Semantic Network Skeletons - A Tool to Analyze Spreading Activation Effects

Kerstin Hartig and Thomas Karbe

Institute for Software Engineering - TU Berlin
10587 Berlin, Germany
Email: {kerstin.hartig, thomas.karbe}@tu-berlin.de

***Abstract*—Spreading Activation algorithms are a well-known tool to determine relevance of nodes in a semantic network. Although often used, the configuration of a spreading activation algorithm is usually very problem-specific, and experience-driven. There are practically no guidelines or tools to help with the task. In this paper, we present semantic network skeletons, which are essentially a structural summary of a semantic network. We show how to derive the skeleton from a given semantic network, and how to derive conclusions about good configurations from it. Our results are then demonstrated in a case study from the automotive domain.**

Keywords–Spreading Activation; Information Retrieval; Semantic Network; Advisory System.

I. INTRODUCTION

Spreading Activation algorithms are a long-known tool to determine relevance of nodes in a semantic network. Originally from psychology, they have been used in many other application areas, such as databases, artificial intelligence, biology, and information retrieval [1].

All spreading activation algorithms follow a basic pattern: chunks of activation are spread pulsewise from nodes to neighbor nodes, which marks those nodes as being relevant to a certain degree. However, practically, each known implementation differs in many details, such as the amount and distribution of activation. Whether a specific configuration for such an algorithm is good or bad depends largely on two factors: the problem to be solved by spreading, and the structure of the underlying semantic network. Although there are many working examples of such algorithms, until now there are almost no guidelines on how to achieve a good configuration.

In this paper, we aim to gain insights on spreading configurations by analyzing the structure of the semantic network. In many cases a semantic network consists of many thousands of nodes and edges, and is therefore hard to comprehend. We present a tool called *semantic network skeleton* that summarizes basic structural information of a semantic network and, thus, focuses on a few essential pieces of information.

In Section II, we will give a short summary about semantic networks and spreading activation. In Section III, we will introduce semantic network skeletons formally and visually, followed by a description of the necessary steps to retrieve skeletons from a semantic network in Section IV. Now, that we have network skeletons available, we can use them to analyze the underlying semantic network and to derive spreading effects for different configuration decisions. In Section V, this skeleton analysis for preconfiguration optimization will be presented. Section VI is dedicated to a case study on an advisory systems in the automotive domain. We finish the paper with conclusions and an outlook on future research potentials regarding semantic network skeletons.

II. BASICS AND RELATED WORK

We apply spreading activation as semantic search technique on semantic networks. Therefore, we shed some light on both concepts.

A. Semantic Network

Historically, the term semantic network had its origin in the fields of psychology and psycholinguistics. Here, a semantic network was defined as an explanatory model of human knowledge representation [2][3]. In such a network, concepts are represented by nodes and the associations between concepts as links. Generally, a semantic network is a graphic notation for representing knowledge with nodes and arcs [4][5]. Notations range from purely graphical to definitions in formal logic.

Technically, among others semantic networks can be described by RDF and RDF Schema (RDFS). The RDF data model [5] is defined to be a set of RDF triples whereas each triple consists of a subject, a predicate and an object. The elements can be Internationalized Resource Identifiers (IRI), blank nodes, or datatyped literals. Each triple can be read as a statement representing the underlying knowledge. A set of triples forms an RDF Graph, which can be visualized as directed graph, where the nodes represent subject and object and a directed edge represents the predicate [5].

According to the RDF Specification [5], resources such as IRI and literals carry a particular meaning whereas blank nodes stand for anything. Therefore, statements containing blank nodes denote the existence of something with the statements predicate. In contrast, statements without blank nodes mean the relationship between concrete resources holds.

For the rest of this paper, we use the terms RDF and semantic network interchangeably.

B. Spreading Activation

Spreading activation, like semantic networks, has a historical psychology and psycholinguistic background. It was used as a theoretical model to explain semantic memory search and semantic preparation or priming [2][3][6].

