Adaptive Semantic Routing in Dynamic Environments

Stefan Jakob, Harun Baraki, Alexander Jahl, Eric Douglas Nyakam Chiadjeu, Yasin Alhamwy and Kurt Geihs

Distributed Systems Group, University of Kassel, Wilhelmshöher Allee 71 - 73, Kassel, Germany

Keywords: Knowledge Representation and Reasoning, Knowledge-based Systems, Semantic Routing.

Abstract: Digital cities increasingly depend on IT and communication infrastructure. They can be seen as large distributed systems consisting of heterogeneous and autonomous participants. Especially during emergency situations and natural disasters, crucial knowledge about injured people and damaged infrastructure has to be discoverable by rescuers. Since knowledge discovery is about finding semantic information, IP-based routing mechanisms have to rely on flooding as they have to query each node to locate matching information. This leads to additional stress on the potentially damaged communication infrastructure. Therefore, we propose a semantic routing mechanism tailored for loosely-coupled networks that can dynamically change and are unstructured. By providing a multi-agent system, devices are enabled as part of the network to support the communication infrastructure. The focus of the network is set on discovering semantically structured knowledge, which can change at run-time. The main contributions of this paper are routing tables that incorporate semantic information to aggregate semantically close knowledge. They are only updated if new knowledge is available or new aggregates are created, to avoid network flooding. Based on a search & rescue scenario, we explain our new routing and update algorithms. Finally, we discuss the message complexity of the routing mechanism and the suitability of the used knowledge representation.

1 INTRODUCTION

Future digital cites will increasingly depend on IT and communication technologies. This includes unmanned aerial vehicles (UAVs), mobile robots, autonomous cars, and smart infrastructures. The city infrastructure, for example, comprises smart street lights, that can be used to charge electric vehicles, camera systems for traffic surveillance and Internet access points. Therefore, a future digital city can be seen as a distributed system with heterogeneous participants that rely on communication infrastructure to interact with each other. Hence, the underlying network and the exchange of information have to be resilient. This is especially important for rescuers in the case of emergency situations or during natural disasters. One example is an earthquake. In this case, parts of the communication infrastructure could be destroyed; thus, the communication and reachability could be limited. Buildings could collapse, gas and water leaks could occur, and humans could be injured.

To overcome such emergency situations, rescuers need an overview of the current situation. Therefore, UAVs and mobile rescue robots are used to detect injured people and dangers like gas leaks.

Additionally, reliable access to knowledge is needed, which includes, among other information: available supplies, free hospital beds, and positions of injured people. Since parts of the communication infrastructure are damaged, access to a central coordinator that manages the necessary knowledge is not always given. Mission-critical knowledge, thus, has to be maintained and stored on the nodes of the communication network. To integrate heterogeneous devices as part of the communication infrastructure, a multi-agent system (MAS) is presented in (Jakob et al., 2020). In this system, agents run on the nodes of the network, e. g., on robots, UAVs, and smart street lights. The agents can join and leave the MAS dynamically. Furthermore, the agents manage and store knowledge, and thus, form an unstructured distributed knowledge base. To associate a semantic meaning with the stored knowledge, an ontology formulated in Answer Set Programming (ASP) (Brewka et al., 2011) has been presented in (Jakob et al., 2021). By applying this ontology, initially classified knowledge is assigned to more general classes in order to group semantically related knowledge. For example, knowledge about patients that have been injured during the emergency situation may be aggregated with knowl-

Jakob, S., Baraki, H., Jahl, A., Nyakam Chiadjeu, E., Alhamwy, Y. and Geihs, K.
Adaptive Semantic Routing in Dynamic Environments.
DOI: 10.5220/0010300609971004
In *Proceedings of the 13th International Conference on Agents and Artificial Intelligence (ICAART 2021) - Volume 2*, pages 997-1004
ISBN: 978-989-758-484-8

edge about other people. This is the case since each patient is a person, and thus, they form a semantic sub-class relation.

Rescuers using this system are mainly interested in the contents of the knowledge base and not in the location of the knowledge. Classical IP-based routing does not support the localisation of specific contents if its address is unknown. Therefore, a semantic routing mechanism is needed. Additionally, several challenges arise. The routing mechanism has to be performant since the network and its communication capability are limited due to the emergency situation. Central servers have to be prevented because they would introduce a bottleneck and single point of failure. Last but not least, knowledge can change dynamically. Hence, we present a novel semantic routing mechanism, which is the main contribution of this paper.

