Slicing microformats for information retrieval*

J. Guadalupe Ramos\textsuperscript{1}, Josep Silva\textsuperscript{2}, Gustavo Arroyo\textsuperscript{2}, and Juan C. Solorio\textsuperscript{1}

\textsuperscript{2} DSIC, Universidad Politécnica de Valencia
Camino de Vera s/n, E-46022 Valencia, Spain.
\{jsilva,garroyo\}@dsic.upv.es
\textsuperscript{1} Instituto Tecnológico de La Piedad
Av. Tecnológico 2000, La Piedad, Mich., México. CP 59300
\{guadalupe@dsic.upv.es, juancso@hotmail.com\}

Abstract. Microformats are a medium to incorporate semantic information into the web by means of standard tags which are enriched with particular attributes. They are a set of simple and open data formats built upon existing and widely adopted standards, hence, they are considered a pragmatic path to the Semantic Web.

In this work, we introduce a new method for information extraction from the semantic web. Basically we model the semantic information, which is contained in a set of web pages, in a formal graph like structure, namely, semantic network. Then, we introduce a novel slicing based technique for information extraction from semantic networks. In particular, the technique allows us to extract a portion—a slice—of the semantic network with respect to some criterion of interest. The slice obtained represents relevant information retrieved from the semantic network and thus from the semantic web. Our approach can be used to design novel tools for information retrieval and presentation, and for information filtering that was distributed along the semantic web.

1 Introduction

The Semantic Web is considered an evolving extension of the World Wide Web in which the semantics of information and services on the web is made explicit by adding metadata. Metadata provides the web contents with descriptions, meaning and inter-relations. The Semantic Web is envisioned as a universal medium for data, information, and knowledge exchange.

Two important technologies for developing the Semantic Web are already in use: The eXtensible Markup Language (XML) and the Resource Description Framework (RDF) among others \cite{1}. Nevertheless, efforts to extend the Web

\textsuperscript{*} This work has been partially supported by the Spanish \textit{Ministerio de Ciencia e Innovación} under grant TIN2008-06622-C03-02, by the \textit{Generalitat Valenciana} under grant GVPRE/2008/001, by the \textit{Universidad Politécnica de Valencia} (Programs PAID-05-08 and PAID-06-08) and by the Mexican \textit{Dirección General de Educación Superior Tecnológica}. 
with meaning have gained little traction. These initiatives have been bogged down by complexity and over-ambitious goals, or have simply been too much trouble to implement at a large scale (see, e.g., the discussion in [2]).

Recently, a new initiative has emerged that looks for attaching semantic data to web pages by using simple extensions of the standard tags currently used for web formatting in (X)HTML\(^1\), these extensions are called microformats [3, 4]. A microformat is basically an open standard formatting code that specifies a set of attribute descriptors to be used with a set of typical tags.

**Example 1.** Consider the following XHTML code that introduces information of a common personal card.

```xml
<h2>Directory</h2>
<p>Vicente Ramos</p>
  Software Development
  118, Atmosphere St.
  La Piedad, México
  59300
  +52 352 52 68499

</p>
</h2>
<h4>His Company</h4>
<a href="page2.html" xmlns="vcard" title="Company Page">Company Page</a>

Now, let us see the same information but taking into account the standard hCard microformat [5], which is useful for representing people, companies, organizations, and places data.

```xml
<h2>Directory</h2>
<div xmlns="vcard"
  class="vcard">
  <span class="fn">Vicente Ramos</span>
  <div class="org">Software Development</div>
  <div class="adr">
    <div class="street-address">Atmosphere 118</div>
    <span class="locality">La Piedad, México</span>,
    <span class="postal-code">59300</span>
  </div>
  <div class="tel">+52 352 52 68499</div>
</div>
<h4>His Company</h4>
<a href="page2.html" xmlns="vcard">Company Page</a>
</div>

The class property qualifies each type of attribute which is defined by the hCard microformat. The code starts with the required main class vcard and classifies the information with a set of classes which are auto-expllicative: fn describes name information, adr defines address details and so on.

\(^1\) XHTML is a sound selection because it enforces a well-structured format.
Microformats are a clever adaptation of semantic XHTML that makes it easier to publish, index, and extract semi-structured information like tags, calendar entries, contact information, and reviews on the web. Microformats have given rise to the so-called semantic web\(^2\) [6]. Indeed, they are considered a pragmatic path towards achieving the vision set forth for the Semantic Web [4].

