Computing Minimal Signature Coverage for Description Logic Ontologies

David Geleta Terry R. Payne Valentina Tamma

Abstract

An ontology *signature* (set of entities used to define terms in the ontology) may facilitate the expression of more than its constituent concept, role and individual names, since *rewriting* permits defined entities to be replaced by syntactically different, albeit semantically equivalent definitions. A signature can support and improve a variety of semantic interoperability scenarios, especially when only a restricted subset of terms is available to facilitate (cover) a knowledge-based task, or when it is beneficial to minimise the size of the cover set. Identifying whether a given signature permits the definition of a particular entity is a well-understood problem, while determining the smallest (minimal) signature that covers a set of entities (i.e. a task signature) poses a challenge: the complete set of alternative definitions, or even just their signature, needs to be obtained, and all combinations of such definition signatures need to be explored, for each of the entities under consideration. In this paper, we present and empirically evaluate our novel approach for efficiently computing an approximation of *minimal signature cover sets*.

1 Introduction

Ontologies in computer science provide a reference vocabulary (*signature*) for some domain of interest, where the meaning of vocabulary members (*entities*) is defined inductively in terms of other entities [1]. In definitorially complete Description Logics ontologies, defined ontological entities are permitted to be expressed (rewritten) into syntactically different, but semantically equivalent forms [14], thus it is possible to convey the meaning of an entity without using the actual entity name. Therefore, a signature may enable the expression of not only its asserted concept, role and individual names, but also those defined entities whose definition is permitted with the given signature.

A carefully composed signature may support and enhance a variety of semantic interoperability scenarios, such as ontology alignment. Semantic interoperability between individually designed ontologies is typically hindered by heterogeneity, as distinct ontologies may differ in their vocabularies and in the meaning they associated with particular entities [3]. Ontology matching resolves heterogeneity between different ontologies by producing an alignment, i.e. a set of correspondences that describe relationships between semantically related entities of distinct ontologies. However, as alignments are typically incomplete, providing only a partial coverage of an ontology vocabulary, often only a *restricted signature* is available to support semantic interoperability. Moreover, in ontology alignment negotiation [9, 11, 12], where an alignment is required to be mutually acceptable for all interacting parties and it is the product of a negotiation process, it is beneficial to *minimise* the considered correspondences (i.e. the aligned part of an ontology signature), in order to reduce the overall cost of the process, and to support emerging constraints (privacy, confidentiality, etc.).

F. van Harmelen et al. have shown [15], that semantic interoperability tasks (amongst other parameters) are characterised by signatures, thus the prerequisite for any knowledge-based task is that it must be *covered* by the signature which is available for the party who intends to carry out the given task. In order to determine whether a given *task signature* is coverable by an available, *restricted signature*, each entity of the task signature must be individually examined; an entity is considered to be covered, either if it appears in the restricted signature, or if it is rewritable using only the restricted signature members. Although task coverage is trivial to establish, determining the smallest, minimal signature that covers a given task signature poses a challenge, as the complete set of rewriting forms (definitions) need to be known, and all combinations of such definitions are required to be explored, for each entity in question.

In our previous work [4], we have presented a pragmatic approach to computing the complete set of *definition signatures* (or DSs, i.e. sets of entities that permit the rewriting of defined concepts or roles) for a given ontology, by exploiting the Beth definability property [14], a well-known property from classical logic which permits the notion of rewriting. In this paper, we introduce and empirically evaluate a novel algorithm, which by assuming the a priori obtained complete set of definition signatures, can efficiently compute an approximation of the smallest entity combination (minimum signature coverage) that covers a given task signature. The remainder of this paper is organised as follows: Section 2 presents the notion of Beth-definability, and recaps our previous work concerning the notion and computation of definition signatures, furthermore, it discusses the set coverage problem family, which relates closely to the signature coverage problem, and as has inspired our solution. Section 3 introduces and characterises the signature coverage problem. Section 4 presents our novel approach. Section 5 reports on the empirical evaluation, including the experiment framework, methodology and the results. Section 6 concludes the paper.

2 Background

In this paper, we assume familiarity with basic notions of Description Logics [1] and the Web Ontology Language [6] (OWL). The *vocabulary* of a DL ontology consists of the (disjoint) union of the countably infinite sets of concept names, role names and individual names, where an *entity* e is either a concept, role or an individual. A *signature* is an arbitrary set of role and concept names,

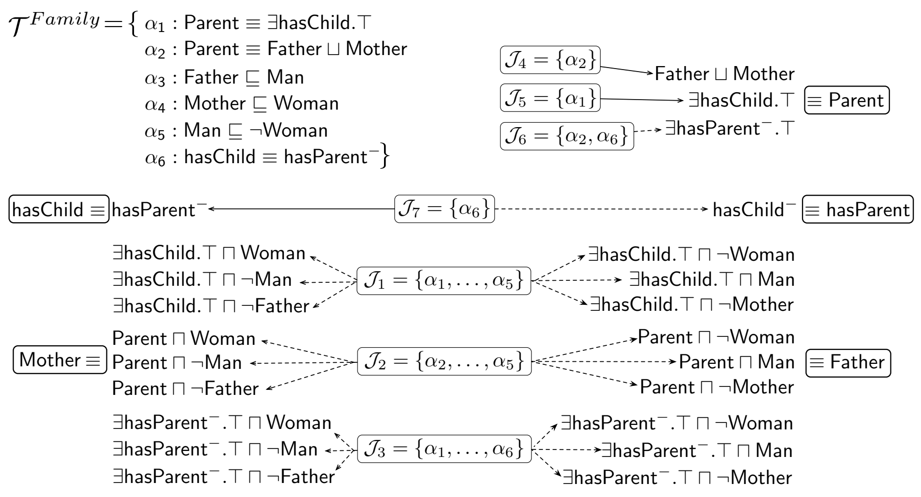


Figure 1: This small ontology describes a family domain. Concepts Mother and Father are only implicitly defined in $\mathcal{T}^{Family}$, hence these are also explicitly definable, while concept Parent is both explicitly and implicitly defined in $\mathcal{T}^{Family}$, as shown by their definition axioms. Each definition axiom is explained by a justification ($\mathcal{J}_1 - \mathcal{J}_7$), where dashed line denotes implicit, normal line denotes explicit definability.

and individuals; by $\mathsf{Sig}(\mathsf{C})$ we denote the signature of the a complex concept C, while $\mathsf{Sig}(\mathcal{T})$ denotes the signature of a TBox. In this paper, Σ refers to a *definition signature* (DS) or its minimal variant (MDS), i.e. the set of entities that implicitly define a given concept or role.

2.1 Beth Definability

A DS is used to characterise *implicitly definable* concepts in terms of their explicit definability, by exploiting *Beth definability theorem*. The theorem, initially studied for first-order logic [2], states that a concept is *implicitly definable* with respect to a theory if and only if it is also *explicitly definable*. Given that explicit definability implies implicit definability, the Beth definability property holds for some logic language $\mathcal{L}$ if the converse also holds, i.e. if implicit definability implies explicit definability. Consequently, if a term is implicitly defined then it is always possible to define it explicitly. As there are several variants of Beth definability [14], we focus on *Projective Beth definability* which is a stronger formulation [7] with the ability to specify a signature, thus permitting us to restrict the vocabulary that can be used in definitions. Beth definability has also been studied in the context of DLs [14], where it has been used to compute explicit definitions based on implicit definitions. We thus assume a general DL language $\mathcal{L}$ for which the Beth definability property holds. We define an explicit definability concept as:

Definition 1 (Explicitly defined concept) *Let* C *be a concept name, and* $\mathcal{T}$

a TBox, where $\mathsf{C} \in \mathsf{Sig}(\mathcal{T})$*.* C *is explicitly defined under* $\mathcal{T}$*, if and only if there is an axiom* $\alpha : \mathsf{C} \equiv \mathsf{D}$*, such that* $\alpha \in \mathcal{T}$*, where* D *is either a concept name in* $\mathcal{T}$*, or a complex concept such that* $\mathsf{Sig}(\mathsf{D}) \subseteq \mathsf{Sig}(\mathcal{T}) \setminus \{\mathsf{C}\}$*.*

For example, let us consider $\mathcal{T}^{Family}$, a small $\mathcal{ALC}$-TBox describing the family domain, shown in Figure 1 (upper). The concept Parent, defined by the axiom α_1 or α_2, is the only explicitly defined concept in the ontology. Similarly, we can define implicitly definable concepts:

Definition 2 (Implicitly definable concept) *Let* C *be a concept name,* $\mathcal{T}$ *a TBox, and* Σ *a signature, where* $\mathsf{C} \in \mathsf{Sig}(\mathcal{T})$*, and* $\Sigma \subseteq \mathsf{Sig}(\mathcal{T}) \setminus \{\mathsf{C}\}$*.* C *is implicitly definable from* Σ *under* $\mathcal{T}$*, if and only if for any two models* $\mathcal{I}$ *and* $\mathcal{K}$ *of* $\mathcal{T}$*,* $\Delta^{\mathcal{I}} = \Delta^{\mathcal{K}}$*, and for all predicate* $P \in \Sigma$*,* $P^{\mathcal{I}} = P^{\mathcal{K}}$*. Then it holds that* $\mathsf{C}^{\mathcal{I}} = \mathsf{C}^{\mathcal{K}}$*.*

Given the example, it can be seen that both Mother and Father are implicitly defined concepts in $\mathcal{T}^{Family}$, and each has nine syntactically different, but semantically equivalent definitions (Figure 1, lower). Furthermore, the explicitly defined concept Parent, is also implicitly defined by axioms $\{\alpha_2, \alpha_6\}$.