Instead of annotated files, we focus on semantically structured knowledge, which can be located in remote parts of the network and can dynamically change during run-time. Moreover, we assume that the structure of the network may also change dynamically in disaster scenarios. Thus, the semantic routing has to dynamically adapt to these changes. Additionally, the semantic routing applies ASP ontologies presented as in (Jakob et al., 2021). It incorporates parts of the ontology to aggregate semantically close knowledge under a common base class. These common base classes are used to create and provide semantic routing tables, that support access to knowledge based on its content instead of its location. These tables dynamically adapt if the network structure changes. To prevent network flooding, agents aggregate semantically close knowledge and tables are updated if new knowledge is available or new aggregates are created.

The remainder of the paper is structured as follows. Section 2 introduces Answer Set Programming, which is the used non-monotonic reasoning and knowledge representation formalism. The semantic routing is presented in Section 3. Section 4 discusses the message complexity of the presented semantic routing. Related work is shown in Section 5. Finally, Section 6 concludes this paper.

2 ANSWER SET PROGRAMMING

Answer Set Programming (ASP) has its origin in logic programming, constraint satisfaction, and knowledge representation. Furthermore, ASP is a declarative and non-monotonic logic programming language. An ASP program contains rules, which are divided into three parts. The rule head, the positive body, and the negative body. The head is derived if all predicates in the positive body hold and all predicates in the negative are false. A typical example is shown in Listing 1.

```
1 bird(tweety).
2 penguin(tux).
3 bird(X) :- penguin(X).
4 fly(X) :- bird(X), not penguin(X).
```

Listing 1: Example ASP Program.

Line 1 and 2 of Listing 1 are facts. They do not have a body and thus are unconditionally true. Additionally, these facts state that `tweety` is a bird and that `tux` is a penguin. Line 3 and 4 are rules. Informally speaking, Line 3 states that penguins are a bird, too, and Line 4 states that all birds can fly if they are not a penguin. The solution (Answer Set) of this program is: {penguin(tux), bird(tux), bird(tweety), fly(tweety)}. Since ASP is a declarative logic language, a solver is needed to determine the solution of an ASP program. The interested reader is referred to (Brewka et al., 2011) for a comprehensive specification of ASP. Clingo (Gebser et al., 2014) is a state-of-the-art ASP solver. It introduces several features, which include multi-shot solving, External Statements, and Program Sections. Multi-shot solving enables the usage of Clingo as a knowledge base since already derived predicates can be queried and reused. External Statements are predicates marked with the keyword `#external` and their truth value can be dynamically adapted at run-time. Finally, by defining Program Sections, ASP programs can be divided into parameterisable and reusable sub-programs. Hence, using Clingo provides mechanisms for the creation of a dynamically adaptable and extendable knowledge base.

Queries given to this knowledge base are formulated as ASP programs, which consists of rules and optimisation statements. A typical query in a search & rescue scenario is the closest patient that has to be saved. An example of a semantic query is shown in Listing 2.

```
1 {patient("A", 1);patient("B", 2);
      patient("C", 1)} = 1.
2 #minimize {Y@1 : patient(X, Y)}.
```

Listing 2: Example Semantic Query.

The first line lists all known patients with their name and distance parameter by using a choice rule (Brewka et al., 2011), which creates three possible solutions containing exactly one patient. The second line is a query containing an optimisation state-

ment, which selects the patients which are closest. Solving this query results in two optimal solutions `A` and `C`. To retrieve knowledge about all known patients, the query has to be efficiently routed and deployed to the corresponding parts of the distributed knowledge base.

3 CREATING SEMANTIC ROUTING TABLES

To locate contents in an unstructured distributed knowledge base, the most simple way is to rely on flooding. However, in scenarios with impaired communication like emergency situations or natural disasters, flooding would introduce further stress on the network. Therefore, an intelligent way of routing is needed. One way would be the use of structured P2P like Distributed Hash Tables (DHT). However, DHTs are not suited for highly dynamic networks, since the content is equally distributed on their nodes. This could lead to a broad distribution of contents and queried contents could be on remote nodes of the network and thus often not reachable in highly dynamic networks. A suitable approach is the use of routing tables. They provide information about routes to network destinations. Furthermore, they can easily be updated by adding, altering or removing entries. Yet, in many application scenarios, like search & rescue, the emphasis is on the content instead of its location. Therefore, we have developed a new way to create routing tables that focus on contents and their semantic.

