Pruning Cycles in UMLS Metathesaurus: A Neuro Symbolic AI Approach

Richard Wallace[1], Ravi Bajracharya[1], Jans Aasman[1] and Craig Norvell[1]

[1]*Franz Inc, Lafayette, CA, USA.*

Abstract

The Unified Medical Language System (UMLS) is a set of files and software tools that brings together many health and biomedical vocabularies and standards to enable interoperability between computer systems. Maintained by the National Library of Medicine, UMLS includes over 3 million concepts and 11 million unique terms from over 200 vocabularies, including SNOMED CT, ICD-10, LOINC, MeSH, and RxNorm. The UMLS aims to facilitate the development of computer-based systems that can effectively understand and use biomedical information from various sources.

However, several authors have diagnosed significant problems with the predicate relationships in UMLS, including inconsistencies, ambiguities, redundancies and, in the case of transitive relationships, cycles. This last issue, cycles, became our concern when we attempted to search for clinical codes in transitive closures of narrower UMLS concepts in an NQF (National Quality Forum) [1] query for early-onset diabetes quality metrics.

To address the challenges within UMLS, the study leverages a neuro-symbolic knowledge graph framework. This innovative approach integrates logic-based semantic reasoning, classical machine learning, and Large Language Models (LLMs) to clean the UMLS graph and eliminate cycles effectively. By validating existing hierarchical and equivalence relations through this framework, the project successfully prunes a significant number of cycles (66 %) from UMLS. This outcome demonstrates the potential of Neuro-symbolic AI in enhancing the reliability and precision of biomedical information systems. A cleaned up UMLS saves time for data scientists that can have more trust in the outcome of their queries and might even save lives by being more precise in analytics underlying treatments.

Keywords

UMLS, LLM, Neuro-Symbolic AI, Knowledge Graph

1. What is UMLS?

The Unified Medical Language System (UMLS) is a comprehensive and systematic effort initiated by the National Library of Medicine (NLM) to integrate disparate biomedical terminologies and ontologies [1]. It serves as a unifying resource by mapping relationships and concepts across various vocabularies, promoting semantic interoperability in the field of health analytics. UMLS knowledge sources underpin a wide variety of consequential informatics research and applications. The UMLS Metathesaurus (2020AA edition) now contains 4.28 million concepts and 15.5 million concept names (including some in 25 different spoken languages) from 214 vocabulary sources; the Semantic Network has 127 types and 54 relationships; the SPECIALIST Lexicon includes 983,420 lexical items; and there are many associated lexical programs and tools. Seen in hindsight, some features that add complexity to the UMLS Metathesaurus also make terminology data more FAIR (Findable, Accessible, Interoperable, and Reusable).

At its core, the UMLS employs a Metathesaurus, a large, multi-terminology database that aligns different biomedical vocabularies, such as MeSH, SNOMED CT, and LOINC. UMLS establishes associations between synonymous or related terms, enhancing the comprehension and retrieval of biomedical information.

The UMLS currently helps thousands of system developers and researchers to overcome variations in the way concepts are expressed, a task that remains critical to effective retrieval, analysis, aggregation,

[1]National Query Forum or NQF is a not-for-profit, nonpartisan, membership-based organization that works to improve healthcare outcomes, safety, equity, and affordability.

NLP4KGC'24: International Workshop on Natural Language Processing for Knowledge Graph Creation, September 17–19, 2024, Amsterdam, Netherlands

and semantically interoperable exchange of biomedical and health-related information and data. Use of the UMLS resources underpins systems collectively used by millions of scientists, health professionals, patients, and consumers—and by thousands of computer programs—every day.

2. Exposing Cycles with Synthea

Synthea is an open-source synthetic clinical data generator that simulates the medical history and health status of patients in a population [2]. It is designed to support the development, testing, and evaluation of health data analytics tools, models, and applications. Synthea generates synthetic patient data that includes demographics, encounters, allergies, conditions, laboratory results, medications, and claims. The realistic generated data represents a diverse patient population, making it useful for training machine learning algorithms, testing healthcare applications, and conducting research.