Both the Semantic Web and the semantic web require new formal models, methods and tools to represent and query the embedded information. In the Semantic Web setting the semantic model is based on the notion of Ontology. An ontology defines and categorizes classes of concepts and their relations [1].

In contrast, in the semantic web setting, there does not exist a widely accepted model, and thus, the scientific community must do an effort to propose new approaches and formal methods. In this paper we propose the use of semantic networks which is a convenient simple model for representing semantic data; and we define a slicing technique for this formalism in order to analyze and filter the semantic web. A semantic network is often used as a form of knowledge representation; and it is formalized as a graph whose vertices represent concepts, and whose edges represent semantic relations between the concepts [7].

Once the information is modeled in a semantic network, formal methods for information extraction are needed to ensure a systematic and sound treatment of the information. Because semantic networks are implemented as a data structure that contains a considerable amount of information, its treatment is not a trivial task. We use a slicing technique to reduce the complexity of such a data structure.

Program slicing is basically a decomposition technique for the extraction of those program statements—the slice—that (potentially) affect the values computed at some point of interest. Program slicing was originally introduced by Weiser [8] and has now many applications such as debugging, program specialization [9], and XML filtering [10], see [11] for a survey.

Slicing techniques are (usually) based on a data structure called Program-Dependence Graph (PDG) [12]. The PDG allows slicers to find out which sentences of a program are related to some criterion (the so called slicing criterion) and thus they belong to the slice.

Therefore, program slicing could be a very convenient way to retrieve information from semantic networks with respect to some slicing criterion. Based on this idea, we introduce a program slicing inspired technique for information extraction from the semantic web. Our technique is based on an extension of semantic network, the indexed semantic network, that we conveniently formalize. This new notion of semantic network contains indexes that allow us to extract sub-graphs which are related to a specific topic. Roughly, the technique proceeds as follows: Firstly, an indexed semantic network is built from a collection of web pages. Then, we extract from the indexed semantic network the sub-net which is related to the slicing criterion. Finally, a slice is extracted from the semantic sub-net. The slices extracted from the sub-net represent the semantic information.

\(^2\) Note the different use along the paper of Semantic Web (in capital letters) and semantic web (in lowercase letters).
tion associated to the slicing criterion which, in turn, is the required information by the user.

The main contributions of this paper can be summarized as follows:

- We propose the use of semantic networks to represent semantic webs through the use of microformats, and show its usefulness.
- We extend standard semantic networks with indexes. This extension acts as an interface for the semantic network.
- We introduce a formal slicing based method for information recovering in semantic networks.

The rest of the paper is organized as follows. In Section 2, we overview the topic of semantic networks and recall the basic concepts related to them. In Section 3, we describe how semantic networks can be built from the semantic web. Furthermore, our slicing method for information extraction is formally introduced in Section 4. Finally, in Section 5 we review some related work and conclude.

2 Semantic Networks

The concept of semantic network is fairly old—in fact, the term of semantic network dates back to Ross Quillian’s works [13] where he introduced it as a way of talking about the organization of human semantic memory—in the literature of cognitive science and artificial intelligence. Nevertheless, it is a common structure for knowledge representation, which is useful in modern and different problems of artificial intelligence. For instance, in the recent Semantic Network Analysis Workshops [14, 15] many applications of this formalism were discussed, e.g., for social networks or hypertext networks.

A semantic network is a directed graph consisting of nodes which represent concepts and edges which represent semantic relations between the concepts. Sowa [16, 7] introduced a classification of semantic networks, in which the type of definitional networks emphasizes the subtype of is-a relation between a concept type and a newly defined subtype. This is the kind of semantic network that we will use in this paper. In Figure 1, we present a typical example.

3 From the semantic web to the semantic network

Roughly speaking, our method for semantic web information extraction is composed by two main steps:

1. Representing the information in the semantic web with a semantic network
2. Slicing the semantic network

In this section we focus on the first step, while the second one is subject of formal treatment in Section 4.
3.1 Constructing the semantic network from the semantic web

In order to represent semantic information in a semantic network we should decide what is the relevant information to be gathered and what we expect from a web information extraction query. In this work, we consider the microformats, i.e., classes as convenient entities for modeling, and then, for indexing or referencing.