A major challenge is to provide a routing of semantic queries in unstructured networks. The first fundamental requirement for solving semantic queries are knowledge bases that provide semantic information. Commonsense knowledge sources provide relations between their classes or concepts to structure the represented knowledge. One of these sources is ConceptNet 5 (CN5) (Speer et al., 2017), which has been created from verified sources like Wiktionary, WordNet, and Wikipedia. It consists of a hypergraph that connects concepts by directed edges that are annotated with predefined relations. For example, the relation `IsA` indicates a subclass relationship and the relation `Synonym` is used to connect concepts with similar meaning. Furthermore, each edge in CN5 is annotated with a weight that indicates the reliability of the represented knowledge. A weight greater than `1.0` indicates, that the edge has been extracted from a verified source.

We leverage the structure of commonsense knowledge to extract a taxonomy of classes. Taxonomies can be used to support a hierarchical aggregation of knowledge. An extraction of a taxonomy is shown in Figure 1. The taxonomy is marked in green and has been identified by relying on a subset of relation types and a selection of the highest edge weights. In the given example, this leads to selecting the edge between `patient` and `person` instead of `patient` and `human`. Additional knowledge provided by CN5 is marked in red. As you can see in this figure, the hypergraph provides a mesh-like structure and can contain additional types of edges, while taxonomies provide tree-like structures. Since tree-like structures ease the aggregation of semantically close knowledge, we apply taxonomies to automatically generate semantic routing tables that are used to distribute semantic queries to the corresponding nodes of an unstructured network.

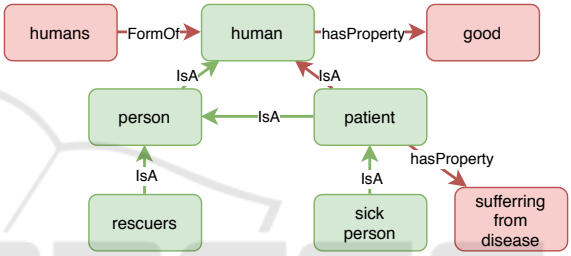

Figure 1: Taxonomy Extracted from a Hypergraph[1].

In other works, the Web Ontology Language (OWL) is used to model ontologies or taxonomies since it is the de facto standard for knowledge representation in the semantic web. Since our focus is set on constantly changing networks and information, OWL is inapplicable. Its full specification is undecidable. The resulting ontologies are monotonous, which prevents the invalidation of already derived knowledge if contradicting knowledge arises. In addition, OWL does not allow the definition of defaults, which prevents to incorporate commonsense knowledge that can be overwritten in individual cases. In contrast to this, ASP is decidable and non-monotonous. Thus, already derived knowledge like defaults can be invalidated. Furthermore, current ASP solvers, for example, Clingo, enable a dynamic adaption of ASP programs and thus supports dynamic ontologies and taxonomies.

For the extraction and transformation of commonsense knowledge into ontologies and taxonomies in ASP, we provide a framework (Jakob et al., 2021) that enables their automatic generation but also the manual adaptation and extension with properties. Using this framework, a taxonomy can be created by limiting the considered relations of CN5 to `IsA`, `FormOf`, and `Synonym`. Furthermore, only edges with a weight of at least `1.0` are considered to extract knowledge

supported by a verified source. The general idea of the taxonomy extraction shown in (Jakob et al., 2021) is to provide a starting concept and applying an adapted breadth-first search that uses a set of relations to select edges and a minimum weight as a stopping criterion. Following the search & rescue example presented above, a taxonomy is needed that provides classes for human patients. During design time, a rescuer can use our modelling framework to start the generation of the ontology. Therefore, the concept `human` is selected as the base concept and the weight of `1.0` is chosen as a stopping criterion. This results in the following excerpt of the resulting taxonomy:

```
1 #external isA("patient","person",1).
2 weight(1,100,0)
      :- isA("patient","person",1).
3 #external isA("person","human" ,2).
4 weight(2,100,0)
      :- isA("person","human",2).
5 weight(2,200,1)
      :- isA("person","human",2).
```

Listing 3: Excerpt from the Taxonomy.

