Parameterisation of MAP Inference on Temporal Markov Logic Networks

Anonymous Author(s)
Submission Id: 797

ABSTRACT

Reasoning on uncertain and inconsistent temporal data is crucial for several multi-agent tasks. Markov Logic Networks (MLN) have been for instance used for improving plan or goal recognition. However, incorporating temporal information into MLNs may still be improved. We propose TMLNs (Temporal Markov Logic Networks) to extend the knowledge representation capacity of MLNs to temporal data. This extension offers weighted facts and rules, and validity intervals for predicates. We then introduce a temporal and parametric semantics (TPS), allowing flexible reasoning through several functions (enforcing temporal consistency constraints and knowledge selection and aggregation). Finally, we present the new NeoMaPy tool, to compute Maximum A Posteriori (MAP) inference on MLNs and TMLNs with several TPS. We compare our performance with state-of-the-art inference tool and exhibit faster results.

KEYWORDS

Temporal Markov Logic Networks, Maximum A-Posteriori Inference, Temporal Parametric Semantics, Knowledge Graph, Goal Recognition.

ACM Reference Format:

Anonymous Author(s). 2023. Parameterisation of MAP Inference on Temporal Markov Logic Networks. In *Proc. of the 22nd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2023), London, United Kingdom, May 29 – June 2, 2023*, IFAAMAS, 9 pages.

1 INTRODUCTION

Reasoning on uncertain and complex data to infer the most probable outcome is a crucial task in several multi-agent settings, such as guessing an agent's goal or recognising their plan through a sequence of observed actions [11]. Excelling at these two tasks involves examining logical links between actions, surfacing the conflicts among facts, and filtering them out to reach a consistent state. Knowledge graphs (KGs) are a key tool for modelling actions, represented as *facts* with a (subject, predicate, object) triplet. Several approaches have been developed to reason on KGs, and one of them, Markov Logic Networks [6, 18], propose probabilistic reasoning in a unified formalism supporting uncertainty, which is key for goal recognition. However, only few MLN-based KG reasoning frameworks handle both uncertainty and temporal data [5], and none can handle a fully uncertain universe where any fact or rule may be uncertain.

For goal recognition, reasoning under uncertainty and inconsistency is at the basis of methodology: the validity of the knowledge of any agent remains questionable. Temporal information is also crucial: outside of a temporal interval, a fact becomes false. By using time intervals in knowledge, it is possible to refine the reasoning of agents during a sequence of events. Once the scenario has been modelled, predicting a goal amount to inferring the most likely consistent goal. In [3, 10, 11, 13] the authors propose MLN-based learning approaches and in [14, 15] deep Learning approaches to enhance goal recognition. However, those representations hardly handle a fully uncertain universe (facts and rules). Moreover, they are not explainable, one cannot trace back the facts that leads to a given conclusion (goal).

We propose a novel and general approach, which introduces a new representation of knowledge, called *temporal MLN* (TMLNs), built on a temporal many-sorted logic, for reasoning on both time and uncertainty. We enlarge the notion of uncertainty to rules, and adapt the reasoning, to deal with both uncertain facts and rules. We exemplify our method on a goal recognition case, from the educational game `Crystal Island` [19]. We then define a new temporal semantics and a temporal extension to *Maximum A-Posteriori* (MAP) inference [17]. This MAP inference produces *instantiations*, *i.e.,* extended sets of facts maximising the score *w.r.t.* a temporal semantics: it leads to the agent's goal. The proposed temporal semantics is parametric: it allows combining several sub-functions for various consistency validations (*i.e.,* highlighting different probable goals). Finally, we present our `NeoMaPy` process that computes the MAP inference, an approach based on conflict graphs (with Labelled Property Graphs) which allows the process to be efficient and explainable.

2 BACKGROUND

In a seminal work [5], Chekol et al. formalise the *Uncertain Temporal Knowledge Graphs* (UTKG) approach, which integrates both time and uncertainty in KGs to reach a *certain world maximisation*. However, they do not take into account the possibility to have uncertain rules. We enlarge their vision by putting *time* at the heart of reasoning. We formalise the notion of *temporal uncertainty*, by combining certain and uncertain formulae. And our novel representation of TMLN leads to easier manipulations and better analyses.

2.1 Many-Sorted First Order Logic

We start by presenting the Many-Sorted First-Order Logic [20].

Definition 2.1 (Many-Sorted FOL). Let $\mathbf{So} = \{s_1, \dots, s_n\}$ be a set of sorts. A *Many-Sorted First-Order Logic* `MS-FOL`, is a set of formulae built up by induction from: a set $\mathbf{C} = \{a_1, \dots, a_l\}$ of constants, a set $\mathbf{V} = \{x^s, y^s, z^s, \dots \mid s \in \mathbf{So}\}$ of variables, a set $\mathbf{P} = \{P_1, \dots, P_m\}$ of predicates, a function $\mathtt{ar} : \mathbf{P} \rightarrow \mathbb{N}$ which tells the arity of any predicate, a function `sort` s.t. for $P \in \mathbf{P}$, $\mathtt{sort}(P) \in \mathbf{So}^{\mathtt{ar}(P)}$, and for $c \in \mathbf{C}$, $\mathtt{sort}(c) \in \mathbf{So}$, the usual connectives ($\neg, \vee, \wedge, \rightarrow, \leftrightarrow$), Boolean constants ($\top$ and $\bot$) and quantifier symbols ($\forall, \exists$). A *ground formula* is a formula without any variable.

