Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence (IJCAI-23)
Demonstrations Track

`NeoMaPy`: A Framework for Computing MAP Inference on Temporal Knowledge Graphs

Victor David[1] , **Raphaël Fournier-S'niehotta**[2] and **Nicolas Travers**[3,2]
[1]Department of Mathematics and Computer Science University of Perugia
[2]Conservatoire National des Arts et Métiers
[3]Léonard de Vinci Pôle Universitaire, Research Center
victor.david@unipg.it, fournier@cnam.fr, nicolas.travers@devinci.fr

Abstract

Markov Logic Networks (MLN) are used for reasoning on uncertain and inconsistent temporal data. We proposed the TMLN (Temporal Markov Logic Network) which extends them with sorts/types, weights on rules and facts, and various temporal consistencies. The `NeoMaPy` framework integrates it in a knowledge graph based on conflict graphs, which offers flexibility for reasoning with parametric Maximum A Posteriori (MAP) inferences, efficiency thanks to an optimistic heuristic and interactive graph visualization for results explanation.

1 Introduction

Markov Logic Networks (MLNs) [Richardson and Domingos, 2006; Domingos and Lowd, 2019] are a very useful conceptual tool for reasoning over uncertain facts. They combine Markov networks and First Order Logic, by attaching weights to logic formulae. Several MLNs extensions have been devised to work on different types of data [Snidaro *et al.*, 2015; Chekol *et al.*, 2016; Rincé *et al.*, 2018]. Those uncertain temporal facts generate *conflicts*. Reasoning on those facts often requires to resolve those conflicts, *i.e.,* to find consistent sets of facts useful for multi-agent tasks, production of hypotheses in history, global analysis, etc.

MLNs help find *the most probable state of the world*, gathering a set of facts whose weights have maximal probabilities with a process called *Maximum A-Posteriori* inference (MAP) [Niu *et al.*, 2011; Riedel, 2012; Noessner *et al.*, 2013; Sarkhel *et al.*, 2014]. However, the state of the art integrating temporal information into MLN is insufficient, and computing the MAP inference usually relies on a heavy data mining process which checks rules application on possible facts [Chekol *et al.*, 2017b]. It may be optimized by aggregating some formulae and by parallelizing the mining, but the complexity of those pessimistic approaches remains highly dependent on the number of possibilities.

We have recently introduced an extension of MLNs called Temporal Markov Logic Networks (TMLN) [David *et al.*, 2022], along with a temporal semantics which may be configured through 3 categories of functions. We have devised key principles on the semantics for MAP inference, to reach desirable properties, and examined total and partial (in)consistency relations between temporal formulae. Our completely different approach to MAP inference relies on building compatible worlds instead of mining valid worlds.

In this paper, we now introduce the `NeoMaPy` framework, a complete implementation of our approach for TMLN reasoning[1]. To achieve this, we extract a conflict graph between facts [Bertossi, 2011; Hipel *et al.*, 2020], based on the rules and the nodes weights. Thus, the MAP inference now searches combinations of non-conflict graphs. This optimistic approach allows to parameterize MAP inferences with various semantics, computing efficiently with a heuristic and interacting with results for explaining choices of facts.

2 Background Concepts

2.1 Temporal Markov Logic Networks

Temporal Markov Logic Networks are based on a Temporal Many-Sorted First-Order Logic `TF-FOL` which combines formulae and temporal predicates from a temporal domain, to represent temporal facts and rules (more details are presented in [David *et al.*, 2022]). Temporal Markov Logic Networks associate a degree of certainty to each formula.