Our project utilized Synthea to build a patient-centric knowledge graph composed of 10 million patients in the state of California. The process was streamlined by AllegroGraph's FedShard [1] feature which was purpose-built to store instance data, such as EMR patient healthcare events, in combination with large knowledge bases such as UMLS. FedShard facilitates having core entities of interest (e.g. Patients) at the core with several layers of knowledge linked to the patient as 'events'. The events represent activities that transpire in a temporal context over their healthcare journey. By linking the diagnosis and procedure codes in the Synthea patient data with the UMLS knowledge base, we now have a Patient 360 Entity-Event Knowledge Graph (EEKG), that can predict patient outcomes and readily scales through sharding and parallelization of queries [3].

Our challenge started when we assumed that the UMLS structure could facilitate queries such as "Select all the patients with a more specific diagnosis than **nervous system disorder (C0027765)**." Unexpectedly, the transitive closure of the narrower relation from the concept "nervous system disorder" included some oddities such as "Heart failure","Burn Injury" and "Renal dysplasia". Eventually we diagnosed the problem to our assumption that the UMLS graph was acyclic, which led to our review of previous literature and research on the causes of and prior attempts to audit and eliminate the cyclic links in UMLS.

3. UMLS has many cycles

Errors and inconsistencies within UMLS are well documented. Even the Wikipedia article on UMLS [4] says:

> *Given the size and complexity of the UMLS and its permissive policy on integrating terms, errors are inevitable.*
>
> *Errors include ambiguity and redundancy, hierarchical relationship cycles (a concept is both an ancestor and descendant to another), missing ancestors (semantic types of parent and child concepts are unrelated), and semantic inversion (the child/parent relationship with the semantic types is not consistent with the concepts).*
>
> *These errors are discovered and resolved by auditing the UMLS. Manual audits can be very time-consuming and costly.*

Our discussion is focused entirely on how we identified and resolved the problem of cycles using LLMs in combination with our Patient Knowledge Graph. UMLS is intended to be a directed, acyclic graph, but the existence of cycles invalidates this assumption. More significantly, cycles limit the usefulness of UMLS in symbolic reasoning applications. Cycles can lead not only to nonsense conclusions, but combinatorial explosion as well.

[1] AllegroGraph is a horizontally distributed, multi-model (document and graph), entity-event knowledge graph technology solution from Franz Inc.[https://allegrograph.com/products/allegrograph/]. FedShard is a patented sharding and federation feature of AllegroGraph.

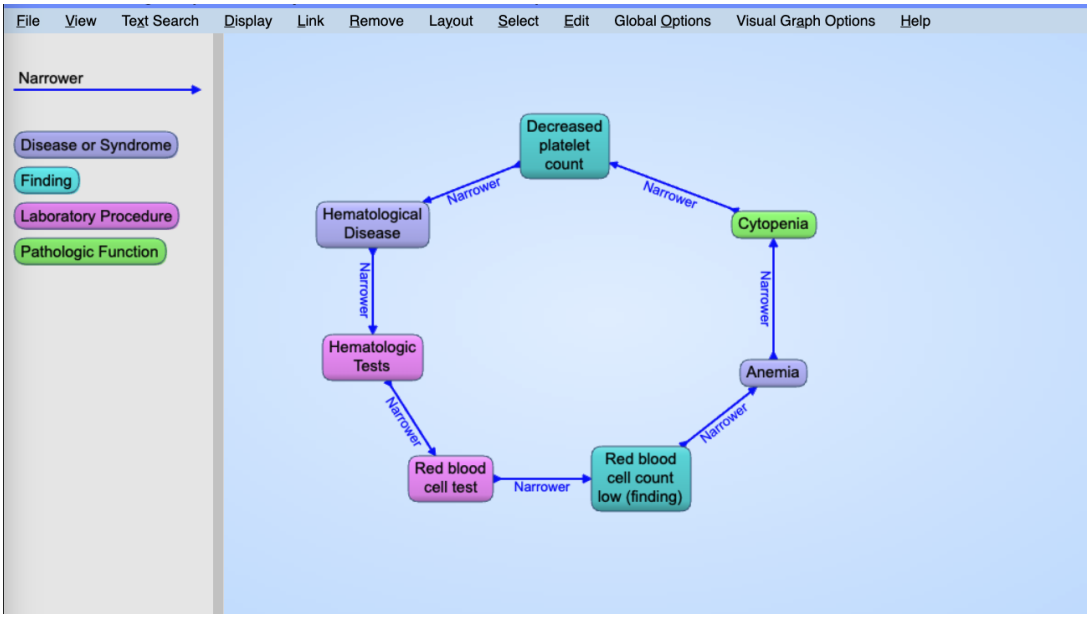

Figure 1: A sample cycle found in UMLS. The predicate "narrower" followed through a series of concepts, returns to the original concept. Such cycles make it difficult to use UMLS for automated deduction and reasoning.

