{TH}(a,b) = |\Xi a| + |\Xi b| - 2|\Xi a \cap \Xi b|$$

d_{TH} is a direct analog of d_l and d_u, but it is not a true metric.

We need to normalize distance to the size of the structure, so that we are measuring the relative proportion of the overall structure two nodes are apart, or in other words, what proportion of their potential maximum distance. These normalized forms are (all $\in [0,1]$):

$$\bar{d}_u(a,b) := \frac{d_u(a,b)}{|P|-1}, \quad \bar{d}_l(a,b) := \frac{d_l(a,b)}{|P|-1}, \quad \bar{d}_{TH}(a,b) = \frac{d_{TH}(a,b)}{|P|-2}.$$

3.2 Annotation Collection and Measurement

The metrics above measure distances between classes in an ontology, and in particular, for any two individuals x, y in our KB, it can measure the distance between its classes $C(x), C(y)$. But in order to compare the *entire* semantic context of an individual x, we need to identify not just its classes $C(x)$, but additionally the classes $C(z), C(w)$ of any individuals z, w which x is related to, that is, to which x is connected via object properties. We may not wish to invoke *all* object properties in such connections, but perhaps particular ones. And these should be transitive, so that through transitive closure we can find both directly and indirectly related individuals.

The SAO has two transitive object relations , `sao:depends_on` and `sao:has_part`, to play this role. But we don't wish to restrict ourselves to just `sao:depends_on` and `sao:has_part` proper, but rather all of their sub-relations as well, for example `sao:has_corpus`, which inherits from `sao:depends_on`. So, for an individual x, let $A(x)$ be the class annotations of x, defined as: $A(x)$ is the set of all classes which either x or some other individual y, to which x is either directly or indirectly linked by a particular, transitive object property, or some sub-relation of one, are members of. From our example from Fig. 2, we have:

$$A(\texttt{FASTAs\_Linux}) = \{ \texttt{sao:alphabet, sao:corpus, sao:cyber\_sequence,} \\ \texttt{sao:executable\_file, sao:FASTA\_file\_set, sao:linux\_executable\_file,} \\ \texttt{sao:map\_file, sao:opcode, sao:symbol\_set} \} \tag{1}$$

Consider an individual x with classes $C(x)$ and full set of class annotations $A(x)$. We wish to have measures of $A(x)$ for various x, and to compare $A(x)$ and $A(y)$ for different individuals x, y. We first wish to capture the "spread" of $A(x)$ within the ontology. Given a distance d, we use

$$E_d(x) = \frac{\sum_{C_1 \in A(x)} \sum_{C_2 \in A(x)} \bar{d}(C_1, C_2)}{\binom{|A(x)|}{2}} \in [0,1],$$

as the **extent** of x, which is normalized by the number of pairs of classes drawn from $A(x)$. Next, given two individuals x, y with classes $C(x), C(y)$ and annotations $A(x), A(y)$, then we are interested in comparing the aggregate distance $d(A(x), A(y))$ between those *sets* of annotations. To do so, we use a Hausdorff distance [7] as a standard method. Given a distance d, then we have

$$H_d(x,y) = \max\left\{ \max_{C \in A(x)} \min_{D \in A(y)} \bar{d}(C,D), \max_{D \in A(y)} \min_{C \in A(x)} \bar{d}(C,D) \right\} \in [0,1].$$

If d is a true distance function on $\mathcal{P}$ (positive definite, symmetric, satisfies triangle inequality) then H_d is also a distance function.

4 Metrics Behavior

The left side of Fig. 3 shows distributions of the $\binom{169}{2} = 14,196$ distances $d(C_1, C_2)$ between the 168+1 (for the bottom node) classes in the SA-KB. The right side shows the distributions of the $\binom{116}{2} = 6,670$ Hausdorff distances $H_d(x, y)$ between the individuals. The left side of Fig. 4 shows the distribution of the extents $E(x)$ of the individuals, while the right side shows the distribution of $|A(x)|$, the number of classes annotating each individual. We note the following:

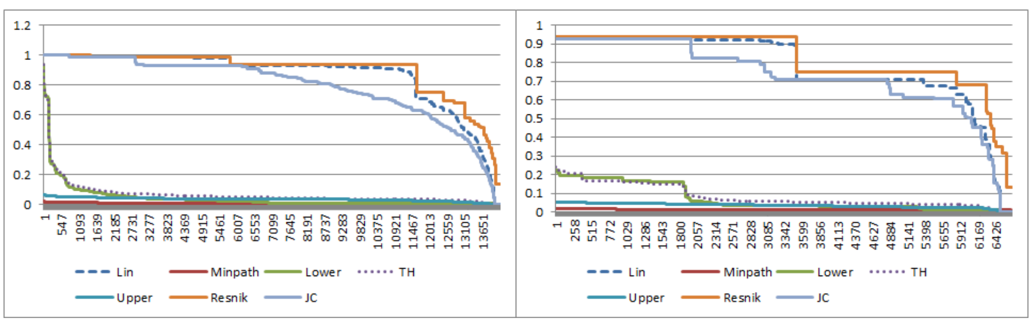


Fig. 3. (Left) Distances $d(C_1, C_2)$ between classes C_1, C_2. (Right) Hausdorff distance $H_d(x, y)$ between annotation sets of individuals $A(x), A(y)$.