A TMLN $\mathcal{M} = (\mathbf{F}, \mathbf{R})$ is a set of weighted temporal facts and rules where $\mathbf{F}$ and $\mathbf{R}$ are sets of pairs such that:
– $\mathbf{F} = \{(\phi_1, w_1),\ldots, (\phi_n, w_n)\}$ with $\forall i \in \{1,\ldots, n\}$, $\phi_i \in$ `TF-FOL` such that it is a ground formula (*i.e.*, without variable, see Table 1) and $w_i \in [0, \infty[$,
– $\mathbf{R} = \{(\phi'_1, w'_1),\ldots, (\phi'_k, w'_k)\}$ with $\forall i \in \{1,\ldots, k\}$, $\phi'_i \in$ `TF-FOL` such that it is not a ground formula and in the form (premises, conclusion), *i.e.,* $(\psi_1 \wedge \ldots \wedge \psi_l) \rightarrow \psi_{l+1}$ where $\forall j \in \{1,\ldots, l + 1\}$, $\psi_j \in$ `TF-FOL`, and $w_i \in [0, \infty[$.
The universe of all TMLNs is denoted by `TMLN`.

2.2 MAP Inference

After obtaining the representation of facts and rules in a TMLN, to select the most probable and consistent set of ground formulae with a MAP inference, we proceed to an *instantiation* of the TMLN, to obtain the ground rules (when possible) by replacing variables in rules by constants.

A TMLN instantiation $I \subseteq$ `MI`$(\mathcal{M})$ is a TMLN only composed of ground formulae, I is also called *a state* of the TMLN $\mathcal{M}$, and `MI`$(\mathcal{M})$ is its maximal instantiation, *i.e.,*

[1]A companion video for this paper is available at https://www.youtube.com/watch?v=c8AzFQMs1I4

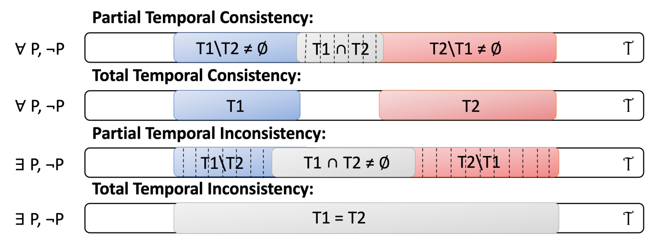

Figure 1: Temporal in/consistency information. P and $\neg P$ are conflicting predicates, with time T1 (distinct time in blue) and T2 (distinct time in red) respectively (common time is in grey). Hatched zones are not necessary time zones.

the set of all ground formulae from $\mathcal{M}$. An instantiation can be inconsistent. The universe of all TMLN instantiations is denoted by $\mathtt{TMLN}^*$. To compute the strength of a TMLN state, we resort to a *semantics*. We denote the universe of all semantics by $\mathtt{Sem}$, such that for any $\mathcal{S} \in \mathtt{Sem}$, $\mathcal{S} : \mathtt{TMLN}^* \to [0, +\infty[$. It computes a strength above 0 (not a probability between 0 and 1). One semantics may maximize the amount of information and quality for another one. States computed by MAP inference are relative to a given semantics. Given a TMLN $\mathcal{M} \in \mathtt{TMLN}$ and a semantics $\mathcal{S} \in \mathtt{Sem}$, a method solving a MAP problem is denoted by: $\mathtt{map} : \mathtt{TMLN} \times \mathtt{Sem} \to \mathcal{P}(\mathtt{TMLN}^*)$, where $\mathcal{P}(X)$ denotes the powerset of X, such that: $\mathtt{map}(\mathcal{M}, \mathcal{S}) = \{I \mid I \in \underset{I \subseteq \mathtt{MI}(\mathcal{M})}{\operatorname{argmax}} \mathcal{S}(I) \text{ and } \nexists I' \in \underset{I' \subseteq \mathtt{MI}(\mathcal{M})}{\operatorname{argmax}} \mathcal{S}(I') \text{ s.t. } I \subset I'\}$.