Example 2. Let us consider again the semantic microformatted web page code of Example 1. We see that the semantic information is classified by using predefined classes which can embed other classes. For instance the main class vcard embeds the org class (to define an organization), the adr class (to indicate addressing data), etc. The next code shows a semantic web page composed by two main classes, i.e., vcard and vevent (for events microformatting [17]):

```xml
<h2>Staff</h2>
<div class="vcard">
  <span class="fn"><strong>Jessica Pechuck</strong></span>,
  <p class="role">COO</p>
  <div class="org">Software Development</div>
  <div class="adr">
    <div class="street-address">Atmosphere 118</div>
    <span class="locality">La Piedad, México</span>,
    <span class="postal-code">59300</span>
  </div>
  <div class="tel">+52 352 52 68499</div>
</div>

<h2>Personal Events</h2>
<div class="vevent">
  <span title="2009-02-25" class="dtstart">
    February 25, 2009
  </span>
  <span class="summary">Microformats use</span> at
  <span class="location">Main Street 126</span>
</div>
```
In this meeting we will discuss the use of microformats
</div>
</div>

In the example we see that microformats use classes to hierarchize the information; thus, classes should be the basic units of our semantic model. If we focus on the relations between classes we identify two kinds of relations, namely:

**strong relations** that are the relations which come from hypertext links between pages or sections of a page by using anchors.

**weak relations** that can be embedding relationships, for classes that embeds other classes or semantic relationships among classes of the same type, for instance, between two vcard.

*Example 3.* Consider again the microformatted code of Examples 1 and 2. From their classes we can build the semantic network depicted in Figure 2 (the grey parts of the figure do not belong to the semantic network and thus they can be ignored for the moment).

In the figure, the nodes of the first page are labeled with P1 and the nodes of the second page are labeled with P2. Thus, nodes (i.e., concepts) are unique. We observe three kinds of edges: The locality class from Example 1 is embedded in the adr class. Thus, there is an embedding relationship from node adr to node locality. Furthermore, vcard in P1 and vcard in P2 of the semantic web of Example 2 are linked by a semantic relationship. Besides, there is one strong hyperlink to P2 generated by the microformatted tag `<a class="url" href="page2.html">`. Observe that the graph only contains semantic information and their relations; and it omits content or formatting information such as the `<strong>` labels. Observe that we add to the graph two additional concepts, P1 and P2, which refer to web pages. This is very useful in practice in order to make explicit the embedding relation between microformats and their web page containers.

It is important to note that, in the previous example, similar classes participate in a cyclic relation. This is needed and useful in order to preserve semantic relations among information which is located in many source pages. The source pages to be analyzed in order to build the semantic network should be defined by the user or by the system, for instance, they could be the answer from a web searching engine. Another important design decision is related to the classes to be semantically linked. In the above example we took only main classes, i.e., vcard and vevent. It was a design decision not to link other classes such as adr.

4 A technique for information retrieval

In this section we formalize the notions related to semantic networks. Firstly, we define the semantic networks, then we introduce an extension called indexed
Fig. 2. Semantic network of Example 1 and Example 2.
semantic network. In addition, we define the notion of semantic sub-net. Once, the needed graph structures have been defined, we introduce the concept of backward and forward slicing of such a graphs, and enunciate our fundamental result of semantics preservation of slices. Finally, we show an algorithmic view of our slicing based method for information extraction. Without loss of generality, we only consider weak links (i.e., only semantic relations), thus we analyze semantic networks without taking into account the labels associated to the edges.

4.1 Extending semantic networks

We introduce first some preliminary definitions.

Definition 1 (semantic network). A directed graph is an ordered pair $G = (V, E)$ where $V$ is a finite set of vertices or nodes, and $E \subseteq V \times V$ is a set of ordered pairs $(v \rightarrow v')$ with $v, v' \in V$ called edges. A semantic network is a directed graph $S = (V, E)$ in which nodes have been labeled with names of web pages and microformatting classes of these pages.

As an example of semantic network consider the directed graph in Figure 2 (omitting the grey parts) where nodes are the set of microformatted classes provided by two semantic web pages.