4. Previous Work

Short of a labor-intensive audit of all hierarchical relationships in UMLS by expert medical ontologists, the previous solutions include a special issue of Journal of Biomedical Informatics focused on structural issues in the Unified Medical Language System (UMLS) [5]. The UMLS, designed by the National Library of Medicine, integrates biomedical terminologies into a unified knowledge representation. The UMLS, a complex knowledge base, resulted from collaborative efforts with external contributors. While much research has focused on content-related aspects, the editorial emphasizes the importance of addressing structural issues, including mathematical definitions and design considerations. The UMLS, with over 100 terminological sources, is a significant project, and the editorial directs readers to related research and bibliographies for further exploration. Several authors have published papers about the problem of inconsistencies and cycles in UMLS. They have proposed a variety of techniques to address the problem. We can classify these techniques into 3 broad classes: Auditing and Visualization, Rule-Based approaches, and Natural Language Processing.

4.1. Visualization and Auditing Tools

Perhaps the most sophisticated example, explored by Morrey et al. [6], is a graphical user interface offering several different visualization options to explore the very large graph of UMLS relationships.

The paper by Bodenreider et al [7] explores visualization techniques for assessing a partition of the Semantic Network (SN) they previously designed. Their methods reveal issues and confirm the validity of certain groups in the partition.

The approach described in Halper et al [8] introduces a methodology for auditing cycles in the parent relationship hierarchy of the UMLS. The goal is to identify and delete erroneous relationships within cycles, rather than simply breaking the cycles by deleting parent relationships. The focus is on cycles involving three concepts, and the paper presents hypotheses that suggest a high probability of locating an erroneous parent relationship within a cycle. These hypotheses are confirmed with statistical confidence, supporting the proposed auditing approach. The paper also suggests that cycles can serve as indicators of other non-structural inconsistencies that are difficult to detect automatically. An extensive auditing example is provided to illustrate how a cycle can reveal further inconsistencies. Overall, the approach presents a systematic and focused method for auditing cycles in the UMLS parent

relationship hierarchy.

4.2. Rule-based Analysis

Geller et. al developed an automated technique to uncover inconsistent and suspicious relationships in UMLS [9]. Cimino et al [10] explore inconsistencies in the correspondence between the hierarchies of UMLS Metathesaurus and UMLS Semantic Network, revealing potential errors in either hierarchy or in the assignment of concepts [10].

The approach described by Mougin et al [11] addresses the issue of inconsistencies in hierarchical relations within the UMLS Metathesaurus. Previous work proposed a formal approach to identify and eliminate circular hierarchical relations, resulting in a directed acyclic graph. However, this approach is only partially automated and implementation is complex. An alternative approach suggested here is to avoid loops by preventing nodes from being visited twice. The objective is to compare the effectiveness of the formal approach in eliminating cycles versus the simpler approach of avoiding them. The study found that 12% of concepts with descendants had differences between the two approaches, with the formal approach significantly reducing the number of descendants in these cases. The benefits in terms of semantic coherence are more nuanced and require further exploration.

4.3. Natural Language Processing

A few authors have reported natural language processing to improve UMLS. For example, the paper by Rindflesch et al [12] addresses Natural Language Processing challenges in recognizing IS-A relationships in biomedical texts. They demonstrate that combining underspecified syntactic analysis with UMLS leads to effective processing in the area of Chemicals and Drugs.

5. Rationale Behind Using LLM to Prune Cycles

Our approach may be framed as a hybrid of rule-based (symbolic) and natural language processing by Large Language Models (neural).

LLMs show remarkable textual reasoning capabilities including deductive reasoning which involves recognizing relationships between concepts, structural or semantic. With the capability of LLMs increasing rapidly, it can be a perfect tool to audit relationships between concepts. In this case, we are interested in the ability of LLM to determine if a particular concept is a narrower or broader concept in relation to a second concept. This is a perfect example of integration of neural and symbolic computation to enable reasoning from data.