Over the years, spreading activation evolved into a highly configurable semantic search algorithm and found its application in different fields. In a comprehensive survey, Crestani examined different approaches to the use and application of spreading activation techniques, especially in associative information retrieval [1]. Spreading Activation is capable of both identifying and ranking the relevant environment in a semantic network.

The processing of spreading activation is usually defined as a sequence of one or more iterations, so-called pulses. Each node in a network has an activation value that describes its current relevance in the search. In each pulse, activated nodes

 ISBN: 978-1-61208-472-5

spread their activation over the network towards associated concepts, and thus mark semantically related nodes [1]. If a termination condition is met, the algorithm will stop. Each pulse consists of different phases in which the activation values are computed by individually configured activation functions. Additional constraints control the activation. Fan-out constraints limit the spreading of highly connected nodes because a broad semantic meaning may weaken the results. Path constraints privilege certain paths or parts of them. Distance constraints reduce activation of distant nodes, because distant nodes are considered to be less associated to each other. There are many other configuration details such as decays, thresholds, and spreading directions.

A challenge, mentioned in spreading activation related research is the tuning of the parameters, e.g., values associated with the different constraints as well as weighting or activation functions [7]. For evaluation of the prototype WebSCSA (Web Search by Constrained Spreading Activation) in [7], values and spreading activation settings are identified experimentally, empirically, or partly manually according to the experiments requirements. Álvarez et al. developed a framework for the application and configuration of spreading activation over RDF Graphs [8]. They state that a deep knowledge of the domain and the semantic network is necessary and domain-specific customization configuration is needed. It is a known fact that spreading activation configuration has a huge impact on the quality of the spreading results. Currently, there exists no systematic approach for the determination of proper configuration settings. Moreover, not even guidelines for the appropriate configuration are available to potential users. There is a lack of systematic analyses of the impact and interaction of different settings and parameters. The semantic network skeleton presented in this paper aims at facilitating such analyses in order to gain helpful insights and support appropriate configurations.

III. SEMANTIC NETWORK SKELETON

As stated before, proper configuration of a spreading activation algorithm is a challenging task. One important influencing factor for a good configuration is the structure of the underlying semantic network. Often however, semantic networks tend to be very large, and therefore hard to comprehend.

In this paper, we propose a tool called *semantic network skeleton*, which is supposed to summarize the structure of a semantic network. Therefore, using a skeleton shall make it easier to comprehend their structural properties and draw conclusions for configurations.

A. Skeleton Introduction

A skeleton of a semantic network is a directed graph that has been derived from a semantic network. We will call the semantic network from which the skeleton has been derived the *source (network)*.

Generally spoken, the skeleton shall represent the semantic structure of the source. Therefore, similar nodes and edges are grouped and represented by single node representatives and edge representatives in the skeleton. Thus, the skeleton hides all the parts of the source which are similar, and it makes the structural differences in the network more explicit.

Often, a semantic network contains also nodes and edges that carry little semantic value and therefore should be ignored by a spreading activation algorithm. An example are blank nodes, which by definition carry no specific meaning. Therefore, before creating a skeleton from a source, one first has to define the *semantic carrying* set of nodes and edges. This choice is very problem-specific, and therefore cannot be generalized. We call the semantic carrying subnetwork of the source the *spread graph*.

Since the skeleton is based on the spread graph, it represents only semantic carrying nodes and edges. The skeleton usually contains three types of node representatives: those classes, instances, and literals. Since the relationships between instance node representatives carry the most structural information about the semantic network, we call this part the *skeleton core*.

B. Types of Semantic Network Skeletons

We distinguish between two types of skeletons regarding their completeness and detail level: the maximum and the effective skeleton of a network.

A *maximum skeleton* contains all potential nodes and relations of the source. It is comparable with a UML class diagram in the sense that it shows everything that is theoretically possible in that network. However, it does not transport any information about the actual usage of classes/instances in the source network. Therefore, the maximum skeleton might contain nodes and relationships that have never been instantiated in the source.

