SEMAPRO 2021

The Fifteenth International Conference on Advances in Semantic Processing

ISBN: 978-1-61208-888-4

October 3 - 7, 2021

Barcelona, Spain

SEMAPRO 2021 Editors

Oana Dini, IARIA, USA/EU

Claus-Peter Rückemann, Westfälische Wilhelms-Universität Münster (WWU) /

DIMF / Leibniz Universität Hannover, Germany

Efstratios Kontopoulos, Catalink Limited, Cyprus

SEMAPRO 2021

Forward

The Fifteenth International Conference on Advances in Semantic Processing (SEMAPRO 2021) continued a series of events focused on the progress on ontology, web services, semantic social media, semantic web, deep web search /deep semantic web/, semantic deep web, semantic networking and semantic reasoning.

The inaugural International Conference on Advances in Semantic Processing, SEMAPRO 2007, was initiated considering the complexity of understanding and processing information. Semantic processing considers contextual dependencies and adds to the individually acquired knowledge emergent properties and understanding. Hardware and software support and platforms were developed for semantically enhanced information retrieval and interpretation. Searching for video, voice, and speech [VVS] raises additional problems to specialized engines with respect to text search. Contextual searching and special patterns-based techniques are current solutions.

We take here the opportunity to warmly thank all the members of the SEMAPRO 2021 technical program committee, as well as all the reviewers. The creation of such a high-quality conference program would not have been possible without their involvement. We also kindly thank all the authors who dedicated much of their time and effort to contribute to SEMAPRO 2021. We truly believe that, thanks to all these efforts, the final conference program consisted of top-quality contributions. We also thank the members of the SEMAPRO 2021 organizing committee for their help in handling the logistics of this event.

SEMAPRO 2021 Chairs

SEMAPRO 2021 Steering Committee
Sandra Lovrenčić, University of Zagreb, Croatia
Tim vor der Brück, Lucerne University of Applied Sciences and Arts, Switzerland
Michele Melchiori, Università degli Studi di Brescia, Italy
Wladyslaw Homenda, Warsaw University of Technology, Poland
Fabio Grandi, University of Bologna, Italy
Sofia Athenikos, Twitter, USA

SEMAPRO 2021 Publicity Chairs
Lorena Parra, Universitat Politecnica de Valencia, Spain
José Miguel Jiménez, Universitat Politecnica de Valencia, Spain

SEMAPRO 2021
Committee

SEMAPRO 2021 Steering Committee

Sandra Lovrenčić, University of Zagreb, Croatia
Tim vor der Brück, Lucerne University of Applied Sciences and Arts, Switzerland
Michele Melchiori, Università degli Studi di Brescia, Italy
Wladyslaw Homenda, Warsaw University of Technology, Poland
Fabio Grandi, University of Bologna, Italy
Sofia Athenikos, Twitter, USA

SEMAPRO 2021 Publicity Chairs

Lorena Parra, Universitat Politecnica de Valencia, Spain
José Miguel Jiménez, Universitat Politecnica de Valencia, Spain

SEMAPRO 2021 Technical Program Committee

Witold Abramowicz, Poznan University of Economics, Poland
Harry Agius, Brunel University London, UK
Riccardo Albertoni, Istituto per la Matematica Applicata e Tecnologie Informatiche "Enrico Magenes" - Consiglio Nazionale delle Ricerche (IMATI-CNR), Italy
Jose María Alvarez Rodríguez, Carlos III University of Madrid, Spain
Abdel-Karim Al-Tamimi, Higher Colleges of Technology, UAE
Sofia Athenikos, Twitter, USA
Giuseppe Berio, Université de Bretagne Sud | IRISA, France
Floris Bex, Utrecht University & University of Tilburg, Netherlands
Loris Bozzato, Fondazione Bruno Kessler, Trento, Italy
Zouhaier Brahmia, University of Sfax, Tunisia
Okan Bursa, Ege University, Turkey
Ozgu Can, Ege University, Turkey
Rodrigo Capobianco Guido, São Paulo State University (UNESP), Brazil
Damir Cavar, Indiana University, USA
Alberto Cetoli, QBE Europe, UK
David Chaves-Fraga, Universidad Politécnica de Madrid, Spain
Ioannis Chrysakis, FORTH-ICS, Greece / Ghent University, Belgium
Ademar Crotti Junior, Trinity College Dublin, Ireland
Cláudio de Souza Baptista, Federal University of Campina Grande, Brazil
Christian Dirschl, Wolters Kluwer Deutschland GmbH, Germany
Milan Dojchinovski, InfAI | Leipzig University, Germany / Czech Technical University in Prague, Czech Republic
Julio Cesar Duarte, Instituto Militar de Engenharia, Rio de Janeiro, Brazil
Enrico Francesconi, IGSG - CNR, Italy
Rolf Fricke, Condat AG, Berlin, Germany
Panorea Gaitanou, Greek Ministry of Justice, Athens, Greece
Bilel Gargouri, MIRACL Laboratory | University of Sfax, Tunisia
Fabio Grandi, University of Bologna, Italy
Damien Graux, ADAPT Centre - Trinity College Dublin, Ireland
Jingzhi Guo, University of Macau, Macau SAR, China

Bidyut Gupta, Southern Illinois University Carbondale, USA
Prabhakar Gupta, Parth Group, India
Shun Hattori, Muroran Institute of Technology, Japan
Ulrich Heid, Iwist | Universität Hildesheim, Germany
Tobias Hellmund, Fraunhofer Institute of Optronics, System Technologies and Image Exploitation IOSB, Germany
Tracy Holloway King, Amazon, USA
Timo Homburg, Mainz University of Applied Sciences, Germany
Wladyslaw Homenda, Warsaw University of Technology, Poland
Tzung-Pei Hong, National University of Kaohsiung, Taiwan
Helmut Horacek, DFKI/Saarland University, Germany
Thomas Hubauer, Siemens AG Corporate Technology, Germany
Sergio Ilarri, University of Zaragoza, Spain
Agnieszka Jastrzebska, Warsaw University of Technology, Poland
Young-Gab Kim, Sejong University, Korea
Stasinos Konstantopoulos, Institute of Informatics and Telecommunications, NCSR "Demokritos", Greece
Petr Kremen, Czech Technical University in Prague, Czech Republic
Jaroslav Kuchař, Czech Technical University in Prague, Czech Republic
Chun-Ming Lai, Tunghai University, Taiwan
André Langer, Chemnitz University of Technology, Germany
Kyu-Chul Lee, Chungnam National University, South Korea
Els Lefever, LT3 | Ghent University, Belgium
Antoni Ligęza, AGH-UST Kraków, Poland
Johannes Lipp, Fraunhofer Institute for Applied Information Technology FIT, Germany
Usha Lokala, University of South Carolina, USA
Giuseppe Loseto, Polytechnic University of Bari, Italy
Sandra Lovrenčić, University of Zagreb, Croatia
Federica Mandreoli, Universita' di Modena e Reggio Emilia, Italy
Miguel A. Martínez-Prieto, University of Valladolid, Segovia, Spain
Miguel Felix Mata Rivera, UPIITA-IPN, Mexico
Michele Melchiori, Università degli Studi di Brescia, Italy
Dimitri Metaxas, Rutgers University, USA
Mohamed Wiem Mkaouer, Rochester Institute of Technology, USA
Luis Morgado da Costa, Nanyang Technological University, Singapore
Fadi Muheidat, California State University San Bernardino, USA
Yotaro Nakayama, Technology Research & Innovation Nihon Unisys, Ltd., Tokyo, Japan
Nikolay Nikolov, SINTEF Digital, Norway
Fabrizio Orlandi, ADAPT Centre | Trinity College Dublin, Ireland
Peera Pacharintanakul, TOT, Thailand
Peteris Paikens, University of Latvia - Faculty of Computing, Latvia
Panagiotis Papadakos, FORTH-ICS | University of Crete, Greece
Silvia Piccini, Institute Of Computational Linguistics "A. Zampolli" (CNR-Pisa), Italy
Vitor Pinheiro de Almeida, Pontifícia Universidade Católica do Rio de Janeiro (PUC-Rio), Brazil
Luiz André Portes Paes Leme, Universidade Federal Fluminense, Brazil
Livia Predoiu, Otto-von-Guericke-Universität Magdeburg, Germany
Matthew Purver, Queen Mary University of London, UK
Francisco José Quesada Real, Universidad de Cádiz, Spain
Irene Renau, Pontificia Universidad Católica de Valparaíso, Colombia

Copyright Information

Table of Contents

Semantic Reasoning with Differentiable Graph Transformations 1
Alberto Cetoli

An Extensible Semantic Data Fusion Framework for Autonomous Vehicles 5
Efstratios Kontopoulos, Panagiotis Mitzias, Konstantinos Avgerinakis, Pavlos Kosmides, Nikos Piperigkos, Christos Anagnostopoulos, Aris S. Lalos, Nikolaos Stagakis, Gerasimos Arvanitis, Evangelia I. Zacharaki, and Konstantinos Moustakas

AWAPart: Adaptive Workload-Aware Partitioning Knowledge Graphs 12
Amitabh Priyadarshi and Krzysztof J. Kochut

Towards an Architecture for Policy-Aware Decentral Dataset Exchange 18
Sebastian Neumaier, Giray Havur, and Tassilo Pellegrini

A Reference Ontology for Collision Avoidance Systems and Accountability 22
David Martin-Lammerding, Jose Javier Astrain, and Albert Cordoba

Temporal Information and Event Markup Language: TIE-ML Markup Process and Schema Version 1.0 29
Damir Cavar, Billy Dickson, Ali Aljubailan, and Soyoung Kim

Semantic Reasoning with Differentiable Graph Transformations

Alberto Cetoli
QBE Europe
London, UK
Email: alberto.cetoli@uk.qbe.com

***Abstract*—This paper introduces a differentiable semantic reasoner, where rules are presented as a relevant set of graph transformations. These rules can be written manually or inferred by a set of facts and goals presented as a training set. While the internal representation uses embeddings in a latent space, each rule can be expressed as a set of predicates conforming to a subset of Description Logic.**

***Keywords*–Semantic Reasoning, Semantic Graphs, Graph Transformations, Differentiable Computing.**

I. Introduction

Symbolic logic is the most powerful representation for building interpretable computational systems [1]. In this work, we adopt a subset of Description Logic [2] to represent knowledge and build a semantic reasoner, which derives new facts by applying a chain of transformations to the original set.

In the restricted context of this paper, knowledge can be expressed in predicate or graph form, interchangeably. Thus, semantic reasoning can be understood as a sequence of graph transformations [3], which act on a subset of the original knowledge base and sequentially apply the matching rules.

In this paper, we show that rule matching can be made differentiable by representing nodes and edges as embeddings. After building a one-to-one correspondence between a sequence of rules and a linear algebra expression, the system can eventually train the embeddings using a convenient loss function. The rules created in this fashion can then be applied during inference time.

Our system follows the recent revival of hybrid neuro-symbolic models [1], combining insights from logic programming with deep learning methods. The main contribution of this work is to show that reasoning over graphs is a learnable task. While the system is presented here as a proof of concept, we show that differential graph transformations can effectively learn new rules by training nodes, edges, and matching thresholds through backpropagation.

In Section II we describe in detail the fundamentals of our reasoner, with working examples shown in Section III. Section IV reviews specific connections with prior works and finally a few remarks in Section V conclude the paper. The relevant code can be found at [4].

II. Problem statement

The system presented here is a semantic reasoner inspired by the early STRIPS language [5]. It creates a chain of rules that connects an initial state of *facts* to a final state of inferred predicates. Each rule has a set of pre- and post-conditions, expressed here using a subset of Description Logic (DL). In the following, we restrict our DL to Assertional Axioms (ABox). Thus, each fact can be represented as a set of predicates, or - equivalently - as a graph with matching rules as described below.

A. Rules as graph transformations

We use a predicate form to represent facts, rules, and intermediate states, as shown in Figure 1. For example, the semantics for "Joe wins the election in the USA" is captured in the following form

```
joe(a), win(a,b), election(b), in(b,c), USA(c)
```

In the prior example, joe, $election$, and USA are nodes of the semantic graph, whereas win and in are convenient relations to represent the graph's edges.

The rules are specified with a MATCH/CREATE pair as below

```
MATCH person(a), win(a,b), election(b)
CREATE (a), be(a,b), president(b)
```

The MATCH statement specifies the pre-condition that triggers the rule, while the CREATE statement acts as the effect - or post-condition - after applying the rule. The result of applying this rule is shown in Figure 1, where a new state is created from the original fact. Notice that the name joe (which matches $person$) is propagated forward to the next set of facts.

By applying rules in sequence one builds an inferential chain of MATCH and CREATE conditions. After each rule the initial facts graph is changed into a new set of nodes and edges. This chain of graph transformations builds a path in a convenient semantic space, as shown in Figure 2. One of this paper's main result is to show that there is a one-to-one correspondence between the chain of matching rules and a chain of linear algebra operations.

B. Nodes and edges as embeddings

Both nodes and edges are represented as embeddings in a latent space. For convenience, in the current work the vocabulary of possible nodes matches the Glove 300dim dataset [6], whereas edges are associated random embeddings linked to the relevant ontology.

C. Matching nodes and edges

A rule is triggered if the pre-condition graph is a sub-isomorphism of the facts. Each node and edge of the pre-conditions has a learnable threshold value t. Two items match if the dot product between their embeddings is greater than a specific threshold. In the predicate representation, we make explicit these trainable thresholds by adding the symbol $>$ to the predicate's name. In this way, the rule in Section II-A becomes

```
MATCH person>0.6(a), win>0.7(a,b), election>0.6(b)
CREATE (a), be(a,b), president(b)
```

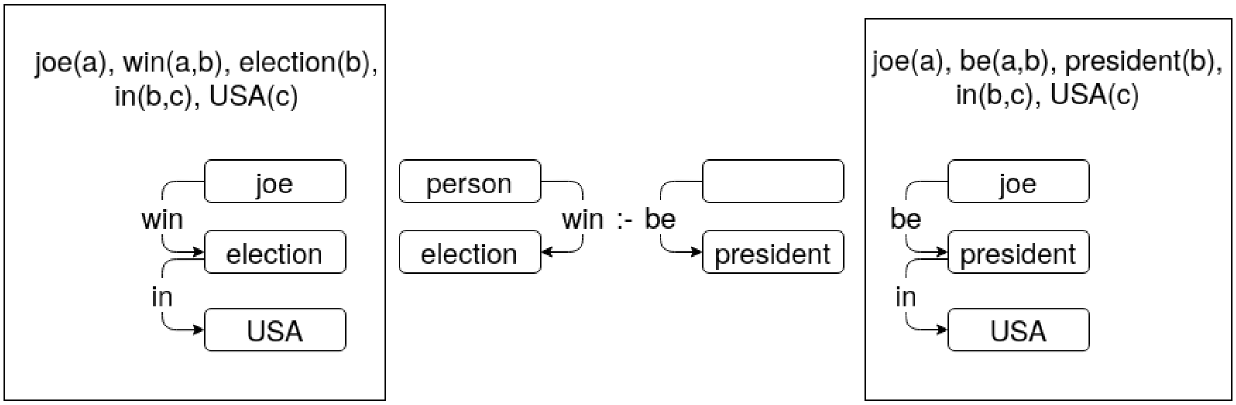


Figure 1. A matching rule example, as explained in II-A. The facts on the left are "transformed" into the ones on the right following the application of the relevant rule. This picture makes explicit the dual nature of the predicates/graph representation.

indicating that - for example - joe and $person$ would only match if their normalized dot product is greater than $t = 0.6$. In the Description Logic framework this is equivalent to an *individuality assertion*

$$joe \approx person \iff \text{embedding}(joe) \cdot \text{embedding}(person) > t$$

Matching facts and pre-conditions creates a *most general unifier* (MGU) that is propagated forward on the inference chain.

D. Creating a trainable path

During training, the final state is a *goal* of the system, as shown in Figure 2. The system learns how to create rules given a set of template *empty rules*, where the embeddings for each node and edge are chosen randomly. These templates are specified prior to the training using $*$ to indicate a random embedding, as in the following

```
MATCH *(a), *(a,b), *(b)
CREATE (b), *(b,d), *(d)
```

In the current state of development, the algorithm generates all possible paths - compatibly with boundary conditions - and then applies to each of them the training algorithm explained below. A more efficient method will be pursued in future works.

E. Training of the embeddings and thresholds

At every step of the inference chain the collection of predicates changes according to the order of transformations. At every step i, we employ a vector f_i that signals the truth value of each predicate. For computational reasons, the dimensions of this vector must be fixed in advance and set to the maximum size of the predicate set. The first value f_0 is a vector of ones, as every predicate in the knowledge base is assumed to be true.

At the end of the resolution chain there is a "goal" set of predicates, usually less numerous than the initial set of facts. A vector g indicates the truth conditions of the goal predicates. This vector - also of size n - contains a number of ones equal to the number of goal nodes and is zero otherwise. The application of a rule can then be described by two matrices: the similarity matrix S and the rule propagation matrix R.

A **similarity matrix** describes how well a set of facts matches the pre-conditions.

$$S_i = M_i \odot \text{Softmax}(P_i^T F_i - T_i) \quad (1)$$

Where P_i is the matrix with the pre-conditions's nodes as colums, F_i is the matrix with the fact nodes as columns at step i. M_i is the matrix of the matches, bearing value of 1 if two nodes match and vanishing otherwise. For example, if the first node of the pre-conditions matches the second node of the facts, the matrix will have value 1 at position (1, 0).

The matrix T_i is a bias matrix whose columns are the list of (trainable) thresholds for each predicate in the pre-conditions $T_i = \left[t_1^i, t_2^i, ... \, t_n^i\right]$. This bias effectively enforces the matching thresholds: A negative value as an argument to Softmax will lead to an exponentially small result after the operation.

All the matrices M, P, and F are square matrices $\in \mathcal{R}^{n \times n}$. Equation (1) is reminiscent of self-attention [7], with an added bias matrix T and a mask M.

A **rule propagation matrix** R puts into contact the left side of a rule with the right side. The idea behind R is to keep track of how information travels inside a single rule. In this work we simplify the propagation matrix as a fully connected layer with only one trainable parameter. For example, if the chosen size n is 4, a rule with three pre-conditional nodes and two post-conditional nodes has an R matrix as

$$R_i = w \begin{bmatrix} 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad (2)$$

where w is the "weight" of the rule. Given a first state f_0, the set of truth condition after n steps is

$$f_n = S_{n-1} ... R_1 S_1 R_0 S_0 f_0 \quad (3)$$

This final state f_n is compared against the goal's truth vector g to create a loss function.

The training of the relation embeddings follows the same sequence of operations as for the nodes. A set of truth vectors f^r

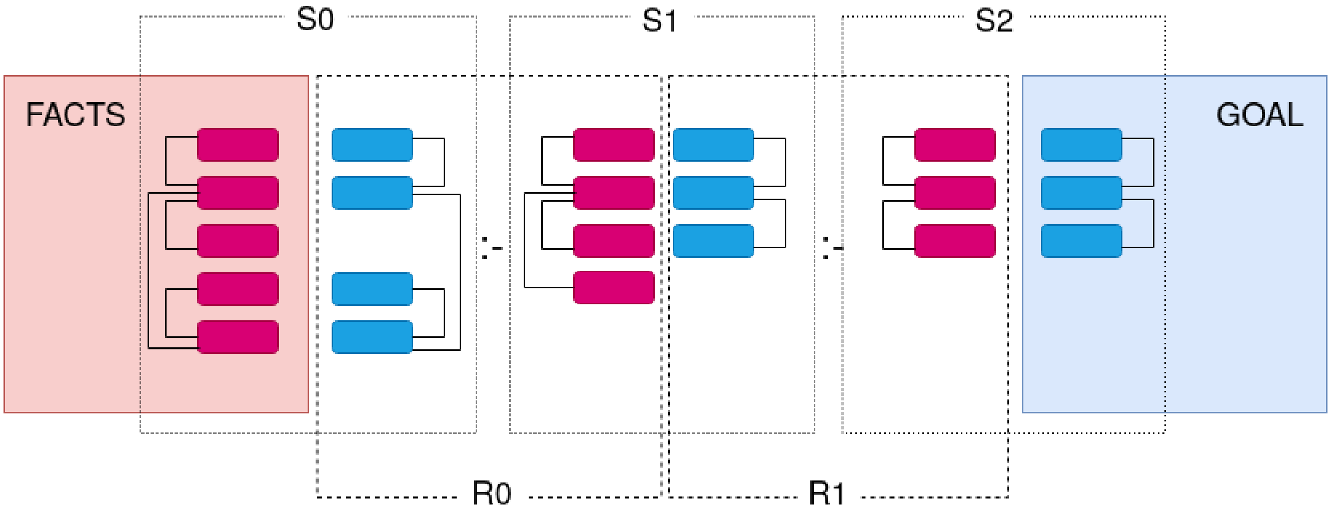


Figure 2. Example of a graph matching path. The facts on the left (represented as a graph of connected embeddings) are transformed through a chain of pre- and post-conditions into the goal on the right. This chain of rules is equivalent to a sequence of linear algebra operations, where the truth values of each predicated are propagated forward through a set of S and R matrices.

and states F^r is acted upon the relation similarity matrix

$$S_i^r = M_i^r \odot \mathrm{Softmax}({P_i^r}^T F_i^r - t_i^r), \tag{4}$$

and the corresponding rule propagation matrix for relations R_i^r, leading to the final truth vector for relations

$$f_n^r = S_{n-1}^r ... R_1^r S_1^r R_0^r S_0^r f_0^r. \tag{5}$$

Following the example of the nodes, a goal vector for the relations is named g^r, containing the desired truth conditions for relations at the end of the chain.

The system learns the node and edge embeddings of the rules, while the initial facts and the goal are frozen during training. The system also learns the *matching thresholds* t and each rule's *weight* w. Following (3) and (5), the final loss function is computed as a binary cross entropy expression

$$\mathcal{L} = g\ \log(f_n) + g^r\ \log(f_n^r). \tag{6}$$

The system can in principle be trained over a set of multiple facts and goal pairs, in which case the loss function is the sum of all the pairs' losses. For simplicity, in this paper we limit the training to a single pair of facts and goal.

In order to avoid the Sussman anomaly, the same rule can only be used once in the same path.

III. Examples and Discussion

A. One-rule learning

As a toy example we want the system to learn that if someone is married to a "first lady", then this person is president. The facts are

```
person(a), spouse(a,b), person(b), be(a,c), first-lady(c)
```

and the goal is

```
person(a), profession(a,b), president(b)
```

Given the empty rule

```
MATCH *(a), *(a,b), *(b), *(a,c), *(c)
CREATE (b), *(b,d), *(d)
```

The system correctly learns the rule that connects the facts with the goal.

```
MATCH person>0.6(a), first-lady>0.6(b), person>0.6(c),
      be>0.63631916(a,b), spouse>0.6338593(a,c)
CREATE (b), president(d), profession(b,d)
```

While trivial, this is a fundamental test of the capacity of the system to learn the correct transformation. The matching thresholds have been clipped and cannot go below 0.6 in training.

While a successful result is almost guaranteed by choosing a rule that closely matches the boundary conditions, the system is proven capable of converging onto the correct embeddings and thresholds using just backpropagation.