The LLM is prompted to basically answer whether concept A is a narrower subset of concept B and generate a response of “yes” or “no”. Following quoted text is the actual prompt text submitted to an LLM, which in this case is a GPT 3.5 model, to determine membership of “Congestive Heart Failure” within “Heart Diseases” Concept:

> *“Does Congestive Heart Failure describe a narrower subset of Heart Diseases? Answer yes or no ”*

We can prompt LLM to answer and validate whether each of the relationships in the cycle represents a valid hierarchical relationship between them. If the answer is “no” for a particular edge, that edge can be removed to break the cycle.

6. Methodology

Neuro-symbolic AI represents an advanced approach in artificial intelligence that integrates the strengths of neural networks (for handling unstructured data, learning from vast datasets, and adapting to new information) with symbolic AI (for logical reasoning, understanding complex relationships, and

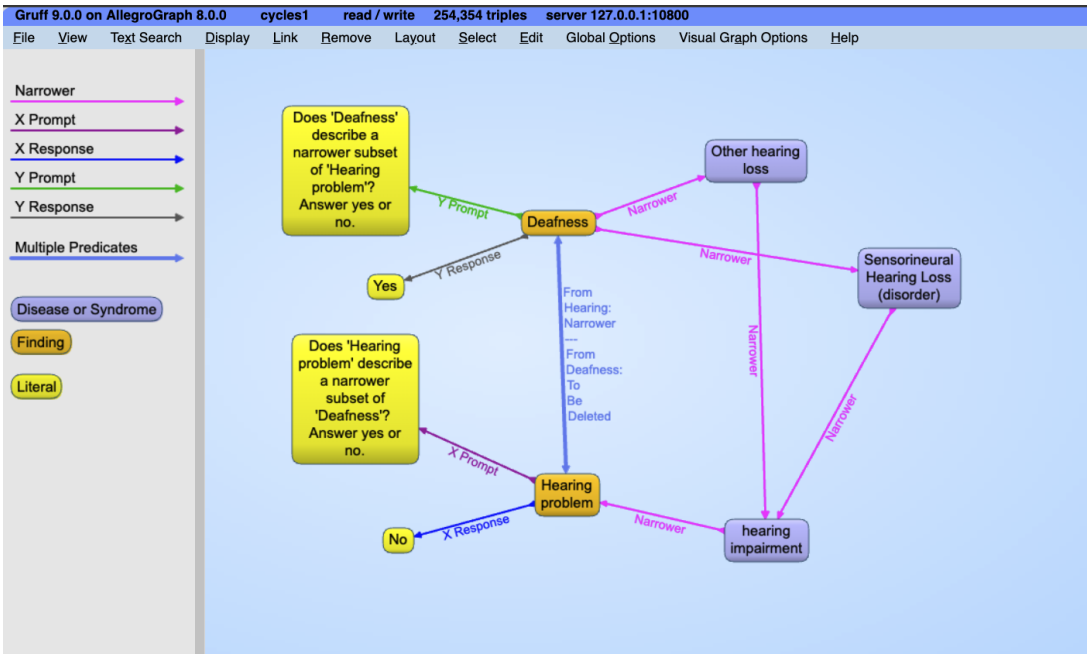

Figure 2: The 1-cycle between the concepts "Hearing Problem" and "Deafness" removed by the neuro-symbolic process. The LLM provides evidence that Deafness is narrower than Hearing Problem, but not vice-versa, so we remove one relation and remove the cycle. This removal has the side-effect also of breaking a longer 4-cycle.

processing structured data). This hybrid methodology is particularly relevant to the paper's focus on pruning cycles within the Unified Medical Language System (UMLS) to address issues of inconsistencies, ambiguities, and redundancies. By leveraging a Neuro-symbolic framework, the paper illustrates how combining machine learning, logical reasoning, and Large Language Models can effectively declutter and validate the complex web of biomedical terminologies and relationships in UMLS.

For the purpose of this paper, we will be looking at the transitive closure of the narrower relationships and using an LLM response to validate each of the relationships that form the cyclic narrower path in the graph. Following bullets highlight the steps involved in setting up the UMLS knowledge graph, counting cycles in the graph and then using a LLM in the mix to then mark the narrower relationships that need to be deleted to break the cycles:

1. Convert UMLS to RDF triples and store them in Allegrograph.
2. Select 1404 MTH concepts exactly matching 1404 SNOMED codes found in Synthea data.
3. Starting with 1-cycles, explore the UMLS graph for cycles starting with the MTH concepts, following all the 'narrower' links recursively. When a cycle is found, save all of its participating triples with a graph-id unique to that cycle, associated with a property indicating the cycle length.
4. Keep count of the number of cycles graph-ids.
5. For each narrower edge in the cycle, from concept Y to concept X, construct LLM prompts of the form "Does 'X' describe a narrower subset of 'Y'? Answer yes or no." and "Does 'Y' describe a narrower subset of 'X'? Answer yes or no."
6. Using a SPARQL query, mark the edge for deletion when the answer to the first question is 'Yes' and the second one is 'No.'
7. Remove the marked edges and re-run the procedure to count the number of cycles.
8. Repeat the process from step 2 by deleting edges from 2-cycles, 3-cycles and so on.

Following subsections elaborate more on the bullets above:

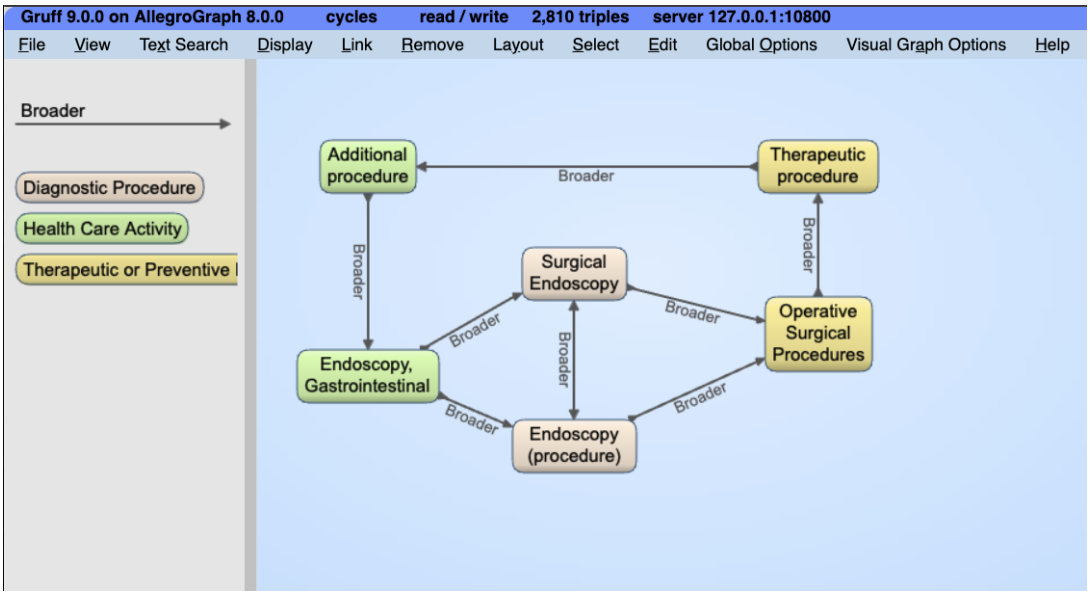

Figure 3: The nuances of counting cycles: cycles contain other cycles, cycles share relations with other cycles, and cycles often have multiple paths from a node back to itself.

6.1. Building UMLS Knowledge Graph

UMLS Metathesaurus is a component of UMLS that contains concept mappings, definitions, hierarchical relationships between concepts and also non-hierarchical semantic relationships between concepts as well. As such, a Knowledge Graph built using the standards based Resource Description Framework (RDF) representation of UMLS, such as AllegroGraph, is the optimal way to represent the network of relationships within UMLS. This representation also helps us to easily explore the transitive closures within the hierarchical relationships of UMLS.

To build the Knowledge Graph, we map UMLS metathesaurus to a SKOS (Simple Knowledge Organization System) ontology [13], which is a great fit for a concept-oriented vocabulary like UMLS. As of 2023AA release of UMLS, our Knowledge Graph has 3 million plus concepts in the core metathesaurus source and over 7 million concepts from other component sources like ICD10 and SNOMED. Of important note because we are using RDF, these concepts could be loaded as SKOS concepts in a SKOS scheme corresponding to the original source.

This approach is primarily based on the earlier work of P. Mirhaji at the University of Texas, Houston [14]. Following bullets highlight the method of creation of a UMLS SKOS knowledge graph according to this method:

1. UMLS concepts become SKOS concept with skos:prefLabel set to Metathesaurus preferred term of the concept.
2. PAR/CHD and "RB"/"RN" relationships from UMLS Metathesaurus are interpreted as SKOS:broader and SKOS:narrower relationships respectively.
3. UMLS semantic network types are used as a rdfs:type for concepts.
4. Concept matches and related concepts across SKOS schemes are connected using the SKOS:exactMatch and SKOS:relatedMatch predicates respectively.