Our approach introduces a semantics decomposed in three functions: a *validation of instantiations* function Δ integrating various consistency relations, ii) a *selecting* function σ that modifies formulae's weights in instantiations and iii) an *aggregate* function Θ returning the final strength. A temporal parametric semantics is a tuple $\mathtt{TPS} = \langle \Delta, \sigma, \Theta \rangle \in \mathtt{Sem}$, st.: $\Delta : \mathtt{TMLN}^* \to \{0, 1\}$, $\sigma : \mathtt{TMLN}^* \to \bigcup_{k=0}^{+\infty} [0,1]^k$, $\Theta : \bigcup_{k=0}^{+\infty} [0,1]^k \to [0, +\infty[$. And for $\mathcal{M} \in \mathtt{TMLN}$, $I \subseteq \mathtt{MI}(\mathcal{M})$, the strength of a temporal parametric semantics $\mathtt{TPS} = \langle \Delta, \sigma, \Theta \rangle$ is computed by: $\mathtt{TPS}(I) = \Delta(I) \cdot \Theta\Big(\sigma(I)\Big)$.

An example of a selection function is a threshold function, an aggregation function may be different types of sums, and for validation functions we have defined the notions of partial and total temporal in/consistencies, which depend on the temporal intersection of conflicting information (Figure 1).

2.3 Reasoning Example

In Table 1, we present some facts about the Brazilian football player *Pelé*, formalized with a TMLN. It is certain that he was a football player between 1956 and 1977 (F_1). Other facts state some information about teams he may have played with, or not. Then, we introduce a rule indicating that, in general, a player can only play for one team at a time (R_1). Variables are typed into two sorts, α and β, respectively indicating a general concept or a temporal point. Table 2 shows examples of rule R_1 instantiated with some facts from Table 1. For instance, GR_{11} is instantiated from facts F_1, F_2 and F_4.

$\mathbf{F_1}$	$(Footballer(Pele, 1956, 1977)$	$, +\infty)$
$\mathbf{F_2}$	$(PlayFor(Pele, NYC, 1975, 1977)$	**, 0.6)**
$\mathbf{F_3}$	$(PlayFor(Pele, Santos, 1956, 1974)$	**, 0.8)**
$\mathbf{F_4}$	$(PlayFor(Pele, Santos, 1973, 1976)$	**, 0.4)**
$\mathbf{F_5}$	$(\neg PlayFor(Pele, Santos, 1972, 1990)$	**, 0.7)**
$\mathbf{F_6}$	$(PlayFor(Pele, Brazil, 1958, 1970)$	**, 0.9)**
$\mathbf{R_1}$	$(\forall x^\alpha, y^\alpha, z^\alpha, t_1^\beta, t_1'^\beta, t_2^\beta, t_2'^\beta, t_3^\beta, t_3'^\beta (Diff(y^\alpha, z^\alpha) \wedge Footballer(x^\alpha, t_1^\beta, t_1'^\beta) \wedge PlayFor(x^\alpha, y^\alpha, t_2^\beta, t_2'^\beta) \wedge \neg Disjoint(t_2^\beta, t_2'^\beta, t_3^\beta, t_3'^\beta)) \to \neg PlayFor(x^\alpha, z^\alpha, t_3^\beta, t_3'^\beta)$	**, 0.7)**

Table 1: Example of a TMLN for the football player Pele.

$\mathbf{GR_{11}}$	$((Diff(NYC, Santos) \wedge \neg Disjoint(1975, 1977, 1973, 1976) \wedge PlayFor(Pele, NYC, 1975, 1977) \wedge Footballer(Pele, 1956, 1977)) \to \neg PlayFor(Pele, Santos, 1973, 1976),$	**0.6)**
$\mathbf{GR_{12}}$	$((Diff(Santos, NYC) \wedge \neg Disjoint(1973, 1976, 1975, 1977) \wedge PlayFor(Pele, Santos, 1973, 1976) \wedge Footballer(Pele, 1956, 1977)) \to \neg PlayFor(Pele, NYC, 1975, 1977),$	**0.7)**
$\mathbf{GR_{13}}$	$((Diff(Brazil, Santos) \wedge \neg Disjoint(1958, 1970, 1956, 1974) \wedge PlayFor(Pele, Brazil, 1958, 1970) \wedge Footballer(Pele, 1956, 1977)) \to \neg PlayFor(Pele, Santos, 1956, 1974),$	**0.7)**

Table 2: Some ground rules instantiating R_1 (from Table 1).