B. Chained two-rule learning

While a single-rule transformation can be useful in a few edge cases, the real power of semantic reasoning comes from combining rules together. In this section we show - using another toy example - that the system can learn two rules at the same time. The simplified task is as in the following: to learn that "if a fruit is round and is delicious, then it is an apple." The facts are

```
fruit(a), be(a,b), round(b), be(a,c), delicious(c)
```

and the goal is

```
fruit(a), be(a,b), apple(b)
```

The system is given the two template rules to fit

```
MATCH *(a), *(a,b), *(b), *(a,c), *(c)
CREATE (b), and(b,c), (c)

MATCH *(a), and(a,b), *(b)
CREATE *(c), *(c,d), *(d)
```

Notice the "and" relations in the templates. These relations are frozen during training and constitute another constraint for the system to satisfy. In the end, our model learns the correct rules

```
MATCH fruit>0.6(a), round>0.6(b), delicious>0.6(c),
      be>0.6953449(a,b), be>0.6957883(a,c)
CREATE (b), (c), and(b,c)

MATCH round>0.6(a), delicious>0.6(b), and>0.9(a,b)
CREATE fruit(c), apple(d), be(c,d)
```

which satisfy the goal when chained.

Here, we forced the system to apply two rules since no single template would fit the boundary conditions. Of particular interest is the fact that the system learned the pre-conditions of the second rule $round > 0.6(a)$, $delicious > 0.6(b)$, $and > 0.9(a,b)$. This is not a trivial task, given that it started training with random embeddings and the only information about the correct values is the one propagated forward from the first rule.

IV. Related works

Neuro-symbolic reasoning has been an intriguing line of research in the past decades [8][9]. Some recent results make use of a Prolog-like resolution tree as a harness where to train a neural network [10], [11], [12], [13]. Our work is similar to theirs, but builds upon a STRIPS-like system instead of Prolog. A different approach employs a Herbrand base for inductive logic programming in a bottom-up solver [14].

Finally, one can see our method as a sequence of operations that create or destroy items sequentially. Each (differential) transformation brings forward a new state of the system made by discrete elements. These types of algorithms have already been investigated in the Physics community, for example in [15].

V. Conclusions

In this work we presented a semantic reasoner that leverages on differential graph transformations for rule learning. The system is built through a one-to-one correspondence between a chain of rules and a sequence of linear algebra operations. Given a set of facts, a goal, and a set of rules with random embeddings, the reasoner can learn new rules that satisfy the constraints. The rules are then written as a set of predicates with pre- and post-conditions, a more interpretable representation than embeddings and weights.

The system presented here is limited in speed and - as a consequence - volume of training data. This is mostly due to our path-creation algorithm, which generates all possible paths given a set of rules. A more efficient algorithm would employ a guided approach to path creation, similar to the method in [13]. A different and possibly novel efficiency gain could be found in a Monte Carlo method, where the path converges to the correct one through means of a Metropolis algorithm.

Using a more efficient algorithm the system would be able to leverage on a higher number of templates, thus making the system useful outside the set of toy examples presented here. In this scenario, another topic that needs addressing is how to best generate the templates from the available data.

Finally, an open question resides on whether the system is able to generalize, given multiple sets of facts and goals. This last inquiry will need a faster algorithm and will be pursued in a future work.

References

[1] A. d'Avila Garcez and L. C. Lamb, "Neurosymbolic AI: The 3rd Wave," arXiv e-prints, Dec. 2020, p. arXiv:2012.05876.

[2] M. Krötzsch, F. Simancik, and I. Horrocks, "A description logic primer," ArXiv, vol. abs/1201.4089, 2012.

[3] H. Ehrig, C. Ermel, U. Golas, and F. Hermann, Graph and Model Transformation. Berlin, Heidelberg: Springer-Verlag, 01 2015.

[4] "Source code," 2021. [Online]. Available: https://github.com/fractalego/dgt/tree/semapro2021

[5] R. E. Fikes and N. J. Nilsson, "Strips: A new approach to the application of theorem proving to problem solving," Artificial Intelligence, vol. 2, no. 3, 1971, pp. 189–208. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0004370271900105

[6] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in Empirical Methods in Natural Language Processing (EMNLP), 2014, pp. 1532–1543. [Online]. Available: http://www.aclweb.org/anthology/D14-1162

[7] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is all you need," in Advances in Neural Information Processing Systems 30, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds. Curran Associates, Inc., 2017, pp. 5998–6008. [Online]. Available: http://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf

[8] A. S. d. Garcez, D. M. Gabbay, and K. B. Broda, Neural-Symbolic Learning System: Foundations and Applications. Berlin, Heidelberg: Springer-Verlag, 2002.

[9] A. Garcez, L. Lamb, and D. Gabbay, "Neural-symbolic cognitive reasoning," in Cognitive Technologies. Berlin, Heidelberg: Springer-Verlag, 2009.

[10] T. Rocktäschel and S. Riedel, "End-to-end differentiable proving," in Advances in Neural Information Processing Systems, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates, Inc., 2017. [Online]. Available: https://proceedings.neurips.cc/paper/2017/file/b2ab001909a8a6f04b51920306046ce5-Paper.pdf

[11] P. Minervini, M. Bosnjak, T. Rocktäschel, and S. Riedel, "Towards Neural Theorem Proving at Scale," in ICML Workshop on Neural Abstract Machines and Program Induction (NAMPI), 2018.

[12] L. Weber, P. Minervini, J. Münchmeyer, U. Leser, and T. Rocktäschel, "NLProlog: Reasoning with weak unification for question answering in natural language," in Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. Florence, Italy: Association for Computational Linguistics, Jul. 2019, pp. 6151–6161. [Online]. Available: https://aclanthology.org/P19-1618

[13] P. Minervini, M. Bosnjak, T. Rocktäschel, S. Riedel, and E. Grefenstette, "Differentiable Reasoning on Large Knowledge Bases and Natural Language," in Proceedings of the Association for the Advancement of Artificial Intelligence Conference on AI (AAAI), 2020.

[14] R. Evans and E. Grefenstette, "Learning explanatory rules from noisy data," J. Artif. Int. Res., vol. 61, no. 1, Jan. 2018, p. 1–64.

[15] A. W. Sandvik, "The stochastic series expansion method for quantum lattice models," in Computer Simulation Studies in Condensed-Matter Physics XIV, D. P. Landau, S. P. Lewis, and H.-B. Schüttler, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2002, pp. 182–187.

An Extensible Semantic Data Fusion Framework for Autonomous Vehicles

Efstratios Kontopoulos, Panagiotis Mitzias, Konstantinos Avgerinakis, Pavlos Kosmides
Catalink Limited
Nicosia, Cyprus
email: {e.kontopoulos, pmitzias, koafgeri, pkosmidis}@catalink.eu

Nikos Piperigkos, Christos Anagnostopoulos, Aris S. Lalos
Industrial Systems Institute
Athena Research Center
Athens, Greece
email: {piperigkos, anagnostopoulos, lalos}@isi.gr

Nikolaos Stagakis, Gerasimos Arvanitis, Evangelia I. Zacharaki, Konstantinos Moustakas
Electrical and Computer Engineering
University of Patras
Patras, Greece
email: {nick.stag, arvanitis, moustakas}@ece.upatras.gr
ezachar@upatras.gr

***Abstract*—Fully autonomous vehicles may still be an elusive goal, however, research in the deployment of relevant Artificial Intelligence technologies in the domain is rapidly gaining traction. A key challenge lies in the fusion of all the diverse information from the various sensors on the vehicle and its environment. In this context, ontologies and semantic technologies can effectively address this challenge by semantically fusing heterogeneous pieces of information into a uniform Knowledge Graph. This paper presents CASPAR, an extensible semantic data fusion platform for autonomous vehicles. Two use case scenarios are also presented that demonstrate the framework's versatility.**

Keywords-Autonomous vehicles; ontologies; knowledge graphs; semantic data fusion; AI.

I. INTRODUCTION

Although fully autonomous vehicles are still an elusive goal, research in the field is rapidly gaining traction, with relevant studies estimating the value of the automotive AI market at a little over $10.5 billion by 2025 [1]. Consequently, most major automotive manufacturers are increasingly investing in the field.

The key challenge in this domain lies in the fact that AI systems operating in such dynamic settings must deal effectively with large volumes of streaming data generated by the various sensors on the vehicle (e.g., camera, radar, Light Detection and Ranging – LiDAR, Global Positioning System – GPS, etc.) as well as by the vehicle's environment (e.g., pervasive inputs, weather data, other vehicles, etc.). The fusion of all this diverse information is, thus, a non-trivial and highly error-prone process.

In this context, semantic technologies and, most prominently, ontologies seem like a natural fit for semantically fusing heterogeneous pieces of information into a uniform knowledge representation model, i.e., a Knowledge Graph (KG). This paper presents ongoing work on an extensible semantic data fusion framework for autonomous vehicles, called CASPAR [2]. The framework is part of a larger platform being developed within the context of the CPSoSaware EU-funded project [3].

The rest of the paper is structured as follows: Section 2 gives an overview of related work on deploying semantic technologies in the domain of autonomous and connected vehicles. Section 3 presents the proposed approach, describing the architecture, input sources, as well as the semantic data fusion component. Section 4 presents two use case scenarios and their evaluation, and, finally, Section 5 concludes the paper with some final remarks and directions for future work.

II. RELATED WORK

It was roughly 20 years ago that the issue of data heterogeneity in the automotive industry was gradually emerging as a critical challenge, and the first approaches proposed the addition of a semantic layer on top of the lower-level sensors and systems of the vehicle. In collaboration with the German car manufacturer Audi AG, the authors in [4] proposed an ontology for representing the various parts and sensors of a car. Other early approaches adopted ontology-based representation of the context and the situations surrounding the vehicle, like, e.g., the road network and all detected objects in the scene, an estimation of the behaviours of other traffic participants, as well as the mission goal of the own vehicle [5].

In the same context, the works presented in [6]-[8] propose ontology-based representations of road intersections and road infrastructures that could serve the basis for traffic models and systems that could predict conflicts between vehicles reaching the same intersection.

Extending the scope beyond representing vehicle- and sensor-related aspects, other approaches adopt a user-centred view of the world, also considering aspects like the mental and physiological state of the driver [9][10] or their grip force and alcohol density [11].

In more recent works, bigger players entered the game, and more holistic Advanced Driver Assistance Systems

(ADAS) were proposed, utilizing a wider range of (the now more mature) semantic technologies. [12] and [13] present intelligent decision-making systems, as part of an ADAS, for assisting autonomous vehicles in making appropriate decisions during certain cases. The systems consist of an ontology-based KG, as well as a set of Semantic Web Rule Language (SWRL) rules for representing traffic regulations and spatiotemporal relationships between entities. In both works, thorough evaluations regarding semantic reasoning and result-set retrieval times are conducted, but, arguably, the respective sizes of the KGs are rather small.

The authors in [14] present a more standards-oriented approach, proposing the Vehicle Signal and Attribute Ontology (VSSo) that is based on the Sensor, Observation, Sample, and Actuator (SOSA) ontology [15] for representing sensor measurements, on the Vehicle Signal Specification (VSS) [16] for representing domain-pertinent aspects (i.e., vehicle signals), and on the Web of Things principles [17] for defining technology and protocol-independent interactions with Web Things. This combination facilitates the decorrelation from automotive standards, enabling the collection and analysis of sensor data coming from vehicles of different models and brands, and allows integrating car data with data coming from other Internet-of-Things (IoT) sources from the Web. As suggested by the authors, VSSo can form the basis for various applications like car fleet monitoring, car trajectory mining, contextual representation of a car and interaction between any car and web services.

Compared to the existing approaches presented above, our framework does not ingest raw sensor measurements into the KG, but instead adds the higher-level outputs generated by analyses performed by other components at a lower level, like, e.g., Driver Monitoring Systems (DMSs), driver's wearables, and visual odometers. This approach offers richer insights about various aspects of the vehicle and the driver, like, e.g., the system health or the driver's state during a driving session.

III. Proposed Approach

This section presents our proposed approach, focusing on the architecture, input sources, as well as the CASPAR semantic data fusion component.

A. Architecture

An overview of the system encompassing the semantic data fusion framework is presented in Figure 1. Adopting the microservice methodology [18], we defined a set of independent, replicable services that collaboratively fulfil the system's functionality. For the communications among services, we deployed RabbitMQ [19], a popular open-source message broker that is scalable and industry-ready.

The system components in the monitoring layer periodically collect and analyze data related to the driver, the vehicle and its surroundings. Five monitoring components (DSO, LeGO, CL, DMS and OFE) - all introduced in the next subsection - are currently integrated. However, the modularity provided by the microservice approach, coupled with the straightforward system design, enables the integration of third-party data sources (e.g., weather or traffic condition reports) with minimum effort.

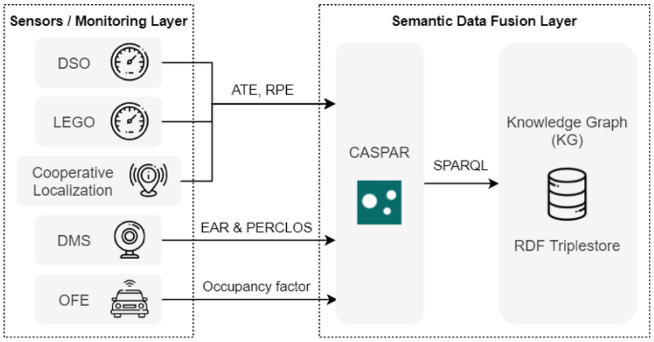


Figure 1. Overview of the system architecture.

The outputs and observations produced by the monitoring layer are communicated to the semantic data fusion layer via a dedicated RabbitMQ exchange. At this stage, they are mapped to ontology concepts, resulting in a unified Knowledge Graph (KG), which is instantiated in a Resource Description Framework (RDF) triplestore by the CASPAR component, which is further described in a next subsection.

B. Input Sources

1) Odometry Algorithms: The quantitative trajectory evaluation of odometry algorithms is an issue which has been examined thoroughly by the research community. A few metrics have been proposed over the last years, of which the *Absolute Trajectory Error* (*ATE*) and the *Relative Pose Error* (*RPE*) are the most popular. More specifically, let us assume that the output of a Simultaneous Localization and Mapping (SLAM) algorithm, thus the estimated trajectory is a set of *n* distinct poses $P_i \in SE_3$, where SE_3 is the Special Euclidean space of rigid body transformations in three dimensional space. Each element of this space can be expressed in the form of a 4x4 matrix:

$$M = \begin{pmatrix} R & T \\ 0 & 1 \end{pmatrix}$$

where $R \in SO_3$ is the rotation part, $T \in R_3$ is the translation part, and SO_3 is the special orthogonal group that contains the rotations. Accordingly, the ground truth trajectory is consisted of n $G_i \in SE_3$ poses in an arbitrary coordinate system.

In order to compute the ATE, which gives us the total consistency of the algorithm, we have to align the two trajectories using an algorithm like Horn's method [20]. Consequently, the ATE error matrix E_i for each of the n estimated poses, can be computed by the following equation:

$$E_i := G_i^{-1} A P_i$$

where A is the alignment matrix. Usually, we use the Root Mean Square Error (RMSE) of ATE which is calculated as follows [21]:

$$E_{rmse} := \sqrt{\frac{1}{n}\sum_{i=1}^{n}\|T_i\|^2}$$

The RPE is a metric which indicates the accuracy of the algorithm over a specific time step. Let us assume that we have a common time step Δ for both the algorithm and the ground truth trajectory, the RPE matrix R_i^{Δ} can be calculated by the following equation [21]:

$$R_i^{\Delta} := (G_i^{-1}G_{i+\Delta})^{-1}(P_i^{-1}P_{i+\Delta})$$

The RMSE for the translation of RPE is calculated as in the case of ATE.

The odometry algorithms used in this paper are presented below:

Direct Sparse Odometry (*DSO*) [22] is a state-of-the-art monocular visual odometry solution, relying on the Camera sensor. Contrast to most related methods, it features the combination of Sparse+Direct: it optimizes the photometric error defined directly on images, without exploiting any geometric prior, using the so-called keypoints from some keyframes. One of the main benefits of keypoints is their robustness to photometric variations. In addition, the main drawback of adding geometry priors is the introduction of correlations between geometry parameters, which render a statistically consistent, joint optimization in real time infeasible. DSO provides a strategy for keyframes and keypoints management, which leads to a windowed optimization problem, solved by Gauss-Newton (GN) method. Only the most useful frames out of consecutives frames are kept (tracking). And then, some active points are determined in order to estimate the pose of the vehicle using GN.

LeGO-LOAM [23] is a lightweight and ground-optimized LiDAR odometry solution, which introduces a two-step Levenberg Marquardt (LM) optimization for pose estimation. As shown in Figure 2, before the feature extraction module, the point cloud from LiDAR is being processed by the segmentation module.

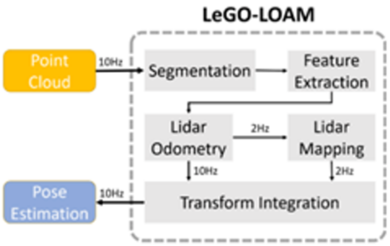


Figure 2. LeGO-LOAM system overview.

The segmentation model is responsible for creating clusters of points from the point cloud. More specifically, it assigns three related values to individual 3D points: (a) label as a ground or segmented point, (b) column and row index in the depth image (created by projecting all points to the image plane), (c) depth value. Feature extraction is responsible for categorizing the points from segmentation to either planar or edge features. And finally, the pose is estimated by performing LM optimization using these two groups of vehicles between consecutive LiDAR scans.

Cooperative Localization (*CL*) is expected to further improve the positioning accuracy of the Localization sub-system of Connected and Autonomous Vehicles. Vehicles, apart from the advanced sensors of LiDAR, Camera, etc., benefit from direct V2V communication and exchange of rich information, for increased perception and scene analysis ability. Graph Laplacian processing [24][25] enables the fusion of heterogeneous measurements from vehicles in a linear and compact way, contributing to efficient location estimation. It makes use of connectivity representation of collaborating vehicles, along with the inter-vehicular measurements (noisy distances, angles, and positions) provided by the Perception sub-system, in order to formulate a linear least-squares estimation problem. Combined with the Extended Kalman Filter, it also exploits the motion properties of vehicles, significantly increasing location accuracy [26]. Note that CL could be useful for mitigating GPS location spoofing cyberattacks [27].

2) Driver Monitoring System (DMS): Our DMS module captures frontal facial images of the driver to assess fatigue levels based on the activity of the eyes. The driver's drowsiness is measured based on two metrics, the *Eye Aspect Ratio* (*EAR*) [28] and the *PERcentage of Eye CLOSure* (*PERCLOS*) [29]. To obtain the facial landmarks we use the Dlib toolkit, which provides us with 68 facial landmarks characterizing various facial features, such as eyes, nose, mouth, etc. From those 68 points, 12 points correspond to the eyes. We use these landmarks to calculate the ratio of the vertical and horizontal lines defined by the eclipse that is fitted to the eye. This ratio is computed as:

$$EAR = \frac{\|p_2 - p_6\| + \|p_3 - p_5\|}{2\|p_1 - p_4\|}$$

Typical values indicating eyelid closure were determined at EAR < 0.2 in [28]. In our tests, we found that an EAR threshold closer to 0.25 performs better in this simulation context.

The PERCLOS measure is defined as the percent of the time the eyelid occludes the pupil (EAR < 0.25) within a *K*-second moving window, where *K* can be tuned by the user. Typical values of K are around 60 seconds. Therefore, PERCLOS is calculated as:

$$\text{PERCLOS}(\%) = \frac{\text{num of frames where eyes are closed}}{\text{frame interval length}} \times 100$$

In literature, the values that have been suggested as representative of low drowsiness state are typically under the 0.25% PERCLOS and 70% or 80% (known as PERCLOS70 and PERCLOS80, respectively) for high drowsiness [30].

3) Occupancy Factor Estimation (OFE): The Occupancy Factor (OF) is an empirical metric, extracted by analysing the point cloud of the scene that has been acquired by the LiDAR device. It indicates how "clear and open" the road is beyond the driver's field of view. Based on this value, the road's condition can be separated into three categories: (a) Safe road without objects, (b) Road with small obstacles (e.g., potholes) or cars at a far distance from the vehicle, (c) Road with a lot of traffic, parked cars, etc.

OF is estimated via point cloud processing. In more detail, after the acquisition of the point cloud by the LiDAR, a geometry processing technique is applied to estimate the saliency map of the point cloud scene [31].

The saliency map extraction assigns at each vertex of the point cloud a value based on its distinctiveness (geometrical importance). To visualize the saliency map of the point cloud, we quantize the range of value into 64 classes which we map to 64 colours, as presented in Figure 3. The lowest saliency values correspond to deep blue, while the highest values correspond to deep red. Vertices that lie in totally flat areas take the lowest value (as not being salient), while vertices lying in very sharp corners take the highest value.

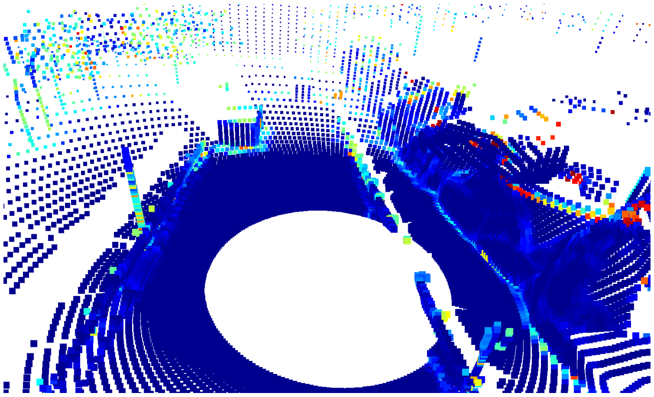


Figure 3. Example point cloud showcasing our color mapping.

The next step addresses the spatial scene analysis. Specifically, we are interested in the segmentation of the road. For estimating OF, we take into consideration only these set of vertices, denoted as *N* below, that: (a) belong to the region of the road and (b) correspond to the lowest saliency value (i.e., totally flat area of the road). Finally, OF is estimated as the sum of the inverse norm2 distance between each vertex, of the previously aforementioned set of vertices, and the point v_1 *(0,0,0)* that represents the centre of the LiDAR sensor.

$$\mathrm{OF} = \sum_{\forall i \in \mathcal{N}} \frac{1}{||v_i - v_1||_2}$$

The higher the OF value, the less occupied the road for driving.

C. Semantic Data Fusion

The integration of inputs (see previous subsection) to the unified KG is handled by *CASPAR* (*Structured Data Semantic Exploitation Framework*), our domain-agnostic tool for the automated retrieval and fusion of structured data from disparate sources into domain-specific semantic models, facilitating the discovery of new knowledge along with the extraction of actionable insights.

CASPAR is based on the ontology population principles presented in recent works of ours [32][33]. In a nutshell, the tool administers a set of interconnected mechanisms for transforming data into knowledge, represented in a machine-interpretable and exploitable format (RDF). These mechanisms incorporate: (a) the automated acquisition of structured data from user-defined sources (APIs, databases, messaging buses, etc.), (b) the mapping of input data fields to semantic entities (concepts, relationships, etc.), (c) the semantic fusion and population of knowledge into a semantic repository, (d) the semantic enrichment of existing knowledge from external Linked Open Data repositories, and, (e) the application of rule-based semantic reasoning to unveil underlying or generate new knowledge.

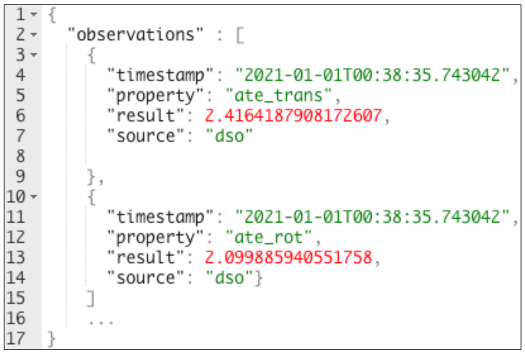
For the purposes of this work, the SOSA ontology [15] serves as the core semantic model for describing sensors and their observations, the studied features of interest and the observed properties. As described in the next section, the core model is populated with the outputs generated by the input sources, i.e., other analysis components in the CPSoSaware project architecture.

IV. Scenarios and Results

This section presents two use case scenarios applying the proposed architecture and semantic data fusion component. Evaluation results are also discussed.