Line 1 of Listing 3 states, that a `patient` is a `person`. Furthermore, the UUID `1` is provided which is used to link the corresponding weight rule that is shown Line 2. Informally speaking, it states the edge with the UUID `1` has a weight of `100` at time `0`. Manually adapting weights in our framework would lead to an additional weight rule containing the same UUID would but a higher timestamp. Line 3 and 4 analogously define the relation between `human` and `person`. After the extraction of the taxonomy and its translation into ASP, a rescuer increases the weight of edge `2` to `200` as shown in Line 5. This is done since the inference rules presented in (Jakob et al., 2021) only apply further base classes if their weight increases to prevent impractical classifications or loops. The resulting taxonomy is then used to classify patients and to filter knowledge not necessary in the rescue missions.

The novel semantic routing consists of four parts, which are explained in the following paragraphs. These are semantic routing tables, semantic aggregation, an update mechanism, and propagation of individuals. In the following, the four parts are presented by the example shown in Figure 2[1]. In this scenario, rescuers try to locate patients after an earthquake. The patients are labelled `A`, `B`, and `C`. Each patient is associated with a corresponding smartphone that provides communication with robots, UAVs, and the smart environment. Furthermore, each node in the presented network represents an agent that has a UUID. The dashed arrow in Figure 2 represents a query to the system. Solid arrows indicate that nodes can communicate with each other.

[1]Created with https://app.diagrams.net/ (December 16, 2020)

```
1 route(path("human","person",
      "patient"), uuid(sp1), dist(2),
      uuid(r2))   :- node(uuid(r2)),
      not -route(path("human","person"
      ,"patient"),uuid(sp1), dist(2),
      uuid(r2)).
2 route(path("human","person",
      "patient"), uuid(sp2), dist(1),
      uuid(sp2)) :- node(uuid(sp2)),
      not -route(path("human","person"
      ,"patient"),uuid(sp2), dist(1),
      uuid(sp2)).
3 route(path("human","person"), uuid(
      uav2), dist(1), uuid(uav2)) :-
      node(uuid(uav2)), not -route(
      path("human","person"), uuid(
      uav2), dist(1), uuid(uav2)).
4 #external node(uuid(r2)).
5 #external node(uuid(sp2)).
6 #external node(uuid(uav2)).
7 #external -route(path("human","
      person","patient"), uuid(sp1),
      dist(2), uuid(r2)).
8 #external -route(path("human","
      person","patient"), uuid(sp2),
      dist(1), uuid(sp2)).
9 #external -route(path("human","
      person"), uuid(uav2), dist(1),
      uuid(uav2)).
```

Listing 4: Semantic Routing Table of Robot 1.

Semantic Routing Tables. The base of the adaptive semantic routing are routing tables that contain semantically aggregated entries. The entries point to the neighbouring nodes, which are closer to a queried semantic piece of information. These tables consist of ASP rules and Externals Statements. While ASP rules are used to model the entries of the table, External Statements are applied to keep track of the availability of a node and the validity of the entries. Furthermore, each entry indicates the type of the knowledge based on the taxonomy, the node of the network that either aggregated the knowledge or is its origin, the distance to the corresponding node, and the neighbouring node that provides access to this knowledge. The distance is measured by the hops. Listing 4 shows the semantic routing table of `Robot 1` (`r1`) according to Figure 2.

The Lines 1 to 3 of this listing are the actual entries of the semantic routing table. As can be seen in Line 1, a `path` indicates the base classes, which have been obtained through aggregation and updating pro-