3 The NeoMaPy Approach

The NeoMaPy framework consists of a two-step MAP inference extraction based on a graph database, and a conflict resolution heuristic. This major contribution introduces a parametric, efficient and interactive process.

Graph of conflicts. This first step transforms a TMLN instantiation into a property graph where constants and predicates become *Concept* nodes. Ground formulae combining those concept nodes are represented as *TF* nodes (*Temporal Formula*) with temporal predicates and weights as properties. Rules are expressed as queries on the graph of interactions between *TF* nodes based on their properties, constants and predicates. They produce conflict relationships between *TF* nodes, labelled with a conflict type. *TF* and *Concept* nodes and relationships are stored in a graph database.

Thanks to this conflict graph, applying semantics corresponds to a pattern query on the graph, searching for conflicts between *TF* nodes. It reduces the MAP inference to the computation of the maximal subset of consistent *TF* nodes.

Infering the MAP. Once the set of conflictual nodes has been obtained, the MAP inference is computed in two steps: 1) we conduct a pre-processing that structures our data into a set of connected components (*i.e.,* if there is no path between two nodes, they are not connected). 2) for each connected component (*i.e.,* a dictionary) we apply in parallel the MAP inference algorithm MaPy which creates a list of solutions by iteratively trying to add each node to the current solutions. We optimize this process by using a heuristic to eliminate the worst solutions and by restricting the size of this solution list, *i.e.,* by keeping the k best solutions.

4 Implementation

Figure 2 illustrates the architecture of the NeoMaPy framework. The first step extracts the knowledge graph by instantiating facts and ground facts with the Neo4j[2] graph database (nodes' size depends on weights). By applying rules, the

[2]https://neo4j.com

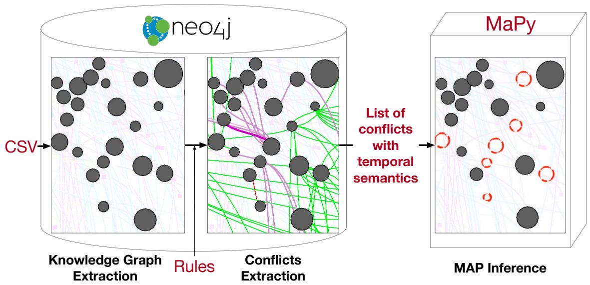

Figure 2: `NeoMaPy` pipeline for the MAP inference.

graph of conflicts is extracted from facts (green and purple links). The second step exploits the list of conflicts with a parameterized semantics and processes it with the `MaPy` algorithm implemented in Python (red circles are removed nodes). The source code and datasets are available on GitHub[3].

4.1 MAP Inference Computation

Conflicts extraction from *TF* nodes has been implemented in `Neo4j`. Facts are imported from CSV files. The graph is composed of *Concept* and *TF* nodes. Rules are applied to instantiate ground rules as *Cypher* queries. Conflicts are then instantiated as "*conflict*" relationships on the graph, by searching for `TF` with corresponding patterns (rules).

The *Cypher* query below illustrates the generation of conflicts for the *pCon* rule (partial temporal consistency). If two *TF tf1* and *tf2* share the same concepts (*s,o,p*) with opposite polarities (positive or negative information) and a timeframe intersection, it produces a *pCon* conflict between *tf1* and *tf2*. For optimization purposes, concept IDs are repeated in *TF* nodes (*e.g.,* tf1.p = tf2.p). Conflict and inference relationships are typed.

```
MATCH (tf1:TF) -[:s]-> (:Concept) <-[:s]- (tf2:TF)
WHERE tf1.p=tf2.p and tf1.o=tf2.o and tf1.polarity <>
  tf2.polarity AND ( (tf1.date_start <= tf2.date_start
  and tf2.date_start <= tf1.date_end) AND (tf1.date_start
  <= tf2.date_end and tf2.date_end <= tf1.date_end) )
MERGE (tf1)-[c:conflict]-(tf2) SET c.pCon=true;
```

The resulting graph eases the tracability of the MAP inference. Moreover, `MaPy` processes the inference with a parametric semantics extracted with a *Cypher* query of corresponding conflicts, and inference rules (inferred *TF* are ignored along with their premises), thresholds on weights, etc.