An *effective skeleton* represents the structure of a specific instance of a semantic network. Therefore, it contains only nodes and relations that are actually part of the source network. This means, that a class that is part of an RDF schema, but that has not been instantiated in a concrete instance of that RDF schema would have a node representation in the maximum skeleton, but not in the effective skeleton.

Comparing maximum and effective skeletons, we find advantages and disadvantages for both of them: The maximum skeleton is the more generalized skeleton version, and therefore it applies to many different network instances of the same RDF schema. However, its generality also means, that it carries less specific information about each single instance, and therefore, conclusions drawn from a maximum skeleton are weaker than those drawn from an effective skeleton. The effective skeleton is specific to one instance of a semantic network. Thus, it cannot be reused for other instances, but it results in more precise conclusions.

C. Annotations

While the skeleton structure helps to understand the basic structure of the source network better, a detailed analysis often requires more information: It might be useful to know, how many node or edges are subsumed by a node or edge representative in the skeleton; The average number of incoming or outgoing edges for all represented nodes could indicate a certain spreading behaviour; Maybe there are 10.000 edges of the same type subsumed by one edge representative, but actually they all originate in only 10 different nodes. To capture such (often numerical) information, skeletons can be enhanced by annotations. Typically, there are four types of annotations: those, that describe node or edge representatives and those that describe the source or target of an edge representative.

Since effective and maximum skeletons carry different information, this also applies to annotations on them. While annotations on an effective skeleton refer to a concrete network instance of an RDF Schema (e.g., the concrete count of instances of a node type), annotations on a maximum skeleton

 ISBN: 978-1-61208-472-5

describe potential values. Thus, an instance count could have the value $*$, meaning that any number of instances is possible.

D. Syntax

Let L be a set of labels. A Semantic Network Skeleton S is defined by

$$S = (N, E, s, t, l),$$

where

- N is a non-empty set of *node representatives*,
- E is a set of *edge representatives*,
- $s : E \to N$ is the *source map*,
- $t : E \to N$ is the *target map*, and
- $l : N \times E \to L$ is the labelling.

The node and edge representatives each represent a set of nodes/edges of the same type from the original semantic network. Each edge representative $e \in E$ has a source node representative $s(e)$ and a target node representative $t(e)$. Furthermore, all node and edge representatives have a label $l(n)/l(e)$ assigned.

Given a semantic network skeleton $S = (N, E, s, t, l)$, and let $n_1, n_2 \in N$, $e \in E$, $s(e) = n_1$, and $t(e) = n_2$. Then the triple

$$T = (n_1, e, n_2)$$

is called a *skeleton triple* of S. A skeleton triple represents all corresponding RDF triples of the source network.

It is often useful to annotate statistical values to node or edge representatives, or sources/targets of edge representatives. For a skeleton S the *skeleton annotation* A_S is defined as

$$A_S = (A_n, A_e, A_s, A_t),$$

where

- $A_n : K \times N \to V$ is the *node annotation*,
- $A_e : K \times E \to V$ is the *edge annotation*,
- $A_s : K \times E \to V$ is the *edge source annotation*, and
- $A_t : K \times E \to V$ is the *edge target annotation*.

Here, K stands for a set of *annotation keys*, and V stands for a set of *annotation values*.

E. Graphical notation

The graphical notation for the skeleton corresponds to the graphical notation of RDF Graphs. In Figure 1, the proposed graphical notation is depicted. A node representative $n \in N$ is represented by a circle with its label $l(n)$ denoted over the circle. An edge representative $e \in E$ is represented by an unidirectional arrow with its label $l(n)$ denoted next to the arrow center. An arrow must connect two circles, with the arrow start connecting to the circle that represents the source and the tip of the arrow connecting to the circle that represents the target. Annotations are denoted in the circles, or near the start, middle, or end of the arrow, depending on their annotation type (node, edge, edge source, or edge target annotation).