A semantic network is a profuse mesh of information. For this reason, we extend the semantic network with an index which acts as an interface between the semantic network and the potential interacting systems. The index contains the subset of concepts that are relevant (or also visible) from outside the semantic net. It is possible to define more than one index for different systems and or applications. Each element of the index contains a key concept and a pointer to its associated node. Artificial concepts such as webpages (see P1 and P2 in Figure 2) can also be indexed. This is very useful in practice because it is common to retrieve the embedded (microformatted) classes of each semantic web page.

Let $K$ be a set of concepts represented in the semantic network $S = (V, E)$. Then, $\text{rnode} : (S, k) \rightarrow V$ where $k \in K$ (for the sake of clarity, in the following we will refer to $k$ as the key concept) is a mapping from concepts to nodes; i.e., given a semantic network $S$ and a key concept $k$, then $\text{rnode}(S, k)$ returns the node $v \in V$ associated to $k$.

Definition 2 (semantic index). Given a semantic network $S = (V, E)$ and an alphabet of concepts $K$, a semantic index $I$ for $S$ and $K$ is any set $I = \{(k, p) \mid k \in K$ and $p$ is a mapping from $k$ to $\text{rnode}(S, k)\}$

We can now extend semantic networks by properly including a semantic index. We call this kind of semantic network indexed semantic network (IS).

Definition 3 (indexed semantic network). An indexed semantic network $IS$ is a triple $IS = (V, E, I)$, such that $I$ is a semantic index for the semantic network $S = (V, E)$. 
Now, each semantic index allows us to visit the semantic network from a well-defined collection of entrance points which are provided by the rnode function.

**Example 4.** An IS with a set of nodes \( V = \{a, b, c, d, e, f, g\} \) is shown in Figure 3 (a). For the time being the reader can ignore the use of colors black and grey and consider the graph as a whole. There is a semantic index with two key concepts \( a \) and \( c \) pointing out to their respective nodes in the semantic network.

Similarly, the semantic network of Figure 2 has been converted to an IS by defining the index with four entries \( P1 \) (page1.html), \( P2 \) (page2.html), wcard and vevent and by removing the strong links. Thus, for instance, wcard entry points to the cycle of vcard nodes.

Given a graph \( G = (V, \mathcal{E}) \) and two nodes \( v_1, v_n \in V \), if there is a sequence \( v_1, v_2, \ldots, v_n \) of nodes in \( G \) where \((v_i, v_{i+1}) \in \mathcal{E}\) for \( 1 \leq i \leq n - 1 \), then we say that there is a path from \( v_1 \) to \( v_n \) in \( G \). Given \( u, v \in V \) we say that the node \( v \) is reachable from \( u \) if there is a path from \( u \) to \( v \).

**Definition 4 (semantic sub-net).** Let \( IS = (V, \mathcal{E}, I) \) be an indexed semantic network. Then, a semantic sub-net of \( IS \) with respect to concept \( k \) with \((k, p) \in I\) for some \( p \) is \( S_k = (V', \mathcal{E}') \) such that \( V' = \{\text{rnode}((V, \mathcal{E}), k)\} \cup \{v | v \in V \text{ and } v \text{ is reachable from } \text{rnode}((V, \mathcal{E}), k)\} \) and \( \mathcal{E}' = \{(u, v) | (u, v) \in \mathcal{E} \text{ and } u \in V'\} \).

**Example 5.** Figure 3 (a) shows in black color the semantic sub-net extracted from the whole IS with respect to concept \( c \).

**Definition 5 (semantic relationship).** Given a semantic network \( S = (V, \mathcal{E}) \) and a node \( v \in V \), the semantic relationships of \( v \) are the edges \( \{v \rightarrow v' | v' \in \mathcal{E}\} \).

We say that a concept \( v \) is semantically related to a concept \( u \) if there exists a semantic relationship \( (u \rightarrow v) \).

The semantic relations in our semantic networks are unidirectional. The semantics associated to the edges of a semantic network is not transitive because edges can have different meanings. Therefore, the semantic relation of Definition 5 is neither transitive.

Given a node \( n \) in a semantic network, we often use the term semantically reachable to denote the set of nodes which are reachable from \( n \) through semantic relationships. Clearly, semantic reachability is a transitive relation.

The following lemma ensures that an extracted sub-net does not change the semantics of its associated semantic network.