Deciding definability. A particular concept name C can either be defined *explicitly* or *implicitly* under an ontology, or be *undefined.* Explicit definability is a syntactic notion; deciding whether C is explicitly defined under an ontology is the trivial process of searching the TBox for a concept equivalence axiom whose left-hand side is C, and the potentially complex concept on the right-hand side does not include C (e.g. $\mathsf{C} \equiv \mathsf{D}$ where $\mathsf{C} \notin \mathsf{Sig}(\mathsf{D})$). In contrast, implicit definability is a semantic notion whose detection requires reasoning[1]. The computational complexity of determining whether a concept is implicitly defined depends on the complexity of the entailment check, which is predicted on the expressivity of the given DL language. Thus it is potentially exponential in the size of the ontology, for expressive DL dialects.

Role definability. The notion of definability also applies to roles, hence a role can be classified either as *defined* (explicitly or implicitly), or as *undefined.* For example, in $\mathcal{T}^{Family}$, the role $\mathsf{hasChild}$ is explicitly defined as the inverse of the role $\mathsf{hasParent}$, which is implicitly defined by axiom α_6. A defined role means that its extensions (set of pairs of individuals) can be unambiguously determined under an ontology, if the individuals of those entities that are used to define the role are known in an interpretation. Concepts are defined in terms of other concept and role names, whereas roles are only defined in terms of other roles. Deciding role definability can be achieved by using the same method as for concepts, however the implicit definability check process must be restricted to the R-Box (a subset of the TBox which contains all role axioms), and the definition signature Σ can only contain role names [4].

The *number of possible rewritings* of a defined concept (or role), regardless of whether it is explicitly or implicitly defined, is potentially exponential in the size of the ontology. Descriptions of defined concepts (i.e. the right-hand side

[1]Implicit definability can be reduced to entailment checking [14].

of a non-primitive concept definition axiom) are built inductively using other, potentially defined concepts. Thus, the number of possible concept rewritings is dependent on the definability of its constituent concepts. As the definability of any defined description member concept is dependent on the definability of its own description, definability is therefore a *recursive notion* [4].

2.2 Minimal Definition Signatures

A definition signature can be defined as:

Definition 3 (Definition Signature (DS)) *Given a TBox $\mathcal{T}$, a set of entities Σ is a definition signature of*

- *the* ***concept*** *C under $\mathcal{T}$, if and only if members of Σ can be used to construct the right-hand side of a definition axiom for C, i.e. there is some complex concept D, such that* $\mathsf{Sig}(\mathsf{D}) \subseteq \Sigma$, *and* $\mathcal{T} \models \mathsf{C} \equiv \mathsf{D}$, *where* $\Sigma \subseteq \mathsf{Sig}(\mathcal{T}) \setminus \{\mathsf{C}\}$*;*
- *the* ***role*** *r under $\mathcal{T}$, if and only if there is some complex role s, such that* $\mathsf{Sig}(\mathsf{s}) \subseteq \Sigma$, *and* $\mathcal{T} \models \mathsf{s} \equiv \mathsf{r}$, *where* $\Sigma \subseteq \mathsf{Sig}(\mathcal{T}) \setminus \{\mathsf{r}\}$*.*

If an entity e is defined in an ontology, then we can entail that there exists some subset of the ontology signature that implicitly defines e. We only focus on acyclic definitions, as definitions with *direct cycles* (where the defined concept appears in its corresponding description) are excluded by this definition, due to the fact that such definition signatures does not permit rewriting without using the defined entity name. We denote that an *entity e is implicitly definable by a signature Σ^e* under an ontology $\mathcal{O}$ as $\mathcal{O} \models_{\Sigma^e} e \equiv \mathsf{C}$, where C is a potentially complex concept or complex role, such that $Sig(\mathsf{C}) = \Sigma$.

As definition signatures may contain *redundant members*, their size could be as large as the ontology signature, thus we introduced the notion of *signature minimality*:

Definition 4 (Minimal Definition Signature (MDS)) *A signature Σ is a minimal definition signature of a defined entity e under a TBox $\mathcal{T}$, if none of its proper subsets are definition signatures of e.*

The *minimality* property of an MDS refers to minimising the size of the signatures, by eliminating superfluous entities. However, a defined concept may have many *unique* MDSs (where the difference of any two MDSs is not an empty set) under an ontology, with the same cardinality. From the definition, it follows that every MDS is also a DS, and any DS may contain at least one, but potentially many MDSs. For example, in the $\mathcal{T}^{Family}$ example (Figure 1), the signature $\Sigma = \{\mathsf{hasChild}, \mathsf{Man}, \mathsf{Woman}\}$ is a DS of all three defined concepts in the TBox. However, this signature is not a minimal DS of Parent, because it can be defined by the following MDSs: $\{\mathsf{hasChild}\}, \{\mathsf{hasParent}\}, \{\mathsf{Mother}, \mathsf{Father}\}$; as formalised by axiom sets $\{\alpha_1\}, \{\alpha_2, \alpha_6\}$ and $\{\alpha_2\}$, respectively.

While rewriting has been extensively studied, the practical applicability of currently existing methods is limited, as they are bounded to particular Description Logics (DLs), and they often present only theoretical results. In our previous work [4], we have presented a pragmatic approach that in practice can efficiently compute the *complete set of definition signatures* of most defined entities described using a DL language for which the Beth definability property holds. Knowing the signature of a definition, even without knowing the actual form of the definition axiom, can support a variety of applications, such as ontology alignment evaluation. Furthermore, we have sampled the prevalence and the extent of definability in real world ontologies by conducting experiments on a large and diverse dataset[2], which included establishing the definability status of concepts and roles and computing all possible minimal definition signatures of defined entities. This has confirmed the hypothesis that definability is prevalent in any type of ontology, although it is more likely to occur in more expressive, and semantically richer ontologies. In addition it was shown, that definability computation is feasible for most real world ontologies, and in some cases, it can be useful in dynamic environments as well, due to the fact that a subset of MDSs can be found in polynomial time.

2.3 The Set Coverage Problem Family

The *set coverage problem* (or minimal set cover problem) is a classic question in computer science, combinatorics, and complexity theory [17]. The set cover problem is, given a set of elements $\mathcal{U}$ (referred to as the *universe*) and a family $\mathcal{S}$ of subsets of $\mathcal{U}$ (whose union equals the universe), to find the smallest, *minimal* sub-collection $\mathcal{C} \subseteq \mathcal{S}$, called the *cover set* such that the union of sets in $\mathcal{C}$ covers $\mathcal{U}$ ($\forall x\{x \in \mathcal{U} | x \in \mathcal{C}\}$). For instance, let us consider Example 1:

Example 1 (Minimal set cover problem) *Let $\mathcal{U}$ be a universe of elements, and $\mathcal{S}$ be a family of sets such that:*

- $\mathcal{U} = \{1, 2, 3, 4, 5\}$
- $\mathcal{S} = \{\{1, 2, 3\}, \{1, 2\}, \{3, 4\}, \{4, 5\}\}$

The union of subsets of $\mathcal{S}$ clearly contains all members of $\mathcal{U}$, thus $\mathcal{U}$ can be covered by an $S' \subseteq \mathcal{S}$. Although there are several other solutions ($\{\{S_1, S_3, S_4\}, \{S_2, S_3, S_4\}\}$), there is only one minimal cover set, $\mathcal{C} = \{S_1, S_2\}$.