The SKOS representation of UMLS not only makes the UMLS datasets FAIR (Findable, Accessible, Interoperable and Reusable) but also forms a baseline knowledge graph that can be enriched through other related data sources for any other specific use-cases.

6.2. Counting Cycles

What counts as a cycle? A→B→C→A is the same as the cycles B→C→A→B and C→A→B→C. UMLS has many cycles, but most of them are 1-cycles of the form A→B→A.

Many cycles contain other cycles as subcycles. The algorithm for counting cycles is depth-first search. It's difficult to optimize because even a node that has been 'visited' may still be explored when another path passes through it. To ensure the accuracy of our results we applied a full brute-force depth-first search for cycles, without optimizations, because we prioritized accuracy of the count over efficiency of the counting procedure. A lot of longer cycles contain 1-cycles as subcycles

6.3. SPARQL Query

The basic idea in step 3 above is to audit all the triples that belong to cycles, and save a new triple with same form as the first,

`?y skos:narrower ?x.`

but save it with unique graph-id so that we can count it and retrieve the cycle length of a cycle length for triples in the graph ?g with

`?g <http://franz.com/cycleLength> ?length.`

Following SPARQL query implements validation of 2-cycles in step 5 and also uses the LLM integrations available in AllegroGraph to query GPT models for "Yes" or "No" answers to the validation question in step 4 [2]:

```
PREFIX llm: <http://franz.com/ns/allegrograph/8.0.0/llm/>
PREFIX franzOption_openaiApiKey: <franz:sk-U01ABc2defGHIJKlmnOpQ3RstvVWxyZABcD4eFG5jiJKlmno>

INSERT { graph ?g {?y <http://franz.com/toBeDeleted> ?x.
                   ?x <http://franz.com/xPrompt> ?xPrompt.
                   ?y <http://franz.com/yPrompt> ?yPrompt.
                   ?x <http://franz.com/xResponse> ?xResponse.
                   ?y <http://franz.com/yResponse> ?yResponse.
                   }
       } WHERE {
  {SELECT ?x ?y ?g {
    graph ?g {
      ?y skos:narrower ?x.
      ?g <http://franz.com/cycleLength> 2.
      }
      FILTER NOT EXISTS {?y <http://franz.com/toBeDeleted> ?x}
  } ORDER BY RAND() LIMIT 100}
?x rdfs:label ?xlabel.
?y rdfs:label ?ylabel.
 bind(concat("Define ",?xlabel,": ") as ?xdefPrompt).
 bind(concat("Define ",?ylabel,": ") as ?ydefPrompt).
 bind(concat(
   "Does '",
   ?xlabel,
   "' describe a narrower subset of '",
   ?ylabel,
   "'?  Answer yes or no.") as ?xPrompt).
 bind(concat(
   "Does '",
   ?ylabel,
   "' describe a narrower subset of '",
   ?xlabel,
   "'?  Answer yes or no.") as ?yPrompt).
   bind(llm:response(?xPrompt) as ?xResponse).
   bind(llm:response(?yPrompt) as ?yResponse).
  FILTER (strstarts(lcase(str(?xResponse)), "no"))
  FILTER (strstarts(lcase(str(?yResponse)), "yes"))
}
```

[2] AllegroGraph 8.0.0 release has new features that integrate LLM features right into SPARQL queries through its flagship magic predicates, which enable extension of SPARQL predicates to support custom functions like LLM prompting for example. [https://franz.com/agraph/support/documentation/8.0.0/llm.html]

7. Result

Following table represents the reduction in cycles over successive execution of the SPARQL INSERT query above for various cycle lengths up to length 6.

Cycle Length	Original UMLS	Removed 2985 1-cycles	Removed 477 2-cycles	Removed 99 3-cycles
1	6014	3009	2882	2866
2	2582	1535	958	956
3	1903	963	637	522
4	1886	743	493	370
5	1796	719	476	235
6	1715	579	427	262
Total Cycles	15896	7548	5873	5211

We can make following observations based on the results table:

1. The SPARQL query is run successively for cycles of lengths from 1 to 6
2. We also notice on the first run we have the biggest reduction of 1-cycles but it also removes cycles of longer length in the process. We hypothesize that the removal of longer cycles makes some of the shorter cycles unreachable.
3. Ultimately with this approach we see a 67% reduction in cycles of length up to 6 in UMLS.