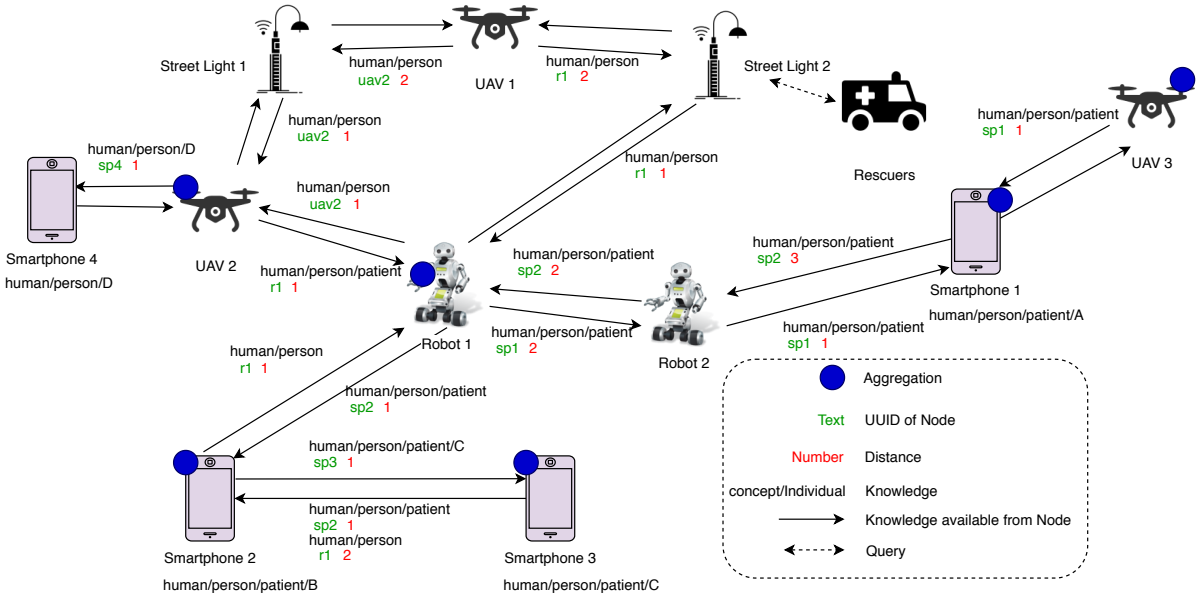

Figure 2: Semantic Routing Information[1].

cesses or are part of the initial classification of an individual. In this line, the `path` states that knowledge about `patients` with the base classes `person` and `human` can be found at `Smartphone 1` (`sp1`), which is reachable in `2` hops via robot `r2`. To derive this `route`, the External Statement in Line 4 has to be set to `true`, which indicates that robot `r2` is reachable. Additionally, the External Statement in Line 7 has to be `false` since it is used to invalidate the `route` in case `r1` receives contrary information. Thus, `routes` can be adapted dynamically during the run-time of the system. The remaining combinations of rules and External Statements work analogously.

In contrast to the `routes` presented in Listing 4, individuals that are managed on a node are modelled as External Statements. This enables a node to remove knowledge on its own, e. g., if it is outdated. For example, `sp1` in Figure 2 manages knowledge about individual `A`. Therefore, the External Statement `route(path("human","person","patient","A"), uuid(sp1), dist(0), uuid(sp1))` is added to the semantic routing table. This External Statement indicates, that knowledge about patient `A` is available on this node, because the distance is `0`. Since the node manages the knowledge about patient `A` itself, it can set the External Statement to false if this is no longer the case. To minimise the size of the tables, semantically close knowledge is aggregated.

Semantic Aggregation. A central aspect of the semantic routing tables is the aggregation of semantically related knowledge. As already discussed above, a taxonomy is used since it provides a tree-like classification which eases the aggregation based on its branches. Furthermore, several distinct taxonomies can be applied which allows managing knowledge originating from several independent base classes. There are two situations when knowledge can be aggregated.

The first one is the aggregation based on the taxonomy. In this case, entries of a routing table are aggregated if they represent individuals that share the same base classes. Figure 2 shows an example for this kind of aggregation. All possible aggregation points of this scenario are marked with a blue dot. `Smartphone 2` (`sp2`), which has knowledge about patient `B` aggregates a routing table entry provided by `Smartphone 3` (`sp3`), which has knowledge about patient `C`. After `sp3` receives the knowledge about patient `C`, it propagates a corresponding routing table entry to `sp2`, which already has knowledge about patient `B`. Since both, `B` and `C`, share the same base classes, the knowledge is aggregated to a new routing table entry, which associates both individuals to a base class. In this case, the route `route(path("human","person","patient"), uuid(sp2), dist(0), uuid(sp2))` is created on `sp2` and is propagated to the neighbouring nodes except the origin of the routing entry that caused the aggregation. Base classes are aggregated analogously. For example, `UAV 2` aggregates knowledge about `path("human","person")` with `path("human","person","patient")`.