Finding the minimal cover set is an NP-hard problem, however, there is a polynomial time greedy algorithm, that is able to find approximations (i.e. not necessarily minimal, but small cover sets) [17].

In the *weighted set cover* problem, each set $S \in \mathcal{S}$ is assigned a *weight* $w(S) \geq 0$, thus in this case, the goal is to find a cover set $\mathcal{C}$ with the minimal total weigh $\sum_{S \in \mathcal{C}} w(S)$. To illustrate the problem, let us consider the following example:

[2]The evaluation corpus was assembled from a number of different datasets, including a corpus which was obtained by crawling the Web, this contained thousands of OWL ontologies [10]

Example 2 (Weighted set cover problem)

- $\mathcal{U} = \{1, 2, 3, 4, 5\}$
- $\mathcal{S} = \{\{1, 2, 3\}, \{1, 2\}, \{3, 4\}, \{4, 5\}\}$
- $W = \langle 5, 2, 2, 2 \rangle$

The union of subsets of $\mathcal{S}$ contains all members of $\mathcal{U}$, but each set $s \in \mathcal{S}$ is assigned a weight value, as given by the vector W. In terms of cover set cardinality, the minimal solution is $\mathcal{C}_1 = (S_1 \cup S_4)$. However, when weights are taken into account the cover set with a minimal total weight is $\mathcal{C}_2 = (S_2 \cup S_3 \cup S_4)$, where $w(\mathcal{C}_1) = 7, w(\mathcal{C}_2) = 6$, thus $w(C_1) > w(C_2)$.

Another relevant classical problem is from relational database theory, which concerns finding the minimal cover for functional dependencies. A *functional dependency (FD)* is a constraint between two sets of attributes in a relation from a database [16]. For example, $X \rightarrow Y$ means that the values of the attribute set Y are determined by the values of X, or in other words, two tuples in a database sharing the same values of X would also share the same values for Y. The *closure* of a set of attributes X with respect to a set of FDs $\mathcal{F}$ is the set X^+ of all attributes that are functionally determined by X using $\mathcal{F}^+$, the closure of $\mathcal{F}$. Before computing the closure, a set of FDs $\mathcal{F}$ is *normalised* by exhaustively applying inference rules. The following *inference rules* [16] are used both in normalisation and in closure computation (where X, Y, Z, W denote attribute sets in some relation R):

Inference Rule	**Condition**	**Action**
Reflexivity	if $Y \subseteq X$	then $X \rightarrow Y$
Augmentation	if $X \rightarrow Y$	then $XZ \rightarrow YZ$
Transitivity	if $X \rightarrow Y$ and $Y \rightarrow Z$	then $X \rightarrow Z$
Union	if $X \rightarrow Y$ and $X \rightarrow Z$	then $X \rightarrow YZ$
Decomposition	if $X \rightarrow YZ$	then $X \rightarrow Y$ and $X \rightarrow Z$
Pseudotransitivity	if $X \rightarrow Y$ and $WY \rightarrow Z$	then $WX \rightarrow Z$
Composition	if $X \rightarrow Y$ and $Z \rightarrow W$	then $XZ \rightarrow YW$

Table 1: Functional dependency inference rules

The next example illustrates the computation of the closure of all attributes:

Example 3 (FD set attribute closures) *Let us consider a set of FDs $\mathcal{F}$ such that*

$$\begin{aligned} \mathcal{F} = \{ \; & (1)\ A \rightarrow B \\ & (2)\ C \rightarrow E \\ & (3)\ E \rightarrow F \\ & (4)\ A, C \rightarrow D\} \end{aligned}$$

$\mathcal{F}$ is already normalised (is in 3NF, the third normal form), i.e. each FD contains exactly one attribute on the RHS (right-hand side), and each FDs' LHS

is irreducible (reducing any one attribute from the LHS set would change the content of $\mathcal{F}$). The closure of all attributes in $\mathcal{F}$ is computed as shown by the following steps:

1. $A^+ : A, B$ *(A by reflexivity, B by (1))*
2. $B^+ : B$ *(B by reflexivity)*
3. $C^+ : C, E, F$ *(C by reflexivity, E by (2), F by transitivity and (2, 3))*
4. $D^+ : D$ *(D by reflexivity)*
5. $F^+ : F$ *(F by reflexivity)*
6. $(A, C)^+ : A, B, C, D, E, F$ *(A, C by reflexivity, B by (1), D by (4), E by (2), F by transitivity and (2, 3))*

3 The Signature Coverage Problem

The ontology signature coverage problem concerns whether a given task *signature* can be covered by another, *restricted signature*, where both signatures are subsets of the same vocabulary. A task signature is said to be covered, if all of its constituent entities are covered. Individual names can only be covered by an asserted entity i.e. *explicitly*, however, definable signature entities (concepts and roles) can also be *covered implicitly*, if the restricted signature contains a corresponding definition signature. We define entity coverage as:

Definition 5 (Explicitly or implicitly covered entity) *Given an ontology $\mathcal{O}$, a task signature $\mathcal{S}$, and restricted signature $\mathcal{R}$ such that $\mathcal{S}, \mathcal{R} \subseteq \mathsf{Sig}(\mathcal{O})$, an entity $e \in \mathcal{S}$ is covered explicitly by $\mathcal{R}$, if $e \in \mathcal{R}$; or covered implicitly by $\mathcal{R}$, if e has a definition signature Σ, such that $\Sigma \subseteq \mathcal{R}$; otherwise it is uncovered.*

A defined concept or role can simultaneously be covered explicitly and implicitly, thus a task signature entity $e \in \mathcal{S}$ may assume one of the four different *coverage status* w.r.t. a restricted signature $\mathcal{R}$:

uncoverable		$e \notin \mathcal{R} \wedge \Sigma \not\subseteq \mathcal{R}$	explicit coverage: $e \in_? \mathcal{R}$
coverable	explicitly only	$e \in \mathcal{R} \wedge \Sigma \not\subseteq \mathcal{R}$	
	explicitly and implicitly	$e \in \mathcal{R} \wedge \Sigma \subseteq \mathcal{R}$	explicit and implicit coverage: $e \in_? \mathcal{R}^+$
	implicitly only	$e \notin \mathcal{R} \wedge \Sigma \subseteq \mathcal{R}$	

Table 2: Entity coverage status

Determining whether a given task signature is coverable by a particular, restricted signature is the trivial process of identifying the coverage status of each task signature entity. This can be achieved in two ways: *i)* either each task signature entity is subjected to an implicit definability check; *ii)* or by assuming that the complete set of minimal definition signatures of each entity is already obtained, for each entity $e_i \in \mathcal{S}$, we search for an MDS Σ^{e_i} such that $\Sigma^{e_i} \subseteq \mathcal{R}$. The set of entities, which covers all members of a task signature $\mathcal{S}$ is called the *cover set* and it is defined as follows:

Definition 6 (Cover set) *Given an ontology $\mathcal{O}$, a task signature $\mathcal{S}$, and restricted signature $\mathcal{R}$ such that $\mathcal{S}, \mathcal{R} \subseteq \mathsf{Sig}(\mathcal{O})$, $\mathcal{C}$ is a cover set of $\mathcal{S}$ with respect to $\mathcal{R}$, if and only if $\forall e\{e \in \mathcal{S} | e \in \mathcal{C} \vee \exists \Sigma^e | \Sigma^e \subseteq \mathcal{C}\}$ and $\mathcal{C} \subseteq \mathcal{R}$.*

In other words, a cover set is a definition signature of all entities of the task signature.

As an ontology signature can cover more than its constituent entities, due to the fact that a given signature may permits some defined entities to be implicitly covered, hence we adopt the notion of *closure* from functional dependency computation (Section 2.3) in order to provide a signature representation which describes *the set of all entities covered* by a given signature. We refer to such an entity set as signature closure, and define it as follows:

Definition 7 (Signature closure) *Given an ontology $\mathcal{O}$, and a signature $\mathcal{X}$ such that $\mathcal{X} \subseteq \mathsf{Sig}(\mathcal{O})$, $\mathcal{X}^+$ (the closure of $\mathcal{X}$), contains all explicitly and implicitly covered entities of $\mathsf{Sig}(\mathcal{O})$ by $\mathcal{X}$, i.e. the set of entities $\forall e\{e \in \mathsf{Sig}(\mathcal{O}) | e \in \mathcal{X} \vee \exists \Sigma^e | \Sigma^e \subseteq \mathcal{C}\}$.*

This permits a more succinct definition of coverage: a task signature $\mathcal{S}$ is covered by a set $\mathcal{C}$ if and only if $\mathcal{S} \subseteq \mathcal{C}^+$.