F. Formal notation of graphical example

A skeleton $S = (N, E, s, t, l)$ that contains among others the node and edge representatives depicted in Figure 1 would be formally denoted by

- the labels *Function, Malfunction, hasMalfunction* $\in L$,
- two nodes $n_1, n_2 \in N$ with $l(n_1) =$ *Function*, and $l(n_2) =$ *Malfunction*,

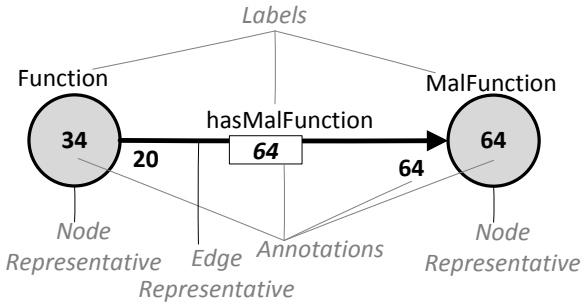

Figure 1. Graphical notation for skeletons.

- an edge $e \in E$ with $l(e) =$ *hasMalfunction*, $s(e) = n_1$, and $t(e) = n_2$.

Additionally, the skeleton annotation $A_S = (A_n, A_e, A_s, A_t)$ would contain the following mappings:

- $A_n(node\_count, n_1) = 34$,
- $A_n(node\_count, n_2) = 64$,
- $A_e(edge\_count, e) = 64$,
- $A_s(src\_rep, e) = 20$, and
- $A_t(tgt\_rep, e) = 64$.

Here, $node\_count$ and $edge\_count$ are the numbers of nodes/edges that have been subsumed by a node/edge representative. The source and target annotations src_rep and tgt_rep are the number of represented nodes, that are part of represented RDF triples. Thus, 20 of the 34 nodes represented by n_1 are connected to nodes represented by n_2 via an edge represented by e.

IV. SKELETON RETRIEVAL

Semantic network structures are as diverse as their potential applications and user-specific design decisions. Generally, semantic networks of all kinds can be subject to skeleton retrieval. However, transformation rules must guarantee that the semantic definition described in Section III-A holds.

In this paper, we focus on retrieving skeletons from semantic networks based on RDF and RDF Schema, more specifically, we utilize the RDF statements from the corresponding RDF Graph. Technically, different approaches are possible from successively parsing RDF Statements to utilizing query languages such as SPARQL [9]. A comprehensive technical description of potential transformations would go beyond the scope of this paper. Therefore, we rather offer an abstract method focusing on semantic compliance.

A. Creating Effective Skeletons

For retrieving the effective node and edge representatives from the spread graph, we apply the following abstract method.

1) Find all resources that are RDF classes. Each class becomes a node representative in the skeleton.
2) For each class find all its instances. All instances of one class are subsumed by one node representative.
3) Find all literals. They are subsumed by one node representative in the skeleton.
4) For each statement, add an edge representative (if not yet existent) for the predicate between the node representative of the statements subject and the node representative of the statements object in the skeleton.

Additionally, during the skeleton retrieval process, the desired annotation values can be computed.

 ISBN: 978-1-61208-472-5

We propose to subsume all literals by one node representative in the skeleton. In RDF, the literals of the class rdfs:Literal contain literal values such as strings and integers. A literal consists of a lexical form, which is a string with the content, a datatype IRI, and optionally a language tag. It is of course possible to further distinguish dependent on datatype or even analysing value equality instead of term equality. However, the content string of the lexical form seems to be most important and sufficient for the application.

B. Creating Maximum Skeletons

For creating a maximum skeleton, we apply the following method to retrieve node and edge representatives from a spread graph.

1) Find all resources that are RDF classes. Each class becomes a node representative in the skeleton. Additionally a node representative for instances of this class must be created. For resources that are classes themselves and subclasses of another class all properties must be propagated from its superclass.
2) Find all properties and their scope (range, domain). For each property add (if not existent yet) an edge representative from the node representative for the instances of the specified domain to the node representative for the instances of the specified range. For each subproperty p_1 of a property p_2 edge representatives must be created between all node representatives connected via p_2.

Again, required annotation values can be computed during the skeleton retrieval process.