The second type of aggregation is based on

distance. In the case that a node receives a routing entry, which has the same base classes, a lower or the same distance, and has the identical origin of information (second UUID entry of a route), an aggregation step is conducted. For example, `r1` receives an entry from `sp2` while its routing table already has an entry from `r2`. Both entries share the same base classes and the entry received from `sp2` has a lower distance, thus, the new aggregation `route(path("human","person","patient"), uuid(r1), dist(1), uuid(r1))` is created and forwarded to the remaining neighbours. The routing table entries are added by an update mechanism.

Updates. Besides the aggregation of routing entries, a reliable update mechanism is needed to keep the semantic routing tables viable. The update mechanism is responsible for three kinds of updates, which are conducted if corresponding messages are received.

The first kind is the arrival of additional routes. During this kind of update, new routes are received via a message containing corresponding ASP rules. First, possible aggregates are checked. If an aggregation is possible, the negative External Statement, for example `-route(path("human","person","patient"), uuid(sp1), dist(2), uuid(r2))`, of the corresponding existing routes are set to true and thus the routes are removed from the routing table. Afterwards, the created aggregate is added to the table and propagated to the neighbouring nodes. In case that no aggregation is possible, the received route is added to the table.

The second kind of updates are overwrites. They occur if new routing table entries arrive which match existing ones, i. e., if they have the same classes and origin. In case that a new routing table entry has a lower distance, the old entry is removed by setting the corresponding external statement to false and by adding the new entry to the routing table. In the last step, it is checked if any aggregation is possible.

The third kind of updates happens if a neighbouring node is no longer reachable. For example, consider that in Figure 2 `r1` is no longer able to reach `r2`, which is detected by periodical ping requests. Since `r2` can no longer provide the knowledge accessible via the corresponding routing table entries, the entries have to be removed. Therefore, `r1` sets the External Statement that represents `r2` (`node(uuid(r2))`) to false. Since this External Statement is set to false, all routing entries are removed from the table, as their rule body positively depends on it. During this kind of update, no aggregation is needed since no new knowledge is received.

Propagation of Individuals. A key aspect of semantic routing is to distribute knowledge about managed individuals. Therefore, their existence needs to be published. Once a node of the system receives knowledge about a new individual, e. g. individual `path("human","person","D")` is introduced to `Smartphone 4` (`sp4`), corresponding routing entries are created. To prevent the distribution of unaggregated routes, the routing table is checked for potential aggregations. If possible, the entries are aggregated and the routing table is updated. Finally, the appearance of a new entry is published to the neighbouring nodes via messages.

By combing these four steps, dynamically adaptable semantic routing tables are created. Queries to these tables are formulated as ASP programs, too. Therefore, the arity of the path predicate can be used to reduce the possible answers. For example, the rule `patientEntry(X) :- route(path("human","person","patient"), uuid(_), dist(_), uuid(X)).` selects all aggregated patient entries of a semantic routing table and returns the UUIDs of the nodes which provides the corresponding knowledge. In case of the semantic routing table of `r1` shown in Listing 4, `r2` and `sp2` are returned, since the corresponding routing entries indicate that they are able to provide the knowledge about patients.

4 DISCUSSION

A central aspect of the evaluation of a routing algorithm is its message complexity. Achieving a low message complexity in a loosely-coupled network with dynamic configuration is particularly challenging, since nodes and information may enter and leave the system at any time. Content-based routing algorithms rely on semantic information and, hence, no distance metric and no clear addressing are given, which adds to the challenge. In the case of content-based routing, queried information can be located on any node of the network. A typical approach is to use flooding to reliably discover all information corresponding to a search query. However, flooding has a high message complexity. Considering a fully connected network, naive flooding, which simply forwards messages to all neighbours except to the origin of the message has a message complexity of $O(n * (n-1))$ in the best and the worst case. Already received messages are ignored. If a mesh network is not fully connected, the message complex-

ity is reduced to $O(n*m)$, where m is the number of neighbours of each node. Again, this complexity holds for the best and the worst case. To further reduce the complexity, selective flooding can be introduced, which has a message complexity of $O(n)$ since it relies on a tree-like structure to forward messages. However, the direction leading to the target, a distance metric, and the approximate location of the target are needed.