A. Scenario #1: Evaluate the Robustness of Odometry Algorithms

In the first scenario we rely on an end-to-end testing framework, based on the CARLA open-source urban driving simulator [34], for generating synthetic sensory data and evaluating the three aforementioned odometry algorithms against different weather and lighting conditions. Each algorithm uses a different modality and our purpose is to study the effect of the changing conditions on the efficiency of each algorithm.



```
{
  "observations" : [
    {
      "timestamp": "2021-01-01T00:38:35.743042",
      "property": "ate_trans",
      "result": 2.4164187908172607,
      "source": "dso"

    },
    {
      "timestamp": "2021-01-01T00:38:35.743042",
      "property": "ate_rot",
      "result": 2.099885940551758,
      "source": "dso"}
    ]
    ...
}
```

Figure 4. Excerpt of the ATE and RPE observations submitted to CASPAR.

Based on the architecture described above, ATE and RPE measurements are sent via the message bus to the CASPAR semantic data fusion framework and are ingested into the KG. Indicatively, 1226 observations were submitted for a driving simulation of 126 seconds. Figure 4 displays an excerpt of the observations, while Figure 5 illustrates the representation of the same sample observations in Graffoo format [35] fused inside the KG. As seen in the latter figure, no conflict resolution considerations are raised, since SOSA facilitates the explicit association of observations to the respective sources via `sosa:madeBySensor`, as well as the representation of different observed properties via `sosa:observedProperty`.

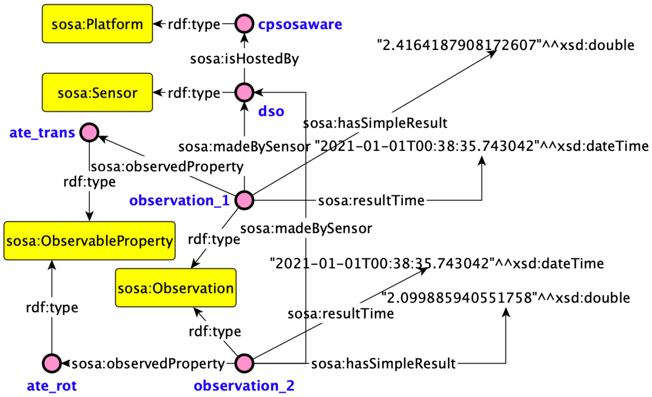


Figure 5. Excerpt of the ATE and RPE observations submitted to CASPAR.

After the population of the KG is complete, useful insights regarding the performance of the algorithm can be extracted. Figure 6 displays such an example, where the LiDAR-based odometry algorithm (LeGO-LOAM) presents better results with regards to RPE and seems to be more robust, constituting thus a better candidate in conditions similar to the specific simulation session.

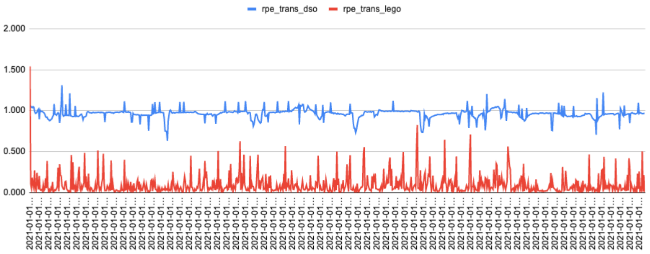


Figure 6. Performance comparison of LeGO vs DSO for a simulation session.

More specifically, as illustrated in Figure 6, the LiDAR odometer outperforms the visual odometer in terms of the relative drift between two consecutive poses, which depicts an indicative use case in which the reduced environmental light (night) resulted in the downgrade of the DSO performance.

Additionally, LIDAR's robustness has been also pointed out in [23]. Specifically, vision-based methods are sensitive to illumination and viewpoint changes. On the contrary, LiDAR functions well even at night and the high resolution of many 3D point-clouds permits the capture of the fine details of an environment at long ranges, over a wide aperture.

However, it must be noted that the superiority of LiDAR has only been identified in a specific set of scenarios. In the future, we plan to extend the set of scenarios and include more use cases (e.g., road bumps, sudden breaks, dynamic objects etc.), in order to further examine the robustness of the algorithms.

B. Scenario #2: Calculate Risk Levels during a Driving Session

In the second scenario, our objective is to inform the driver about potential risks during a driving session. We focus on two factors: The driver's drowsiness and the free available space of the road. For this purpose, two components have been developed (see also Figure 1): (a) the Driver Monitoring System (DMS) component, and (b) the Occupancy Factor Estimation (OFE) component.

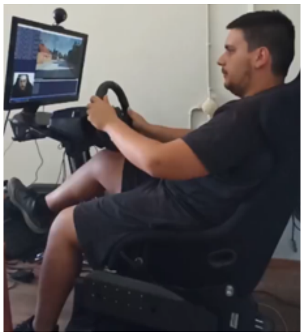


Figure 7. Driving simulation setup for integrating DMS and OEF.

For the evaluation of our implementation, we integrated the DMS with CARLA [34], whose spectacularly photorealistic graphics provide an immersive driving experience. The simulator provides the flexibility to design a variety of driving scenarios under different states of driver's drowsiness and different conditions of the road (e.g., the state of the traffic), in a safe environment for the operator who tests the implementation. The setup of the integration (see Figure 7) uses the Logitech G29 steering wheel for enhancing the driving sense, as well as a static web camera that captures the face of the driver in real-time.

Similar to the previous scenario, the DMS and OFE modules submit their observations, namely the PERCLOS and OF measurements, to CASPAR via RabbitMQ. However, an upgrade compared to scenario #1 entails a set of rules (see Table I) for calculating the risk levels during the simulated driving session. Risk level 1 corresponds to a "low risk" driving situation, where the driver is focused and drives carefully in a full open-eyed state (without any observed drowsiness). Moreover, the road is free from other vehicles, providing thus an unobstructed area for driving. On the other hand, risk level 3 corresponds to a "high risk" driving situation where the driver demonstrates intense drowsiness, as identified by the facial analysis of the DMS component, with intense drowsiness and/or the unobstructed area of the road is restricted (due to obstacles, a lot of traffic, small-ranged road, etc).

TABLE I. SET OF RULES FOR CALCULATING THE RISK LEVEL

	PERCLOS < 0.25	***PERCLOS >= 0.25 & <0.7***	***PERCLOS >= 0.7***
OF > 280	Low risk (1)	Be aware (2)	High risk (3)
OF <= 280 & >200	Low risk (1)	Be aware (2)	High risk (3)
OF <= 200	Be aware (2)	High risk (3)	High risk (3)

After the KG is populated through CASPAR, according to the approach described before (see Figure 5), the above ruleset is executed in the form of a respective SPARQL query "on-top" of the KG. The result is a risk level report, as illustrated in Figure 8. Outputs like this can constitute parts of reports, e.g., after traffic accidents.

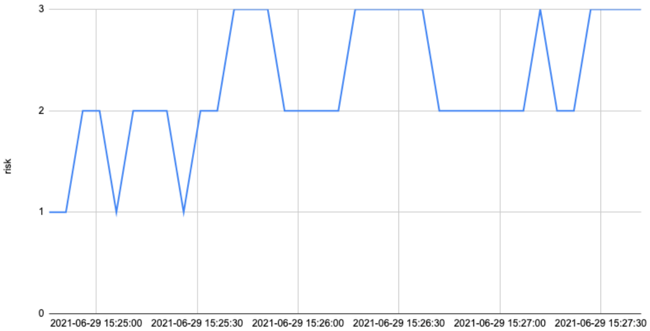


Figure 8. Output graph indicating the risk levels during a driving session.

Observing Table I and Figure 8, we see that when PERCLOS is higher than 0.7 (i.e., intense drowsiness), the risk level is always equal to 3 (i.e., high risk), independently of the value of OF. On the other hand, when PERCLOS is lower than 0.25, then the risk level is 1 (i.e., low risk), and correspondingly when the PERCLOS ranges from 0.25 to 0.7, the risk level is 2 (i.e., be aware). In these cases, the risk level is changed (level up) only when OF is lower than 200 indicating that the driver has to draw extra attention.

V. CONCLUSION

This paper presented CASPAR, a semantic data fusion framework for autonomous vehicle that is part of a larger platform in the context of an EU-funded project. The inputs to CASPAR constitute analysis results generated by other components in the platform and, this way, higher-level and richer insights can be derived regarding various aspects of the vehicle and the driver. In this context, the two scenarios presented in the paper demonstrate the framework's functionality and versatility in the domain.

However, this is largely still a work-in-progress. Thus, our next steps involve testing the semantic data fusion framework in a wider variety of simulation scenarios involving more sources of information (e.g., steering frequency, weather info, biometrics, etc.) and more challenging conditions (e.g., dynamic objects, reduced visibility, sudden braking, etc.). This would also entail extending the rule-base accordingly. A parallel future direction also involves considering the extraction of real-time analytics and insights, which, thus far, was not possible due to challenges in the scalability and performance of the triplestores we considered.

ACKNOWLEDGMENT

This work has received funding from the European Union's Horizon 2020 research and innovation programme under Grant Agreement No 871738 - CPSoSaware: Crosslayer cognitive optimization tools & methods for the lifecycle support of dependable CPSoS.

REFERENCES

[1] Automotive Artificial Intelligence Market by Offering (Hardware, Software), Technology (Deep Learning, Machine Learning, Computer Vision, Context Awareness and Natural Language Processing), Process, Application and Region - Global Forecast to 2025. Available at: https://www.marketsandmarkets.com/Market-Reports/automotive-artificial-intelligence-market-248804391.html. [retrieved: August, 2021]

[2] CASPAR homepage. Available at: https://caspar.catalink.eu/. [retrieved: August, 2021]

[3] CPSoSaware H2020 project homepage. Available at: https://cpsosaware.eu/. [retrieved: August, 2021]

[4] A. Maier, H. P. Schnurr, and Y. Sure, "Ontology-based information integration in the automotive industry" In International Semantic Web Conference, Springer, Berlin, Heidelberg, Oct. 2003, pp. 897-912.

[5] S. Vacek, T. Gindele, J. M. Zollner, and R. Dillmann, "Situation classification for cognitive automobiles using case-based reasoning" In 2007 IEEE Intelligent Vehicles Symposium, IEEE Press, June 2007, pp. 704-709.

[6] B. Hummel, W. Thiemann, and I. Lulcheva, "Scene understanding of urban road intersections with description logic" In Dagstuhl Seminar Proceedings, Schloss Dagstuhl-Leibniz-Zentrum fr Informatik, 2008.

[7] R. Regele, "Using ontology-based traffic models for more efficient decision making of autonomous vehicles" In 4th International Conference on Autonomic and Autonomous Systems (ICAS'08), IEEE Press, March 2008, pp. 94-99.

[8] M. Hülsen, J. M. Zöllner, and C. Weiss, "Traffic intersection situation description ontology for advanced driver assistance" In 2011 IEEE Intelligent Vehicles Symposium (IV), IEEE Press, June 2011, pp. 993-999.

[9] M. Feld and C. Müller, "The automotive ontology: managing knowledge inside the vehicle and sharing it between cars" In Proceedings of the 3rd International Conference on Automotive User Interfaces and Interactive Vehicular Applications, Nov. 2011, pp. 79-86.

[10] M. Madkour and A. Maach, "Ontology-based context modeling for vehicle context-aware services" Journal of Theoretical and Applied Information Technology, vol. 34(2), pp. 158-166, 2011.

[11] S. Kannan, A. Thangavelu, and R. Kalivaradhan, "An intelligent driver assistance system (i-das) for vehicle safety modelling using ontology approach" International Journal of UbiComp, vol. 1(3), pp. 15-29, 2010.

[12] A. Armand, D. Filliat, and J. Ibañez-Guzman, "Ontology-based context awareness for driving assistance systems" In 2014 IEEE intelligent vehicles symposium proceedings, IEEE Press, June 2014, pp. 227-233.

[13] L. Zhao, R. Ichise, S. Mita, and Y. Sasaki, "Core Ontologies for Safe Autonomous Driving" In International Semantic Web Conference (Posters & Demos), Oct. 2015.

[14] B. Klotz, R. Troncy, D. Wilms, and C. Bonnet, "VSSo: The Vehicle Signal and Attribute Ontology" In SSN@ ISWC, Oct. 2018, pp. 56-63.

[15] A. Haller et al., "The modular SSN ontology: A joint W3C and OGC standard specifying the semantics of sensors, observations, sampling, and actuation" Semantic Web, vol. 10(1), pp. 9-32, 2019.

[16] Vehicle Signal Specification: https://genivi.github.io/vehicle_signal_specification/. [retrieved: August, 2021]

[17] M. Kovatsch, R. Matsukura, M. Lagally, T. Kawaguchi, K. Toumura, and K. Kajimoto, "Web of Things (WoT) Architecture" W3C Recommendation 9 April 2020. Available at: https://www.w3.org/TR/wot-architecture/Overview.html. [retrieved: August, 2021]

[18] S. Newman, "Building microservices: designing fine-grained systems" O'Reilly Media Inc, 2015.

[19] M. Rostanski, K. Grochla, and A. Seman, "Evaluation of highly available and fault-tolerant middleware clustered architectures using RabbitMQ" In 2014 federated conference on computer science and information systems, IEEE Press, Sept. 2014, pp. 879-884.

[20] B. K. Horn, "Closed-form solution of absolute orientation using unit quaternions" Josa a, vol. 4(4), pp. 629-642, 1987.

[21] D. Prokhorov, D. Zhukov, O. Barinova, K. Anton, and A. Vorontsova, "Measuring robustness of Visual SLAM" In 16th International Conference on Machine Vision Applications (MVA), IEEE Press, May 2019, pp. 1-6.

[22] J. Engel, V. Koltun, and D. Cremers, "Direct sparse odometry" IEEE transactions on pattern analysis and machine intelligence, vol. 40(3), pp. 611-625, 2017.

[23] T. Shan and B. Englot, "Lego-loam: Lightweight and ground-optimized LiDAR odometry and mapping on variable terrain" In 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE Press, Oct. 2018, pp. 4758-4765.

[24] N. Piperigkos, A. S. Lalos, K. Berberidis, and C. Anagnostopoulos "Cooperative multi-modal localization in connected and autonomous vehicles" In 2020 IEEE 3rd Connected and Automated Vehicles Symposium (CAVS), IEEE Press, Nov. 2020, pp. 1-5.

[25] N. Piperigkos, A. S. Lalos, and K. Berberidis, "Graph based cooperative localization for connected and semi-autonomous vehicles" In 2020 IEEE 25th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD), IEEE Press, Sept. 2020, pp. 1-6.

[26] N. Piperigkos, A. S. Lalos, and K. Berberidis, "Graph Laplacian Extended Kalman Filter for Connected and Automated Vehicles Localization" In 2021 4th IEEE International Conference on Industrial Cyber-Physical Systems (ICPS), IEEE Press May 2021, pp. 328-333.

[27] C. Vitale et al., "CARAMEL: results on a secure architecture for connected and autonomous vehicles detecting GPS spoofing attacks" EURASIP Journal on Wireless Communications and Networking, vol. 2021(1), pp. 1-28, 2021.

[28] F. You, X. Li, Y. Gong, H. Wang, and H. Li, "A real-time driving drowsiness detection algorithm with individual differences consideration" IEEE Access, vol 7, pp. 179396-179408, 2019.

[29] D. F. Dinges and R. Grace, "PERCLOS: A valid psychophysiological measure of alertness as assessed by psychomotor vigilance" US Department of Transportation, Federal Highway Administration, Publication Number FHWA-MCRT-98-006, 1998.

[30] S. T. Lin, Y. Y. Tan, P. Y. Chua, L. K. Tey, and C. H. Ang, "Perclos threshold for drowsiness detection during real driving" Journal of Vision, vol. 12(9), pp. 546-546, 2012.

[31] G. Arvanitis, A. S. Lalos, and K. Moustakas, "Robust and fast 3-D saliency mapping for industrial modeling applications" IEEE Transactions on Industrial Informatics, vol. 17(2), pp. 1307-1317, 2020.

[32] E. Kontopoulos, P. Mitzias, M. Riga, and I. Kompatsiaris, "A Domain-Agnostic Tool for Scalable Ontology Population and Enrichment from Diverse Linked Data Sources" In DAMDID/RCDL, Oct. 2017, pp. 184-190.

[33] M. Riga, P. Mitzias, E. Kontopoulos, and I. Kompatsiaris, "PROPheT–Ontology Population and Semantic Enrichment from Linked Data Sources" In International Conference on Data Analytics and Management in Data Intensive Domains, Springer, Cham, Oct. 2017, pp. 157-168.

[34] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An open urban driving simulator" In Conference on robot learning (PMLR), Oct. 2017, pp. 1-16.

[35] R. Falco, A. Gangemi, S. Peroni, D. Shotton, and F. Vitali. "Modelling OWL ontologies with Graffoo." In European Semantic Web Conference, pp. 320-325. Springer, Cham, 2014.

ISBN: 978-1-61208-888-4

AWAPart: Adaptive Workload-Aware Partitioning Knowledge Graphs

Amitabh Priyadarshi
Department of Computer Science
University of Georgia
Athens, GA, USA
email: amitabh.priyadarshi@uga.edu

Krzysztof J. Kochut
Department of Computer Science
University of Georgia
Athens, GA, USA
email: kkochut@uga.edu

***Abstract*—Large-scale knowledge graphs are increasingly common in many domains. Their large sizes often exceed the limits of systems storing the graphs in a centralized data store, especially if placed in main memory. To overcome this, large knowledge graphs need to be partitioned into multiple sub-graphs and placed in nodes in a distributed system. But querying these fragmented sub-graphs poses new challenges, such as increased communication costs, due to distributed joins involving cut edges. To combat these problems, a good partitioning should reduce the edge cuts while considering a given query workload. However, a partitioned graph needs to be continually re-partitioned to accommodate changes in the query workload and maintain a good average processing time. In this paper, an adaptive partitioning method for large-scale knowledge graphs is introduced, which adapts the partitioning in response to changes in the query workload. Our evaluation demonstrates that the performance of processing time for queries is improved after dynamically adapting the partitioning of knowledge graph triples.**

Keywords-knowledge graphs; adaptive graph partition; query workload.

I. INTRODUCTION

The availability of large-scale knowledge graphs, which often holds hundreds of millions of vertices and edges, such as the ones used in social network systems or in other real-world systems, requires large-scale graph processing. Of-ten, these datasets are too large to be stored and processed in a centralized data store, especially if it is maintained in main memory. Instead, the knowledge graph often needs to be partitioned into multiple sub-graphs, called shards, and transferred to multiple nodes in a distributed system. However, these systems frequently suffer from network latency. One of the techniques to improve the query answering performance is to reduce the inter-process communication between graph processing subsystems. While graph partitioning may be an effective pre-processing technique to improve the runtime performance, the cost of frequent partitioning of the entire large-scale knowledge graph may be prohibitive. In this case, it would be advantageous to partition the graph only once, initially, and make only necessary partitioning adjustments, afterwards. For example, such adjustments are needed in case of changes to the query workload.

First, let us talk about graph partitioning. Given a graph $G = (V, E)$, where V is a set of vertices and E is a set of edges and a number k >1, a *graph partitioning of G* is a subdivision of vertices of G into *subsets* of vertices V_1, ..., V_k that partition the set V. A *balance constraint* requires that all partition blocks are equal, or close, in size. In addition, a common objective function is to minimize the *total number of cuts*, i.e., edges crossing (cutting) partition boundaries.

Our knowledge graph dataset is in the form of Resource Description Framework (RDF) [1]. RDF enables the embedding of machine-readable information on the web. A resource can be represented using a URL on the web. The RDF statement which comprises of three parts called a triple, consists of (s, p, o) resource, property, and value of resource. RDF Schema [2] defines Classes and Properties that create a taxonomy for arranging the RDF data. Web Ontology Language (OWL) [3] is a language to describe complex knowledge about the things and provides a way to represent the relationships between a group of things. The documents in OWL are known as Ontologies [4]. The RDF query language SPARQL is the W3C standard that is used for querying the data in RDF graphs for exploring relationships between resources. SPARQL tries to match a triple pattern in an RDF graph. A SPARQL endpoint accepts SPARQL queries that return the result via HTTP. The partitioned RDF graph can be accessed through different SPARQL endpoints in a single query using Federated SPARQL Query [5]. The SERVICE keyword is used to direct a portion of a query towards a particular SPARQL endpoint. In Table 1 there is an example of LUBM's [22] SPARQL query and its federated query. The federated query processor merges the results coming from the various SPARQL endpoints.

TABLE I. ORIGINAL AND FEDERATED QUERY OF LUBM 9TH QUERY

Original Query	**Federated Query**

SELECT ?X ?Y ?Z FROM lubm
WHERE{
?X rdf:type ub:Student.
?Y rdf:type ub:Faculty .
?Z rdf:type ub:Course .
?X ub:advisor ?Y .
?Y ub:teacherOf ?Y .
?X ub:takesCourse ?Z .
}
	SELECT ?X ?Y ?Z FROM lubm
	WHERE {
	?X rdf:type ub:Student.
	SERVICE <Sparql endpoint> {?Y rdf:type ub:Faculty .}
	?Z rdf:type ub:Course .
	SERVICE <Sparql endpoint> {?X ub:advisor ?Y .
	SERVICE <Sparql endpoint> {?Y ub:teacherOf ?Y .}
	?X ub:takesCourse ?Z .
	}

This paper is outlined as follows. Section 2 provides an overview of related work. Section 3 discusses the partitioning method. Section 4 is about the architecture and workflow of the system. Section 5 is dedicated for the experiments, and Section 6 concludes the paper.

II. RELATED WORK

Usually, partitioning a large-scale graph decreases query processing efficiency. However, this decrease can be mitigated if the partitioning is adjusted to a query workload

ISBN: 978-1-61208-888-4

and tuned to reduce the workload demands for inter-partition communication. Related work on graph partitioning and its implication on query processing is addressed in this section.

The graph partitioning problem is NP-complete [6]. Many practical techniques have been developed to address this issue, including spectral partitioning methods [7] and geometric partitioning methods [8]. Barnard and Simon [9] proposed the multilevel method to graph partitioning, and Hendrickson and Leland [10] enhanced it. Coarsening, initial partitioning, and uncoarsening are the three basic phases of the multilevel technique. Karypis et al. [11] employ a recursive multilevel bisection method for graph bisection to generate a k-partition on the coarsest level in their partitioning approach.

Workload-aware, distributed RDF systems include DREAM [12], WARP [13], PARTOUT [14], AdPart [15], and WISE [16]. DREAM [12] only partitions SPARQL queries into subgraph patterns, not the entire RDF dataset. The RDF dataset is replicated among nodes. It is designed in a master-slave architecture, with each node using RDF-3X [17] on its assigned data for statistical estimation and query evaluation. WARP [13] assigns each vertex of the RDF graph to a partition using the underlying METIS system. The triples are subsequently assigned to partitions, which are then stored in a triple store on dedicated hosts (RDF-3X). WARP uses an n-hop distance to compute the query's center node and radius. If the query is within n-hops, WARP sends the query to all partitions to be executed in parallel. A complex question is broken down into multiple sub-queries, which are then run in parallel, and the results are merged. PARTOUT [14] uses normalization and anonymization to extract representative triple patterns from a query workload by substituting infrequent URIs and literals with variables. Frequent URIs (above a frequency threshold) are normalized. PARTOUT uses an adapted version of RDF-3X as a triple store for their n hosts. AdPart [15] is an in-memory RDF system that incrementally re-partitions RDF data. In an in-memory data structure, each worker stores its local set of triples. AdPart provides an ability to monitor and index the workloads in the form of hierarchical heat maps. It introduces Incremental ReDistribution (IRD), which is a query workload-guided combination of hash partitioning and k-hop replication. WISE [16] is a workload-aware, runtime-adaptive partitioning system for large-scale knowledge graphs. Based on changes in the workload, a partitioning can be modified incrementally by trading triples. The frequencies of SPARQL queries are kept in a Query Span structure. When migrating the triples, a cost model that maximizes the migration gain while preserving the balanced partition is applied.