Once it has been established, that a restricted signature $\mathcal{R}$ covers a given task signature $\mathcal{S}$, the problem is to identify the *smallest subset* $\mathcal{C} \subseteq \mathcal{R}$ which covers $\mathcal{S}$. This is called the minimal cover set (or cover), and defined as follows:

Definition 8 (Minimal cover set) *Given an ontology $\mathcal{O}$, a task signature $\mathcal{S}$, a restricted signature $\mathcal{R}$ such that $\mathcal{S}, \mathcal{R} \subseteq \mathsf{Sig}(\mathcal{O})$, and the set $\mathcal{C}$ which covers $\mathcal{S}$ with respect to $\mathcal{R}$, $\mathcal{C}$ is minimal if and only if there is no other cover set $\mathcal{C}' \subseteq \mathcal{R}$ such that $|\mathcal{C}'| < |\mathcal{C}|$.*

There can be more than one minimal cover set, i.e. two sets with the same cardinality whose overlap is not an empty set.

Finding a minimal cover set is an NP problem, because it requires all cover sets to be identified, by exhaustively testing each subset of the power set of the ontology signature ($\mathcal{P}(\mathsf{Sig}(\mathcal{O}))$), in order to find all covers and select the one with the minimum cardinality. This complexity can be reduced by considering the module of a given task signature, instead of the entire ontology signature. As it was shown in our previous work [4], all possible MDSs of a defined entity is contained in a module of the entity, thus the module of a task signature contains all possible minimal cover sets. However, finding the minimal cover set in a module is still an NP problem, as in this case, each subset of the power set of the module signature needs to be considered.

Approximation algorithms are commonly used for problems with NP time complexity, such as the set cover problem, in order to provide sub-optimal solutions in polynomial-time. The *greedy algorithm design* is one of the standard techniques for approximation algorithms [17]. In the context of the minimal signature cover problem, the optimal solution is the smallest possible cover set. The following example illustrates the signature coverage problem:

Example 4 (Signature Coverage) *Let $\mathcal{O}$ be an ontology, $\mathcal{S}$ a task signature, $\mathcal{R}$ a restricted signature, and M the complete set of MDSs of each defined entity of $\mathcal{S}$ (an MDS $m \in M$ is represented as the tuple $\langle e, \Sigma \rangle$, where e is the defined concept or role name,and Σ denotes the definition signature), where $\mathcal{S}, \mathcal{R} \subseteq \mathsf{Sig}(\mathcal{O})$, such that*

- $\mathsf{Sig}(\mathcal{O}) = \{\mathsf{A}, \mathsf{B}, \mathsf{C}, \mathsf{D}, \mathsf{E}, \mathsf{F}, \mathsf{r}, \mathsf{s}, \mathsf{q}\}$
- $M = \{\langle \mathsf{C}, \{\mathsf{A}, \mathsf{B}\}\rangle, \langle \mathsf{C}, \{\mathsf{E}, \mathsf{r}\}\rangle, \langle \mathsf{C}, \{\mathsf{q}\}\rangle, \langle \mathsf{B}, \{\mathsf{D}\}\rangle, \langle \mathsf{D}, \{\mathsf{B}\}\rangle, \langle \mathsf{s}, \{\mathsf{r}\}\rangle\}$
- $\mathcal{S} = \{\mathsf{B}, \mathsf{C}, \mathsf{D}, \mathsf{E}, \mathsf{s}, \mathsf{q}\}$
- $\mathcal{R} = \{\mathsf{A}, \mathsf{B}, \mathsf{C}, \mathsf{D}, \mathsf{E}, \mathsf{r}, \mathsf{q}\}$

Without accounting for definability, i.e. by only considering explicit coverage, the restricted signature does not cover the task signature because $\mathcal{S} \setminus \mathcal{R} \neq \emptyset$. However, considering implicit coverage shows that the closure of the restricted signature is $\mathcal{R}^+ = \{\mathsf{A}, \mathsf{B}, \mathsf{C}, \mathsf{D}, \mathsf{E}, \mathsf{r}, \mathsf{s}, \mathsf{q}\}$, thus $\mathcal{S}$ can be covered by $\mathcal{R}$, because $\mathcal{S} \subseteq \mathcal{R}^+$. Following a naive, greedy approach, one may select those entities that appear both in $\mathcal{S}$ and $\mathcal{R}$ as these entities can be covered explicitly, i.e. $C_1 = \mathcal{S} \cap \mathcal{R} = \{\mathsf{C}, \mathsf{D}, \mathsf{E}, \mathsf{q}\}$; then attempt to cover the remaining task signature entities, by adding a correponding MDSs for each uncovered task signature entity; in this case covering s by adding r to C_1, as there is an MDS $\langle s, \{r\}\rangle$ thus s is implicitly definable by the signature $\{\mathsf{r}\}$. As a result, $C_1 = \{\mathsf{B}, \mathsf{C}, \mathsf{D}, \mathsf{E}, \mathsf{r}, \mathsf{q}\}$ covers $\mathcal{S}$. However, the smallest cover set is $C_2 = \{\mathsf{B}, \mathsf{E}, \mathsf{r}, \mathsf{q}\}$, because $C_2^+ = \{\mathsf{B}, \mathsf{C}, \mathsf{D}, \mathsf{E}, \mathsf{r}, \mathsf{s}, \mathsf{q}\}$, $\mathcal{S} \subseteq C_2^+$, and $|C_1| > |C_2|$.

In example 4, a naive, greedy approach (*Greedy #1*) has produced a non-minimal cover set C_1, which was an approximation of the minimal cover C_2. The cover set C_1 can be improved by removing redundant entities (resulting in the set $C_1' = C_2$), i.e. producing a non-redundant cover set:

Definition 9 (Non-redundant cover set) *Given an ontology $\mathcal{O}$, a task signature $\mathcal{S}$, a restricted signature $\mathcal{R}$ such that $\mathcal{S}, \mathcal{R} \subseteq \mathsf{Sig}(\mathcal{O})$, and the set $\mathcal{C}$ which covers $\mathcal{S}$ with respect to $\mathcal{R}$, $\mathcal{C}$ is a non-redundant cover set if and only if none of its no proper subsets are cover sets of $\mathcal{S}$.*

Non-redundant cover sets are typically small, however, as there can be more than one non-redundant cover set (with different cardinality), a non-redundant cover set is not necessarily minimal. It is worth noting that every minimal cover set is also a non-redundant set.

4 Approximating Minimal Cover Sets

In order to tackle the NP time complexity of the minimal signature cover problem, we introduce a greedy, approximation algorithm (*Greedy #2*), which provides a sub-optimal solution in polynomial-time. The resulting cover set is always non-redundant.

The basic idea behind the approach is that the cover set is built incrementally until all task signature members are covered, however, instead of selecting individual entities from the restricted signature, at each iteration the approach selects an entity set. The entity sets that are being considered are MDSs, because individual entities are typically only cover entities explicitly, while MDSs can cover defined entities implicitly (in addition to explicitly covering all those task signature entities that appear in the MDS as well). The selection is made by assigning a *cost* and *value* score to each MDS, and then picking the set which provides the maximum value and the minimum cost with respect to the task signature and the incomplete cover set, prioritising on the value score. The cost quantifies the number of entities required to be added to the cover set (i.e. the set difference of the cover set and the particular MDS), while the value represents the number of entities that the given signature covers (an MDS can be a DS for more than one defined entity, thus it can cover several task signature entities). In case there are more than one MDSs with the same cost and value, a random MDS is selected.