- In Figs. 3 and 4, the distributions shown are not mutually correlated, rather each is sorted separately in descending order, and then overlaid in the figure.
- The SS-based distances are convex, and of the SS-based distances, JC is likely the best in the sense of the most discriminative, although it has large regions in the high distances where it fails to distinguish class pairs.
- For the poset distances, lower dominates upper (as discussed above), and is concave. When combined with lower, the resulting hourglass distance d_{TH} is less concave and more linear, providing a wide range of discriminatory values, although less in the lower values. Below we will show that SS-based distances provide poor rank orderings for recommendations, wo while we will return to this issue, in the sequel d_{TH} will be considered preferentially.

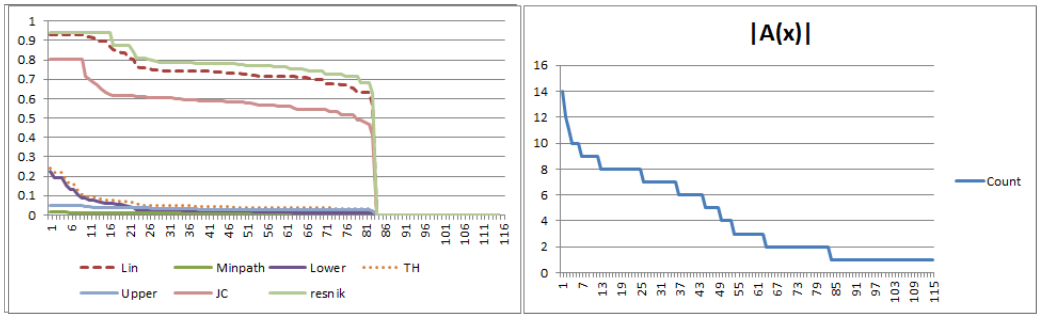


Fig. 4. (Left) Extents $E(x)$ of the individuals. (Right) The number of classes $|A(x)|$ annotating each individual.

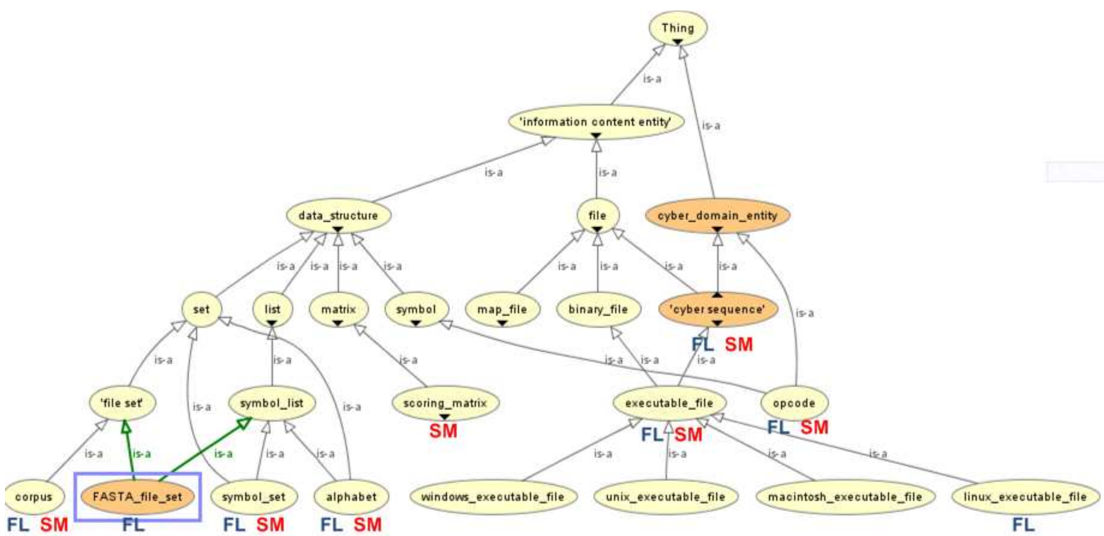


Fig. 5. A portion of the inferred SAO class hierarchy showing class annotations for just `FASTAs_Linux` (marked with 'FL' in blue), just `SM_Executable` (marked with 'SM' in red), and both (marked with both).