AWAPart, presented in this paper, is a query-adaptive workload-aware knowledge graph partitioning algorithm that extracts features from both the query workload and the dataset. These features are utilized to create a distance matrix between queries and then cluster similar queries together using hierarchical agglomerative clustering. From the knowledge graph data, subgraphs (partitions) associated with these features are produced and distributed as shards in a computing cluster. The partitioning of the graph will be adjusted in response to changes in the workload, e.g., if some queries are replaced or their execution frequencies change. This is done by updating metadata with new features from new queries. These new features are being clustered again. Scoring helps the system to swap data associated with features from one shard to another. This swapping is done to reduce the edge cuts and minimize query runtime. Importantly, unlike the related systems, ours does not rely on a specialized data store implementation and uses an *off-the-shelf* knowledge graph storage and query processing system (Virtuoso [18]) and relies on standard SPARQL queries for distributed processing.

III. Workload-Aware Adaptive Knowledge Graph Partitioning

As the workload changes over time, an optimized (current) graph partition eventually becomes inefficient for the modified workload. AWAPart's goal is to adapt an existing knowledge graph partitioning to changes in the query workload, to optimize the workload processing time. Critical features from the current and modified workloads are extracted and analyzed. Features of queries in the changed workload are clustered, based on the similarity measures. The features in the new and old clusters are compared and a new optimized partition is created. The system then dynamically adjusts the deployed partitioning (shards) by exchanging triples belonging to the modified features between shards in the cluster. However, the analysis of the workload and the resulting adjustment of the partitions (shards) is infrequent. It can be performed in the background, without interrupting the process of querying. Queries in the workload are re-written to form federated SPARQL queries for processing on the cluster. Adjusting the partitioning (shards) aims to limit the number of distributed joins (utilizing triples from different shards), which decreases workload processing time.

A. *Query Feature Extraction*

The query feature metadata maintains the information about the triple patterns, which is referred as features in this paper, present in a set of triples. This metadata is maintained for each shard to describe the current set of triples in the shard.

The following features are used to describe various triple patterns, which are identified for the purpose of query workload clustering.

- Property (P): This feature represents all triples which share a given predicate P (triple's property).
- Property-Object (PO): This feature represents all triples sharing the same predicate P and the object (triple's property and object).

Other feature types used for query analysis are:

- Subject-Subject Join (SSJ): Triples sharing the same subject.
- Object-Object Join (OOJ): Triples sharing the same object.
- Object-Subject Join (OSJ): Triples connected on an entity which is the object in one triple and the subject in the other (it is referred as an "elbow" join in this paper).

We created the QueryAnalyzer which extracts the above features from the queries and creates the feature metadata.

This metadata represents the features, their frequencies, neighboring features, related data sizes and distributed joins in that query. This helps the system to optimize the partition by re-adjusting the partitioning based on the updated features. Currently, our QueryAnalyzer is built for the SPARQL query language, but it can be easily adapted to a different graph pattern-based query language, such as Cypher, used in the Neo4j [19] graph database.

Triples in the entire knowledge graph are indexed based on their subject, predicate and object, and the graph can be searched using any of them. For instance, it is easy to materialize the predicate feature and locate all triples with a given property P or any other triple pattern using a feature discussed above. For indexing the initial dataset of N-Triples, Apache Lucene API [20] is used to accelerate searching for triple features, while creating an initial partition [21] tailored to the initial query workload.

B. *Query Workload Clustering and Knowledge Graph Adaptive Partitioning*

The distance matrix is used as an input data for data mining, such as multi-dimensional scaling, hierarchical clustering, etc. To measure the similarity between queries in a workload, based on their features, Jaccard similarity is used which generates a distance matrix. Clustering uses this distance matrix. The Jaccard similarity of sets A and B is the ratio of the intersection of sets A and B to the union of sets A and B. $J_{SIM} = |A \cap B| / |A \cup B|$.

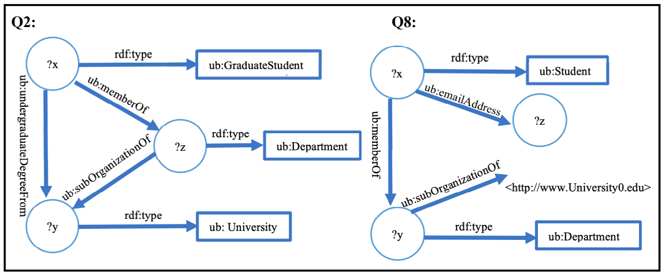
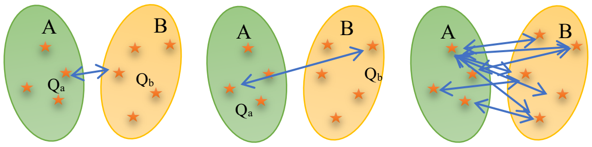


Figure 1. Distance between Q2 and Q8 is $1\text{-}J_{sim} = 1\text{-} (|Q2 \cap Q8|/|Q2 \cup Q8|) = (1\text{-}3/8) = 0.625$

In Figure 1, query 2 has 6 features: (3 *PO* features: *rdf:type→ ub:GraduateStudent, rdf:type→ ub:Department, rdf:type→ ub:University* and 3 *P* features: *ub:memberOf, ub:subOrganizationOf, ub:underGraduateDegreeFrom*) while query 8 has 5 features (2 *PO* features: *rdf:type→ub:Student, rdf:type→ ub:Department,* and 3 *P* features: *ub:emailAddress, ub:subOrganizationOf, ub:memberOf*). The Jaccard similarity, which is the ratio of the intersection of both sets to the union of both sets, is 3/8. Now, the distance between two similar sets should be 0 and the Jaccard similarity of two identical sets returns 1. Therefore, the distance between queries Q2 and Q8 is $(1\text{-}J_{SIM}\,(Q2, Q8)) = 1 - 3/8 = 0.625$.

We used the Hierarchical agglomerative clustering (HAC) algorithm (Figure 4), which is a method of creating a hierarchy of clusters in a bottom-up fashion. The creation of clusters is based on the measure of similarity between clusters and the selection of linkage method. The shortest pairwise distance between queries determines the grouping. The distance matrix is recalculated once the two most similar clusters are being grouped together. Jaccard is used to create this distance matrix. This distance matrix is used to start the HAC. Recalculation of the distance matrix is based on the choice of linkage from single, complete, or average (Figure 2). Single linkage is the proximity between two nearest neighbors, complete linkage is the proximity between the farthest neighbor and average linkage is arithmetic mean of all proximities between each object on each cluster with every object on another cluster. Running HAC using a single linkage on LUBM queries gives a dendrogram (Figure 3). Clustering is computed periodically, based on the changes in the query workload and generates new dendrograms.



a. Single Linkage (SL) b. Complete Linkage (CL) c. Average Linkage (AL)

Figure 2. a) $SL(A,B) = min(D(Q_a,Q_b)$, b) $CL(A,B) = max(D(Q_a,Q_b)$ and c) $AL(A, B) = \frac{1}{n_A n_B} \sum_{a=1}^{n_A} \sum_{b=1}^{n_B} D(Q_a, Q_b)$

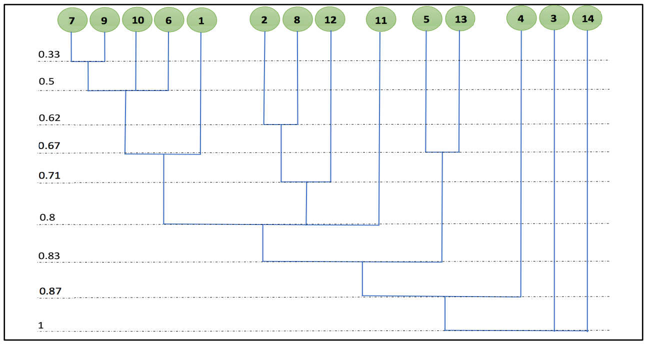


Figure 3. HAC Dendrogram of LUBM's 14 Queries

```
Input    Feature Distance Matrix D of workload Query
Output   HAC Dendrogram I
         Assign for each D[n][n] into C[m] where m = n*n
         while C.size > 1 do
           for i = 1 to  C.size  do
             if $(c_a,c_b)$ = min d$(c_a,c_b)$ in C  //Distance funct. d$(c_1,c_2)$
                 delete $c_a$ and $c_b$ from C
                 add min d$(c_a,c_b)$ in C
           assign I = (old,  $c_a c_{b,}$, min d$(c_a,c_b)$)
           recalculate proximity matrix using (SL/CL/AL)
           P = modifyDistance( $c_a$ ,$c_b$ ,min d$(c_a,c_b)$)
           for each P , $c_m$ = P[i][j]
           Update C = $c_1, c_2, …, c_m$
         Output I
```

Figure 4. Hierarchical Agglomerative Clustering of Queries

The adaptive partitioning algorithm (Figure 5) takes the initial partitioning and a new query workload as input and outputs the partition minimizing the distributed joins, based on the new workload, with its new sets of features. To eliminate replication of data, only one copy of query features is stored in the shards. For removal of the replication and to decide in which shard the only copy of triples associated with the selected features will be transferred, the algorithm compares the statistics for each P or PO feature in each shard.

```
Input    Initial Partition P, features $F_G$, New Queries workload $Q_{new}$
Output   Adaptive Partition A
Add queries $Q_{new}$ and its frequency $f$ in $Q_{old}$
Avg query execution time($T_{base}$) =$(\sum_{Q=1}^{n}(\frac{\sum_{i=1}^{f} T_{Qi}}{f}))/n$
Analyze Query $Q_{new}$ for features $F_{Qnew}$
Run HAC on $F_Q$, where $F_Q = F_{Qold} + F_{Qnew}$
Create Feature set g based on HAC at similarity distance d
Statistics (g, $F_Q$)
  Find key features $F_K$ in g.
  Find distributed joins of workload $D_{Q(old+new)} = (D_Q * f)$
  Find stats $S_K$ for each $F_K$
    Find p, q, s for shard $C_i$ and complete dataset T
    $S_K = (p_c w_1+q_c w_2+s_c w_3) + (p_t w_4+q_t w_5+s_t w_6)$ //key features in p (peer features), in q (query), s (triple size) and $w_1$ to $w_6$ are weights. c and t are cluster and total.
   Score for each $F_K = [min (D_{QR})*w * f]+ S_K$   //$D_{QR}$(distributed joins of $F_K$ in all query in every shard), $w$ (weight) and $S_K$ (key feature stat score).
Balance_Partition (Score, g, $F_G$)
  select all $F_K$ from g with highest scores for each $F_K$
  Assign data associated to features set g into P'.
  Proximity_Query ()
    Find $F_{prox}$= proximity of $F_{Unclustered}$ with $F_{Clustered}$
    Assign max($F_{prox}$) in cluster $P_i$' with their neighbor F.
    Assign $F_X = F_X$ + remaining $F_U$
  while $F_X$ not empty do
    $P'_{min}$ = Find min(P') by size of data
    $F_{max}$ = Find max(F in $F_X$) by size of data
   -Assign $F_{max}$ into $P'_{min}$
Avg execution time($T_{new}$) = $(\sum_{Q=1}^{p+n}(\frac{\sum_{i=1}^{f} T_{Qi}}{f}))/(p+n)$
if avg($T_{new}$) < avg($T_{base}$) then  A = P'
else Revert back and no change in P, A=P
Output A
```

Figure 5. Knowledge Graph Adaptive Partitioning Algorithm.

The statistics use other feature patterns, such as SSJ, OOJ and OSJ and distributed joins in queries. The statistics comprise of (1) out degree sequence (hops) starting from the key feature (q) in a query graph pattern and its successive (peer) feature (p) present in the sequence, (2) triple size ratio (s) of the key feature and its successive (peer) features in shards and in the complete dataset, and (3) distributed joins in the queries. To balance the partition, the algorithm uses the statistics to determine the out degree of other features in the query to the key feature. It also uses features that are not involved in the workload, but present in the dataset. The algorithm monitors the query execution time and stores the statistics. It outputs the changes to shard compositions, based on the above information. Triples associated with the selected features are moved between shards and the partition metadata is updated. This operation is infrequent, and we assume that the system adjusts the partitioning only after identifying a significant change in the workload processing (the system monitors the execution time for each query). Typically, once the execution time increases significantly (given a threshold) the current partitioning is modified and an exchange of triples takes place. Queries from the new workload run according to the updated partition metadata and the runtime of the queries are being recorded.

IV. IMPLEMENTATION

AWAPart stores an RDF dataset by partitioning it into sub-graphs, based on the initial query workload, and distributing the sub-graphs as shards among the nodes in a cluster. As the query workload changes, AWAPart establishes a new partitioning optimized for the new workload and dynamically adjusts the shards by triggering exchanges of subsets of triples between shards. The system is deployed on a single Master Node which controls the adaptive partitioning and a set of independent, share-nothing Processing Nodes, each with an installed triple store and a SPARQL query processor. The Master Node (Figure 6) is responsible for the overall workload analysis. It also controls the movement of triples subsets among the nodes in the cluster to adjust the partitioning. As the Master Node receives the query, the QueryAnalyzer and Feature Extractor (QAFE) starts the query feature extraction and updates the feature metadata. The Partition Manager (PM) uses the Hierarchical Agglomerative Clustering (HAC) module to cluster the extracted features. Using this HAC information, the Partition Metadata (PMeta) is updated. The dataset is indexed (IS) and according to PMeta, triples are searched and stored as shards. These shards are being uploaded to the processing nodes for the first time. A new query is sent to Query Rewriter and Processor (QRP), which rewrites the query into a federated query, based on the Partition Metadata (PMeta).

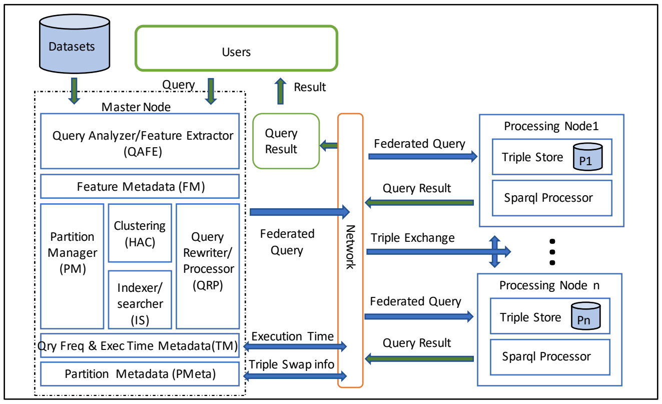


Figure 6. AWAPart System Architecture

This federated query is then sent to the processing node where it is going to be executed. The node where the query is executed is called the Primary Processing Node (PPN). The PPN is selected to minimize the distributed joins by selecting the shard with the highest number of features for the query. Adjustment of the partitioning of the RDF data is triggered by the Partition Manager (PM), due to changes in the workload query set and/or query frequency. The PM computes a new partition and, if the current shards require modifications, triples with selected features are exchanged between Processing Nodes to achieve a desired partitioning. The metadata of each Processing Node that was involved in triple swaps is updated to reflect the current state of triples in the shards. The PM uses the information stored in the Query Frequency and Execution Time Metadata (TM) and in the Feature Metadata (FM) with clustering information given by the Clustering Unit (HAC) to update Partition Metadata (PMeta). TM stores the information of every unique query and its average runtime.

V. EXPERIMENTS

The synthetic dataset and queries in the Lehigh University Benchmark (LUBM [22]) were used for the evaluation of AWAPart, our knowledge graph adaptive partitioning method based on a query workload. LUBM includes basic information organized as a knowledge graph about a set of universities and related entities. It includes a set of 14 SPARQL queries intended for benchmarking of knowledge graph storage/query systems. The experiments were conducted on a cluster of Intel i5-based systems running Linux Ubuntu 18.04.4 LTS 64-bit OS. A relatively small cluster was selected to focus on the effects of repartitioning of the datasets of manageable sizes. There are many available RDF triple stores that provide the functionality of storing and querying the RDF data, such as Redland [23], Sesame, Jena [24], Virtuoso, etc. In the experiments, an instance of OpenLink Virtuoso [18] was installed on each node in the cluster. The knowledge graph partitioning and adaptive repartitioning systems, as well as the experiments were coded in Java with the use of the Apache Jena framework.

Two experiments were used to evaluate the effects of adaptive knowledge graph partitioning system, based on workload. (1) The first experiment was designed to evaluate the effectiveness of the adaptive partitioning to accommodate the changes in the set of queries in the workload. (2) The second experiment was created to evaluate the adaptive partitioning in response to the changes in the frequency of specific queries in the workload (the set of queries in the workload is unchanged, but some queries are executed more often than initially). An LUBM dataset of 10 universities, which included 1,563,927 triples was created and used. The initial partition [21] is created based on the initial query workload. The experiments show that the AWAPart system offers significant performance improvements over a system where the initial partitioning was unchanged.

Experiment 1: This experiment demonstrates the effects of changes in the composition of the workload query set on their performance, when executed on the initial partition and then on the adaptive partitioning. The changes to the workload included additions of new and/or deletions of existing queries. The modified workloads runtime on the initial partition and on our adaptive partitioning for the LUBM dataset were evaluated. Figure 7 shows 10 extra queries [25] EQ1 to EQ10 and 14 old queries Q1 to Q14 for the LUBM dataset and their runtimes. EQ1 to EQ10 are a mixture of linear, star, snowflake, and complex queries. The figures show the improvement in runtime performance for queries from EQ1 to EQ10 in milliseconds. Except for Q9, the performance of the other 13 original queries does not change. Figure 8 shows the average runtime of all 24 queries on the initial partition versus the adaptive partition in milliseconds. An overall improvement of 2 seconds of the adaptive partition over the initial partition is shown. Despite the drop in performance of a single query, the overall performance gains are clearly visible. If Q9 were replaced in the new workload composition (Figure 7), the performance gain would be even higher. Figure 9 shows that the improvement of the average runtime of the 10 new queries (EQ) on the initial partition is approximately 56 seconds, while the adaptive partition decreases it to 21 seconds. It is an improvement of 63% in the average runtime of the newly introduced queries on the adaptive partition over the initial partition. This experiment shows that the system can successfully adapt the partitioning with changes in the workload. At regular intervals, the system takes a snapshot of the current query workload and adapts the partitioning, which improves the workload runtime performance.

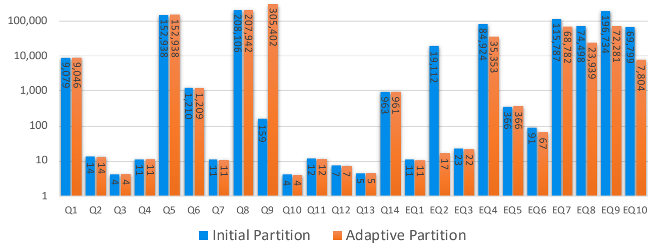


Figure 7. LUBM's 24 queries runtime in milliseconds

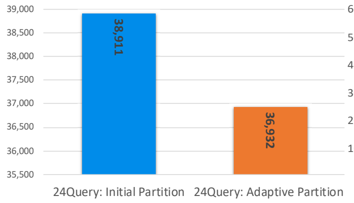


Figure 8. LUBM all 24-query average runtime in milliseconds

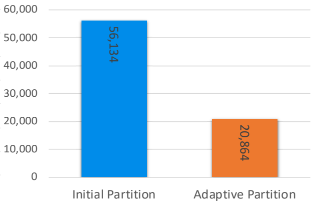


Figure 9. LUBM 10 new queries average runtime in milliseconds

Experiment 2: This experiment examined the effects of the changes in the relative frequency of queries in the workload executed on the initial partition as compared to the adaptive partitioning and so the workload query frequency distribution

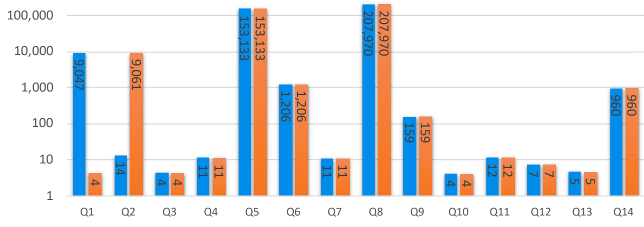


Figure 10. LUBM all queries average runtime of Initial vs. Adaptive partition in milliseconds

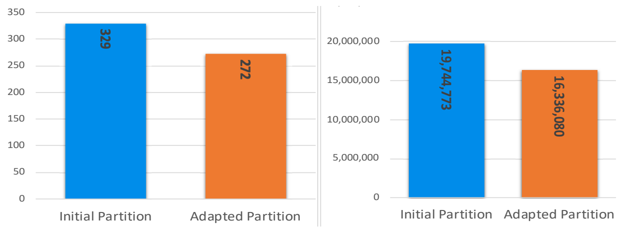


Figure 11. LUBM all query average runtime when frequency of Query1 is 50% of total workload a) Total runtime in minutes. b) Total runtime in milliseconds.

was altered. For example, if Q1 in LUBM is executed more frequently than the other 13 queries. The workload frequency share of query Q1 was increased to 50% of the whole workload. Figure 10 shows the changes in the runtime of queries Q1 and Q2. Queries 1 and 2 shares the same features. Our system swaps the queries based on score. This swapping reduces the distributed joins of Q1 but increases the distributed joins in the less frequently executed Q2, while maintaining the average runtime for the workload with evenly distributed queries. However, when the workload frequency is biased towards Q1, Figure 11 shows the improvement in the average workload performance by comparing the average runtime of the initial partition with biased workload frequency and adaptive partitioning with the biased workload frequency. The figure shows an improvement of approximately 17% of the adaptive partitioning over the initial partition, when the workload frequency is biased towards Q1.

The experiment shows that, when a query has a higher frequency than others, the performance of the adaptive partition against the initial partition is improved. Consequently, the system is adaptive to the changes in the workload. Again, at regular intervals, e.g., daily or after a set number of queries, the system takes a snapshot of the current query workload and query frequencies and, if needed, adapts the partitioning, which improves the average performance of the workload.

VI. Conclusion And Future Work

In this paper, a system is proposed which is a distributed knowledge graph query processing system that adaptively partitions the graph according to changing workload. It aims to reduce the number of distributed joins during query execution that eventually leads to a reduced run-time for the queries achieving better performance. The system is adaptive with the new workload and the system learns the workload regularly and modifies the partition, which eventually improves the partition's overall runtime performance. Our experiments show the runtime comparison of workload aware initial partition versus adaptive partition. The results depict a significant increase in the performance of the queries. There is no need for replication of the data while optimizing the runtime of the workload queries.

In the future, a study of an evolving knowledge graph in terms of its schema and instances should be undertaken. Also, it will be interesting to examine how the adaptive partitioning handles the evolving datasets along with the evolving workload queries.

References

[1] RDF Working Group. "Rdf - semantic web standards." https://www.w3.org/RDF/ accessed July 1, 2021.

[2] World Wide Web Consortium. "Rdfs - semantic web standards." https://www.w3.org/2001/sw/wiki/RDFS accessed July 1, 2021.

[3] World Wide Web Consortium. "Owl - semantic web standards." https://www.w3.org/OWL/ accessed July 1, 2021.

[4] World Wide Web Consortium. "Ontologies - w3c." https://www.w3.org/standards/semanticweb/ontology accessed July 1, 2021.

[5] World Wide Web Consortium. "Sparql 1.1 federated query." https://www.w3.org/TR/sparql11-federated-query/ accessed July 1, 2021.