**Lemma 1.** Let \( N \) be the semantic sub-net extracted from the semantic indexed network \( IS = (V, \mathcal{E}, I) \) with respect to concept \( k \). Let \( n = \text{rnode}((V, \mathcal{E}), k) \). Then \( N \) is formed by \( n \) and all and only the semantically reachable nodes from \( n \), and all and only the semantic relationships of its nodes.

**Proof.** The claim trivially holds from the fact that \( N \) is a subset of \( IS \), i.e., \( N \) does not add new nodes nor edges to the semantic network; and also, the nodes and edges of \( N \) are all those nodes and edges in all the paths starting at the node \( n = \text{rnode}((V, \mathcal{E}), k) \). Therefore, \( n \) and all the semantically reachable nodes from \( n \) belong to \( N \); and all the semantic relationships of \( n \) and the nodes in the paths are preserved because the paths are only traversed forwards.
Fig. 3. a) A semantic sub-net. b) The sub-net’s adjacency matrix. c) A backward slice.

4.2 Semantic sub-net slicing

In this section we present a procedure that allows us to extract a portion of a semantic sub-net according to some criterion. The procedure uses an adjacency matrix to represent the semantic sub-net.

The adjacency matrix $m$ of a directed graph $G$ with $n$ nodes is the $n \times n$ matrix where the non-diagonal entry $m_{ij}$ contains 1 if there is an edge such that $m_i \rightarrow m_j$.

Example 6. Consider the semantic sub-net in Figure 3 (a). Node c has two directed edges, one to node d and other to node f. Thus, in the entry $m_{cd}$ and $m_{cf}$ we write 1, and 0 in the other cells.

Now, we are in a position to introduce our slicing based method for information recovering from semantic sub-nets. In traditional program slicing, the user selects a variable in a sentence of a program, and the slicer extracts the part of the program that has an influence over this variable. This can be done thanks to the use of a Program Dependence Graph [12] that stores the control and data dependences in a program. In our context, the slicing criterion is different to the standard program slicing technique which consists in a single point. Thanks to the introduction of indexes we can enrich our notion of slicing criterion by adding an extra level of information which allows us to perform slicing at two different levels. Firstly, we can select a concept in the index. From this concept we can extract a semantic sub-net as described before. Next, in the resultant semantic subnet we can select the node of interest. Hence, our slicing criterion consists of a pair formed by a key concept and a node. Formally:

Note that we could write a label associated to the edge in the matrix instead of 1 in order to also consider other relationships between nodes.
Definition 6 (slicing criterion). Let IS = (V, E, I) be an indexed semantic network. Then a slicing criterion \( C \) for IS is a pair of elements \((k, v)\) such that \((k, p) \in I\) for some \( p \), \( v \in V'\) and \( S_k = (V', E')\) is the semantic sub-net of IS with respect to concept \( k \).

Intuitively, the slicing criterion contains the concept of interest, from which we can extract a relevant sub-net, and a single node of the computed sub-net. This node is a particular microformatting class with the semantic information of interest reachable through semantic relations. Given a semantic sub-net, we can produce two different slices by traversing the sub-net either forwards or backwards from the node pointed out by the slicing criterion. Each slice gives rise to different semantic information.

Example 7. Consider the slicing criterion \((c, d)\) for the IS in Figure 3 c). The first level of slicing uses \( c \) to extract the semantic sub-net highlighted with black color. Then, the second level of slicing performs a traversal of the semantic sub-net either forwards or backwards from \( d \). In Figure 3 c) the backward slice contains all nodes whereas the forward slice would only contain \( \{d, f, g\} \).

Both backward slicing [8] and forward slicing [18] are well-known and widely used techniques in the literature (see, e.g., [10, 19, 20]). In our context, they can be used to distinguish between two different semantic relations: While backward slicing produces more general information (i.e., the classes to which the slicing criterion belongs), forward slicing produces specialized semantic relations (i.e., the classes which belong to the slicing criterion).

Example 8. Consider the semantic network in Figure 2 together with the slicing criterion \((P1, adr:P1)\). With \( P1 \) we can perform the first level of slicing to recover a semantic sub-net which is composed by the nodes \( \{P1, vcard:P1, vcard:P2\} \) and all of their descendant (semantically reachable) nodes. Then, from node \( adr:P1 \) we can go forwards and collect the information related to the address or backwards and collect nodes \( vcard:P1, P1 \) and \( vcard:P2 \). The backward slicing illustrates that the node \( adr:P1 \) is semantically reachable from \( P1, vcard:P1, \) and \( vcard:P2 \), and thus, there are semantic relationships between them. Hence, we extract a slice from the semantic network and, as a consequence, from the semantic web.