In order to evaluate the actual value of an MDS (i.e. the set of all entities of the task signature that the MDS covers either explicitly or implicitly), its *closure* needs to be identified, similarly to functional dependencies, thus we represent MDSs in the form of functional dependencies to facilitate this notion. There is a strong resemblance between the concept of FD and MDS, meaning that an MDS can be thought of as functional dependency between entities of an ontology, where the relation between the signature of the left-hand side (LHS) and the entity on the right-hand side (RHS) is implicit definability. For example, the minimal definition signature $\Sigma^{\mathsf{C}} = \{\mathsf{A}, \mathsf{B}\}$ which defines concept C using entities $\{\mathsf{A}, \mathsf{B}\}$ may be represented as $m : (\mathsf{A}, \mathsf{B} \rightarrow \mathsf{C})$, we refer to such MDS as fMDS, and define it as follows:

Definition 10 (fMDS) *Given a defined entity e, and its minimal definition signature Σ, the corresponding fMDS is the function $m : (\Sigma \rightarrow e)$, which given Σ covers e.*

A given fMDS closure is computed from the *set of all fMDSs*, by identifying all relevant definition signatures:

Definition 11 (fMDS-closure) *Given an fMDS $m : (\Sigma \rightarrow e)$, and a set of fMDSs $\mathcal{M}$ where $m_i \in \mathcal{M}$, the closure of m_i is $m_i^+ : (\Sigma \rightarrow E)$ such that*

$$E = \Sigma \cup \{e\} \cup (\bigcup \forall m_j^{+RHS} \{m_j \in \mathcal{M} | m_j^{+LHS} \subseteq m_i^{+LHS}\})$$

where m^{+LHS} denotes the signature Σ, and m^{+RHS} refers to signature E.

For example, given $\mathcal{M} = \{m_1 : (\mathsf{A}, \mathsf{B} \rightarrow \mathsf{C}), m_2 : (\mathsf{B} \rightarrow \mathsf{D})\}$, the closure is $\mathcal{M}^+ = \{m_1^+ : (\mathsf{A}, \mathsf{B} \rightarrow \mathsf{A}, \mathsf{B}, \mathsf{C}, \mathsf{D}), m_2^+ : (\mathsf{B} \rightarrow \mathsf{D})\}$ because the m_1^{+LHS} signature, in addition to implicitly covering concept C, also explicitly covers concepts A, B, furthermore, it implicitly covers D as $m_2^{+LHS} \subseteq m_1^{+RHS}$ thus $\mathsf{D} \in m_1^{+LHS}$.

Now we formalise the cost and value calculation of an fMDS:

Definition 12 (fMDS value and cost) *Given an fMDS m, an ontology $\mathcal{O}$, a task signature $\mathcal{S}$, a cover set $\mathcal{C}$, and M the complete set of fMDSs in $\mathcal{O}$, where $m, \mathcal{S} \subseteq \mathcal{O}$ the value and cost of m with respect to $\mathcal{S}$ and $\mathcal{C}$ is given by*

```
Algorithm 1: ComputeMinimalSignatureCover($\mathcal{O}, \mathcal{R}, \mathcal{S}, \mathcal{M}$)
   Input  : $\mathcal{O}$: ontology; $\mathcal{S}$: task signature; $\mathcal{R}$: restricted signature;
            $\mathcal{M}$: the complete set of $f$MDSs of each defined entity $e \in \mathcal{O}$
   Output: $C$: cover set of $\mathcal{S}$ w.r.t. $\mathcal{R}$
 1 $C \leftarrow C \ \cup \ \forall e\{e \in \mathcal{S} \cap \mathcal{R} | e \notin m_i^{RHS} | m_i \in \mathcal{M}\}$
 2 Initalise($M$)
 3 $C^+ \leftarrow$ ComputeSignatureClosure($C, M$)
 4 $M' \leftarrow M \setminus \forall m_i\{m_i \in M' | m_i^{LHS} \subseteq C^+\}$
 5 while $(\mathcal{S} \setminus C^+) \neq \emptyset$ do
 6     $\mathcal{V} \leftarrow$ ComputeValueCostVector($M', \mathcal{S}, C^+$)
 7     $m_{selected} \leftarrow$ select an $m \in M'$ according to $\mathcal{V}$, with max $value(m)$,
       and min $cost(m)$
 8     $C \leftarrow C \cup m_{selected}^{LHS}$
 9     $C^+ \leftarrow$ ComputeSignatureClosure($C, M$)
10     $M' \leftarrow M' \setminus (\{m_{selected}\} \cup \forall m_i\{m_i \in M' | m_i^{LHS} \subseteq C^+\})$
11 end
12 return C
```

- *the value function $v(m)$, which assigns a natural number $i \in \mathbb{N}_0$ to m such that $v(m) = |\mathcal{R} \setminus C^+ \cap m^{+RHS}|$*

- *the cost function $c(m)$, which assigns a natural number $i \in \mathbb{N}_0$ to m such that $c(m) = |C^+ \setminus m^{LHS}|$*

Algorithm 1 present the approach that approximates a minimal cover set for an ontology signature. The algorithm uses two subroutines (both described in Section 4.1), Algorithm 2 that computes the *closure of fMDSs*, and Algorithm 3 which computes the *closure of signatures*.

Algorithm 1 assumes the *precondition*, that the task signature $\mathcal{S}$ is coverable by the restricted signature $\mathcal{R}$. The process starts by applying an optimisation heuristic, that initialises the cover set C with all entities of the task signature that can only be covered explicitly (line 1). Next M, which is used as the search space, is initialised with the complete set of fMDSs $\mathcal{M}$. In addition, we generate and add a 'faux' MDS to M, for each entity that can be covered both explicitly and implicitly. For instance, the concept A would have the faux fMDSs ($\mathsf{A} \rightarrow \mathsf{A}$), i.e. the entity can cover itself as it is permitted by the restricted signature. By including these fMDSs, we ensure that the search space is complete, i.e. for each task signature entity it includes all possible ways of cover. Finally we replace M with its closure (line 2).

Before the process begins the search, C^+, the cover closure is computed. This is used as the termination condition of the search process (line 5), i.e. the algorithm concludes when task signature is covered. M' is created as a copy of M, the former is the actual search space which is continuously pruned at each iteration (in order to optimise the process by reducing the size of the search space), while the latter is the complete set of fMDSs closures which is left intact for the purpose of computing signature closures during the search. The pruning of M' is carried out by removing any fMDSs which has no value and cost w.r.t. the cover set (line 4).

During the search (line 5-11), the value and cost of each fMDSs in M' is evaluated w.r.t. the cover set (line 6), then the best fMDSs is selected (line 7) and added to the cover (i.e. the LHS of the fMDS, which is the definition signature, line 8). The cover is then reevaluated, by updating its closure (line 9), finally M' is pruned according to the updated cover set. These steps are repeated until the task signature is covered.

The algorithm always finds a non-redundant cover set for a given task signature, thus it is complete and correct. At the worst case, the process covers at least one entity at each iteration, thus the maximum number of steps performed by the algorithm is n, where $n = |\mathcal{S}|$. As both subroutines used by Algorithm 1 are polynomial time, the overall time complexity of Algorithm 1 is polynomial as well.

(Greedy #3) In Example 4, we have outlined an algorithm (*Greedy #1*), which produces redundant cover sets. The main difference between approach #1 and #2 is that the former narrows the non-deterministic part of the search, as it explicitly covers all of those entities that can be covered explicitly, i.e. it includes each explicitly and implicitly coverable entity as well. Thus approach #1 is typically faster than #2, however #2 may produces a considerably more optimal cover set compared to approach #1. The redundancy issue of approach #1 is trivial to resolve, as redundant entities can be filtered out from the cover set in polynomial time:

$$C \setminus \forall e\{e \in C | \exists m \in M | e \in m^{RHS} | m^{LHS} \subseteq C\}$$

i.e. by removing every entity from the cover set C, that is definable by C. Thus we introduce approach #3, that is expected to produce more optimal solutions than #1, while still performing faster than approach #2.

4.1 Computing Closures

Algorithm 2 computes the closure of an fMDS set, by exhaustively applying functional dependency inference rules (Section 2.3). The closure of an fMDS consists of all entities that can be covered using the LHS signature of m. For example, given $M = \{m_1 = (\mathsf{A}, \mathsf{B} \rightarrow \mathsf{C}), m_2 = (\mathsf{C} \rightarrow \mathsf{D})\}$, first we apply the identity rule, i.e. every entity explicitly covers itself, hence $m_1 : (\mathsf{A}, \mathsf{B} \rightarrow \mathsf{A}, \mathsf{B}, \mathsf{C})$ and $m_2 : (\mathsf{C} \rightarrow \mathsf{C}, \mathsf{D})\}$. Then we apply the transitivity rule: $m_2^{LHS} \subseteq m_1^{RHS}$, thus $m_1 : (\mathsf{A}, \mathsf{B} \rightarrow \mathsf{A}, \mathsf{B}, \mathsf{C}, \mathsf{D})$. Finally the MDS closure is given by the set $M^+ = \{m_1 : (\mathsf{A}, \mathsf{B} \rightarrow \mathsf{A}, \mathsf{B}, \mathsf{C}, \mathsf{D}), m_2 : (\mathsf{C} \rightarrow \mathsf{C}, \mathsf{D})\}$. At the worst case, the each fMDS is applied to every other fMDS, at most once, thus the algorithm has quadratic time complexity.