[6] M. R. Garey, D. S. Johnson, and L. Stockmeyer, "Some simplified NP-complete problems," in *Proceedings of the sixth annual ACM symposium on Theory of computing*, 1974: ACM, pp. 47-63

[7] W. Donath and A. Hoffman, "Algorithms for partitioning of graphs and computer logic based on eigenvectors of connections matrices," *IBM Technical Disclosure Bulletin,* vol. 15, no. 3, pp. 938-944, 1972.

[8] J. R. Gilbert, G. L. Miller, and S.-H. Teng, "Geometric mesh partitioning: Implementation and experiments," *SIAM Journal on Scientific Computing,* vol. 19, no. 6, pp. 2091-2110, 1998.

[9] S. T. Barnard and H. D. Simon, "Fast multilevel implementation of recursive spectral bisection for partitioning unstructured problems," *Concurrency: Practice and experience,* vol. 6, no. 2, pp. 101-117, 1994.

[10] B. Hendrickson and R. Leland, "A multi-level algorithm for partitioning graphs,", SC 95, no. 28, pp. 1-14, 1995.

[11] G. Karypis and V. Kumar, "A fast and high quality multilevel scheme for partitioning irregular graphs," *SIAM Journal on scientific Computing,* vol. 20, no. 1, pp. 359-392, 1998.

[12] M. Hammoud, D. A. Rabbou, R. Nouri, S.-M.-R. Beheshti, and S. Sakr, "DREAM: distributed RDF engine with adaptive query planner and minimal communication," *Proceedings of the VLDB Endowment,* vol. 8, no. 6, pp. 654-665, 2015.

[13] K. Hose and R. Schenkel, "WARP: Workload-aware replication and partitioning for RDF," in *Data Engineering Workshops (ICDEW), 2013 IEEE 29th International Conference on*, 2013: IEEE, pp. 1-6.

[14] L. Galárraga, K. Hose, and R. Schenkel, "Partout: a distributed engine for efficient RDF processing," in *Proceedings of the 23rd International Conference on World Wide Web*, 2014, pp. 267-268.

[15] R. Harbi, et al. "Accelerating SPARQL queries by exploiting hash-based locality and adaptive partitioning," *The VLDB Journal,* vol. 25, no. 3, pp. 355-380, 2016.

[16] X. Guo, H. Gao, and Z. Zou, "WISE: Workload-Aware Partitioning for RDF Systems," *Big Data Research,* vol. 22, p. 100161, 2020.

[17] T. Neumann and G. Weikum, "RDF-3X: a RISC-style engine for RDF," *Proceedings of the VLDB Endowment,* vol. 1, no. 1, pp. 647-659, 2008.

[18] OpenLink Software Documentation Team. "Openlink Virtuoso." https://virtuoso.openlinksw.com/ accessed July 1, 2021.

[19] "Neo4j." http://www.neo4j.org accessed July 1, 2021.

[20] A. Lucene, "Apache Lucene-Overview," *Internet:* http://lucene. apache. org/java/docs/[Jan. 15, 2009], 2010.

[21] A. Priyadarshi and K. J. Kochut, "WawPart: Workload-Aware Partitioning of Knowledge Graphs," Cham, 2021: Springer International Publishing, in Advances and Trends in Artificial Intelligence. Artificial Intelligence Practices, pp. 383-395.

[22] Y. Guo, Z. Pan, and J. Heflin, "LUBM: A benchmark for OWL knowledge base systems," *Journal of Web Semantics,* vol. 3, no. 2-3, pp. 158-182, 2005.

[23] "Redland rdf libraries." http://librdf.org/ accessed July 1, 2021.

[24] B. McBride, "Jena: A semantic web toolkit," *IEEE Internet computing,* vol. 6, no. 6, pp. 55-59, 2002.

[25] C. Basca and A. Bernstein, "x-Avalanche: Optimisation Techniques for Large Scale Federated SPARQL Query Processing.", 2009.

Towards an Architecture for Policy-Aware Decentral Dataset Exchange

Sebastian Neumaier
St. Pölten University of Applied Sciences
St. Pölten, Austria
orcid.org/0000-0002-9804-4882
email: sebastian.neumaier@fhstp.ac.at

Giray Havur
Siemens AG Österreich
Vienna, Austria
orcid.org/0000-0002-6898-6166
email: giray.havur@siemens.com

Tassilo Pellegrini
St. Pölten University of Applied Sciences
St. Pölten, Austria
orcid.org/0000-0002-0795-0661
email: tassilo.pellegrini@fhstp.ac.at

***Abstract*—In the production of digital artefacts, components, such as software libraries, datasets, data streams, and content items are typically provided and used under various policies, such as licenses, terms of trade, or disclaimers. Ensuring policy compliance is a mandatory requirement for legally secure commercialization. However, manual clearance of rights is time-consuming, costly, and error-prone, especially when multiple stakeholders and contractual dependencies are involved. In this *position paper* we present an architecture for a trusted exchange in a shared data ecosystem. This includes the modelling of transparent, interoperable, and customizable data sharing policies; methods for collection and monitoring of metadata against the respective policies; and the automated validation and compliance checking of the modelled policies in a secure and trusted environment.**

***Keywords*—multi-lateral data sharing, policy-aware systems, policy languages**

I. Introduction

New data-sharing practices stimulated by phenomena like open data, open innovation, and crowdsourcing initiatives as well as the increasing interconnectivity of services, sensors, and (cyber physical) systems have nurtured an environment, in which the effective handling of policies has become key to legally secure innovation, productivity and value creation. Herein, policies shall be understood as a documented set of guidelines for ensuring the accountable management and intended usage of information. Policy-compliant data sharing becomes especially challenging when multiple stakeholders are involved. From the *user's perspective*, general problems associated with policy compliance are: (1) a massive information overload and high efforts/costs in acquiring and understanding the service provider's policy; (2) a lack of interoperability between policies due to device, application and service dependent frameworks; (3) a loss of transparency and control over data; and (4) a loss of trust into the data provider. From the *data provider's perspective*, problems associated with policy management are: (5) high efforts in ensuring legal compliance and accountability as conforming with regulations; (6) missed opportunities to use data usage preferences for service and business model innovation; and (7) missed opportunities to use the user's data sensitivity for service improvements and customer relationship management.

To tackle the problems (1-7), we aim to develop a decentralized, trustable policy negotiation framework which enables transparent, flexible and legally compliant creation and processing of data usage policies in a service ecosystem.

In Section II, we argue for the necessity of various policy types to facilitate data exchange. In Section III, we identify key challenges of policy-aware data exchange. In Section IV, we introduce three policy types (cf. Section IV-A) processed by our envisioned architecture model (cf. Section IV-B). In Section V, we provide the related work. In Section VI, we conclude with an outlook on the next research steps.

II. Policy Representation and Policy-Types

Rights Expression Languages (RELs) are a subset of Digital Rights Management technologies that are used to explicate machine-readable policies for the purpose of automated Digital Asset Management. Recent research conducted on the genealogy of RELs indicates that since 1989 more than 60 RELs have been developed from which just a small fraction is constantly maintained [1]. Among these, the most prominent RELs used to represent policies are the MPEG-21 Rights Expression Language [2], the W3C Open Digital Rights Language (ODRL) [3] and the Creative Commons Rights Expression Language (ccREL) [4]. Chong et al. [5] distinguish six policy types that appear in the context of asset management: 1) revenue policies, 2) provision policies, 3) operational policies, 4) contract policies, 5) copyright policies, and 6) security policies. While general-purpose RELs, such as MPEG-21 or ODRL support all of these policies but come with limitations concerning semantic expressivity, complementary special-purpose RELs allow to express more complex policies [6].

Enabling automated policy-based data exchange requires at least three preconditions: (i) policies, such as dataset usage licenses should be available *trust-based*; (ii) policy validation should be achieved through *proactive monitoring, control and access mechanisms* [7][8]; and (iii) reactive checks should be applied to prevent policy violations [7][8] i.e., by applying dataset watermarking techniques [9]. We can conclude that automated policy clearance requires various policies types and compliance mechanisms to specify the conditions under which digital assets are being utilized and exploited, especially when multiple stakeholders are involved in the commercialization strategy [10][11].

III. Challenges

Challenge 1 – Policies for external data exchange in scalable, multilateral settings: The first challenge we identified

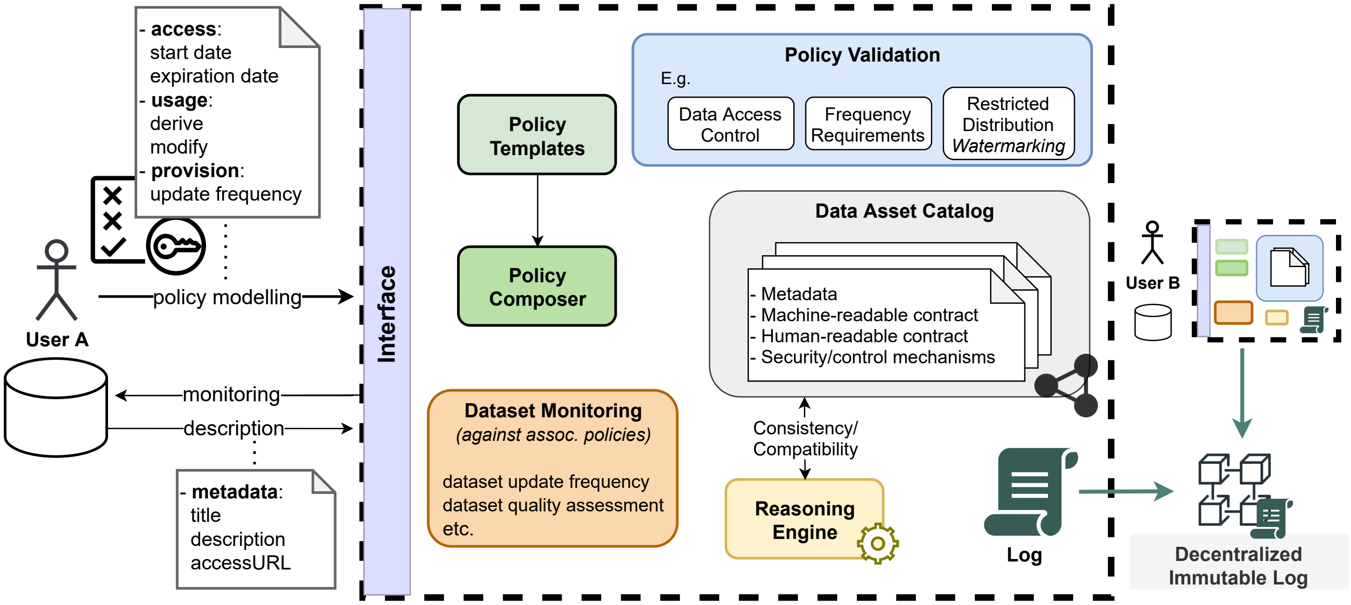


Figure 1. Architecture model of the components and interactions.

is the need for extensible machine-readable but also verbalisable/understandable policies that allow both automated contracting and compliance checking approved by legal experts. This requires auditable processes for policy modelling, adaption and modification. In particular, the process of policy modelling gets increasingly complex when more than two parties are involved: many data contracting and policy reasoning frameworks so far have focused on bilateral contracts only.

Challenge 2 – Develop and extend reasoning routines to support policy creation and ensure policy conformance: A set of formalised and modelled policies can be translated into rules derived from their machine-readable representations (e.g., RDF). These rules (often conditionally) permit or prohibit the execution of an action on certain subjects and may affect other rules, e.g., that govern the execution of the same action on the other subject(s). Accordingly, a declarative (logic-programming-style) reasoning mechanism is required to infer conformance of a created policy and test the compliance with defined terms and conditions.

Challenge 3 – Metadata catalogues for data exchange under specified policies: Current data catalogues so far only organise basic descriptive metadata, i.e., they allow a listing of datasets, provide metadata (in standard vocabularies) and offer search functionalities over the metadata; however, they do not integrate any policy management. The challenge is to incorporate machine-readable policies and contracts in current data catalogues.

Challenge 4 – Automated policy checking and service-level validation: An essential requirement for data users is a guaranteed high quality and reliability of data sources. Quality control and policy management within a data catalogue governed by well defined and modelled machine-readable policies would allow to automate the control and checking of these agreements and policies. The challenge that we identify is the use of monitoring information, such as quality measurements and collected metadata in policies.

Challenge 5 – Towards a framework for decentral data exchange: Current data sharing platforms have mainly centralised and monolithic architectures and potentially build complex environments to serve datasets. These platforms need efficient and scalable management of policies and data access to manage data exchange between multiple partners under several policies and agreements. However, to ensure the synchronisation of the relevant information between the stakeholders (e.g., policies and monitoring results), the architecture model needs to consider a decentral "logging" component.

IV. Solution Approach

Herein, we present our envisioned policy-aware dataset exchange platform (depicted in Figure 1). It processes three policy types, which we derived from the above-stated challenges.

A. Policy Types

In the following, we identify and discuss three different policy types: *(i) usage policies* that regulate distribution and modification of the resource; *(ii) provision policies*, such as a service-level agreement where the provider supplies data, compliant with a specific schema and defined quality metrics (e.g., availability and up-to-dateness); *(iii) access policies* applied to the data by the dataset provider, such as restricted access based on time constraints, version, anonymisation, or subsetting of data.

(i) Usage policies – agreements wrt. permissions, prohibitions and obligations:

Usage policies typically state *trust-based aspects*, as the transmission of data always implies some loss of control over the resource. Any further modification and distribution are possible without the knowledge of the publisher, and it is open for research what is actually (technically/contractually) enforceable in this respect. The example given below depicts a usage policy – using the ODRL vocabulary and RDF Turtle syntax – which prohibits re-distribution of a dataset:

```
<http://example.com/usagePolicy> a odrl:Agreement ;
  odrl:prohibition [
    odrl:action odrl:distribute ;
    odrl:assigner <http://ex.com/OrgaA> ;
    odrl:assignee <http://ex.com/OrgaB> ;
    odrl:target <http://ex.com/doc1> ] .
```

There is recent research on watermarking [9] and fingerprinting [12] of digital resources, which allows a reactive checking of the stated usage policies.

(ii) Provision policies – guaranteed Quality-of-Service / Quality-of-Data: High quality of data – and equally important, metadata – is a crucial requirement for successful data publishing and data sharing via platforms. Provision policies, such as data quality agreements, can be modelled by using (and potentially extending) standard vocabularies. To support an automated validation of provision policies the data-sharing platform needs quality control based on monitoring and quality assessments of the data sources. The following example of a provision policy contains an obligation clause which requires daily updates to the dataset (expressed by using the “odrl:modify” property):

```
<http://example.com/provisionPolicy> a odrl:Agreement ;
  odrl:obligation [
    odrl:action [
      rdf:value odrl:modify ;
      odrl:refinement [
        odrl:leftOperand odrl:elapsedTime ;
        odrl:operator odrl:lt ;
        odrl:rightOperand "P1D" ;
        odrl:unit xsd:duration
                      ]
                ] ;
    odrl:assigner <http://ex.com/OrgaC> ;
    odrl:assignee <http://ex.com/OrgaA> ;
    odrl:target <http://ex.com/doc1> ] .
```

In a real-world setting, such provision policies need additional provenance information, such as a validity period and applicable region.

(iii) Access policies – restricted and monitored access control: In a conditional data sharing scenario, the data provider needs to explicate the access and authorisation conditions. Defining a set of access policies allow the automation of such authorisation and access requirements. Example access policies are time-restricted data access, subsetting or aggregation of data, anonymisation of attributes, etc. Here we give an example of an access policy which permits read-access for a restricted time period:

```
<http://example.com/accessPolicy> a odrl:Agreement ;
  odrl:permission [
    odrl:assigner <http://ex.com/OrgaA> ;
    odrl:assignee <http://ex.com/OrgaD> ;
    odrl:action odrl:read ;
    odrl:constraint [
      odrl:leftOperand odrl:dateTime ;
      odrl:operator odrl:lt ;
      odrl:rightOperand "2022-01-01"^^xsd:date
                    ] ;
    odrl:target <http://example.com/document1> ] .
```

B. Platform Architecture

Figure 1 displays *Data Owner* (User A, at the left of the figure) – potentially also a data user – who interacts with the system in three ways: first, the owner brings in metadata descriptions of the datasets, second, allows monitoring of the datasets, and third, describes the policies under which the dataset is entered into the framework, e.g., restricted access by a start and expiration date, modification policies, and guaranteed update frequency of the resource. The *Policy Composer* and *Policy Templates* components support modelling and ingestion of new policies.

To process the policies (i.e., to check the consistency and compatibility of new entries), there is a *Reasoning Engine* component required, supporting logical reasoning operations. The *Dataset Monitoring* component collects information, such as quality assessments and monitoring results. The central component of the architecture depicted in Figure 1 is the catalogue: it holds the descriptions of the resources, the machine-readable policies and agreements, and the associated control and validation mechanisms that are applied.

Eventually, if Data Consumer (User B, at the right of Figure 1) wants to access a dataset, there is a *Policy Validation* layer which tests and validates the defined policies. For instance, the layer consists of a control mechanism that restricts access based on the defined constraints. To ensure the synchronisation of the relevant information in the *Data Asset Catalog* between the stakeholders (e.g., policies and monitoring results), the architecture includes a shared log component, which synchronises with a *decentralised immutable ledger*.

V. Related Work

There have been several initiatives and approaches to enable efficient and new use of data for small and medium sized companies, to generate new products and services in recent years. Data Markets try to solve these needs: the goal is to enable the distribution and transfer of data – raw, processed, anonymised, etc. – and therefore support a business model based on the exchange of data. A prominent example is the Data Market Austria (DMA) [13] that devised a national-level Data-Services Ecosystem supported by algorithms, tools, and methods for data analytics along the data value chain, and providing data curation, discovery and preservation services through the use of cloud-based approaches. However, in DMA, standard – *non-machine-processable* – licenses for data use and re-use can be defined when datasets are added to the system; and if data providers provide data that is licensed by third parties, they are responsible for disclosing and specifying the licensing terms. Our architecture aims at vastly reducing the tedious contracting efforts.

A survey by Kirrane et al. on existing access control models and policy languages can be found in [10]; a very recent overview of existing policy languages and vocabularies in the context of data protection and GDPR in [14] (under review).

Regarding license management, proof of concepts combining software and data licenses were provided by the Ontology Engineering Group [15] of the University of Madrid and the

IPTC working group on RightsML [16]. Both approaches are still in an experimental phase and lack a sufficient level of usability and legal validation to be suitable for commercial purposes. Villata and Gandon [17] and Governatori et al. [18] describe the formalization of a license composition tool for derivative works. They also provide a demo called Licentia [19] that exemplifies the practical value of such a service. The pitfall of their approach is that license compatibility can just be checked against a bundle of selected permissions, obligations and prohibitions and not against a selection of two or more other licenses containing these or other conditions. Additionally, their compatibility check assumes a reciprocal relationship between licenses instead of a directed relationship as given under real-world circumstances.

In prior work, we developed a framework for automated compatibility checks of these licenses: the DALICC software framework [20] supports the automated license clearance of rights issues in the creation of derivative digital assets (e.g., datasets, software, images, videos, etc.). However, extending these to customized usage policies, such as the examples given above, and provide an automated clearance of these, is still an open research question. The proposed architectures extends DALICC in three main points: (i) it provides a domain-specific licence contract management environment specialized for data sharing among multiple parties, (ii) it focuses on permanence and enforceability of contracts via a distributed trusted environment and an immutable log and (iii) aims at the validation of service-level policies, such as the checking of data quality agreements.

VI. Conclusion and Future Work

In this position paper, we have proposed an architecture that allows stakeholders (users, service providers and third parties) to define customised, machine-processable policies for data exchange that supports automated clearance of usage restrictions, automated validation of data provision and quality agreements, and enforcement and control of data restriction requirements.

Future work will be dedicated to developing methods to validate provision policies to *enforce access restrictions*, and to validate usage policies (e.g., based on digital fingerprinting [12]). Eventually, the results will lead to a platform that allows defining usage, access and provision policies for their resources, to make the resources available to others in decentral organised instances, and to check for potentially conflicting policies and validate the compliance if available ones.

References

[1] T. Pellegrini *et al.*, “A genealogy and classification of rights expression languages–preliminary results,” in *Data Protection/LegalTech-Proceedings of the 21st International Legal Informatics Symposium IRIS*, 2018, pp. 243–250.

[2] *MPEG-21*, https://mpeg.chiariglione.org/standards/mpeg-21, [Online; accessed 19-August-2021].

[3] *ODRL Information Model 2.2*, https://www.w3.org/TR/odrl-model/, [Online; accessed 19-August-2021].

[4] *ccREL: The Creative Commons Rights Expression Language*, https://www.w3.org/Submission/ccREL/, [Online; accessed 19-August-2021].

[5] C. Chong *et al.*, “LicenseScript: A logical language for digital rights management,” *Annales des Télécommunications*, vol. 61, pp. 284–331, Apr. 2006. DOI: 10.1007/BF03219910.

[6] J. Prados, E. Rodriguez, and J. Delgado, “Interoperability between different rights expression languages and protection mechanisms,” in *International Conference on Automated Production of Cross Media Content for Multi-Channel Distribution*, IEEE Computer Society, 2005, pp. 145–152. DOI: 10.1109/AXMEDIS.2005.28.

[7] B. Agreiter *et al.*, “A technical architecture for enforcing usage control requirements in service-oriented architectures,” in *Proceedings of the 4th ACM Workshop On Secure Web Services*, ACM, 2007, pp. 18–25. DOI: 10.1145/1314418.1314422.

[8] S. Pearson and M. C. Mont, “Sticky policies: An approach for managing privacy across multiple parties,” *Computer*, vol. 44, no. 9, pp. 60–68, 2011. DOI: 10.1109/MC.2011.225.

[9] A. S. Panah, R. G. van Schyndel, T. K. Sellis, and E. Bertino, “On the properties of non-media digital watermarking: A review of state of the art techniques,” *IEEE Access*, vol. 4, pp. 2670–2704, 2016. DOI: 10.1109/ACCESS.2016.2570812.

[10] S. Kirrane, A. Mileo, and S. Decker, “Access control and the resource description framework: A survey,” *Semantic Web*, vol. 8, no. 2, pp. 311–352, 2017. DOI: 10.3233/SW-160236.

[11] M. Hilty, A. Pretschner, D. A. Basin, C. Schaefer, and T. Walter, “A policy language for distributed usage control,” in *12th European Symposium On Research In Computer Security*, ser. Lecture Notes in Computer Science, vol. 4734, Springer, 2007, pp. 531–546. DOI: 10.1007/978-3-540-74835-9_35.

[12] P. Kieseberg, S. Schrittwieser, M. Mulazzani, I. Echizen, and E. R. Weippl, “An algorithm for collusion-resistant anonymization and fingerprinting of sensitive microdata,” *Electron. Mark.*, vol. 24, no. 2, pp. 113–124, 2014. DOI: 10.1007/s12525-014-0154-x.

[13] B.-P. Ivanschitz, T. J. Lampoltshammer, V. Mireles, A. Revenko, S. Schlarb, and L. Thurnay, “A semantic catalogue for the Data Market Austria,” in *Proceedings of the Posters and Demos Track of the 14th International Conference on Semantic Systems - SEMANTiCS2018*, 2018.

[14] B. Esteves and V. Rodrıguez-Doncel, “Analysis of ontologies and policy languages to represent information flows in GDPR,” 2021, *Under review*. [Online]. Available: http://www.semantic-web-journal.net/system/files/swj1280.pdf.

[15] *ODRLAPI: A Java API to manipulate ODRL2.0 RDF expressions*, http://oeg-upm.github.io/odrlapi/, [Online; accessed 19-August-2021].

[16] *RightsML - Implementation Examples*, http://dev.iptc.org/RightsML-Implementation-Examples, [Online; accessed 19-August-2021].

[17] S. Villata and F. Gandon, “Licenses compatibility and composition in the web of data,” in *Third International Workshop on Consuming Linked Data (COLD2012)*, 2012.