4.2 Scenarios

A Graphical User Interface was developed using *GraphStream* [4] [Dutot *et al.*, 2007], to improve the reasoning process on uncertain temporal knowledge graphs. The dataset we use contain football facts and rules from [Chekol *et al.*, 2017a]. The demonstration will show all `NeoMaPy` steps:

Graph import and conflict extraction. *Concept* and *TF* nodes are imported from CSV files into the *Neo4j* database and inference rules are applied. Then, a set of rules expressed as *Cypher* queries are applied on the graph.

[3]https://github.com/cedric-cnam/NeoMaPy_Daphne

[4]A Java library for graphs: https://graphstream-project.org/.

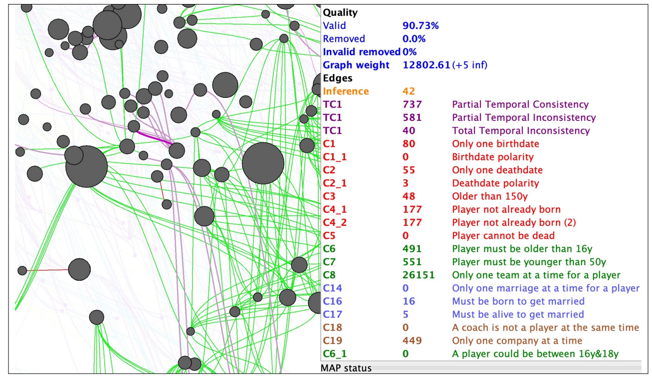

Figure 3: DataViz & statistics of the Knowledge graph with conflicts. Left: initial graph with the conflicts colored according to their type. Right: statistics, in the upper part (below "Quality"); and the list of inference and conflict types (below "Edges").

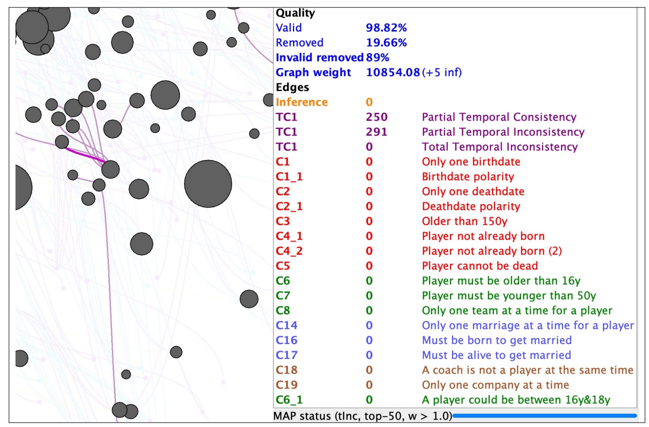

Figure 4: MAP inference DataViz with the *tInc* (Total Temporal Inconsistency) semantics. We obtain this graph without: conflicts from business rules (from C6 to C19), temporal conflicts based on *tInc*, and conflicting nodes dropped by the MAP inference.

Conflict-graph visualization. The produced conflict graph is visualized as shown in Figure 3. Several interactions are offered to users to explore the knowledge graph, such as node search, clusters of conflict nodes, zooming features. Moreover, graph statistics are computed (graph weights and conflict statistics in Figure 3).

Parametric MAP inferences. Eventually, several MAP inference computations are applied to show the impact on the graph. Simple strategies are compared with different Parametric Temporal Semantics showing the maximization of the argmax $\mathcal{S}(I)$ defined in Section 2.2 (*e.g.,* the *tInc total inconsistency* semantics in Figure 4).

5 Conclusion

Our framework `NeoMaPy` demonstrates the computation of a MAP inference on TMLNs with formalized uncertain temporal facts and rules. The two-step extraction of the graph of conflicts and their resolution with a MAP inference heuristic provides a parametric, interactive and efficient process.