Algorithm 3 computes the closure of an ontology signature w.r.t. to a set of fMDSs, similarly to Algorithm 2, by exhaustively applying inference rules. X^+, the closure of a signature consists of all entities that can be covered either explicitly or implicitly by the set X. For example, given a signature $X = \{\mathsf{A}, \mathsf{B}\}$, and

Algorithm 2: ComputeMDSClosure(M)

```
$M^+ \leftarrow M$
for $m \in M^+$ do
    $m^{RHS} \leftarrow (m^{LHS} \cup m^{RHS})$
end
Updated $\leftarrow$ true
while Updated do
    Updated $\leftarrow$ false
    for $m \in M^+$ do
        for $m' \in M^+$ do
            if $m' \neq m \wedge m^{LHS} \subseteq m'^{RHS} \wedge m^{RHS} \not\subseteq m'^{RHS}$ then
                $m' := (m'^{LHS} \rightarrow m'^{RHS} \cup m^{LHS})$
                Updated $\leftarrow$ true
            end
        end
    end
end
return $M^+$
```

Algorithm 3: ComputeSignatureClosure(X, M^+)

```
$X^+ \leftarrow X$
Updated $\leftarrow$ true
while Updated do
    Updated $\leftarrow$ false
    for $m^+ \in M^+$ do
        if $m^{+LHS} \subseteq X^+$ then
            $X^+ \leftarrow X^+ \cup m^{+RHS}$
            $M^+ \leftarrow M^+ \setminus \{m^+\}$
            Updated $\leftarrow$ true
        end
    end
end
return $X^+$
```

the set of fMDSs closures $M^+ = \{m_1^+ = (\mathsf{A}, \mathsf{B} \rightarrow \mathsf{A}, \mathsf{B}, \mathsf{C}, \mathsf{D}), m_2^+ = (\mathsf{C} \rightarrow \mathsf{D})\}$ the closure of X is given by the set $X^+ = \{\mathsf{A}, \mathsf{B}, \mathsf{C}, \mathsf{D}\}$. The algorithm has polynomial time complexity, as at most every fMDS is applied to the signature exactly once.

5 Empirical Evaluation

In this section, we empirically determine how effective our approximation approach is in finding task signature cover sets. The experiments tested the *hypoth-*

Ontology	$\mathcal{DL}$ Expressivity	Axioms	$\mathcal{C}$			$\mathcal{R}$			$\mathcal{C} \cup \mathcal{R}$	
			$\lvert\mathcal{C}\rvert$	$Def\%$	$\mathcal{M}$	$\lvert\mathcal{R}\rvert$	$Def\%$	$\mathcal{M}$	$Def\%$	$\mathcal{M}$
Conference corpus										
cmt	$\mathcal{ALCIN}(\mathcal{D})$	226	29	13.79%	2.00	59	67.80%	1.00	50.00%	1.09
conference	$\mathcal{ALCHIF}(\mathcal{D})$	285	59	49.15%	2.31	64	65.63%	1.00	57.72%	1.54
confOf	$\mathcal{SIN}(\mathcal{D})$	196	38	18.42%	4.00	36	5.56%	1.00	12.16%	3.33
edas	$\mathcal{ALCOIN}(\mathcal{D})$	739	103	11.65%	7.00	50	56.00%	1.00	26.14%	2.80
ekaw	$\mathcal{SHIN}$	233	73	0.00%	0.00	33	90.91%	1.00	28.30%	1.00
iasted	$\mathcal{ALCIN}(\mathcal{D})$	358	140	11.43%	2.50	41	39.02%	1.00	17.68%	1.75
sigkdd	$\mathcal{ALEI}(\mathcal{D})$	116	49	16.33%	2.38	28	42.86%	1.00	25.97%	1.55
	AVG.	**307.57**	**70.14**	**17.25%**	**2.88**	**44.43**	**52.54%**	**1.00**	**31.14%**	**1.87**
LargeBio corpus										
NCI_fma	$\mathcal{ALC}$	9083	6551	30.27%	1.32	63	1.59%	0.00	30.00%	1.32
NCI_snomed	$\mathcal{ALCH}$	30411	24040	28.60%	1.39	82	1.22%	0.00	28.50%	1.39
SNOMED_fma	$\mathcal{ALER}$	20243	13430	21.44%	1.10	18	5.56%	0.00	21.47%	1.10
SNOMED_nci	$\mathcal{ALER}$	71042	51128	57.31%	1.09	21	4.76%	0.00	57.87%	1.09
	AVG.	**32694.75**	**23787.25**	**34.40%**	**1.22**	**46.00**	**3.28%**	**0.00**	**34.46%**	**1.22**

Table 3: Evaluation corpus

esis, that the presented approximation approach, by considering both explicit and implicit coverage, produces a cover set which is albeit not minimal, but still considerably smaller than cover sets obtained by only explicit coverage. Thus approximations of minimal cover sets are typically smaller than explicit covers, if the given task signature contains defined entities w.r.t. a restricted signature (clearly, for a task signature which lacks defined entities, only explicit coverage is possible). In addition, by measuring the computation *time* and the *cardinality* of the resulting cover sets, we compare those two versions of the approach (*Greedy #2* and *#3*) that produce more optimal approximations, i.e. non-redundant cover sets. Greedy #2 considers a larger search space, thus it is expected to provide a more optimal solution (i.e. smaller cover set) than Greedy #3, consequently, the latter approach is likely to perform faster.

The *evaluation corpus* was assembled from two OWL datasets that are commonly used for empirical evaluation in the ontology alignment literature. We have selected 7 small ontologies (average 70.14 concepts and roles, and 307.57 axioms per ontology) from the *Conference* dataset [3], which describes the conference organisation domain; and 4 large (average 23787.25 entities, and 32694.75 axioms) ontologies from the *Large biomedical ontology* dataset[4]. Thus the corpus is diverse in size, moreover, it is appropriate to assess implicit coverage as all ontologies contain some defined concepts and roles. For every concept and role in each ontology of the evaluation corpus, we have pre-computed the definability status and the complete set of MDSs. Table 3 presents a summary of the corpus, showing the DL expressivity, the number of logical axioms, number of concepts ($|\mathcal{C}|$), roles ($|\mathcal{R}|$) and their union ($\mathcal{C} \cup \mathcal{R}$) in the ontology signature, the ratio of defined concepts and roles ($Def\%$), and the average number of different minimal definition signatures per defined entity ($\mathcal{M}$). Both datasets contain ontologies with varying level of definability, as shown by the ratio of defined ontology signature entities and the number of different MDSs per entity.

The *experimental framework* was implemented in Java; the OWL API [8] was

[3]`http://oaei.ontologymatching.org/2014/conference/index.html`
[4]`http://oaei.ontologymatching.org/2014/largebio/index.html`

Ontology	$(\mathcal{C} \cup \mathcal{R})$	$Def\%$	*Ideal Cover*	*Greedy #3*		*Greedy #2*	
			cov	*cov*	Time	*cov*	Time
			Conference corpus				
cmt	88	50.00%	50.00%	72.73%	0.48 ms	72.73%	3.62 ms
conference	123	57.72%	42.28%	69.92%	2.54 ms	70.73%	11.79 ms
confOf	74	12.16%	87.84%	89.19%	0.25 ms	89.19%	0.43 ms
edas	153	26.14%	73.86%	85.62%	3.05 ms	86.27%	8.70 ms
ekaw	106	28.30%	71.70%	85.85%	0.20 ms	85.85%	2.30 ms
iasted	181	17.68%	82.32%	87.85%	0.96 ms	87.85%	3.18 ms
sigkdd	77	25.97%	74.03%	81.82%	0.43 ms	81.82%	1.12 ms
AVG.	**114.57**	**31.14%**	**68.86%**	**81.85%**	**1.13 ms**	**82.06%**	**4.45 ms**
			LargeBio corpus				
NCI_fma	6551	29.98%	70.02%	81.30%	1.53 s	70.02%	3.09 s
NCI_snomed	24040	28.50%	71.50%	82.61%	27.68 s	71.50%	56.42 s
SNOMED_fma	13430	21.47%	78.54%	85.82%	4.36 s	78.56%	8.64 s
SNOMED_nci	51128	57.87%	42.13%	62.18%	709.76 s	42.45%	838.30 s
AVG.	**23787.25**	**34.46%**	**65.55%**	**77.98%**	**185.83 s**	**65.63%**	**226.61 s**

Table 4: Comparing cover size and computation of time of approach #2 and #3, for full covering the entire ontology signature.

used for ontology manipulation and for interacting with the reasoners[5]. Entity definability status, and corresponding MDSs were computed using the OntoDef API [4]. All of the experimental software and data are available online[6].