[18] G. Guido, L. Ho-Pun, R. Antonino, V. Serena, and G. Fabien, “Heuristics for licenses composition,” *Frontiers in Artificial Intelligence and Applications*, vol. 259, pp. 77–86, 2013. DOI: 10.3233/978-1-61499-359-9-77.

[19] *Licentia*, http://licentia.inria.fr/, [Online; accessed 19-August-2021].

[20] T. Pellegrini *et al.*, “DALICC: A license management framework for digital assets,” *Internationales Rechtsinformatik Symposion (IRIS)*, vol. 10, 2019.

A Reference Ontology for Collision Avoidance Systems and Accountability

David Martín-Lammerding
Department of Stats. Comput. Sci. Math
Public University of Navarre (UPNA)
Pamplona, Spain
email:david.martin@unavarra.es

José Javier Astrain
Department of Stats. Comput. Sci. Math
Public University of Navarre (UPNA)
Pamplona, Spain
email:josej.astrain@unavarra.es

Alberto Córdoba
Department of Stats. Comput. Sci. Math
Public University of Navarre (UPNA)
Pamplona, Spain
email:alberto.cordoba@unavarra.es

***Abstract*—Unmanned Aerial Systems (UASs) are deployed in Intelligence, Surveillance, and Reconnaissance (ISR) applications with less cost and more flexibility rather than manned aircraft. An increasing number of UAS missions requires an improvement of their safety capabilities by equipping them with Collision Avoidance Systems (CASs). It is recognized that the use of small UAS at lower altitudes is now a driving force of economic development, but a safety risk when its number increases. UAS generates heterogeneous data from multiple sources, like the Flight Control Unit (FCU), the Global Navigation Satellite System (GNSS), a radio receiver, an onboard-camera, etc. Each CAS implementation receives this data and processes it to avoid collisions. There are many CAS implementations, but each one has a specific design and data repository structure. There is a lack of standards that simplify their development and homologation. This paper presents a reference knowledge model for any CAS for UAS implemented as a novel application ontology called Dronetology-cas. It transforms data to knowledge by combining heterogeneous telemetry and onboard-sensor data using linked-data and an ontology for semantic interoperability across heterogeneous UAS traffic management systems. Dronetology-cas provides a unified semantic representation within an ontology-based triplet store designed to run in a low cost computer. Its semantic model provides advantages, such as interoperability between systems, machine-processable data and the ability to infer new knowledge. It is implemented using semantic web standards, which contribute to simplify an operational safety audit.**

***Keywords*—Semantic reasoning, ontology, UAS, knowledge, conflicts, anti-collision, sensor, embedded, air traffic.**

I. Introduction

The use of Unmanned Aerial Systems (UASs) improves efficiency in logistics applications, infrastructure inspection, emergency situations, etc. and avoids pilot risk. However, their flights are limited to certain areas of the airspace to avoid encountering other aircraft. Air traffic management must evolve to allow the introduction of large numbers of mass-market UAS. Each UAS must be equipped with new safety systems, like Collision Avoidance Systems (CASs).

CASs are developed to detect airplanes in airspace, to discover potential collision hazards and to perform maneuvers to avoid collisions. An increased use of UAS requires autonomous capabilities for safety purposes. However, UAS autonomy involves ensuring accountability. The accountability principle requires UAS operators to take responsibility for what their UAS do in a mission and how they comply with traffic management authorities. UAS operators must have appropriate records to be able to demonstrate their compliance. The accountability of an UAS flight must be ensured because any incident or accident must be able to be investigated by surveyors or authorities. In the worst case, a collision may occur, which must be investigated to determine the cause and to improve CAS.

CAS design factors are showed in Figure 1. Multiple CAS's typologies can be obtained combining different design factors. CAS should not depend on pilots or communications with centralized systems, as any delay in making a decision increases the risk of collision.

Each CAS studied has its own internal data implementation with specific structures. So, data generated by CAS have proprietary formats that are not easily inter-operable. To solve this issue we are integrating and structuring data from CAS using ontologies, linked data, and semantic integration techniques.

Ontologies [2] are formal and explicit specifications of certain domains and are shared between large groups of stakeholders. These properties make ontologies ideal for machine processing and enabling inter-operation. The use of standards for data encoding, structuring and description simplifies an audit task.

In this paper, we present a novel ontology, denoted Dronetology-cas [3], that is suitable for structuring any data generated in a CAS. Dronetology-cas includes a Knowledge Base (KB) which consists of triplets of data collected and inferred knowledge during the UAS mission.

The rest of the paper is structured as follows. Section II presents the state of the art of CAS and accountability systems, Section III defines the problem statement and Section IV describes our contribution. The ontology design is presented in Section V. Section VI formulates ontology Competency Questions (CQs) and Section VII summarizes experimental simulations results. Section VIII presents the conclusions and references end the paper.

II. Related Work

The use of UAS for data gathering is becoming increasingly widespread thanks to high quality and cost-effective sensors. Therefore, the Semantic Sensor Network (SSN) [4] ontology

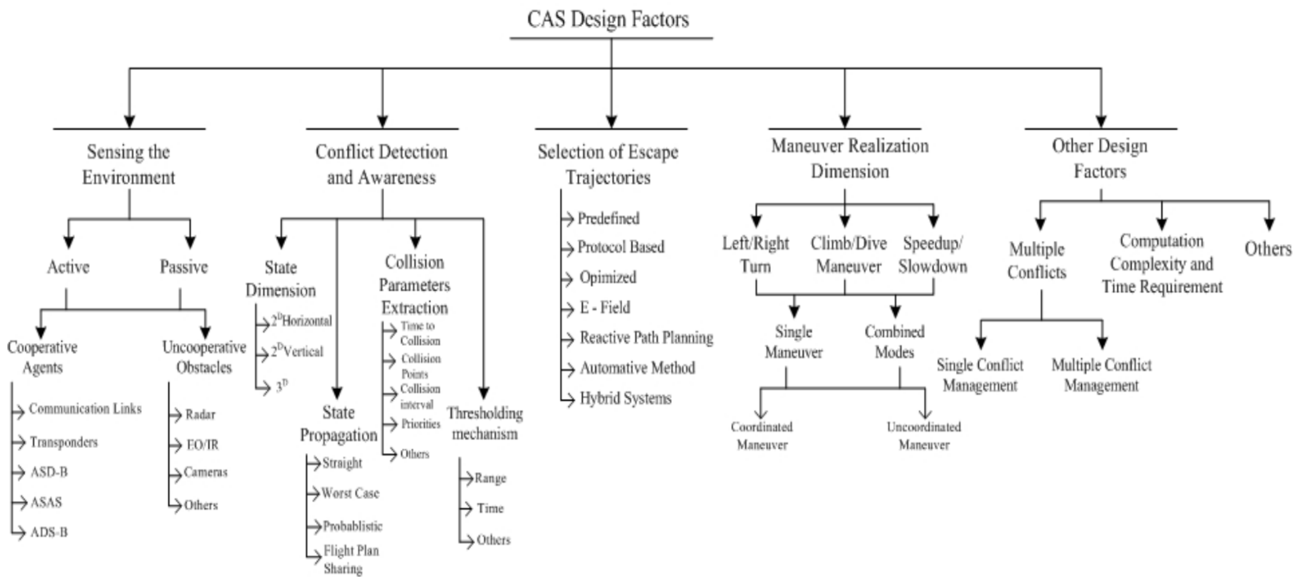


Figure 1. CAS design factors. Taken from [1].

can be used to model UAS as sensors. However, it has the limitation of not having concepts to model the UAS mission.

[5] applies semantic technologies to air traffic in order to unify heterogeneous data from multiple sources. The ontology implementation is performed centralized. However, our proposal is a decentralized ontology implemented in each UAS to serve as a knowledge base for any CAS.

[6] presents a *light-weight* ontology for embedded systems whose design reduces concepts, complexity and query times, compared to the SSN ontology. It is intended for the sensor domain and therefore has limitations for modeling an UAS.

ACAS-Xu [7] and Daidalus [8] are two reference CAS implementations whose source code is available for review. Each CAS requires a specific configuration for the same scenario. Given the same scenario, their output formats are different as shown in [9]. A limitation of both CAS is that they do not share a common conceptual model.

The accountability of an UAS flight must be ensured because any incident or accident should be able to be investigated by surveyors or authorities. There are systems similar to black boxes for UAS, [10]–[12]. They store the UAS's route and the CAS's status. However, the decision-making process prior to a maneuver is complex and its recording is not provided in these systems.

III. Problem statement

The data required by a CAS depends on how the main design factors presented in Figure 1 are combined. The main concepts of CAS used in the design of Dronetology-cas are described below.

A conflict between two UAS occurs when minimum separation, defined as the protection distance d_p, is lost. Figure 2 shows a conflict between *local UAS* and *remote UAS*. A loss of separation does not always predict a future collision, but it is a key safety indicator. A CAS deployed in an UAS is aimed at maintaining a minimum safe separation between UASs. Once a conflict is detected, a CAS diverts the UAS to a new safe path. The number of simultaneous conflicts are denoted as N_C. Time to collision t_{tc} is the time required to collide two UAS if an UAS continues at their current speed and on the same path. Lower t_{tc} values correspond to higher risk of collision. It is used to prioritize conflicts. Very Low Level airspace (VLL) is the space below 500 ft. above ground level. It is the part of the airspace intended for new UAS applications and it will concentrate most UAS conflicts.

CASs are based on different technologies that collect data from the surroundings using sensors and/or collaborative elements based on radio receivers/transmitters. UAS can deploy collaborative elements and non-collaborative sensors. A collaborative element receives and transmits position and bearing data with any other element within its coverage. Automatic Dependent Surveillance – Broadcast (ADS-B) [13] is one of the standards for collaborative systems, based on sharing location information obtained from the Global Positioning System (GPS). A non-collaborative sensor detects obstacles without any external system. There are multiple technologies applied to non-collaborative systems, such as vision cameras [14], LIDAR [15], SONAR [16], Radar [17], etc. [18] details the main technologies applied to sensors for conflict detection.

Most CASs for UASs are distributed, so they run in an onboard computer. However, the size of the UAS limits the weight of the payload, which limits the type and power of processor that can be used. Any software component used in a distributed CAS implementation should be non-compute-intensive to ensure effective real time performance.

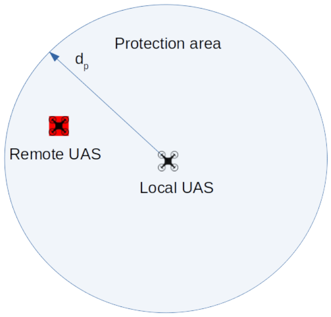


Figure 2. Conflict between local UAS and remote UAS.

IV. Contribution

Dronetology-cas is a novel ontology intended for UAS, whose domain is anti-collision knowledge management and air safety compliance. It provides knowledge-based conflict resolution capabilities. Dronetology-cas is defined as a reference model that improves communications, inter-operation and automation of some air traffic management tasks.

Dronetology-cas defines the foundations to implement a knowledge-based CAS. It provides two modes of integration with a CAS: *repository mode* or *knowledge mode*. The *repository mode* stores data in a semantic structure, so others systems can understand and use it. The *knowledge mode* provides additional knowledge using reasoning from current data.

Dronetology-cas offers key advantages over other repository or log storage implementations. This is achieved by the web semantic technologies used in its implementation. Dronetolog-cas key features are performance, modifiability, ease of maintenance, built-in inference capabilities and potential for reuse.

V. Dronetology-cas: The application ontology

Dronetology-cas is an application ontology derived from the domain ontology Dronetology [19]. The domain of Dronetology is UASs. Dronetology-cas formal specification is based on the design factors shown in Figure 1.

A. Dronetology: The domain ontology

The purpose of Dronetology is to describe concepts that define the components of any UAS, the missions it performs and the environment that surrounds it. Its main applications are the management of bill of materials, the improvement of flight efficiency and autonomous decision making. Dronetology imports external ontologies to avoid repeating concepts from other domains. Another advantage of importing widespread ontologies is that there are data repositories (sources in Resource Description Framework (RDF) format) designed with these models that can be integrated into Dronetology.

B. Dronetology-cas description

We derive the Dronetology-cas application ontology from the Dronetology domain ontology. The domain of Dronetology-cas is CAS for UAS. The aim of Dronetology-cas is to be the KB of any CAS implementation. Therefore, Dronetology-cas is generic and extensible. Dronetology-cas simplifies auditing the CAS decision making process. Its design allows queries in the KB history to retrieve the CAS status at different times. The KB stores the temporal evolution of conflicts with other UAS and the status of the CAS. Dronetology-cas consists on a KB where knowledge is stored. It also has an inference engine that generates new knowledge by applying semantic rules to the KB. The rules are expressed in SPARQL Protocol and RDF Query Language (SPARQL) statements [20], [21]. Rules inference a conflict's attribute, an evasive trajectory method, a maneuver attribute, etc. Knowledge is obtained from data recollected from sensor systems and collaborative elements. Data sources are sensors, the Flight Control Unit (FCU) and the Global Navigation Satellite System (GNSS). Inference improves the CAS decisions thanks to knowledge derived from the data.

A CAS runs in a loop with a operation frequency. This is modeled in Dronetology-cas with the concept of *Iteration*. Dronetology-cas stores CAS status, UAS telemetry and conflicts for each *Iteration* to audit the system. Data collected from sensors are also related to the *Iteration* to provide a complete picture of the environment and the CAS. Dronetology-cas simplifies the integration of data from different sources. It integrates data from any sensor system by defining generic classes, which are not directly dependent on the technology and the implementation. These classes are *NoCollaborativeData* and *CollaborativeData* and both extend *InputData*.

A CAS estimates future positions of conflicts to obtain a maneuver that avoids a collision. In *knowledge mode*, the CAS asks the KB for knowledge to perform a specific function, such as selecting the method to estimate the position. Figure 4 shows several methods to estimate the position. Dronetology-cas stores data about conflicts and also interrelates this data to discover new connections and knowledge. This knowledge can be used to improve the prediction method of the conflict's location. For example, if the conflict has been detected only through a vision camera, the uncertainty about the heading of the conflict is higher, so the most appropriate method of estimating may be the *Worst Case* method. However, if the conflict has been detected by a collaborative element, the heading is known and there is less uncertainty. In this case, the *Straight Projection* method is the most appropriate.

When the CAS makes a maneuver to avoid a collision, Dronetology-cas stores every UAS position and groups them with a individual of class *Maneuver*. Thus, Dronetology-cas replaces multiple specific-maneuvers concepts, like *left-turn*, with a set of positions, which allows any combination of trajectories, altitudes and speeds. Full trajectory prediction made by the CAS are not stored in Dronetology-cas. The dynamics of 3D conflicts are modeled in Dronetology-cas as different positions at different times.

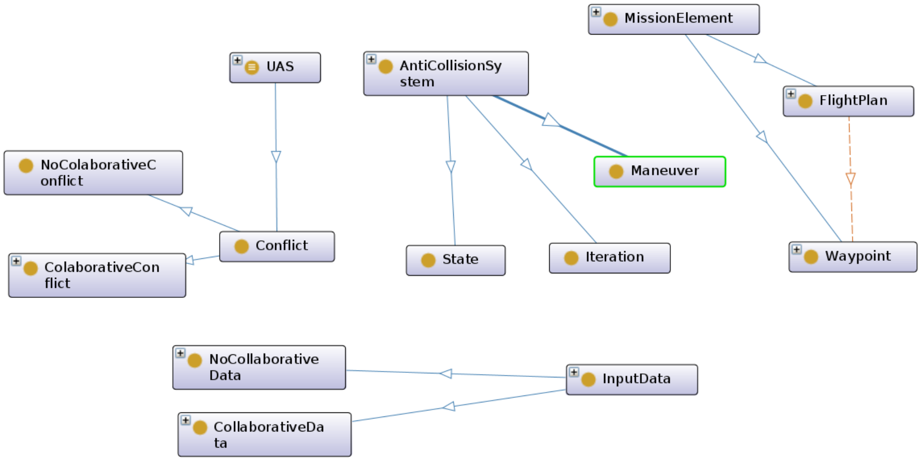


Figure 3. Dronetology-cas main classes

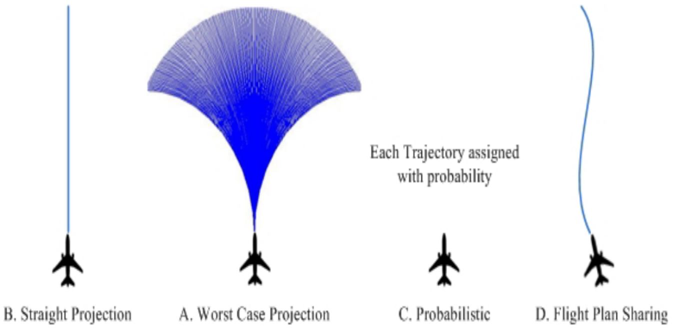
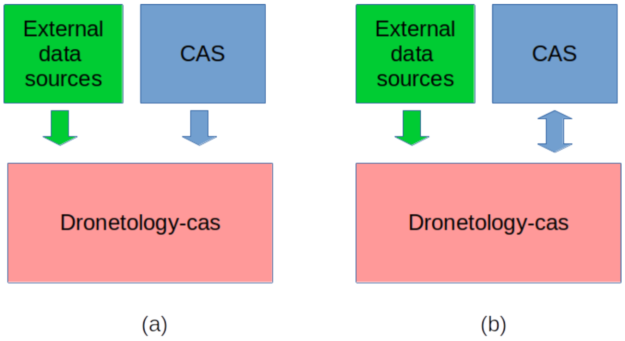


Figure 4. Methods used for projecting current encounter's information. Taken from [1].



(a)

(b)

Figure 5. Dronetology-cas integration alternatives: (a) repository mode (b) knowledge mode

C. Dronetology-cas integration with a CAS

Any CAS can integrate Dronetology-cas in two ways: *repository mode* or *knowledge mode*. The Dronetology-cas repository mode requires that the CAS implements some data source specific code to translate UAS data from its original source format to Dronetology-cas ontology triplets. In this way, Dronetology-cas stores any conflict's data obtained from the onboard sensors into the KB.

The *repository mode* integration implies that the CAS inserts data as triplets into the KB. The CAS stores data in the KB, but it does not query it. Data stored are available for any audit process. External data sources, like Ground Surveillance Radars (GSRs), can add additional conflicts to the KB not detected by onboard sensors, although they depend on network connectivity during the UAS flight. The *knowledge mode* extends the features of the *repository mode*. It adds implicit knowledge inference and reasoning capabilities to some CAS functions, such as conflict detection or new path selection. In the *knowledge mode* integration, the CAS inserts data in the KB and also performs queries. These queries enhance CAS functions, such as classifying conflicts, prioritizing conflicts, selecting a trajectory calculation techniques according to the type of conflicts, etc. The CAS queries the KB for a specific result depending on its decision making implementation requirements. Figure 5 shows the relation between the CAS and Dronetology-cas for each integration mode.

Dronetology-cas is defined using the Web Ontology Language (OWL) language [22]. The main languages used to develop CAS (C, C++, Python) have implementations to process RDF triples [23] and ontologies in OWL format.

D. Dronetology-cas design

Dronetology-cas is an application ontology whose concepts are taken from CAS. The CAS design factors shown in Figure

1 are also considered. It is accessible at [3].

In order to integrate Dronetology-cas with a CAS, Dronetology-cas's concepts are defined with a high level of abstraction. The first design factor is the type of onboard sensors. They are classified into collaborative and non-collaborative sensors. Dronetology-cas models every onboard sensor as an abstract data source, instead of defining detailed concepts related to *sensors*.

Another design aspect to be considered of CAS is the method used to detect conflicts. The main differences between them are the data needed and the criteria followed to classify a nearby UAS as a conflict. Dronetology-cas integrated in *repository mode* stores the CAS and the conflict status. In *knowledge mode*, it improves the CAS capabilities for conflict-classification aggregating data from multiple sensors or linking the conflict detection sensor with the conflict estimated path. When a conflict‘s attribute are not available, like speed, it can be inferred from the conflict past locations. The inference of conflict attributes also improves the CAS decisions.

Finally, the method to calculate an evasive trajectory and the associated maneuver are a CAS's design choice. Dronetology-cas integrated in *repository mode* stores a maneuver as a sequence of UAS locations. In *knowledge mode*, the CAS could query Dronetology-cas to select a maneuver calculation method using knowledge about the conflict.

Dronetology-cas has been designed considering the computational limitations of onboard systems. Thus, memory usage has been reduced by limiting the number of classes in the model and avoiding importing auxiliary ontologies.

The main classes of Dronetology-cas are *UAS*, *MissionElement*, *InputData*, *AntiCollisionSystem* and *Conflict*. Figure 3 shows the main Dronetology-cas classes.

TABLE I
DRONETOLOGY-CAS COMPETENCY QUESTIONS

CQ$_1$	How many conflicts are detected?
CQ$_2$	Which UAS has the highest priority among the UAS in conflict?
CQ$_3$	Which conflict has the shortest time to collision?
CQ$_4$	Has the number of conflicts increased or decreased?
CQ$_5$	How has been detected the conflict with a given UAS?
CQ$_6$	How long it has taken to resolve a conflict?
CQ$_7$	Has the distance flown been increased with respect to the flight plan?
CQ$_8$	In which locations have there been conflicts?
CQ$_9$	Where and when was the collision?
CQ$_{10}$	How many UAS were in conflict before the collision?
CQ$_{11}$	What UAS has it collided with?
CQ$_{12}$	What maneuver was the UAS performing before the collision?

The class *UAS* describes unmanned aircrafts including the communication systems and the ground base. The class *Conflict* is a subclass of *UAS* so in our model only UAS can be conflicts. *MissionElement* is a class that enclose all the elements of a mission. The classes *Waypoint* and *FlightPlan* derive from *MissionElement*.

The class *InputData* represents any data collected from a sensor (non-collaborative), from a collaborative element (radio receiver), from the GNSS or from the FCU. The concepts *NoColaborativeData* and *ColaborativeData* are derived from *InputData* to identify a conflict and its source type. The property *drone:detect* is an object property that relates individuals of *NoColaborativeData* or *ColaborativeData* with individuals of *Conflict*.

Some classes in Dronetology-cas have geographic data defined as datatype properties. The latitude and longitude are relative to the World Geodetic System 1984 (WGS84) coordinate system. The altitude is relative to Mean Sea-Level (MSL). To improve interoperability, the Conflict class uses *geo:wktLiteral* datatype with a WGS 84 geodetic latitude-longitude. This allows Dronetology-cas to implement a geospatial web service that could be reused and recombined to fulfill a user query.

The class *AntiCollisionSystem* groups elements of any CAS. The classes *State*, *Maneuver*, *NextIterationLocation* and *Iteration* are derived from it. The state of the CAS are represented as instances of the class *State* with an attribute that codifies the state and a timestamp. The class *Iteration* relates all the knowledge stored in the KB at an instant of time.

The class *Maneuver* defines a set of locations of the UAS when the CAS is active. CAS calculates multiple location alternatives of the UAS to avoid the collision and stores them in the KB as instances of the class *NextIterationLocalUASLocation*. It also selects one locations that best resolve the conflict from the previous set of locations. An individual of class *Maneuver* groups every individual of class *NextIterationLocalUASLocation* through an object-property. Every iteration, the CAS sends to the FCU the individual of class *NextIterationLocalUASLocation*.

VI. COMPETENCY QUESTIONS

We define a set of CQs that specify what knowledge has to be entailed in Dronetology-cas. This questions has been used to validate Dronetology-cas. Some CQs are suitable for an UAS mission audit process. Others can assist the CAS in a decision making process, when Dronetology-cas is integrated in *knowledge mode*. Table I shows a list of some CQs considered. There are CQs that are intended to find out how the conflict has been resolved, e.g., CQ$_6$, CQ$_7$ and CQ$_8$. Some CQs help to find out what happened and how when a collision happens, e.g. CQ$_9$, CQs CQ$_{10}$, CQs CQ$_{11}$ and CQ$_{12}$.