Acknowledgments

This work was funded by French ANR project DAPHNE and by Project GIUSTIZIA AGILE, CUP J89J22000900005. An author is a member of the INdAM Research group GNCS.

References

[Bertossi, 2011] L. Bertossi. *Database Repairs and Consistent Query Answering*. Synthesis Lectures on Data Management. Morgan & Claypool Publishers, 2011.

[Chekol *et al.*, 2016] Melisachew Wudage Chekol, Jakob Huber, Christian Meilicke, and Heiner Stuckenschmidt. Markov logic networks with numerical constraints. In *Proceedings of the Twenty-Second European Conference on Artificial Intelligence*, ECAI'16, page 1017–1025, NLD, 2016. IOS Press.

[Chekol *et al.*, 2017a] Melisachew Chekol, Giuseppe Pirrò, Joerg Schoenfisch, and Heiner Stuckenschmidt. Marrying uncertainty and time in knowledge graphs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31, 2017.

[Chekol *et al.*, 2017b] Melisachew Wudage Chekol, Giuseppe Pirro, Joerg Schoenfisch, and Heiner Stuckenschmidt. Tecore: temporal conflict resolution in knowledge graphs. *Proceedings of the VLDB Endowment*, 10:Iss–12, 2017.

[David *et al.*, 2022] Victor David, Raphaël Fournier-S'niehotta, and Nicolas Travers. Parameterisation of reasoning on temporal markov logic networks, 2022. arXiv: https://arxiv.org/abs/2211.16414.

[Domingos and Lowd, 2019] Pedro Domingos and Daniel Lowd. Unifying logical and statistical ai with markov logic. *Communications of the ACM*, 62(7):74–83, 2019.

[Dutot *et al.*, 2007] Antoine Dutot, Frédéric Guinand, Damien Olivier, and Yoann Pigné. GraphStream: A Tool for bridging the gap between Complex Systems and Dynamic Graphs. In *Emergent Properties in Natural and Artificial Complex Systems. Satellite Conference within the 4th European Conference on Complex Systems (ECCS'2007)*, Dresden, Germany, October 2007.

[Hipel *et al.*, 2020] Keith W. Hipel, Liping Fang, and D. Marc Kilgour. The graph model for conflict resolution: Reflections on three decades of development. *Group Decision and Negotiation*, 29(1):11–60, 2020.

[Niu *et al.*, 2011] Feng Niu, Christopher Ré, AnHai Doan, and Jude Shavlik. Tuffy: Scaling up statistical inference in markov logic networks using an rdbms. *arXiv preprint arXiv:1104.3216*, 2011.

[Noessner *et al.*, 2013] Jan Noessner, Mathias Niepert, and Heiner Stuckenschmidt. Rockit: Exploiting parallelism and symmetry for map inference in statistical relational models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 27, pages 739–745, 2013.

[Richardson and Domingos, 2006] Matthew Richardson and Pedro Domingos. Markov Logic Networks. *Machine Learning*, 62(1):107–136, Feb 2006.

[Riedel, 2012] Sebastian Riedel. Improving the accuracy and efficiency of map inference for markov logic. *arXiv preprint arXiv:1206.3282*, 2012.

[Rincé *et al.*, 2018] Romain Rincé, Romain Kervarc, and Philippe Leray. Complex event processing under uncertainty using markov chains, constraints, and sampling. In Christoph Benzmüller, Francesco Ricca, Xavier Parent, and Dumitru Roman, editors, *Rules and Reasoning*, pages 147–163, Cham, 2018. Springer.

[Sarkhel *et al.*, 2014] Somdeb Sarkhel, Deepak Venugopal, Parag Singla, and Vibhav Gogate. Lifted map inference for markov logic networks. In *Artificial Intelligence and Statistics*, pages 859–867. PMLR, 2014.

[Snidaro *et al.*, 2015] Lauro Snidaro, Ingrid Visentini, and Karna Bryan. Fusing uncertain knowledge and evidence for maritime situational awareness via markov logic networks. *Information Fusion*, 21:159–172, 2015.