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 [Snidar 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 \(\mathcal{M} = (\mathcal{F}, \mathcal{R})\) is a set of weighted temporal facts and rules where \(\mathcal{F}\) and \(\mathcal{R}\) are sets of pairs such that:

- \(\mathcal{F} = \{ (\phi_1, w_1), \ldots, (\phi_n, w_n) \} \) with \(\forall i \in \{1, \ldots, n\}, \phi_i \in \text{TF-FOL}\) such that it is a ground formula (i.e., without variable, see Table 1) and \(w_i \in [0, \infty[\)

- \(\mathcal{R} = \{ (\phi_1', w_1'), \ldots, (\phi_k', w_k') \} \) with \(\forall i \in \{1, \ldots, k\}, \phi_i' \in \text{TF-FOL}\) such that it is not a ground formula and in the form (premises, conclusion), i.e., \((\psi_1 \land \ldots \land \psi_l) \rightarrow \psi_{l+1}\) where \(\forall j \in \{1, \ldots, l\}, \psi_j \in \text{TF-FOL}\) and \(w_i \in [0, \infty[\).

The universe of all TMLNs is denoted by \(\mathcal{TMLN}\).

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

\(^1\)A companion video for this paper is available at https://www.youtube.com/watch?v=cSAzFQMs1I4

7123
of rule general concept or a temporal point. Table 2 shows examples are typed into two sorts, $R$ and $S$. To state some information about teams he may have played with, Pelé.

### 2.3 Reasoning Example

![Figure 1: Temporal in/consistency information. $P$ and $\neg P$ are conflicting predicates, with time $T_1$ (distinct time in blue) and $T_2$ (distinct time in red) respectively (common time is in grey). Hatched zones are not necessary time zones.](https://example.com/figure1)

The set of all ground formulae from $M$. An instantiation can be inconsistent. The universe of all TMLN instantiations is denoted by $\text{TMLN}^*$. To compute the strength of a TMLN state, we resort to a semantics. We denote the universe of all semantics by $\text{Sem}$, such that for any $S \in \text{Sem}$, $S : \text{TMLN}^* \rightarrow [0, +\infty]$. 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 $M \in \text{TMLN}$ and a semantics $S \in \text{Sem}$, a method solving a temporal problem is denoted by: $\text{map} : \text{TMLN} \times \text{Sem} \rightarrow \mathcal{P} (\text{TMLN}^*)$, where $\mathcal{P} (X)$ denotes the powerset of $X$, such that: $\text{map}(M, S) = \{ I | I \in \text{argmax} \ S(I) \}$ and $\exists I' \in \text{argmax} \ S(I') \text{ s.t. } I \subset I'$.

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

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, $\alpha$ and $\beta$, 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 $F_4$.

<table>
<thead>
<tr>
<th>$\forall P, \neg P$</th>
<th>$\forall P, \neg P$</th>
<th>$\exists P, \neg P$</th>
<th>$\exists P, \neg P$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\text{TPS}(I) = \Delta(I) \cdot \Theta(\sigma(I))$</td>
<td>$\text{TPS}(I) = \Delta(I) \cdot \Theta(\sigma(I))$</td>
<td>$\text{TPS}(I) = \Delta(I) \cdot \Theta(\sigma(I))$</td>
<td>$\text{TPS}(I) = \Delta(I) \cdot \Theta(\sigma(I))$</td>
</tr>
</tbody>
</table>

| $\text{Table 1: Example of a TMLN for the football player Pelé.}$ |
|-------------------|-------------------|--------------------|--------------------|
| $\text{GR}_{11}$ | $\text{GR}_{11}$ | $\text{GR}_{12}$ | $\text{GR}_{13}$ |
| $\langle (\text{Diff}(\text{NYC, Santos}) \land \neg \text{Disjoint}(1975, 1977, 1975, 1976) \land \text{PlayFor}(\text{Pele, NYC, 1975, 1977}) \land \text{Footballer}(\text{Pele, 1956, 1977})) \rightarrow \neg \text{PlayFor}(\text{Pele, Santos, 1975, 1973}), \sigma(\alpha) \rangle$ | $\langle (\text{Diff}(\text{NYC, Santos}) \land \neg \text{Disjoint}(1973, 1975, 1977, 1976) \land \text{PlayFor}(\text{Pele, NYC, 1975, 1977}) \land \text{Footballer}(\text{Pele, 1956, 1977})) \rightarrow \neg \text{PlayFor}(\text{Pele, Santos, 1973, 1975}), \sigma(\alpha) \rangle$ | $\langle (\text{Diff}(\text{NYC, Santos}) \land \neg \text{Disjoint}(1973, 1975, 1977, 1976) \land \text{PlayFor}(\text{Pele, NYC, 1975, 1977}) \land \text{Footballer}(\text{Pele, 1956, 1977})) \rightarrow \neg \text{PlayFor}(\text{Pele, Santos, 1975, 1973}), \sigma(\alpha) \rangle$ | $\langle (\text{Diff}(\text{NYC, Santos}) \land \neg \text{Disjoint}(1973, 1975, 1977, 1976) \land \text{PlayFor}(\text{Pele, NYC, 1975, 1977}) \land \text{Footballer}(\text{Pele, 1956, 1977})) \rightarrow \neg \text{PlayFor}(\text{Pele, Santos, 1975, 1973}), \sigma(\alpha) \rangle$ |

| $\text{Table 2: Some ground rules instantiating } R_1 \text{ (from Table 1).}$ |
|-------------------|-------------------|--------------------|--------------------|
| $\forall P, \neg P$ | $\forall P, \neg P$ | $\exists P, \neg P$ | $\exists P, \neg P$ |
| $\text{TPS}(I) = \Delta(I) \cdot \Theta(\sigma(I))$ | $\text{TPS}(I) = \Delta(I) \cdot \Theta(\sigma(I))$ | $\text{TPS}(I) = \Delta(I) \cdot \Theta(\sigma(I))$ | $\text{TPS}(I) = \Delta(I) \cdot \Theta(\sigma(I))$ |

### 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 parameteric, 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.

**Inferring 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 graph database (nodes’ size depends on weights). By applying rules, the
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.

```
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:TF)-[c:conflict]-(tf2:TF)
```

The resulting graph eases the traceability 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\textsuperscript{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.

\textsuperscript{1}https://github.com/cedric-cnam/NeoMaPy_Daphne
\textsuperscript{2}A Java library for graphs: https://graphstream-project.org/.
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