In the worst case, our semantic routing mechanism has the same message complexity of $O(n*m)$ like flooding in a mesh network. There are two possible cases in which this complexity can arise. The first case is the introduction of the first individual of a completely new taxonomy path to the system since there are no possibilities to form aggregates. The second case can occur during the first aggregations, as the aggregated path has to be disseminated. Since there are no other aggregation points, all routing tables have to be updated. However, after the first aggregations have been introduced, the message complexity is reduced since aggregation points summarise routes, and thus, may stop the propagation. For example, this is the case if routes containing more specific taxonomy paths are received. In the best case, this results in a message complexity of $O(0)$ if a new individual is introduced at an aggregation point, which already contains a routing entry with the corresponding taxonomy path. Furthermore, aggregation points reduce the message complexity of additional aggregations to $O(m)$ if neighbouring nodes already possess routes with the corresponding aggregate.

Besides the creation of routing tables, the message complexity of query resolution is of high relevance. The worst message complexity of $O(n)$ emerges if individuals that match a given concept query are distributed on the nodes of the whole network, for example, a query for all `humans` in Figure 2. In contrast, a complexity of $O(0)$ is achieved if the answer solely exists in the own routing table. Adding a new node to the network has a complexity of $O(1)$, due to transfer of the routing table from a neighbouring node.

A prototypical implementation of the semantic routing is available at Bitbucket[2]. For the scenario given in Figure 2 the presented semantic routing tables are roughly 10 times faster than naive flooding.

The suitability of ASP for the creation of ontologies and taxonomies has been shown in (Jakob et al., 2021). Ontologies and taxonomies are extracted from a hypergraph-based knowledge source and formulated in a way that enables dynamic adaptations at runtime. Furthermore, it is shown that several hundreds of thousands ASP rules are usable in a reasonable time. Typical queries can be solved in a few milliseconds. Clearly, the more complex the query, the higher is the run-time. Nonetheless, queries to a routing table have a simple structure.

[2]https://bitbucket.org/sjakob872/semantic_routing (December 16, 2020)

5 RELATED WORK

Typical well-known P2P search and routing mechanisms, such as flooding, random walks, distributed hashing, and central repositories, do not provide an efficient solution for knowledge discovery in an unstructured or unstable network. Extensive flooding of an unstable network by many users, authorities and rescue teams may lead to a complete communication breakdown in disaster situations. Random walks cannot guarantee to find information, irrespective of whether it is available in the network or not. Central repositories constitute a bottleneck and a single point of failure. Distributed hashing approaches such as (Visala et al., 2009) violate the autonomy of nodes and are not designed for unstable and dynamic networks. The key of a document or information may map to a remote node that is not reachable.

Related works that try to optimize knowledge discovery in dynamic networks can be found in the areas of Semantic Query Routing and Content-Based Routing in P2P and mobile ad hoc networks. In (Gómez Santillán et al., 2010), Gomez et al. extend and combine different approaches known from unstructured P2P systems to address the Semantic Query Routing Problem. The problem can be summarized with locating textual information, more precisely documents annotated with topics, in a P2P network. They apply an Ant Colony algorithm to adapt the pheromone level of routes depending on the hit rates and hops for a certain topic query. The routing for a certain query converges, the more often the query is sent through the network. Since a learning phase is required, rather static networks with recurring topic queries will benefit from this approach. Michlmayr et al. (Michlmayr et al., 2007) apply additionally a taxonomy to search for similar concepts. Concepts are considered as similar, if they share the same super concepts in the taxonomy. However, in contrast to our work and to (Pireddu and Nascimento, 2004), taxonomies are not used to aggregate entries in routing tables. Pireddu and Nascimento utilise taxonomies to summarise the number of documents each neighbour has in each category. A node is aware of these numbers for each direct neighbour, whereas only the aggregated numbers for upper-level categories are known for the neighbours of neighbours.

If the number of hops to a remote node exceeds the depth of the taxonomy tree, no information is kept about this remote node. The taxonomy thus serves here as a summarised directory of documents in the local neighbourhood and is not designed for routing queries to far distant nodes.