In all experiments, for each task signature, we have computed cover sets by using the two approximation approaches, *Greedy #2* and *#3*. A naive approach, used as the *baseline* in all experiments, which considers only explicit coverage of signatures, always provides a cover that is the same set as the task signature (i.e. it is a constant $cov = 100\%$). We have only considered *coverable* task signatures (i.e. $\mathcal{S} \subseteq \mathcal{R}^+$), thus in all cases, the restricted signature $\mathcal{R}$ was equivalent to the T-Box signature, while the task signature $\mathcal{S}$ was allocated several differently sized T-Box signature subsets (i.e. $\mathcal{R} = \mathsf{Sig}(\mathcal{T})$, and $\mathcal{S} \subseteq \mathsf{Sig}(\mathcal{T})$). Varying the composition of only one of the two signatures simplified both the experiment conduct and the result analysis process, while it provided the same overall results. Experiments were conducted with 8GB maximum memory allocated for the Java Virtual Machine, running on a machine equipped with 16GB RAM and a 16 core processor architecture.

(Experiment 1: Ideal Covers.) In this experiment, we have compared the cover set obtained by the different approximation approaches, to the actual minimal cover set. The only minimal signature cover, which is not an approximation and can be computed efficiently, is only obtainable, when the task requires the entire signature of an ontology to be covered, i.e. $\mathcal{S} = \mathsf{Sig}(\mathcal{O})$. This special case provides the opportunity to evaluate the difference between an actual, and an approximated minimal signature cover set. The *ideal cover* is obtained by removing all non-redundant, defined concept and role names from the ontology signature[7]. Results are presented in Table 4 (where the baseline method

[5]We used both the HermiT [5] and Pellet [13] reasoners. In most datasets HermiT performs faster, however Pellet was able to load and process some ontologies that HermiT could not (due to ontologies using datatypes that are not part of the OWL 2 datatype map and no custom datatype definition was given).

[6]`http://www.csc.liv.ac.uk/~dgeleta/ontodef.html`

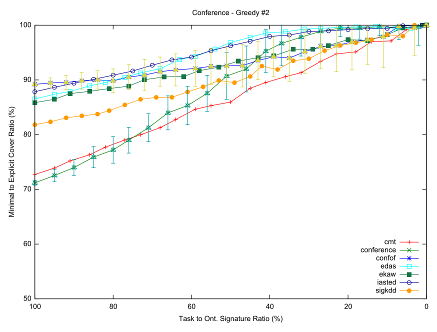
[7]Considering non-redundancy in entity removal is necessary in order to avoid removing those entities that are both defined, and provide the only definition signature to another

is omitted). The partition labeled ideal cover shows the size of the minimal cover set in ratio to the an explicit cover, which is always equivalent to the task signature (i.e. $cov = \frac{|C|}{|\mathcal{S}|}$), while the two right hand side partitions present the result obtained with the greedy algorithms, in terms of the cover size ratio, and the computation time. In the Conference corpus, which contains small ontologies, neither approaches have come close to the ideal cover set (which was, on average 68.86%). The cover sets produced by greedy #2 and #3, on average were 81.85% and 82.06% of the task signature, respectively. With the exception of two cases (ontologies conference and edas), where #2 produced slightly larger cover sets than #3 (the difference in their average is 0.21%), the two algorithms have produced the same results. In the LargeBio corpus, on average, the size of the minimal cover set was 65.55% of task signature, thus with a 65.63% average, approach #2 performed significantly better than #2 (77.98%), and with only a 0.08% difference, it has nearly achieved the optimal solution in all large ontologies, i.e. the minimal cover. In terms of computation time, as expected, greedy #3 performed considerably faster in both datasets: in the Conference corpus, on average, greedy #3 has took 1.13 milliseconds to compute a cover set, while greedy #2 has completed the same task in 4.45 milliseconds; in the LargeBio corpus, greedy #3 needed 185.83 seconds, while #3 required 226.61 seconds to complete the search process.

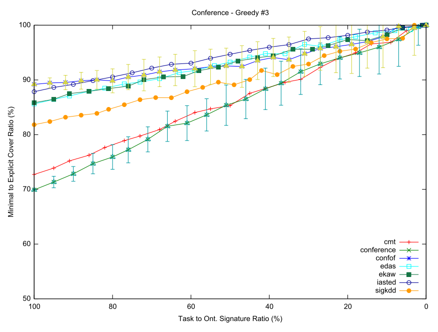
(Experiment 2: Varying signatures.) In the second experiment, we have varied the size of the task signature size, in order to assess the reduction provided by a minimal cover in comparison to the baseline (explicit cover), and to evaluate the size and the computation time difference between the two approaches, on a wider scale of possible tasks size settings. This experiment included 20 test cases for each ontology, where the task to ontology signature ratio ranged between 100% and 5%. Due to the fact that both approaches include an non-deterministic part, where a random choice is made to select an MDS from a set of equally good options (MDSs with the same value and cost scores), each test case was repeated 100 times[8]. Figure 2 presents the *cover set cardinality* results, where the y-axis represents the approximated minimal cover to task signature ratio ($\frac{|\mathcal{C}|}{|\mathcal{S}|}$), and the x-axis shows the task to ontology signature ratio ($\frac{|\mathcal{S}|}{|\mathsf{Sig}(\mathcal{O})|}$). Figure 2 shows the Conference dataset results, computed by approach #2 (a), and approach #3 (b); while the results of the LargeBio dataset are shown in (c) and (d), for approach #2 and #3, respectively (for brevity, error bars are only shown for the ontologies with the highest and lowest covers). In the small ontologies of the Conference corpus, approach #3 performed slightly better than #2, with all task signature sizes, while in LargeBio corpus, approach #2 performs considerably better for larger task signatures, however, with smaller task signatures (under 40%) approach #3 still provides better results. This is reinforced by Figure 2 (e) and (f), which shows the cover cardinality results for the same

entity, for example, the axiom $\mathsf{r} \equiv \mathsf{s}$ implies that both roles r and s are defined, however removing both entities from the ideal cover would make them both uncoverable.

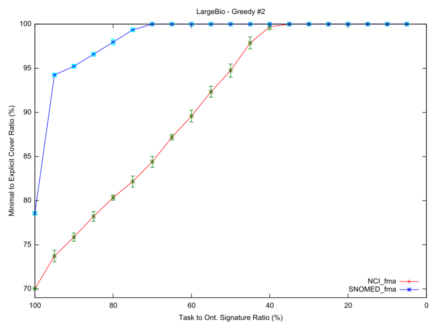
[8]We have tested several different repetition counts, and by comparing their relative standard deviation established that 100 repetition was sufficient for both datasets.



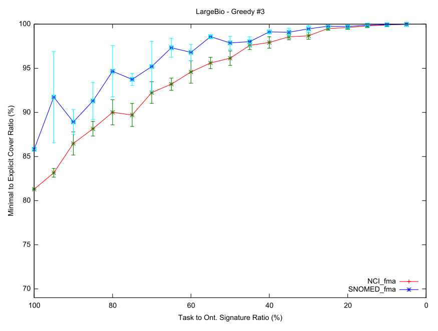
(a)



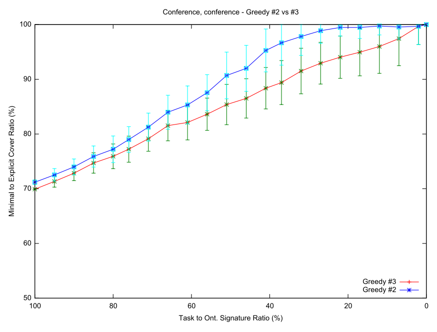
(b)



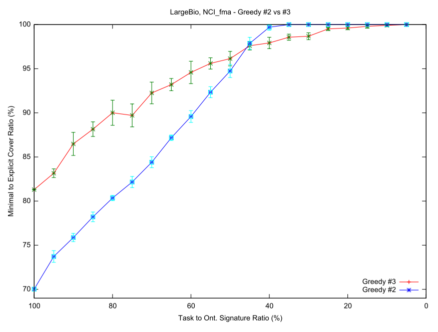
(c)



(d)



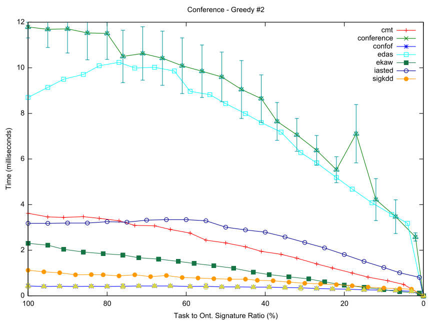
(e)



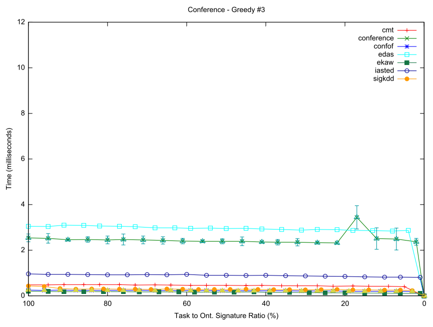
(f)

Figure 2: Cover set sizes, in the Conference (*a, b*) and LargeBio (*c, d*) corpus, obtained by the Greedy #2 (*a, b*) and #3 (*b, d*) approaches. Greedy #2 and #3 are compared in a Conference (*d*), a LargeBio ontology (*e*).