- The Hausdorff distances are a "selection" of the values appearing in the class distances, in virtue of its max-min composition, effectively acting as a coarsened representation.
- Extents $E_d(x)$ and counts $|A(x)|$ are shown separately, and our normalization factor is effective, with generally small correlations coefficients between E_d and $|A|$, especially for the poset distances, with Pearson correlation $\rho(E_{TH}(x), |A(x)|) = 0.23$, while for the SS-based distances $\rho > 0.5$.

5 A Detailed Recommendation Example

In our example in Fig. 2, from Equ. 1 we have that `FASTAs_Linux` is annotated by $|A(x)| = 9$ classes and, which are marked by 'FL' in blue in Fig. 5, and has extent $E_{TH}(x) = .0381$. This is relatively low extent, but a relatively high count, as can be seen in Fig. 4.

Referring back to Fig. 1, consider that we are at the portion of the workflow where a BLASTer component has been identified, and connected to its input FASTA file using the output of the upstream encoder component. In order to complete the instantiation of the BLASTer component, we need to identify its one lacking input, namely a scoring matrix, and we need one which is as semantically compatible as possible. The five available scoring matrices are shown in Table 3, each of which has its own set of annotations, as shown in Table 4, which is arranged with gaps to show where annotations sets overlap and differ.

H_d(`FASTAs_Linux`, y)	Minpath	TH	Resnik	Lin	J+C
sao:SM_Executable	0.012	0.041	0.749	0.676	0.607
sao:SM_Server_Logfiles	0.012	0.053	0.749	0.676	0.607
sao:SM_Logfiles	0.012	0.059	0.749	0.676	0.607
sao:SM_Genetics	0.012	0.148	0.749	0.676	0.607

Table 3. Hausdorff distances between the annotating classes from `FASTAs_Linux` and all the scoring matrices in SA-KB, by different distances d. Sorted up by TH.

Consider the one example scoring matrix `SM_Executable`. On the one hand, it differs from `FASTAs_Linux` in that it is annotated to `sao:scoring_matrix`, since it is a scoring matric, but to neither `sao:FASTA_file_set`, since it is not a FASTA file, nor to `sao:linux_executable_file`, since it can be used with any executable, not being specific to any particular operating system. The right side of Fig. 5 shows this annotation set in the dual 'FL SM' markings.

In measuring the difference in the annotation sets A(`FASTAs_Linux`) and A(`SM_Executable`), the first thought is to cast them as sets X, Y and measure their symmetric difference $|X \Delta Y| = |(X \setminus Y) \cup (Y \setminus X)|$, the number of different elements. We have $|A(\texttt{FASTAs\_Linux}) \Delta A(\texttt{SM\_Executable})| = 3$ for these three differing classes. Table 4 shows Δ for all the scoring matrices, along with H_{TH}, the Hausdorff for the hourglass measure. But we actually observe that H_{TH} is strongly preferred to Δ, in that it not only gets the correct rank ordering, but can also distinguish `SM_Server_Logfiles` as being preferable to `SM_Logfile`, in virtue of the fact that it doesn't just count the differences, it can weight them by semantic distance, in this case that `server_log_file` is more specific

x =FASTAs_Linux	y =SM_Executable	y=SM_Server_Logfiles	y=SM_Logfiles	y =SM_Genetics
$H_{TH}(x, y)$	0.041	0.053	0.059	0.148
$A(x)\Delta A(y)$	3	8	8	12
$E(y)$	0.042	0.038	0.041	0.076
$\|A(y)\|$	8	8	8	9
				edam:data_2976
sao:alphabet	sao:alphabet	sao:alphabet	sao:alphabet	sao:alphabet
				sao:codon
sao:corpus	sao:corpus	sao:corpus	sao:corpus	sao:corpus
sao:cyber_sequence	sao:cyber_sequence	sao:cyber_sequence	sao:cyber_sequence	
sao:executable_file	sao:executable_file			
sao:FASTA_file_set				
				sao:file
				sao:gene_sequence
sao:linux_executable_file				
		sao:log_event	sao:log_event	
			sao:log_file	
sao:map_file	sao:map_file	sao:map_file	sao:map_file	sao:map_file
sao:opcode	sao:opcode			
	sao:scoring_matrix	sao:scoring_matrix	sao:scoring_matrix	sao:scoring_matrix
		sao:server_log_file		
sao:symbol_set	sao:symbol_set	sao:symbol_set	sao:symbol_set	sao:symbol_set

Table 4. Annotation sets for `FASTAs_Linux` vs. all scoring matrices.

than `log_file`. Finally `SM_Genetics` is the least recommended scoring matrix, coming from another domain entirely, it is most divergent in both Δ and H_{TH}.