In (Koloniari and Pitoura, 2004), knowledge is kept in XML files by nodes that are organised in a hierarchy. Each node possesses a local Bloom filter that is triggered whenever an incoming query potentially matches with locally stored knowledge. Nodes also contain a merged Bloom filter that encompasses the Bloom filters of the child nodes. A query matching to a merged filter is redirected to child nodes. If no positive local or child matching is obtained, the query is redirected to the parent node. The approach is rather suited for static P2P networks as nodes have to stay in a hierarchy. The resource capacities of the root node have to be sufficiently dimensioned since it has to handle the most search and update queries. In addition, the usage of Bloom filters may lead to a high number of false positive matchings and thus routings, if they are not configured and adapted well.

In (Jacobson et al., 2009) and (Gritter and Cheriton, 2001), hierarchical names, similar to the taxonomy paths in our work, are used for content-based routing. Also, routing entries are merged to reduce the size of routing tables. However, taxonomies are not applied. Instead, a hierarchical name could start with the URL of the owner of a content. Consistent names and annotations of content are not given. Additionally, typical semantic queries cannot be used as no solver is included in the routing procedure.

6 CONCLUSION

In this paper, we have presented a novel routing mechanism for loosely coupled networks. The main contribution are the creation of semantic routing tables and an efficient update mechanism. Their entries are based on semantic aggregations, which rely on taxonomies and a distance measure to obtain the shortest path to queried individuals. The usage of ASP enable the application of the same formalism for knowledge representation, reasoning, and query resolution. The applied taxonomies are automatically extracted from a commonsense knowledge source and are provided as an ontology in ASP. Additionally, they can be dynamically adapted to the current situation. The framework for the extraction has been presented in (Jakob et al., 2021).

In our future work, we plan to integrate the presented semantic routing mechanism into the distributed and multi-agent-based knowledge base introduced in (Jakob et al., 2020). To prevent that all routing entries are merged to the same base class, we plan to provide methods to adjust the depth of the aggregation. Finally, we want to comprehensively evaluate the proposed routing algorithm in a large scale search & rescue scenario simulation[3] and aim to compare the presented approach with other semantic or content-based approaches.

REFERENCES

Brewka, G., Eiter, T., and Truszczyński, M. (2011). Answer Set Programming at a Glance. *Communications of the ACM*, 54(12):92–103.

Gebser, M., Kaminski, R., Kaufmann, B., and Schaub, T. (2014). Clingo=ASP+Control: Extended Report. Technical report, Knowledge Processing and Information Systems.

Gómez Santillán, C., Cruz Reyes, L., Meza Conde, E., Schaeffer, E., and Castilla Valdez, G. (2010). A Self-Adaptive Ant Colony System for Semantic Query Routing Problem in P2P Networks. *Computación y Sistemas*, 13(4):433–448.

Gritter, M. and Cheriton, D. R. (2001). An architecture for content routing support in the internet. In *USITS*, volume 1, pages 4–4.

Jacobson, V., Smetters, D. K., Thornton, J. D., Plass, M. F., Briggs, N. H., and Braynard, R. L. (2009). Networking named content. In *Proceedings of the 5th international conference on Emerging networking experiments and technologies*, pages 1–12.

Jakob, S., Jahl, A., Baraki, H., and Geihs, K. (2020). A Self-Organizing Multi-Agent Knowledge Base. Accepted for publication at IEEE ICWS 2020.

Jakob, S., Jahl, A., Baraki, H., and Geihs, K. (2021). Generating Commonsense Ontologies with Answer Set Programming. Accepted for publication at ICAART 2021.

Koloniari, G. and Pitoura, E. (2004). Content-based Routing of Path Queries in Peer-to-Peer Systems. In *International Conference on Extending Database Technology*, pages 29–47. Springer.

Michlmayr, E., Pany, A., and Kappel, G. (2007). Using Taxonomies for Content-based Routing with Ants. *Computer Networks*, 51(16):4514–4528.

Pireddu, L. and Nascimento, M. A. (2004). Taxonomy-Based Routing Indices for Peer-to-Peer Networks. In *Workshop on Peer-to-Peer Information Retrieval*.

Speer, R., Chin, J., and Havasi, C. (2017). ConceptNet 5.5: An Open Multilingual Graph of General Knowledge. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, pages 4444–4451.

Visala, K., Lagutin, D., and Tarkoma, S. (2009). LANES: An Inter-Domain Data-Oriented Routing Architecture. In *Proceedings of the 2009 workshop on Re-architecting the internet*, pages 55–60.

[3] https://www.emergencity.de/ (December 16, 2020)