8. Conclusion

We used an LLM model to validate the transitive closure of the narrower relationships in the UMLS knowledge graph particularly with the aim to weed out cycles within the graph. We use AllegroGraph's latest integration of LLM features within a SPARQL query through magic predicates to validate narrower relationships within cycles of length up to 6 cycle lengths. Applying the validation results, we can prune the graph to make it more consistent. This process removed the number of cycles in our sample by 2/3 from almost 16,000 to just over 5,000.

This Neuro-symbolic AI pruning approach leverages the strengths of logic-based semantic reasoning, classical machine learning, and Large Language Models (LLMs) to declutter the UMLS graph by pruning a significant number of cycles. Specifically, the methodology combines the generation of synthetic patient data using Synthea, the construction of a patient-centric Knowledge Graph, and the utilization of LLMs to audit and validate hierarchical and equivalence relationships within UMLS. This results in a more reliable and precise system that could enhance data analytics in healthcare, potentially leading to better patient outcomes.

In healthcare applications, this methodology addresses a critical challenge: ensuring that the complex network of medical terminology and relationships within UMLS is accurate and free from logical inconsistencies. By successfully pruning cycles from UMLS, this approach not only improves the efficiency and trustworthiness of data queries within the system but also contributes to the broader goal of enhancing healthcare analytics and patient care through better data integrity and semantic interoperability.

9. Future Work

Future efforts on this topic shall focus on following areas:

1. Quantitative evaluation of how well the cycle deletions worked through evaluation of false positives and false negatives and other relevant metrics.
2. Apply the process to all narrower and broader edges in UMLS.
3. Apply the methodology to more broader types of relationships that form a transitive closure, semantic or otherwise.
4. Look into which other UMLS relationships can be validated with LLMs.

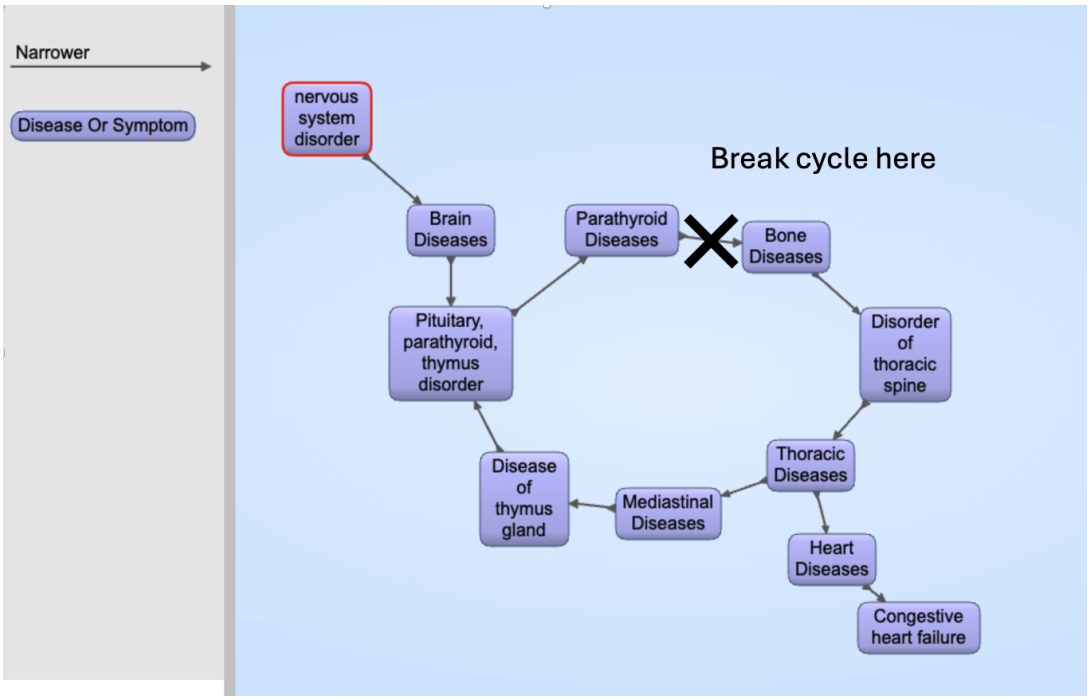

Figure 4: Complete Example illustrating the correction of deductive reasoning by breaking a cycle in UMLS. Initially the concept "Congestive Heart Failure" is, by transitivity, a narrower concept than "Nervous System Disorder". By removing a single 1-cycle, the inconsistency disappears and the network becomes suitable for symbolic reasoning.

