# Trace Link Recovery Using Semantic Relation Graphs and Spreading Activation

Aaron Schlutter[1], Andreas Vogelsang[2]

**Abstract:** The paper was first published at the 28th IEEE International Requirements Engineering Conference in 2020. Trace Link Recovery tries to identify and link related existing requirements with each other to support further engineering tasks. Existing approaches are mainly based on algebraic Information Retrieval or machine-learning. Machine-learning approaches usually demand reasonably large and labeled datasets to train. Algebraic Information Retrieval approaches like distance between tf-idf scores also work on smaller datasets without training but are limited in providing explanations for trace links. In this work, we present a Trace Link Recovery approach that is based on an explicit representation of the content of requirements as a semantic relation graph and uses Spreading Activation to answer trace queries over this graph. Our approach is fully automated including an NLP pipeline to transform unrestricted natural language requirements into a graph. We evaluate our approach on five common datasets. Depending on the selected configuration, the predictive power strongly varies. With the best tested configuration, the approach achieves a mean average precision of 40% and a Lag of 50%. Even though the predictive power of our approach does not outperform state-of-the-art approaches, we think that an explicit knowledge representation is an interesting artifact to explore in Trace Link Recovery approaches to generate explanations and refine results.

**Keywords:** Traceability; Natural Language; Semantic Relation Graph; Spreading Activation

Trace Link Recovery (TLR) is a common problem in software engineering. While many tasks profit from links between related development artifacts, these are laborious to maintain manually and therefore rarely exist in projects. Automatic approaches aim for supporting engineers in finding related artifacts and creating trace links. Information Retrieval (IR) approaches build upon the assumption that if engineers refer to the same aspects of the system, similar language is used. Thus, tools suggest trace links based on Natural Language (NL) content.

State-of-the-art approaches use algebraic IR models (e.g., VSM, LSI), probabilistic models (e.g., LDA), or machine-learning approaches. These approaches rely on implicit models of key terms in documents (e.g., as points in a vector space or as probability distribution). Trace links are recovered based on similarity notions defined over these models. Therefore, it is hard to analyze and explain *why* specific trace links are identified in the model. Another drawback of machine-learning approaches is the need to train the models on reasonably large datasets. However, datasets usually consists of less than 500 artifacts (at least the ones used in scientific publications). [BRA14]

[1] Technische Universität Berlin, Ernst-Reuter-Platz 7, 10587 Berlin, Deutschland aaron.schlutter@tu-berlin.de
[2] Universität zu Köln, SSE, Weyertal 121, 50931 Köln, Deutschland vogelsang@cs.uni-koeln.de

doi:10.18420/SE2021_35

In our paper [SV20], we present a novel approach for TLR using semantic relations between parts of NL, stored in a semantic relation graph, and search trace links by Spreading Activation, a semantic search graph algorithm. While the approach is fully automated, it does not have any prerequisites with regard to the format or content of natural language and is scalable to various sizes of corpora. To improve the user confidence, we are able to generate an explanation between each query and target requirement by identifying and highlighting the contributing text passages.

The semantic relation graph is an explicit model of the knowledge represented in requirements. Our pipeline translates them automatically into vertices and edges that depict semantic parts of common NL (e.g., words and phrases within sentences, but also documents and corpora). The structure supports that single words have a greater distance (i.e., are less relevant) to a certain specification than phrases or whole statements. We use Spreading Activation [Ha19] to identify related requirements. The graph algorithm spreads activation in pulses over the vertices starting from a query vertex. Vertices with higher activation indicate higher relevance. Thus, we build a candidate list to sort all (reachable) targets based on their relations.

We applied the approach on 5 datasets [HHPL19] from different domains commonly used in TLR research, and evaluated in terms of *mean average precision* and *Lag* for answer sets of 5, 10, and 30 candidates. With the best tested configuration, our approach achieves an average precision around 40% and a Lag around 50%. While this performance does not outperform existing state-of-the-art approaches, the explicit representation of requirements content allows to "follow" a trace link through a chain of statements that may serve as an explanation why a trace link exists.

## Bibliography

[BRA14] Borg, Markus; Runeson, Per; Ardö, Anders: Recovering from a decade: A systematic mapping of information retrieval approaches to software traceability. Empirical Software Engineering (EMSE), 19(6):1565–1616, 2014. https://doi.org/10.1007/s10664-013-9255-y.

[Ha19] Hartig, Kerstin: Entwicklung eines Information-Retrieval-Systems zur Unterstützung von Gefährdungs- und Risikoanalysen. PhD thesis, Technische Universität Berlin, 2019. https://doi.org/10.14279/depositonce-8408.

[HHPL19] Huffman Hayes, Jane; Payne, Jared; Leppelmeier, Mallory: Toward Improved Artificial Intelligence in Requirements Engineering: Metadata for Tracing Datasets. In: Artificial Intelligence for Requirements Engineering (AIRE). IEEE, pp. 256–262, 2019. https://doi.org/10.1109/REW.2019.00052.

[SV20] Schlutter, Aaron; Vogelsang, Andreas: Trace Link Recovery using Semantic Relation Graphs and Spreading Activation. In: Requirements Engineering. IEEE, pp. 20–31, 2020. https://doi.org/10.1109/RE48521.2020.00015.