V. SKELETON ANALYSIS FOR PRECONFIGURATION OPTIMIZATION

Structural network properties as well as spreading activation constraints and configuration settings affect spreading activation results. Each particular network property and spreading activation setting may have a particular effect. In combination they even have mutual effects. Knowledge about effects and their causes allows for pre-configuration analyses in order to optimize the settings to retrieve the desired effects.

Therefore, in this section, we present some basic skeleton pre-configuration analysis. First, we examine network properties as well as node and edge properties, that can be derived from skeleton annotation analysis. Furthermore, we introduce two potential effects. The analysis can easily be extended and deepened by attaching other useful annotations to the skeleton, developing different metrics and measures.

A. Network Properties - Distance Analysis

Due to its abstraction from a very complex background, the skeleton view gives a clear overview about potential spreading routes through the network. Routes between specific nodes are of special interest. For example, spreading allows for searching for specific node types initiated from some starting node(s). Thus, the *distance* between representatives of starting and search goal nodes denotes the minimal number of spreading pulses required to at least arrive at and possibly distribute any activation to search goal nodes and, therefore, show relevance between both nodes. In distance analysis, the distance between interesting pairs of node representatives can be calculated. Usually, consideration of the environment semantically contributes to the results and is wanted because a straight route may neglect additional useful relationships. Therefore, in order to gain a proper recommendation result, we propose balancing the pulse step configuration based on the distance. Moreover, the *diameter* of the skeleton (maximum path length between any pair of node representatives) stands for the minimal number of spreading pulses necessary to at least spread the entire source represented by this skeleton. With this knowledge, it is possible to prearrange an appropriate spreading pulse count, which may decrease efforts made for a well-spread solution graph.

B. Node and Edge Properties

The presented semantic network skeleton allows for extensible annotation options for customized analyses. Annotations presented in this paper aim at analyzing how and where nodes and edges are connected. The provided statistical information can be utilized for calculation measures describing the state of specific zones in the network, e.g., the zone around a specific node representative or the zone of a skeleton triple.

Common basic annotations are those presented in Section III-F, e.g., $node\_count$. From those, one can derive further more advanced annotations, of which we will present some of the most useful ones. The presented annotations mostly relate to a specific skeleton triple, but usually a global version is possible, too.

For a skeleton triple $T = (n_1, e, n_2)$, the *branching probability* of a node representative n_1 denotes the probability that a represented node of n_1 connects with any represented node of n_2 in the underlying network.

$$b_{prob}(n_1) = \frac{A_s(src\_rep, e)}{A_n(node\_count, n_1)} \tag{1}$$

The *effective average degree* $deg_{eff}(n_1)$ of a node representative n_1 is the average number of edges to which each represented node of n_1 is connected to.

For a skeleton triple T, the *branching ratio* of a node representative n_1 denotes the average number of edges each represented node of n_1 connects with any represented node of n_2 in the underlying network.

$$b(n_1) = \frac{A_e(edge\_count, e)}{A_n(node\_count, n_1)} \tag{2}$$

Node representatives with high branching ratios can be considered to be *high connectors*, which means their represented nodes are highly connected to neighbor nodes in the source network. In contrast, node representatives with low branching ratios can be considered to be *low connectors*, which means their represented nodes are sparsely connected to neighbor nodes in the source network. Branching ratio 1 indicates a *simple connector*. It means that averagely one edge per represented node connects with a neighbor.

C. Potential Effects of Network Properties on the Spreading Result

Spreading Activation effects result from the impact of configuration settings on network properties. An effect describes a behavior specific nodes or edges have while spreading, with respect to the given input.

Two important effects for pre-configuration analysis are the *distributor effect* and the *sink effect*. If a node generously spreads activation to a number of neighbor nodes above average the node operates as a *distributor*. If a node does not

(or only sparsely) spread activation to neighbor nodes the node operates as a *sink* (one-way street).