References

[1] O. Bodenreider, The Unified Medical Language System (UMLS): integrating biomedical terminology, Nucleic Acids Research 32 (2004) 267D–270. URL: https://doi.org/10.1093/nar/gkh061. doi:10.1093/nar/gkh061.

[2] J. Walonoski, M. Kramer, J. Nichols, A. Quina, C. Moesel, D. Hall, C. Duffett, K. Dube, T. Gallagher, S. McLachlan, Synthea: An approach, method, and software mechanism for generating synthetic patients and the synthetic electronic health care record, Journal of the American Medical Informatics Association 25 (2017) 230–238. URL: https://doi.org/10.1093/jamia/ocx079. doi:10.1093/jamia/ocx079.

[3] R. Bajracharya, R. Wallace, J. Aasman, P. Mirhaji, Entity event Knowledge Graph for powerful health informatics, 2022 IEEE 10th International Conference on Healthcare Informatics (ICHI) (2022). URL: https://doi.org/10.1109/ichi54592.2022.00068. doi:10.1109/ichi54592.2022.00068.

[4] Wikipedia contributors, Unified medical language system, https://en.wikipedia.org/w/index.php?title=Unified_Medical_Language_System&oldid=1195686984, 2024.

[5] Y. Perl, J. Geller, Research on structural issues of the UMLS—past, present, and future, Journal of biomedical informatics 36 (2003) 409–413. URL: https://doi.org/10.1016/j.jbi.2003.11.005. doi:10.1016/j.jbi.2003.11.005.

[6] C. P. Morrey, J. Geller, M. Halper, Y. Perl, The Neighborhood Auditing Tool: A hybrid interface for auditing the UMLS, Journal of biomedical informatics 42 (2009) 468–489. URL: https://doi.org/10.1016/j.jbi.2009.01.006. doi:10.1016/j.jbi.2009.01.006.

[7] O. Bodenreider, A. T. McCray, Exploring semantic groups through visual approaches, Journal of biomedical informatics 36 (2003) 414–432. URL: https://doi.org/10.1016/j.jbi.2003.11.002. doi:10.1016/j.jbi.2003.11.002.

[8] M. Halper, C. P. Morrey, Y. Chen, G. Elhanan, G. Hripcsak, Y. Perl, Auditing hierarchical cycles to locate other inconsistencies in the UMLS, AMIA Annu. Symp. Proc. 2011 (2011) 529–536.

[9] J. Geller, C. P. Morrey, J. Xu, M. Halper, G. Elhanan, Y. Perl, G. Hripcsak, Comparing inconsistent relationship configurations indicating UMLS errors, AMIA Annu. Symp. Proc. 2009 (2009) 193–197.

[10] J. Cimino, H. Min, Y. Perl, Consistency across the hierarchies of the UMLS Semantic Network and Metathesaurus, Journal of biomedical informatics 36 (2003) 450–461. URL: https://doi.org/10.1016/j.jbi.2003.11.001. doi:10.1016/j.jbi.2003.11.001.

[11] F. Mougin, O. Bodenreider, Approaches to eliminating cycles in the UMLS metathesaurus: naïve vs. formal. AMIA, in: Annual Symposium proceedings. AMIA Symposium, 2005, pp. 550–554.

[12] T. C. Rindflesch, M. Fiszman, The interaction of domain knowledge and linguistic structure in natural language processing: interpreting hypernymic propositions in biomedical text, Journal of biomedical informatics 36 (2003) 462–477. URL: https://doi.org/10.1016/j.jbi.2003.11.003. doi:10.1016/j.jbi.2003.11.003.

[13] B. Miles, SKOS Simple Knowledge Organization System Reference, 2009. URL: https://www.w3.org/TR/skos-reference.

[14] P. Mirhaji, UMLS-SKOS: A Semantic Web Framework for Representation of Biomedical Terminological Knowledge using Simple Knowledge Organization System (SKOS), 2009. URL: https://ncbo.bioontology.org/umls-skos.