We can now formalize the notion of forward/backward slice for semantic sub-nets. In the definition we use \(-\rightarrow^*\) to denote the reflexive transitive closure of \(-\rightarrow\).

Definition 7 (forward/backward slice). Let IS = (V, E, I) be an indexed semantic network with \((k, p) \in I\) for some \( p \). Let \( S_k = (V', E')\) be the semantic sub-net of IS with respect to \( k \) and \( C = (k, node) \) a slicing criterion for IS. Then a slice of IS is \( S' = (V_1, E_1) \) such that

- **forward**: \( V_1 = \{node\} \cup \{v | v \in V' \text{ and } (node \rightarrow^* v) \in E'\} \)
- **backward**: \( V_1 = \{node\} \cup \{v | v \in V' \text{ and } (v \rightarrow^* \text{ node}) \in E'\} \)
**Input**: An indexed semantic network $IS = (V, E, I)$ and a slicing criterion $C = (k, \text{node})$ where $(k, p) \in I$ for some $p$

**Output**: A slice $S' = (V', E')$

**Initialization**: $V' := \{\text{node}\}, E' := \emptyset, \text{Visited} := \emptyset$

**Begin**

- Compute $S_k = (V_k, E_k)$ a semantic sub-net of $IS$ whose adjacency matrix is $M$

**Repeat**

- let $s \in (V' \setminus \text{Visited})$
- let $e := \text{column}(s, M)$

**For each** $s' \in V_k$ with $r = \text{row}(s', M)$ and $M_{r,e} = 1$

- $V' := V' \cup \{s'\}$
- $E' := E' \cup \{(s' \rightarrow s)\}$
- $\text{Visited} := \text{Visited} \cup \{s\}$

**Until** $V' = \text{Visited}$

**End**

**Return**: $(V', E')$

---

**Fig. 4.** An algorithm for semantic network backward slicing.

and $E_1 = \{(u \rightarrow v) \mid (u \rightarrow v) \in E' \text{ with } u, v \in V_1\}$

The algorithm of Figure 4 shows the complete slicing based method for information extraction from semantic networks. Roughly speaking, given an $IS$ and a slicing criterion, (i) it extracts the associated semantic sub-net, (ii) it computes the sub-net’s adjacency matrix, and (iii) it extracts (guided by the adjacency matrix) the nodes and edges that form the final slice.

The algorithm uses two functions $\text{row}(s, M)$ and $\text{column}(s, M)$ which respectively return the number of row and column of concept $s$ in matrix $M$. It proceeds as follows: Firstly, the semantic sub-net associated to $IS$ and the adjacency matrix of the sub-net are computed. Then, the matrix is traversed to compute the slice by exploiting the fact that a cell $M_{i,j}$ with value 1 in the matrix means that the concept in column $j$ is semantically related to the concept in row $i$. Therefore, edges are traversed backwards by taking a concept in a column and collecting all concepts of the rows that have a 1 in that column.

Now, we present the main result of the paper which states that the slicing method is correct with respect to semantic relationships.

**Theorem 1.** Let $IS = (V, E, I)$ be an indexed semantic network, $(k, \text{node})$ a slicing criterion such that $v = \text{rnode}((V, E), k)$, and $S'$ the backward slice returned by the semantic network backward slicing algorithm. Then, $S'$ is formed by $\text{node}$ and all the nodes from which $\text{node}$ is semantically reachable in the semantic sub-net induced by $k$. Moreover, all and only the semantic relationships of the nodes in $S'$ that appear in $IS$ also appear in $S'$.

**Proof.** Firstly, $S'$ is extracted from the semantic sub-net $S_k = (V', E')$ computed with respect to $k$ with $\text{rnode}((V, E), k)$. Moreover, by Lemma 1 we know that all
the nodes in $S_b$ are nodes of $IS$ and they all keep their semantic relationships. In addition, we know by Definition 6 that $n$ belongs to $V'$. Then, since the algorithm only collects nodes which are transitively connected to $n$, we can ensure that node and all the nodes from which it is semantically reachable are in $S'$. Moreover, all edges in the paths also belong to the slice, and hence, all the semantic relationships of the nodes in $S'$ that appear in $IS$ also appear in $S'$. Furthermore, $S'$ only collects the relations participating in the paths from node and thus only the semantic relationships of the nodes in $S'$ that appear in $IS$ also appear in $S'$.