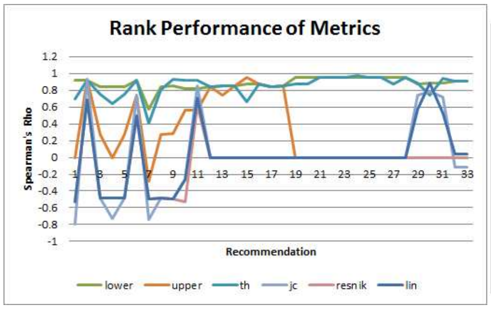
Considering the other metrics available, and referring back to Table 3, we can see the Hausdorffs for other distances, and that H_{TH} is preferable to any of the SS-based measures, in that it can distinguish all of these examples, and rank-orders them correctly, where the SS-based distances cannot.

6 Overall Comparison

Broadening from our one example, we can consider an evaluation of the performance of order metrics relative to SS-based distances. To accomplish this, we considered all possible recommendations of a candidate individual against a fixed source individual in the four cases implied by the upper parts of our workflow in Fig. 1, specifically: 1) recommend symbol sets (candidates) against a mapfile (target); 2) recommend a map file against a fixed symbol set; 3) a scoring matrix against a FASTA file; and finally 4) a FASTA file against a scoring matrix.

There were 33 targets across all four groups, and for each we independently established a "ground truth" for the ordering of the candidates. For example, in Table 3, ground truth for candidate scoring matrices against the FASTAs_Linux target is the rank ordering shown, but with `Server_Logfiles` and `Logfiles` reversed. Then for each recommendation, we measured the Spearman rank correlation r_s (averaging ranks over ties) for ground truth using each of the metrics.

Results are shown in Fig. 6. Note that $r_s = 1$ indicates complete agreement on rank order, while $r_s = -1$ is a complete *reversal* of that order. Also, r_s does not exist when all scores are the same, as exemplified by the SS-based metrics in Table 3. This was *always* true for minpath d_{MP}, which was thereby eliminated, and was frequently true for the SS-based distances, as can be seen in Fig. 3.



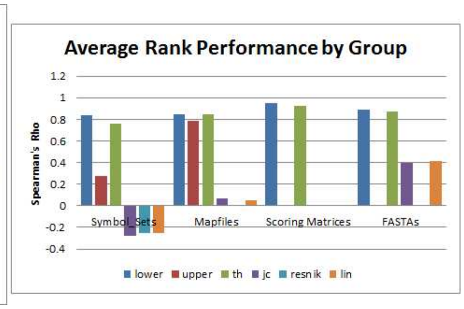


Fig. 6. (Left) Spearman's rank correlation r_s for each of the 33 recommendations; (Right) Average r_s for each group of recommendations.

While our results only hold within our test use case and KB, these were constructed independently of our evaluation. Poset metrics, especially lower and hourglas, provided strong performance for top-k semantic ranking, with high rank correlations above 80%. They were almost always preferable to SS-based metrics, and sometimes remarkably so, including negative r_s values for some SS-based metrics.

Acknowledgments

This research is part of the Signature Discovery Initiative (http://signatures.pnnl.gov) at Pacific Northwest National Laboratory. It was conducted under the Laboratory Directed Research and Development Program at PNNL, a multi-program national laboratory operated by Battelle for the U.S. Department of Energy.

References

1. N Baker, JL Barr, G Bonheyo, CA Joslyn, K Krishnaswami, M Oxley, R Quadrel, LH Sego, MF Tardiff, AS Wynne. Research towards a systematic signature discovery process. In *Wshop. on Signature Discovery for Intelligence and Security, IEEE Intelligence and Information 2013*, 2013.
2. Alexander Butanitsky and Graeme Hirst. Evaluating wordnet-based measures of lexical semantic relatedness. *Computational Linguistics*, 32:1:13–47, 2006.
3. BA Davey and HA Priestly. *Introduction to Lattices and Order*. Cambridge UP, Cambridge UK, 1990.
4. C Fellbaum, ed. *Wordnet: An Electronic Lexical Database*. MIT Press, 1998.
5. PW Lord, Robert Stevens, A Brass, and CA Goble. Investigating semantic similarity measures across the gene ontology: the relationship between sequence and annotation. *Bioinformatics*, 10:1275–1283, 2003.
6. B Monjardet. Metrics on partially ordered sets - a survey. *Discrete Mathematics*, 35:173–184, 1981.
7. Munkres. James; Topology (2nd edition). *Prentice Hall*, 280–281., 1999.
8. C. Oehmen et al. An organic model for detecting cyber events. In *Proc. Sixth Annual Workshop on Cyber Security and Information Intelligence Research*, 2010.
9. Chris Orum and Cliff A Joslyn. Valuations and metrics on partially ordered sets. Technical report, 2009. http://arxiv.org/abs/0903.2679v1.
10. C Pesquita, D Faria, A Falcão, P Lord, FM Couto. Semantic similarity in biomedical ontologies. *PLOS Computational Biology*, 5:7, 2009.