The distributor and sink effects correlate with connectivity information as well as configuration settings such as fan-out constraints and activation thresholds. As an example, a high connector can operate as a distributor and spread activation values via many connected edges. Assuming restrictive fan-out constraints in combination with a low threshold, a high connector may also operate as a sink because the high branching values affect potential distribution disadvantageously and the threshold may be unmanageable.

VI. CASE STUDY

The research of this paper contributes to the enhancement of recent research on an advisory system for decision-making support for hazard and risk analysis in the automotive domain, called *HARvESTer (Hazard Analysis and Risk assessment dEcision Support Tool)*. This advisory system will be utilized for first experiments with the skeleton presented in this paper. Below, the advisory system will be introduced as well as its skeleton creation and subsequent analyses.

A. Recommendation and Advisory System for Decision-making Support for Hazard and Risk Analysis

Since 2011, automotive companies have to adhere to the functional safety standard ISO 26262 [10]. One important safety activity described in the standard is the hazard analysis and risk assessment (HARA), which is strongly expert-driven, and therefore expensive, time consuming, and dependent from the individual experts opinion. In this analysis, experts examine the system under consideration with respect to its functions, possible malfunctions, and the consequences of those malfunctions in different situations. The result of the analysis is a certain safety level and safety goals to reduce the risk introduced by the new system to an acceptable level. According to [11], the experience of experts is still the main means to conduct a proper HARA. Without automation and tool support, a HARA becomes expensive and its results can become inconsistent with results of earlier analyses conducted by other experts.

Therefore, the advisory system automatically combines finished HARA projects and supporting information in a knowledge base and searches it for useful recommendations during a new HARA. Useful recommendations are found by applying spreading activation on the spread graph of the knowledge base. The algorithm determines the most relevant nodes for a specific user query with predefined starting nodes and a search goal, e.g., finding possible hazards for a specific function.

In our preliminary case study, we examined a spread graph of this advisory system. This network is based on RDF Graph and consists of more than 118.000 edges (representing 45 predicates) and more than 48.000 nodes. It contains data from more than 150 HARA projects. A part of this spread graph is depicted in Figure 2. Basically, some functions, malfunctions, hazards, and safety goals are shown. Our expectation is, that the hazard originating from the unintentional closing of the sun roof, i.e., *contusion of body parts*, may also be relevant for the function *close boot lid*, as well as the associated safety goal. For the recommendation query *Show safety goal recommendations for Ins_Function_2*, we expect the special semantic relevance to be detected.

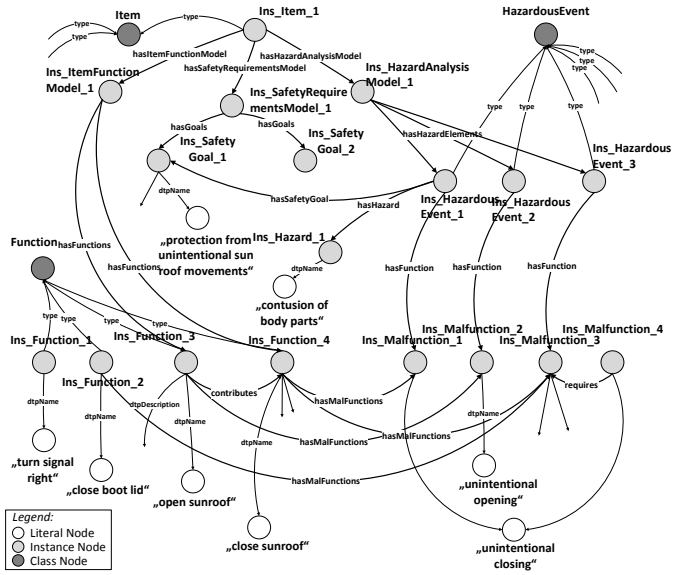

Figure 2. HARA Spread Graph - Extract from the Semantic Network.