Ongoing practical approach: In order to demonstrate the usefulness of our approach we have implemented some tools for discovering and extracting the semantic relationships among web pages. The current prototype is able to analyse a complete web site by traversing its hyperlinks and identifying semantic relations between web pages. As an example, Table 1 shows the collection of microformats found in a set of web pages. Concretely, we launched several queries to the Google web search engine and took the first eight links for each query. The tool automatically analysed Google’s results, and it found out their microformats in the semantic relations between the web pages. Certainly, there are notable efforts to extract microformats from web pages [21], and to filter HTML documents [22]; however current approaches only focus on single web pages, and thus, they ignore the relations between data which is located in different web pages.

<table>
<thead>
<tr>
<th>Google query</th>
<th>web pages</th>
<th>event</th>
<th>word</th>
<th>geo</th>
<th>resume</th>
<th>review</th>
</tr>
</thead>
<tbody>
<tr>
<td>event sport upcoming “New York”</td>
<td>8</td>
<td>14</td>
<td>21</td>
<td>6</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>restaurant Paris food</td>
<td>8</td>
<td>0</td>
<td>20</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>“medical services” madrid hospital</td>
<td>8</td>
<td>0</td>
<td>17</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>hotel London quality room downtown</td>
<td>8</td>
<td>0</td>
<td>15</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>personal service “Los Angeles” street</td>
<td>8</td>
<td>0</td>
<td>13</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>song author</td>
<td>8</td>
<td>0</td>
<td>13</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td><strong>TOTALS</strong></td>
<td><strong>48</strong></td>
<td><strong>11</strong></td>
<td><strong>31</strong></td>
<td><strong>31</strong></td>
<td><strong>0</strong></td>
<td><strong>0</strong></td>
</tr>
</tbody>
</table>

5 Related work and conclusions

In [23], three prototype hypertext systems were designed and implemented. In the first prototype, an unstructured semantic net is exploited and an authoring tool is provided. The prototype uses a knowledge-based traversal algorithm to facilitate document reorganization. This kind of traversing algorithms is based on typical solutions like depth-first search and breadth-first search. In contrast, our IS allow us to optimize the task of information retrieval.

[24] designed a particular form of a graph to represent questions and answers. These graphs are built according to the question and answer requirements. This is in some way related to our work if we assume that our questions are the slicing criteria and our answers are the computed slices. In our approach, we conserve
a general form of semantic network, which is enriched by the index, so, it still
permits to represent sub-graphs of knowledge.

To the best of our knowledge this is the first program slicing based approach
to extract information from the semantic web. The obtained answers are seman-
tically correct, since, the information extraction method follows the paths of the
source semantic tree, i.e., the original semantic relationships are preserved. Fur-
thermore, semantic relationships contained in sets of microformatted web pages
can also be discovered and extracted.

Program slicing has been previously applied to data structures. For instance,
Silva [10] used program slicing for information extraction from individual XML
documents. He also used a graph-like data structure to represent the documents.
However semantic networks are a much more general structure, that could con-
tain many subgraphs, while XML documents are always a tree-like structure. In
contrast to this method, our approach can process groups of web pages.

This method could be exploited by tools that feed microformats. Frequently,
these tools take all the microformats in the semantic web and store them in their
databases in order to perform queries. Our representation improves this behavior
by allowing the system to determine what microformats are relevant and what
microformats can be discarded. Another potential use is related to automatic
information retrieval from websites by summarizing semantic content related to
a slicing criterion. Similarly, web search engines could use this method to be able
to establish semantic relations between unrelated links.

To summarize, we have introduced an approach for information extraction
from the semantic web. This approach is based on program slicing, and has many
potential applications in the design of modern tools for information extraction.

References
2. T. Çelik. What’s the Next Big Thing on the Web? It May Be a Small, Simple
   Thing - Microformats. Knowledge@Wharton, 2005.
   2009.
   WWW ’06: Proceedings of the 15th International Conference on World Wide Web,
5. hCard. Simple, Open, Distributed Format for Representing People, Companies,
   Slicing. In Proc. of the Workshop on Curry and Functional Logic Programming