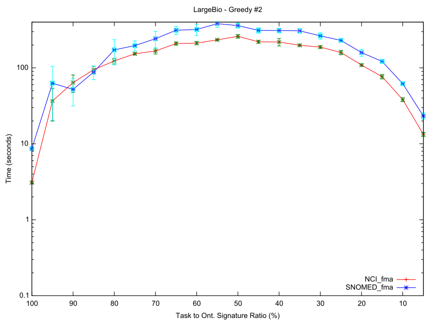
ontologies, produced by the two approaches (in the conference, and NCI_fma ontologies from the Conference, and the LargeBio corpus, respectively). Figure 3 presents the *computation time* results, where the y-axis shows the time, and the x-axis shows the task to ontology signature size ratio. Figure 3 presents the results of approach #2 in the Conference (a), and in the LargeBio corpus (c),



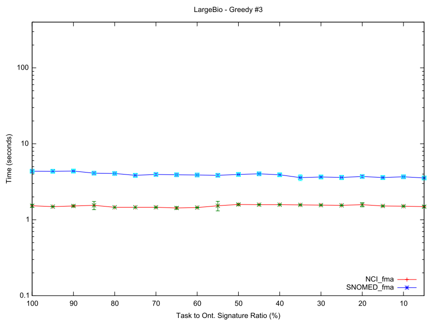
(a)



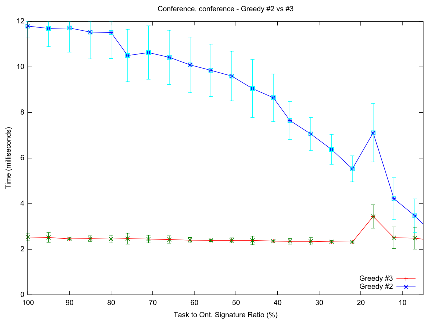
(b)



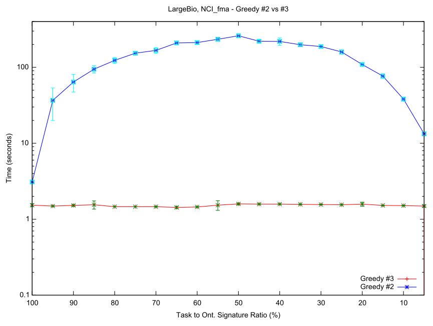
(c)



(d)



(e)



(f)

Figure 3: Computation times, in the Conference (*a, b*) and LargeBio (*c, d*) corpus, obtained by the Greedy #2 (*a, b*) and #3 (*b, d*) approaches. Greedy #2 and #3 are compared in a Conference (*d*), a LargeBio ontology (*e*).

while (b) and (d) shows the results of approach #3 in the two datasets. The computation time of approach #3 is almost a constant for a given ontology, while approach #2 seems to correspond to a bell curve (this is more visible in larger ontologies, as shown by (c)). In all ontologies of both datasets, approach #2 is significantly slower than #3, for example in the NCI_fma ontology (Figure

3, f), at 50% task to ontology signature ratio, #3 required 260.03 seconds, while #2 took only 1.59 seconds to complete. The same trend can be observed in the conference ontology (Figure 3, e).

6 Conclusions

In this paper, we have introduced and characterised the ontology signature coverage problem, which is a non-polynomial time problem that concerns whether an ontology signature can be covered by another signature, under a given ontology. Furthermore, we have presented and empirically evaluated two versions our novel approach, that by exploiting the notion of Beth-definability in Description Logic ontologies and using the pre-computed, complete set of different definition forms, provides a sub-optimal solution to the minimal signature cover problem. The evaluation has confirmed that, although the resulting covers are not always optimal, i.e non-minimal, they are significant improvements on the covers produced by naive approaches considering only explicit coverage.

References

[1] Franz Baader. *The description logic handbook: theory, implementation, and applications.* Cambridge University Press, 2003.

[2] Evert W Beth. On Padoa's method in the theory of definition. *Indagationes Mathematicae*, 15:330 – 339, 1953.

[3] Jérôme Euzenat, Pavel Shvaiko, et al. *Ontology matching*, volume 333. Springer, 2007.

[4] David Geleta, Terry R. Payne, and Valentina Tamma. Computing Minimal Definition Signatures in Description Logic Ontologies. Technical Report ULCS-16-003, University of Liverpool, 2016.

[5] Birte Glimm, Ian Horrocks, Boris Motik, Giorgos Stoilos, and Zhe Wang. Hermit: an owl 2 reasoner. *Journal of Automated Reasoning*, 53(3):245–269, 2014.

[6] Bernardo Cuenca Grau, Ian Horrocks, Boris Motik, Bijan Parsia, Peter Patel-Schneider, and Ulrike Sattler. OWL 2: The Next Step for OWL. *Web Semant.*, 6(4):309–322, November 2008.

[7] Eva Hoogland et al. *Definability and interpolation: Model-theoretic investigations.* Institute for Logic, Language and Computation, 2001.

[8] Matthew Horridge and Sean Bechhofer. The OWL API: A Java API for OWL ontologies. *Semantic Web*, 2(1):11–21, 2011.

[9] Ernesto Jiménez-Ruiz, Terry R Payne, Alessandro Solimando, and Valentina Tamma. Avoiding Alignment-based Conservativity Violations through Dialogue. In *Proc. OWLED*, volume 15, 2015.

[10] Nicolas Matentzoglu, Samantha Bail, and Bijan Parsia. A Snapshot of the OWL Web. In Harith Alani, Lalana Kagal, Achille Fokoue, Paul Groth, Chris Biemann, JosianeXavier Parreira, Lora Aroyo, Natasha Noy, Chris Welty, and Krzysztof Janowicz, editors, *The Semantic Web – ISWC 2013*, volume 8218 of *Lecture Notes in Computer Science*, pages 331–346. Springer Berlin Heidelberg, 2013.

[11] Terry R Payne and Valentina Tamma. Using Preferences in Negotiations over Ontological Correspondences. In *PRIMA 2015: Principles and Practice of Multi-Agent Systems*, pages 319–334. Springer, 2015.

[12] Gabrielle Santos, Valentina Tamma, Terry R Payne, and Floriana Grasso. Dialogue Based Meaning Negotiation. In *The 15th Workshop on Computational Models of Natural Argument (CMNA 2015)*, 2015.

[13] Evren Sirin, Bijan Parsia, Bernardo Cuenca Grau, Aditya Kalyanpur, and Yarden Katz. Pellet: A practical OWL-DL reasoner. *Web Semantics: science, services and agents on the World Wide Web*, 5(2):51–53, 2007.

[14] Balder Ten Cate, Enrico Franconi, and Inanç Seylan. Beth definability in expressive description logics. *J. Artif. Intell. Res.(JAIR)*, 48:347–414, 2013.

[15] Frank Van Harmelen, Annette Ten Teije, and Holger Wache. Knowledge engineering rediscovered: Towards reasoning patterns for the semantic web. In *Foundations for the Web of Information and Services*, pages 57–75. Springer, 2011.

[16] Moshe Y Vardi. *Fundamentals of dependency theory.* IBM Thomas J. Watson Research Division, 1985.

[17] Vijay V Vazirani. *Approximation algorithms.* Springer Science & Business Media, 2013.