B. Effective Skeleton of HARA Spread Graph

The spread graph shown partly in Figure 2 is the input for the skeleton creation process. Since the spread graph is an instance of a network model with concrete data, it does not necessarily contain all potential relations between each pair of nodes. Therefore, we retrieve an effective skeleton of the spread graph. Figure 3 depicts the effective skeleton of the HARA spread graph used for the presented advisory system. The used annotations in this skeleton are those described in Section III-F. The semantic center of the skeleton is the *skeleton core* which only contains representatives of instance nodes.

The effective skeleton of the network only consists of 37 node representatives and 94 edges. The maximum skeleton of the network only consists of 37 node representatives and 103 edges. The difference originates from the fact that in the concrete spread graph not all specified relationships are used at least once.

C. Analysis

Analyses are performed on the effective skeleton in Figure 3.

1) Distance Analysis: The diameter of the skeleton core in the example is 4. The diameter informs us about the necessary spreading steps for reaching each node representative at least once.

For the earlier mentioned query, we would start spreading at some node represented by $Ins\_Function$ and search for goal nodes represented by $Ins\_SafetyGoal$. The distance between those two node representatives is 3, which means that we have to spread for at least 3 pulses before any activation can reach the goal node. However, for more meaningful activation values, the influence of the other node representatives on the result is interesting, too. Therefore, a good spreading configuration ensures, that the activation values reached all node representative in the skeleton (6 pulses) before reaching the goal node representative (again 6 pulses). Altogether 12 pulses are therefore necessary. Of course, other spreading parameters could heavily influence the number of meaningful pulses, but at least the diameter provides some first insights.

 ISBN: 978-1-61208-472-5

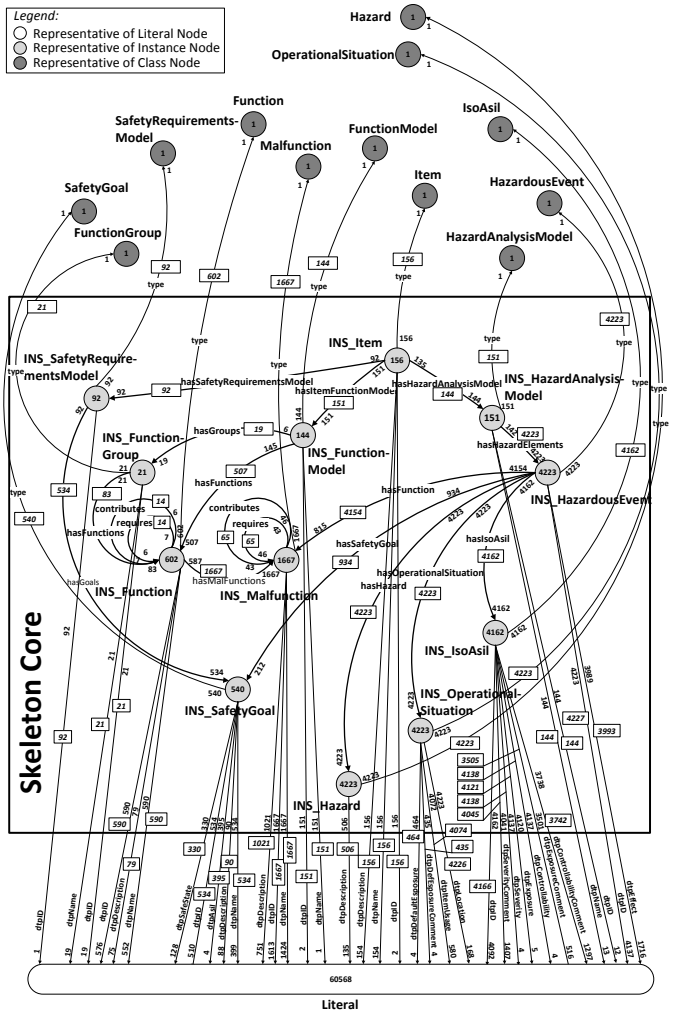

Figure 3. Skeleton of HARA Semantic Network.