In a *knowledge mode* integration, the CAS uses the results of some CQs to make decisions. An example is the CQ *What type of conflict is X?*. With this knowledge about the conflict, the CAS selects the most appropriate way of calculating the future position of the conflict. Other CQs are intended for a security audit of the CAS. An example is the CQ to check when a collision occurred.

VII. PERFORMANCE EVALUATION

The performance of Dronetology-cas is analyzed executing CQs translated to SPARQL in a low cost computer, Pi3 [24]. We simulate a system CAS with a software component developed in Java 8 that inserts triplets with conflicts data in the KB. Response time and memory footprint are measured with different number of triplets stored in the KB. Memory footprint has been measured using the Java 8 API. The number

TABLE II
RESPONSE TIME (IN MILLISECONDS) AND MEMORY FOOTPRINT (IN KILOBYTES) OF REPOSITORY MODE AND KNOWLEDGE MODE IN A PI3.

	Repository mode								*Knowledge mode*							
	CQ_1				CQ_3				CQ_5				CQ_6			
No	Response time		Memory footprint		Response time		Memory footprint		Response time		Memory footprint		Response time		Memory footprint	
triplets	mean	sdev	mean	sdev	mean	sdev	mean	sdev	mean	sdev	mean	sdev	mean	sdev	mean	sdev
100	18.95	4.13	5025.28	1418.13	26.68	7.83	5101.67	1420.78	19.98	5.91	5104.45	1420.83	18.21	7.45	5100.81	1421.06
250	23.54	2.26	5200.98	1308.69	23.38	2.17	5241.25	1308.82	24.12	4.18	5211.77	1308.21	24.03	2.23	5233.40	1305.41
500	38.10	2.97	5441.26	1327.53	38.14	3.10	5441.51	1322.02	52.37	18.03	5377.53	1327.33	39.05	3.56	5491.02	1322.82
1000	74.42	17.78	5986.61	1332.65	67.47	3.10	5983.91	1336.00	68.49	3.18	6061.40	1331.63	68.74	3.10	5969.27	1319.01
2500	165.62	36.42	6997.64	1342.78	171.69	46.86	3362.77	1727.86	173.87	55.31	3875.86	1768.38	160.60	6.81	6972.95	1315.36
5000	320.98	64.36	6004.34	1718.44	320.30	63.49	5391.63	1587.92	355.39	100.29	5309.72	1510.33	324.63	65.44	5935.41	1724.54
10000	662.00	162.01	9545.00	1744.46	662.24	164.92	8440.54	2154.39	654.65	151.11	8560.70	2140.84	670.81	162.15	9595.25	1752.67

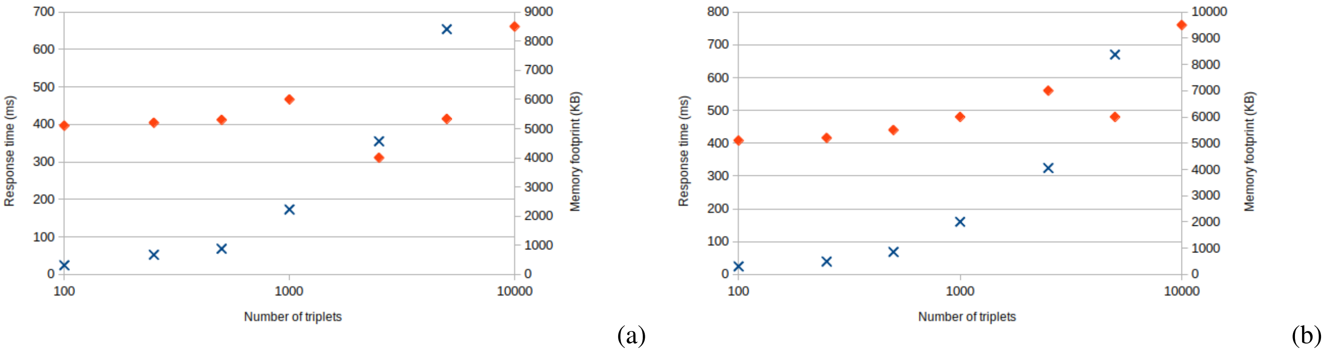


Figure 6. Response time (x) and memory footprint (♦) for *knowledge mode* for CQ_5(a) and CQ_6(b).

of triples with conflicts and CAS data in the KB grows as the UAS flies. Therefore, the flight duration determines the number of triples stored in the KB. In our tests we have simulated up to 10000 triples corresponding to 15 minutes of flight by inserting an average of 10 triples per second.

To measure response times and memory footprint, the most generic CQs have been selected as they are the most likely to be used in any integration mode. CQ_1 and CQ_3 are necessary for any auditing process to review conflicts and their status. CQ_5 and CQ_6 provide knowledge that the CAS can use to modify its response to conflicts. 100 repetitions of each case were performed to calculate the mean and standard deviation. The results obtained from the response times and memory footprint are shown in Table II. CQs considered are translated to SPARQL, available at [25].

The response time affects the CAS depending on the integration type chosen. In *repository mode*, there are no strict response time requirements as it is not required a real-time operation. However, in *knowledge mode*, the response time delays the CAS decisions. For our purpose, a suitable response time should allow to take a decision with the most recent data, before new data is available, that is, the response time should be below the refreshing rate of incoming data. Each sensor system has its refreshing rate ranging from 1 Hz of ADS-B until 20 Hz of a vision camera [26]. The response times of CQ_5 and CQ_6 obtained comply with the previous criteria as long as the number of triplets are below approximately 1000 triplets.

Figure 6 shows that Dronetology-cas response time increases when the number of triples increases. Memory consumption grows as the UAS flies as well. That is, the duration of the UAS flight increases the response time. The worst response time is at the end of a flight. This result is due to our limited implementation of the software components that instantiates and queries the KB. An option to scale up is to have two instances of Dronetology-cas model, each with a different purpose, one instance for the *repository mode* and the other for the *knowledge mode*. The instance for the *repository mode* should store all triplets, but the instance for the *knowledge mode* should keep only triplets needed for the inference process.

VIII. CONCLUSION

In this paper, we described the Dronetology-cas ontology as a value-added component for any CAS. Dronetology-cas integration modes facilitate its application in any CAS. A production-ready implementation of Dronetology-cas should take into a account the performance results and the integration mode required to balance response time and memory consumption.

As the need for UAS safety compliance is expected to increase, reference CAS implementations promoted by government agencies, like Daidalus [8], are candidates to implement advanced audit systems like the proposed in this paper.

Future work will be focused on the implementation of a CAS for UAS using Dronetology-cas and the integration of Dronetology-cas with an existing CAS. Another line of work is to create a *dataset* with semantic mission data to be used for research of UAS air traffic. Further developments of this work have the potential to achieve an ontology standard for autonomous UAS.

References

[1] B. Albaker and N. Rahim, “A survey of collision avoidance approaches for unmanned aerial vehicles,” in *2009 international conference for technical postgraduates (TECHPOS)*, IEEE, 2009, pp. 1–7.

[2] J. Davies, D. Fensel, and F. Van Harmelen, *Towards the semantic web: ontology-driven knowledge management*. John Wiley & Sons, 2003.

[3] D. Martín-Lammerding. (2021). Dronetology-cas, the anti-collision ontology, https://dronetology.net/dronetology-cas, [Online]. Available: https://dronetology.net/dronetology-cas (visited on 02/02/2021).

[4] W. W. Group. (2021). SSN, Semantic Sensor Network Ontology, https://www.w3.org/tr/vocab-ssn/, [Online]. Available: https://www.w3.org/TR/vocab-ssn/ (visited on 02/02/2021).

[5] R. M. Keller, S. Ranjan, M. Y. Wei, and M. M. Eshow, “Semantic representation and scale-up of integrated air traffic management data,” in *Proceedings of the International Workshop on Semantic Big Data*, 2016, pp. 1–6.

[6] H. Rahman and M. I. Hussain, “A light-weight dynamic ontology for internet of things using machine learning technique,” *ICT Express*, 2020.

[7] M. P. Owen, A. Panken, R. Moss, L. Alvarez, and C. Leeper, “Acas xu: Integrated collision avoidance and detect and avoid capability for uas,” in *2019 IEEE/AIAA 38th Digital Avionics Systems Conference (DASC)*, IEEE, 2019, pp. 1–10.

[8] C. Muñoz, A. Narkawicz, G. Hagen, J. Upchurch, A. Dutle, M. Consiglio, and J. Chamberlain, “Daidalus: Detect and avoid alerting logic for unmanned systems,” in *2015 IEEE/AIAA 34th Digital Avionics Systems Conference (DASC)*, IEEE, 2015, 5A1–1.

[9] J. T. Davies and M. G. Wu, “Comparative analysis of acas-xu and daidalus detect-and-avoid systems,” *National Aeronautics and Space Administration NASA Ames Research Center; Moffett Field CA United States Technical Report NASA/TM-2018-219773 ARC-E-DAA-TN50499*, 2018.

[10] Redcat Holdings. (2021). Drone Box, https://www.redcatholdings.com/drone-box, [Online]. Available: https://www.redcatholdings.com/drone-box (visited on 02/02/2021).

[11] Tl-Elektronic. (2021). Black box, https://www.tl-elektronic.com/, [Online]. Available: https://www.tl-elektronic.com/index.php?page=uav&p_id=40&lang=en (visited on 02/02/2021).

[12] UAV Navigation. (2021). Black Box https://www.uavnavigation.com/, [Online]. Available: https://www.uavnavigation.com/sites/default/files/docs/2021-03/UAV%20Navigation%20FDR01%20Brochure.pdf (visited on 02/02/2021).

[13] C. Rekkas and M. Rees, “Towards ads-b implementation in europe,” in *2008 Tyrrhenian International Workshop on Digital Communications-Enhanced Surveillance of Aircraft and Vehicles*, IEEE, 2008, pp. 1–4.

[14] D. Zuehlke, N. Prabhakar, M. Clark, T. Henderson, and R. J. Prazenica, “Vision-based object detection and proportional navigation for uas collision avoidance,” in *AIAA Scitech 2019 Forum*, 2019, p. 0960.

[15] U. Papa, G. Ariante, and G. Del Core, “Uas aided landing and obstacle detection through lidar-sonar data,” in *2018 5th IEEE International Workshop on Metrology for AeroSpace (MetroAeroSpace)*, IEEE, 2018, pp. 478–483.

[16] U. Papa, “Sonar sensor model for safe landing and obstacle detection,” in *Embedded Platforms for UAS Landing Path and Obstacle Detection*, Springer, 2018, pp. 13–28.

[17] N. Gellerman, M. Mullins, K. Foerster, and N. Kaabouch, “Integration of a radar sensor into a sense-and-avoid payload for small uas,” in *2018 IEEE Aerospace Conference*, IEEE, 2018, pp. 1–9.

[18] A. Muraru, “A critical analysis of sense and avoid technologies for modern uavs,” *Advances in Mechanical Engineering ISSN: 2160-0619*, vol. 2, Mar. 2012. DOI: 10.5729/ame.vol2.issue1.23.

[19] D. Martín-Lammerding. (2021). Dronetology, the UAS Ontology, https://dronetology.net/dronetology, [Online]. Available: https://dronetology.net/dronetology (visited on 02/02/2021).

[20] Web Working Group. (2021). SPARQL Query Language for RDF, https://www.w3.org/2001/sw/wiki/sparql, [Online]. Available: https://www.w3.org/TR/sparql11-query/ (visited on 02/02/2021).

[21] ——, (2021). Spin Working Group, Rules for SPARQL, https://www.w3.org/submission/spin-sparql/, [Online]. Available: https://www.w3.org/Submission/spin-sparql/ (visited on 02/02/2021).

[22] ——, (2021). Web Ontology Language (OWL), https://www.w3.org/owl/, [Online]. Available: https://www.w3.org/owl/ (visited on 02/02/2021).

[23] ——, (2021). Resource Description Framework (RDF), https://www.w3.org/2001/sw/wiki/rdf, [Online]. Available: https://www.w3.org/rdf/ (visited on 02/02/2021).

[24] Raspberry Pi Foundation. (2021). Raspberry Pi 3, https://www.raspberrypi.org/, [Online]. Available: https://www.raspberrypi.org/ (visited on 02/02/2021).

[25] D. Martín-Lammerding. (2021). Competency questions in sparql, https://dronetology.net/sim/competency-questions.zip, [Online]. Available: https://dronetology.net/sim/competency-questions.zip (visited on 08/02/2021).

[26] S. Graham, J. De Luca, W.-z. Chen, J. Kay, M. Deschenes, N. Weingarten, V. Raska, and X. Lee, “Multiple intruder autonomous avoidance flight test,” in *Infotech@ Aerospace 2011*, 2011, p. 1420.

Temporal Information and Event Markup Language

TIE-ML Markup Process and Schema Version 1.0

Damir Cavar, Billy Dickson, Ali Aljubailan, Soyoung Kim
Department of Linguistics
Indiana University
Bloomington, USA
email: {dcavar, dicksonb, aaaljuba, sk135}@iu.edu

Abstract—**Temporal Information and Event Markup Language (TIE-ML) is a markup strategy and annotation schema to improve the productivity and accuracy of temporal and event related annotation of corpora to facilitate machine learning based model training. For the annotation of events, temporal sequencing, and durations, it is significantly simpler by providing an extremely reduced tag set for just temporal relations and event enumeration. In comparison to other standards, as for example the Time Markup Language (TimeML), it is much easier to use by dropping sophisticated formalisms, theoretical concepts, and annotation approaches. Annotations of corpora using TimeML can be mapped to TIE-ML with a loss, and TIE-ML annotations can be fully mapped to TimeML with certain under-specification.**

Keywords-TIE-ML; Events; Time; Corpora; Machine Learning.

I. INTRODUCTION

Natural languages use various means to express events and place them in time. Tense, aspect, mood, and modality make up the foundations of this process, and each language utilizes a certain combination of these tools to indicate temporal information.

Tense places an event on the timeline and is most often generated through either verbal inflection, or the use of auxiliaries. Comrie [3] groups tenses into two categories: absolute and relative. Absolute tenses refer to tenses that orient an event with respect to the present (e.g., simple present, simple past, simple future) whereas relative tenses are those that orient an event with respect to a different point in time (e.g., pluperfect, future perfect).

The internal temporal structure of an event can be broken into two categories, grammatical aspect and lexical aspect. Grammatical aspect refers to the marking of aspect through inflection or auxiliaries (e.g., English progressive *-ing*) whereas lexical aspect refers to the inherent temporal properties of a predicate. The original four categories for grouping predicates by their lexical aspectual properties as introduced by Vendler [27] are statives, activities, accomplishments, and achievements, each of which housing differing combinations of telic, dynamic, and durative properties of predicates.

Modality as defined in Portner [13] is “the linguistic phenomenon whereby grammar allows one to say things about, or on the basis of, situations which need not be real” (p. 8). Mood falls under this umbrella and indicates how a proposition expresses modality. Mood can be subdivided into two categories, verbal mood (indicatives and subjunctives) and sentence mood (declaratives, interrogatives, and imperatives). Modal auxiliaries like *may*, *might*, *can*, *should*, etc. express epistemic, deontic, and circumstantial modality.

Understanding these expressions and how they interact and complement each other is crucial toward developing a system for capturing time and event information in natural language. Developing corpora and data sets is essential for quantitative studies of distributional properties of temporal and event logic phenomena and expressions. It also allows us to develop machine learning based automatic annotation and processing of event sequencing and temporal aspect as for example duration.

A. Event Sequencing

Sequencing of events and sub-events is an essential task that we address here. A general observation is that the presentation sequence of events in natural language discourse does not necessarily correspond to the temporal sequence that these events occur in. While in example (1) the presentation of sub-events corresponds to the underlying temporal sequence, in example (2) the presentation sequence does not match with the temporal sequencing.

```
(1) Wash the veggies, chop them, and fry them.
     1                2              3
(2) Before you fry the veggies, wash, and chop them.
               3                1          2
```

Observations suggest that sub-events occupy the same time slot or happen simultaneously, illustrated in (3). To address this aspect, these events or eventualities need to be indexed individually, with an independent time slot index. If integers could represent indices for events and sequence indices for time slots, then the sequencing would be generated with two tiers: the event index tier E, and the time slot tier T. In (3), it is successfully indicated that the event “John lived in Paris” and the event “Susan studied in Berlin” are overlapping in time slot 1.

```
(3) John lived in Paris while Susan studied in Berlin.
    E:   1                          2
    T:   1                          1
```

The reference to time slots in example (3) does not encode the information whether there is a total overlap, subsumption,

ISBN: 978-1-61208-888-4

or partial overlap, but only the information that there is a time slot/span such that the two events 1 and 2 overlap during this time span 1. To simplify the annotation, we could think of events or eventualities expressed by predicates to be tuples indicating the event identifier and a corresponding time slot, e.g., in (3) it is the predicate “live” assigned (1,1), and the predicate “studied” assigned (2,1).

For independent reasons we restrict our exposition here to this simplified perspective of enumerating events or eventualities, referencing time slots or spans using integer identifiers.

B. *Tense*

Reichenbach [18] introduced a theory of tense that presents three time variables that include event time, reference time, and speech time. Event time refers to the time of an event. Reference time is the point of reference along the time axis that an event is related to. Speech time refers to the time of utterance of an event. In absolute tenses, such as English simple past, present and future, the reference time and event time are simultaneous, however in relative tenses such as the pluperfect and future perfect, the reference time and event time are separated. To represent this ordering relationship encoding an event’s specific tense, the three variables E (event time), R (reference time), S (speech time) are ordered on the time axis. For example, the pluperfect would be represented by the event time E preceding the reference time R, and both preceding the speech time S, represented in the sequence E-R-S. Present tense could be described through the simultaneity or overlap of E, R, and S expressed as E,R,S in Reichenbach’s model.

The relative sequence of R and S in Reichenbach’s model reflects the tense categories *present* (S and R overlap), *past* (R precedes S), and *future* (S precedes R). The event is *anterior* to some reference time R, if E precedes R. It is *posterior*, if R precedes E, and it is *simple* if R and E overlap. This system derives tense category labels like Posterior Past or Posterior Future, that do not have correspondence in traditional tense labels like Future Perfect or Pluperfect.

Using the Reichenbach schema to encode tense of simple predicates allows us to correlate the encoding of tense with the sequencing of predicates. In addition, it is essential to encode scope relations between different predicates and tenses in complex clauses when considering assertions about events, whether they are claimed to be facts and true, or hypotheses about some future unfolding of events. Consider the examples in (4) and (5). While the predicate in (4) asserts that Apple merged with Alphabet, the predicate in (5) does not claim to be factual.

(4) *Apple merged with Alphabet.*
(5) *Apple will merge with Alphabet.*

If E precedes S, the event could be asserted to be factual, while S preceding E implies that the event is a hypothetical projection into the future.

The situation changes if a predicate with a tense as in (4) is in the scope of another predicate and specific tense. While example (6) is equivalent to (4) with respect to the embedded predicate, the matrix predicate and tense in (7) render the embedded predicate hypothetical.

(6) *Reuters reported that Apple merged with Alphabet.*
(7) *Reuters will report that Apple merged with Alphabet.*

It is essential to encode the tense of the individual predicates for the correct interpretation of the status of assertions. Syntactic scope relations between the predicates are necessary as well for the correct interpretation of embedded predicates.

The rest of the paper is structured as follows. In Section II, we present the overview of related work in the domain of temporal and event annotations of natural language corpora. In Section III, we describe the core properties of TIE-ML. In Section IV, we present our conclusions and the plan for future research related to TIE-ML. In Section V, we provide information about availability and open access to the TIE-ML standard and sample corpora.

II. Related Work

The demand for data sets and corpora with semantic annotation has grown over the last decades. One of the key types of information for Information Extraction (IE) systems to store, retrieve, and analyze is time and Temporal Expressions (TE).

The need to analyze and interpret event mentions in text sources or spoken language dialogues drive the necessity for deep understanding and models of event logic and temporal reasoning. Building temporally aware software systems can be significantly beneficial for Natural Language Processing (NLP) based information extraction applications, e.g., Question Answering Systems (QAS), Text Mining (TM) techniques, Document Summarization (DS) systems, Medical Documentation Systems (MDS), and other NLP applications such as event characterization and tracking and visualizing events on the timeline.

Accordingly, the automatic recognition to temporal elements in digital texts has recently turned out to be a vast area of research in the field of NLP; several activities and various initiatives were made attempting to develop representations for temporal information annotation in order to obtain more efficient information extraction.

This can account for the proliferation in research in this area, disseminated in theoretical bases and practical applications. The vast majority of work on annotating TEs, however, has been developed by three language technology evaluation programs: Message Understanding Conference (MUC) [34], the succeeding Translingual Information Detection, Extraction and Summarization (TIDES) [35], and The Automatic Content Extraction (ACE) [4]. All of these programs were held under the Defense Advanced Research Project Agency (DARPA) conference, sponsored by the U.S. government. In the remainder of this section, we briefly review the main existing schemes for annotating temporal information.

A. *MUCs*

MUCs are a series of language technology evaluation conferences "in which participating IE systems are rigorously evaluated" [1].

As previously mentioned, a considerable amount of work on IE has been cultivated by MUCs [21][29]. It is no surprise, then, that efforts on devising temporal annotation schemes appear to have begun within the DARPA MUCs [12][29]. More specifically, temporal information was defined as a targeted type for IE starting from MUC-6 and continuing in MUC-7. In MUC-6, one of the required subtasks for annotating Named Entity Recognition (NER) was to identify *absolute* time expressions in documents.

In MUC-7, the requirement of this subtask was extended to include *relative* time expressions [22]. It is worth noting that the distinction between absolute and relative time expressions was first made within MUCs guidelines [26]. However, none of the mentioned subtasks required the consideration of placing events in time or mapping temporal relations between events [22].

During MUC-7, the participating systems were required to extract the TIMEX elements, i.e., the TEs textual span, without being required to describe the inward semantic characteristics of the successfully deciphered TEs. The requirements then were to merely extract the coarse-grained type classification of each recognized TE. That is, the participating systems were required to determine whether a TIMEX represents a DATE or a TIME feature. Examples on such annotation requirement include the following:

- `"twelve o'clock noon"`
  `<TIMEX TYPE="TIME">twelve o'clock noon</TIMEX>`
- `"5 p.m. EST"`
  `<TIMEX TYPE="TIME">5 p.m. EST</TIMEX>`
- `"third quarter of 1991"`
  `<TIMEX TYPE="DATE">third quarter of 1991</TIMEX>`

B. *TIDES*

TIDES was a DARPA-sponsored research program on IE, specifying guidelines that were concerned with the specification and standardization of more detailed semantic representations of TEs than TIMEX had applied in the previous DARPA programs (see MUC 1998). However, TIDES guidelines maintain similarity to MUC's guidelines in handling TEs as separate targets for annotation and/or extraction. Additionally, those standards of TIDES were not aimed at the "hopelessly ambitious goal" of representing the entire varieties of TI expressed in natural language [7].

In the latest version of TIDES [7], *markable* expressions to be annotated must represent an appropriate *lexical trigger*. Also, a trigger must be able to be orientable on a timeline or at least be orientable to a relation to a time (past, present, future). Based on these determinations, lexical triggers that are reliable candidates of markable expressions are:

- nouns (*day*, *month*, *summer*, etc.)
- proper names (*Monday*, *January*, *New Year's Eve*)
- specialized time patterns (*8:00*, *12/2/00*, *1994*)
- adjectives (*recent*, *former*, *current*, *future*, *past*, *daily*, *monthly*, *biannual*, etc.)
- adverbs (*lately*, *hourly*, *daily*, *monthly*, etc.)
- noun or time adverb where adverbs that stem from an adjectival form of a trigger are also triggers. (*now*, *today*, *yesterday*, *tomorrow*, etc.)
- and numbers (*Sixties*, as in referring to the decade "the Sixties")

For temporal annotation format, TIDES developed a special SGML (Standard Generalized Markup Language) tag, i.e., TIMEX2, superseding MUC-7 TIMEX and extending its annotation. TIMEX2 offers a variety of features for more precise capturing of the actual meaning of a TE. TIMEX2 therefore is claimed to be most easily applicable to languages other than English, although all the cases defined and discussed in [5] are related to English.

C. *ACE*

The Automatic Content Extraction (ACE) is a program created by The National Institute of Standards and Technology (NIST) that is driven by and addresses issues identical to MUCs. ACE is a series of evaluation activities that require developing human language technologies capable of understanding natural language, thereby being automatically capable of detecting and extracting the key types of information existed in digital multimedia resources. One of these key types is events with associated entities and their temporal anchoring, which were added to ACE IE efforts in 2004 [4].

By collaboration, the Linguistic Data Consortium (LDC) at the University of Pennsylvania developed annotation guidelines, annotated corpora, and produced other linguistic resources to support the ACE program for research on IE. One of the primary ACE annotation tasks was Event Detection and Characterization (EDC).

In EDC, annotators identified and characterized five types of events in which EDT entities participated. Targeted types included Interaction, Movement, Transfer, Creation and Destruction events. Annotators tagged the textual mention or anchor for each event and categorized them by type and subtype. They further identified event arguments (agent, object, source, and target) and attributes (temporal, locative, instrument, purpose, etc.) according to a type-specific template. In later phases of ACE, annotators identified additional event types as well as characterized relations between events (see [33]).

D. STAG

Sheffield Temporal Annotation Guidelines (STAG), analogous to the development of TIDES, is a TI annotation scheme that was created by Andrea Setzer for her PhD thesis [22][24]; Setzer's work is said to be the first annotation scheme ever to allow for all elements of TI [25]. In her framework, Setzer's objective was to annotate events, TEs, and their temporal relations. This framework is based on four primitive types: *events, states, times* and *relations*.

Event in STAG is intuitively defined as something that happens, must be anchorable in time map, and can be ongoing or conceptually instantaneous [23]. Based on this simple definition, in her scheme Setzer categorizes events into coarse-grained sets, including *occurrence, reporting, perception, attitude*, and *aspectual events*.

For time, instead of viewing times as having extents (intervals), or as being punctual (having a time point), STAG simply applies the notion of *time objects*. Time objects must be replaceable on a timeline and are either fictional or real [24]. Following the broad conventions of MUC's approach in labeling time, time objects in STAG are classified into two types, DATES and TIMES, where times are broadly described as being larger or smaller than a day.

Regarding temporal relations, STAG defines relations between events and other events, and events and times. The framework provided for temporal relations heavily depends on the works on temporal relations and temporal ontology conducted by Allen [2][24]. As a result, in providing a practical framework for temporal relations, the set of relations that connect events to times was reduced to merely five relations: *before, after, includes, included,* and *simultaneous*, the latter being vague to determine [28].

E. TimeML

TimeML [15][17] is a metadata standard proposed for TI annotation, and it is currently the most conventional mark-up language for annotating events and temporal relations [8][16][17].

The framework of TimeML was created based on recommendations from the Time and Event Recognition for Question Answering Systems (TERQUAS) workshop in July 2002. TERQUAS feedback was given on how to enhance temporally aware NLP question answering systems (QAS) [9][11][12].

Pustejovsky and his colleagues proposed the TimeML specifications for annotating events and their temporal anchoring by amalgamating two of the previous TI annotation schemes: TIMEX2 [5][6][7] and STAG, along with other emerging schemes such as in Katz and Arosio [12].

Dissimilarly from the previous attempt at specifying event and time, TimeML separates the representation of event and temporal expressions from the anchoring or ordering dependencies that may exist in a given text.

There are four major structures specified in TimeML [15][17]: EVENT, TIMEX3, SIGNAL, and LINK. The tag <EVENT> is a cover term for the ontological notion of "events": situations that happen or occur, either punctually or as lasting for a period of time. Events in TimeML are broadly expressed by several linguistic formations, including *verbs*, *nominalizations*, *adjectives*, *predicative clauses*, and *prepositional phrases*.

The TIMEX3 tag, which is used for marking up explicit TEs, e.g., times and dates, is based on both the TIMEX [24] and TIDES TIMEX2 tag [6].

The use of signals is another feature of TimeML that was originally borrowed from Setzer's STAG then expanded in TimeML. The tag <SIGNAL> is used to annotate function words, i.e., indicators of temporal relations, such as temporal connectives (e.g., *while*), or temporal prepositions (e.g., *during*).

The fourth tag, <LINK>, said to be a key innovation for TimeML [15], comprises three types of link tags: TLINK, SLINK, and ALINK. The main task of the <LINK> tag is to encode relations between temporal elements in a text. TimeML proposes a set of 13 relations to indicate fine-grained distinctions between TEs and/or between TEs and events. Overall, the features that distinguish TimeML from other previous schemes below are:

- Extends the TIMEX2 attributes
- Introduces Temporal Functions to allow intentionally specified expressions: *three years ago*, *last month*
- Identifies signals determining interpretation of temporal expressions
  - Temporal prepositions: *for*, *during*, *on*, *at*;
  - Temporal Connectives: *before*, *after*, *while*.
- Identifies all classes of event expressions
  - Tensed verbs: *has left*, *was captured*, *will resign*
  - Stative adjectives and other modifiers; *sunken*, *stalled*, *on board*
  - Event nominals: *merger*, *Military Operation*, *Gulf War*
- Creates dependencies between events and times:
  - Anchoring: *John left on Monday*.
  - Orderings: *The party happened after midnight*.
  - Embeddings: *John said Mary left*.

III. TIE-ML

While TimeML represents an approved, very detailed and precise annotation standard for events and temporal relations, it also introduces a high level of complexity for annotators. In our practical lab experience, the time and complexity to annotate basic data sets was prohibitively high. It required experts and well-trained linguistic annotators, and the productivity and quality control turned out to be costly. TIE-ML is a solution for a basic event sequencing corpus with Reichenbach style of tense annotation that reduces the annotation complexity and facilitates much faster output with less errors.

The TIE-ML annotation system is designed to improve the accuracy for annotators by simplifying the annotation task for time and event information. Speeding up the annotation by reducing the complexity of the effort for annotators will hopefully lead to larger data sets in shorter time, reducing costs and annotation errors.

At the same time, the goal of TIE-ML is to facilitate machine learning model development for event sequence annotation and event labeling. To experiment with automatic sequencing, very basic annotations are necessary, as for example a basic event annotation for the presentation sequence and the time sequence.

A. *Event Identification*

Events in the TIE-ML schema are individual predicates that are usually clauses. Each clause or independent predicate is given a numerical event identifier (`eventid`), shown in Figure 1, that serves both to mark relationships between events, as well as track the presentation order of events in text.

Since the temporal ordering of events does not necessarily coincide with the presentation order, tracking this information can provide insight in the intuition and motivations of an author or interlocutor for presenting events in a particular way.

```
<s> <c eventid="1">
    Danny watched the movie
    </c>
    <c eventid="2">
    and ate popcorn
    </c>. </s>
<s> <c eventid="3">
    Josh brought the pizza
    </c>. </s>
```

Figure 1. EventID

B. *Tense, Perfect, Progressive*

For each event, TIE-ML provides the possibility for the tense of the predicate, as well as the presence of perfect and progressive aspect to be explicitly annotated using the *tense* attribute, and Boolean *perfect* and *progressive* attributes as shown in Figure 2.

Progressive provides information on the internal temporal structure of the event, while tense and perfect aspect provide information on the location of the event in time and the point the event is oriented with relationship to.

```
<s> <c tense="PAST" perfect= "TRUE"
       progressive="TRUE">
    The patient had been experiencing
    stomach pain
</c>. </s>
```

Figure 2. Tense, Perfect, Progressive

As described below, Reichenbach's time variables provide a more specific annotation that provides more information than the traditional tense labels.

Some languages utilize morpho-syntactic present tense to refer to future events. In Polish, for example, the use of present tense verb forms is compatible with adverbials that indicate future tense reference, as in (8).

(8) *Jutro* *pracuję* *od* *9 do 5.*
Tomorrow work-1st-sg-present from 9 to 5
"Tomorrow I will work from 9 to 5."

The same is not possible for past tense adverbials, as in the ungrammatical example (9).

(9) **Wczoraj* *pracuję* *od* *9 do 5.*
Yesterday work-1st-sg-present from 9 to 5

The convention in TIE-ML is to encode the semantic temporal properties in such constructions, assuming that morphosyntactic and part-of-speech annotation tools will provide the lexical level annotation, indicating present tense at the lexical and syntactic level.

C. *Reichenbach Annotation Model*

In TIE-ML, Reichenbach's [18] time variables are annotated as E for "event time," R for "reference time," and S for "speech time."

In absolute tenses such as English simple past, present and future, R and E are simultaneous, however in relative tenses such as the pluperfect and future perfect, R and E are separated. To represent this ordering relationship, the three time variables are each assigned an integer value from -2 to 2. These values represent a simple relationship where a variable with a lower value occurs before those with higher values, and variables with equivalent values occur simultaneously.

For the simple past sentence in Figure 3, the event time and reference time are simultaneous in the past and given a value of -1 relative to the speech time given a value of 0.

```
<s> <c E="-1" R="-1" S="0">
    Danny watched the movie.
</c> </s>
```

Figure 3. Reichenbach Simple Past

For the pluperfect sentence in Figure 4, the event occurs in the past relative to the reference time, and the reference time occurs in the past relative to the speech time. The event time is given a value of -2, reference time a value of -1, and speech time a value of 0.

```
<s> <c E="-2" R="-1" S="0">
    Josh had watched the movie.
</c> </s>
```

Figure 4. Reichenbach Pluperfect

In addition to encoding the sequencing of E, R, and S, negative and positive numbers are assigned to the variables to indicate past, present, and future directly in the value. While relative ordering of the values is sufficient for the derivation of tense categories and traditional labels for tense in the Reichenbach model, a negative value indicates past, a positive value future, and 0 corresponds to present tense. We exploit this property in different approaches to corpus analysis and machine learning model training.

D. *Reference Time Anchor*

To be able to capture the concrete reference time for an event, we provide a designated attribute to capture concrete date or time point expressions that anchor R on the real time axis.

The reference tag as in Figure 5 marks explicitly mentioned time and dates of events in text or conversations. This value provides a concrete temporal anchor for R in the Reichenbach model and the TIE-ML schema.

```
<s> <c reference="264 BC">
    The First Punic War broke out on the
    island of Sicily in 264 BC.
</c> </s>
```

Figure 5. Reference

E. *Timeline Sequencing*

To capture the presentation time and the relative timeslot association of events, the TIE-ML schema provides a timeslot attribute representing in its value the relative ordering of events along the time axis.

In Figure 6, we observe *starting the oven* as the second presented event that is assigned a timeslot value of 1, while the first presented event *prepare the vegetables* is assigned a timeslot value of 2. The temporal connective *after* signals this shift in the temporal order of events.

```
<s> <c eventid="1" timeslot="2">
    Prepare the vegetables </c>
    <c eventid="2" timeslot="1">
    after starting the oven
</c>. </s>
```

Figure 6. Timeline Slots

IV. Conclusion and Future Work

The most important goal for TIE-ML was to define an annotation schema that facilitates quick and reliable annotation of events and temporal sequencing, and bootstraps larger corpora in shorter time. The priority was given to solving the alignment of presentation and timeline sequencing. At the same time, clause level scope effects on tense in complex sentences with multiple verbal predicates can be annotated without significant increase of annotation effort. Various other aspects of event and temporal logic have been postponed to future versions and specifications.

Important components that are not yet integrated in TIE-ML are for example duration of events, or continuity and sequencing properties of events.

Continuity or sequencing of events is displayed in the contrast between the two predicates in (10) and (11).

(10) *John was reading the book for two months.*
(11) *John was living in New York for ten years.*

While the "reading" event in (10), based on common sense, would be understood to be a sequence of discontinuous sub-events of "reading." Common sense dictates that "living" in (11) is understood to be continuous. In a future release TIE-ML might provide a simple annotation attribute to indicate continuity of events.

Duration aspects are discussed in more detail in the following sub-section.

A. *Durations*

An additional development to TIE-ML is to mark deeper internal properties of events in the form of durations. One aspect of durations to be accounted for are set relations between event reference times. Consider the following examples. (12) sets a reference time of Monday, while (13), (14), and (15) set reference times that are subsets of 'Monday'.

(12) Event 1: *The test took place on Monday.*
(13) Event 2: *In the morning the students ate breakfast.*
(14) Event 3: *In the afternoon the students arrived.*
(15) Event 4: *In the evening the test finished.*

To capture this relationship, TIE-ML could introduce a tier system whereby each tier represents different sizes of reference times. Figure 7 displays two such tiers, one consisting of days, and the other, times of day. Events would then be linked through combinations of timeslot annotations and markers denoting an event's superset.

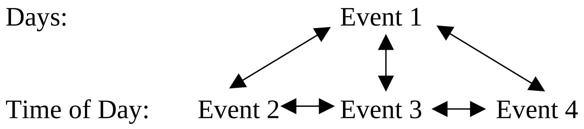


Figure 7. Duration Sets

Building further on capturing properties of durations, TIE-ML intends to incorporate an analysis of typical event duration allowing the execution of common-sense reasoning. For example, in (16), it certainly could be the case that Geoff very much does not like his vegetables, but this edge case aside, we can reason that an 'eating' event typically does not typically take 8 hours.

(16) *Geoff ate his dinner in 8 hours.*

We can also expand logicality prediction to the Reichenbach E, R, S variable values. In example (17) we observe a future tense event time in the past with respect to speech time, and a reference time in the future with respect to speech time. The simple future tense of this event calls for both reference time and speech time to have identical positive values. The inability to satisfy this requirement results in a logical incoherence.

(17) #*Yesterday I will go to the store*. E = 1, R =-1, S=0

Various related aspects of complete or partial temporal overlap of events based on duration cannot be expressed in the current version of TIE-ML. Probabilities or estimates of duration for event and time related common-sense reasoning or models of expectation are not foreseen yet, but might likely enter a future version of TIE-ML.

B. *Cross-linguistic Features*

The current iteration of TIE-ML can be most effectively applied to tensed languages such as German in Figure 8, and the Semitic languages such as Arabic in Figure 9, and Hebrew in Figure 10.

```
<s> <c eventid="1" tense="FUT"
       reference="morgen"
       E="1" R="1" S="0">
    Morgen werde ich ein Buch lesen
    <!-- Tomorrow I will read a book -->
</c>. </s>
```

Figure 8. TIE-ML German

```
<s> <c eventid="1" timeslot="2"
       tense="sp" E="1" R="1" S="2">
            נתן נפגש עם רביקה </c>
    <c eventid="2" timeslot="1"
       tense="sp" E="1" R="1" S="2">
       לאחר שסיים לעבוד אחר הצהריים
<!-- Nathan met with Rebecca after he
finished working this afternoon -->
</c>. </s>
```

Figure 9. TIE-ML Hebrew

```
<s> <c eventid="1" timeslot="2"
       tense="sp" E="1" R="1" S="2">
      التقى أحمد بصديقه </c>
    <c eventid="2" timeslot="1"
       tense="sp" E="1" R="1" S="2">
       بعد أن زار والده الثلاثاء الماضي
<!-- Ahemd met with his friend after he
visited his father last Tuesday  -->
</c>. </s>
```

Figure 10. TIE-ML Arabic

When expressing temporality in Semitic languages, ordinarily Arabic and Hebrew, there is a renewed controversy about whether tense is lacking in terms of grammatical expression, thereby making Semitic languages “aspect languages”. Yet, even if this argument is conceded, not all aspectual dimensions are fully grammaticalized in Semitics [30][31]; that is, the main aspectual dimensions are Perfective (PFV) and Imperfective (IPFV), the latter being used to capture both simple and progressive situations. This is for the progressive aspect is not morphologically encoded thus not distinguished from habitual situation [32]. Rather, it is frequently indicated by other elements such as gerunds, adjectives, and adverbs. To capture these additional features, as well as the time and event information of languages that do not mark for tense requires additional annotations. Mandarin, for example, relies on a multitude of aspectual markers, temporal adverbials, and context to orient events. Mayan has a limited set of temporal adverbials and contains no temporal connectives, relying solely on aspect, mood, and context to order events.

Additional annotations of temporal features will enable TIE-ML to be applied to a wider range of typologically diverse languages. Some of these temporal features will be related to pragmatic cues and general world knowledge applied in general deduction or induction processes, often associated with probabilities or plausible guesses of temporal relations. While there is an obvious need for such annotation levels, in particular encoding of uncertainties and ambiguities, TIE-ML does not yet provide the means for such annotations.

The main motivation for defining the initial version of TIE-ML was to facilitate the development of cross-linguistic data sets with basic event and temporal logic annotation. The annotation process is much simpler than using other annotation standards and processes that certainly are far more sophisticated and powerful. The annotators have to understand only the three Reichenbach variables and event enumeration using event IDs. The training effort for annotators is clearly reduced in TIE-ML when compared to TimeML’s rich tag and concept set. Future evaluations will provide more insight in the annotation error rate and output quantities.

Additionally, the simplified TIE-ML standard should be compatible with other annotation standards, provided core translation possibilities. Translation of TIE-ML annotations to other formats is straightforward, keeping in mind that TIE-ML provides certain levels of under-specification.

V. Availability and Open Access

Sample corpora and data sets are made available at the public GitHub repository “Temporal Information and Event Markup Language” (URL: https://github.com/dcavar/tieml). This repository also hosts conversion scripts and annotation samples for different languages.

The mapping of TIE-ML XML annotations from XML to different formats is straightforward. There are corpus samples of a proposed CoNLL format (URL: https://www.signll.org/conll/) mapping, as well as a JSON format annotation. The TIE-ML project will provide conversion scripts in the public GitHub repo for the unidirectional conversion to these formats and TimeML.

The Apache License Version 2.0 has been chosen as the appropriate license for the XML Schema, sample corpora, scripts, and documentation, facilitating commercial and non-commercial use.

References

[1] M. Albanese, Extracting and summarizing information from large data repositories. University of Naples Federico II, Italy. 2006.

[2] J. F. Allen. Towards a general theory of action and time. Artificial Intelligence, 1984, 23, pp. 123-154.

[3] B. Comrie, *Tense*. Cambridge University Press, Cambridge, 1985.

[4] G. R. Doddington et al. “The automatic content extraction (ace) program-tasks, data, and evaluation.” LREC 2004, 2, pp. 837-840.

[5] L. Ferro, I. Mani, B. Sundheim, and G. Wilson, Tides temporal annotation guidelines version 1.0.2. The MITRE Corporation, McLean-VG-USA, 2001.

[6] L. Ferro et al. “Annotating temporal information: from theory to practice,” in Proceedings of the second international conference on Human Language Technology Research, 2002, pp. 226-230.

[7] L. Ferro, L. Gerber, I. Mani, B. Sundheim and G. Wilson, TIDES: 2003 Standard for the Annotation of Temporal Expressions. MITRE Corp, McLean VA. 2003.

[8] Y. Jeong and S.-H. Myaeng, “Using WordNet hypernyms and dependency features for phrasal-level event recognition and type classification.” European Conference on Information Retrieval, 2013, pp. 267-278.

[9] Y. Karagoel, “Event ordering in Turkish texts.” Time. Middle East Technical University. 2010.

[10] G. Katz, Towards a Denotational Semantics for TimeML. In: F. Schilder, G. Katz, and J. Pustejovsky (Eds.). Annotating, Extracting and Reasoning about Time and Events: International Seminar, Dagstuhl Castle, Germany, April 10-15, 2005. Revised Papers, Springer Berlin Heidelberg, 2007, pp. 88-106.

[11] K. A. Lee, *Taxonomy of Data Categories for Temporal Objects and Relations*. 2005.

[12] I. Mani, J. Pustejovsky and B. Sundheim, “Introduction to the special issue on temporal information processing.” ACM New York, NY, USA, 2004.

[13] P. Portner, *Mood*. 2018. Oxford University Press.

[14] J. Pustejovsky and F. Busa, “A Revised Template Description for Time (v3)”, June 1995, Waltham, MA.

[15] J. Pustejovsky et al. “TimeML: Robust Specification of Event and Temporal Expressions in Text.” New Directions in Question Answering, 2003, pp. 28-34.

[16] J. Pustejovsky et al. “The TimeBank corpus.” *Proceedings of Corpus Linguistics*, 2003.

[17] J. Pustejovsky et al. “The Specification Language TimeML.” In I. Mani, J. Pustejovsky, R. Gaizauskas (eds.) *The Language of Time: A Reader*. Oxford University Press, 2005, pp. 545-558.

[18] H. Reichenbach, *Elements of Symbolic Logic*. The Free Press, New York, 1947.

[19] H. Reichenbach, *The Direction of Time*. University of California Press. 1956, reprinted as Dover 1971.

[20] H. Reichenbach, *The Tenses of Verbs*. In J. C. Meister and W. Schernus (Eds.), Time, 2011, pp. 1–12. De Gruyter.

[21] E. Riloff, “Information extraction as a stepping stone toward story understanding, Understanding language understanding: Computational models of reading,” Citeseer, 1999, pp. 435-460.

[22] A. Setzer and R. Gaizauskas, “Annotating Events and Temporal Information in Newswire Texts.” LREC, 2000, 1287-1294.

[23] A. Setzer and R. Gaizauskas, “Building a temporally annotated corpus for information extraction.” Proceedings of the Workshop on Information Extraction Meets Corpus Linguistics held in conjunction with the Second International Conference on Language Resources and Evaluation (LREC 2000), 2000.

[24] A. Setzer and R. Gaizauskas, „A pilot study on annotating temporal relations in text.” Proceedings of the ACL 2001 Workshop on Temporal and Spatial Information Processing, 2001.

[25] R. Sprugnoli, *Event Detection and Classification for the Digital Humanities*. University of Trento, 2018.

[26] A. Vasilakopoulos, *Temporal Information Extraction*. The University of Manchester (United Kingdom), 2006.

[27] Z. Vendler, Verbs and Times. *The Philosophical Review, 66* (2), 1957, pp. 143-160.

[28] M. Verhagen, Temporal closure in an annotation environment. Language Resources and Evaluation, JSTOR, 2005, 39, pp. 211-241.

[29] K.-F. Wong, Y. Xia, W. Li, and C. Yuan, An overview of temporal information extraction. International Journal of Computer Processing of Oriental Languages, World Scientific, 2005, 18, pp. 137-152.

[30] A. F. Fehri, *Key features and parameters in Arabic grammar*. Vol. 182. John Benjamins Publishing, 2012.

[31] B. A. D. Mudhsh, “A comparative study of tense and aspect categories in Arabic and English.” Cogent Arts & Humanities, 8(1), 1899568. 2021.

[32] K. C. Ryding, *A reference grammar of modern standard Arabic*. Cambridge University press, 2005.

[33] Annotation Task and Specifications. The Linguistic Data Consortium. [retrieved: 09, 2021] https://www.ldc.upenn.edu/collaborations/past-projects/ace/annotation-tasks-and-specifications

[34] R. Grishman, B. Sundheim, “Message Understanding Conference - 6: A Brief History,” In *Proceedings of the 16th International Conference on Computational Linguistics* (COLING), I, Copenhagen, 1996, pp. 466-471.

[35] *Translingual Information Detection, Extraction, and Summarization* (TIDES) Evaluation Site. https://www-nlpir.nist.gov/tides/index.html, [retrieved: 09, 2021]