2) Connectivity Analysis: We distinguish between high and low connector nodes. Low and high connector nodes are characterized by a high and low branching ratio, respectively. A high connector triple would be T_1=*(HazardousEvent, type, INS_HazardousEvent)* with a branching ratio of 4223, whereas T_2=*(INS_HazardousEvent, hasIsoAsil, INS_IsoAsil)* is a simple and low connector with a branching ratio of 1.

With T_1 being such a high connector, any configuration with a strong fan-out constraint would make it a sink, meaning that it would distribute almost no activation. In combination with an activation threshold, T_1 would probably distribute no activation at all. Since a sink does not contribute to the overall spreading result, such a configuration would make T_1 meaningless (except as a search goal).

VII. CONCLUSION AND FUTURE WORK

In this paper, we introduced semantic network skeletons as a new tool for analyzing semantic network properties and spreading activation configuration settings on semantic networks. The skeleton syntax was presented formally and visually. Furthermore, we described how a skeleton can be retrieved from a semantic network based on RDF Graph and explained analyses on network, node, and edge properties. Additionally, we described two effects observable while performing spreading activation and how skeleton analyses may support effect detection. We then presented our results in a case study on an advisory system for hazard analysis and risk assessment in the automotive domain and showed analyses on a real skeleton in order to optimize pre-configuration.

In this paper, only selected analyses were introduced. In future, description of more effects and their correlation with network properties and configuration settings can improve analysis results. Especially, new metric, measure, and ratio definitions can be helpful. A long term goal may be self-configured spreading activation, independently performing spreading activation without manual configuration. A short term goal is providing extensive guidelines for optimally configuring the spreading algorithm with respect to a given network. We furthermore plan on extended case studies experimenting with preconfigured spreading activation. Lastly, potential spreading activation micro-simulation can be examined. Skeletons can be valuable beyond utilization in the context of spreading activation. The compressed and summarized character of network skeletons might be useful for cognate disciplines also dealing with large semantic networks.

REFERENCES

[1] F. Crestani, “Application of Spreading Activation Techniques in Information Retrieval,” Artificial Intelligence Review, vol. 11, 1997, pp. 453–482.

[2] M. R. Quillian, “Semantic Memory,” in Semantic Information Processing, M. Minsky, Ed. MIT Press, 1968, pp. 216–270.

[3] A. M. Collins and E. F. Loftus, “A spreading activation theory of semantic processing,” Psychological Review, vol. 82, no. 6, Nov. 1975, pp. 407–428.

[4] J. F. Sowa, Principles of Semantic Networks: Exploration in the Representation of Knowledge, J. F. Sowa and A. Borgida, Eds. Morgan Kaufmann, Jan. 1991.

[5] “RDF 1.1 Concepts and Abstract Syntax,” W3C, W3C Recommendation, Feb. 2014.

[6] J. R. Anderson, “A Spreading Activation Theory of Memory,” Journal of Verbal Learning and Verbal Behavior, vol. 22, 1983, pp. 261–295.

[7] F. Crestani and P. L. Lee, “Searching the web by constrained spreading activation,” Information Processing and Management, vol. 36, no. 4, 2000, pp. 585–605.

[8] J. M. Álvarez, D. Berrueta, L. Polo, and J. E. Labra, “ONTOSPREAD: A Framework for Supporting the Activation of Concepts in Graph-Based Structures through the Spreading Activation Technique,” in Information Systems, E-learning, and Knowledge Management Research, ser. Communications in Computer and Information Science, vol. 278. Springer Berlin Heidelberg, 2013, pp. 454–459.

[9] “SPARQL 1.1 Query Language,” W3C, W3C Recommendation, Mar. 2013.

[10] ISO 26262 - Road vehicles - Functional safety, International Organization for Standardization, Nov. 2011.

[11] C. Maier, A. Schloske, and S. Bothe, “Studie zur Funktionalen Sicherheit in der Automobilbranche [Survey of Functional Safety in the Automotive Domain (ISO 26262)],” Fraunhofer Institute for Manufacturing Engineering and Automation (IPA), Tech. Rep., Mar. 2013.

 ISBN: 978-1-61208-472-5