Proc. of the 22nd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2023), A. Ricci, W. Yeoh, N. Agmon, B. An (eds.), May 29 – June 2, 2023, London, United Kingdom.

Lowercase (resp. uppercase) greek letters like ϕ, ψ (resp. Φ, Ψ) denote formulae (resp. sets of formulae).

Example 2.2. For instance let $\mathbf{So} = \{s_1, s_2\}$, let $P_1 \in \mathbf{P}$ such that $\texttt{sort}(P_1) = s_2 \times s_1 \times s_1$, let $a_1, a_2, t_1, t_2 \in \mathbf{C}$ such that $\texttt{sort}(a_1) = \texttt{sort}(a_2) = s_2$, $\texttt{sort}(t_1) = \texttt{sort}(t_2) = s_1$ and let $x^{s_2} \in \mathbf{V}$. We can then build the following MS-FOL formulae: $P_1(a_1, t_1, t_2)$, $\forall x^{s_2} P_1(x^{s_2}, t_1, t_2)$. However, $P_1(t_1, t_2, a_1)$ or $\forall x^{s_2} P_1(a_1, a_2, x^{s_2})$ cannot be built because they do not respect the sorts.

MS-FOL formulae are evaluated via a notion of *structure* called *n*-sorted structures [9]. Classical first-order logic formulae are captured as 1-sorted structures.

Definition 2.3 (Structure). A *n-sorted structure* is $\mathbf{St} = (\{D_1, \dots, D_n\}, \{R_1, \dots, R_m\}, \{c_1, \dots, c_l\})$, where $D_1, \dots, D_n$ are the (non-empty) domains, $R_1, \dots, R_m$ are relations between domains' elements, and $c_1, \dots, c_l$ are distinct constants in the domains.

Our running example is presented in Example 2.4. The sentences describe actions and pieces of knowledge obtained during the beginning of a game of `Crystal Island` [19], at given timestamps. In `Crystal Island`, players face an epidemic which takes place on the Island (fictional place). Their objective is to identify the disease that is infecting patients (they submit a hypothesis about that disease to the nurse). Among the possible actions, players can move, read a book or interact with experts. In this context, *goal recognition* involves predicting the next narrative sub-goal that the player will achieve as they solve the interactive mystery.

Example 2.4. At time t_1: we know that *Patient1* has most likely started his illness and the player has started reading a book from t_1 to t_2. At t_2: from the information in the book, we know that it is possible for the player to hypothesise *Disease1* for 5 times, we also know that often patients with *Disease1* are ill for 5 times, and then generally if a patient is no longer ill at the 5th time of the disease then he/she should not have *Disease1*. At t_3: the player moves to the laboratory. At t_4: the player talks with a scientist until t_5. At t_5: after talking to the scientist, the player is likely to get new information about the hypothesis1 of the mysterious disease. At t_6: the player analyses a blood sample and learns that the patient is most likely not ill anymore. Regarding the goals, we know that the higher the probability on the disease hypothesis, the higher the probability the player will want to submit it. And then we know that the actions of moving to the laboratory, reading a book or testing a sample are often correlated to the search for a clue.

In MS-FOL, we define a suitable structure without uncertainty.

Example 2.5. An example of structure associated with the MS-FOL from Example 2.4 is $\mathbf{St}_{island} = (\{Time, Concept\}, \{Knowledge, Action, Goal\}, \{t_1, t_2, \dots, t_{10}, OnsetIllPatient_1, ReadBook, Hypo_1, IllPatient_1, MoveToLab, TalkScientific, TestingSample, SubDiag, FindClue\})$, in which:

- $Time$ is the set of time points (sort s_1), and $Concept$ is the set of all non-temporal objects (sort s_2),
- $Knowledge, Action, Goal$ are the predicate symbols' relations (*e.g.,* $Knowledge \subseteq Concept \times Time \times Time$ indicates which elements are knowledge).
- $t_1, t_2, \dots, t_{10}$ are elements of the domain $Time$ associated with the sort s_1, while $OnsetIllPatient, ReadBook, \dots, SubDiag, FindClue$ are elements of the domain $Concept$ (sort s_2).

2.2 MS-FOL Reasoning

Now, we define MS-FOL formulae for *interpretation*.

Definition 2.6 (Interpretation). An *interpretation* $\mathbf{I_{St}}$ *over a structure* $\mathbf{St}$ assigns to elements of the MS-FOL vocabulary some values in the structure $\mathbf{St}$. Formally,
– $\mathbf{I_{St}}(s_i) = D_i$, for $i \in \{1, \dots, n\}$ (each sort symbol is assigned to a domain),
– $\mathbf{I_{St}}(P_i) = R_i$, for $i \in \{1, \dots, m\}$ (each predicate symbol is assigned to a relation),
– $\mathbf{I_{St}}(a_i) = c_i$, for $i \in \{1, \dots, l\}$ (each constant symbol is assigned to a value).
Then, satisfying formulae is recursively defined by:
– $\mathbf{I_{St}} \models P_i(a_1, \dots, a_k)$ iff $(\mathbf{I_{St}}(a_1), \dots, \mathbf{I_{St}}(a_k)) \in R_i$,
– $\mathbf{I_{St}} \models \exists x^{s_i} \phi$ iff $\mathbf{I}_{\mathbf{St}, x^{s_i} \leftarrow v} \models \phi$ for some $v \in D_i$,
– $\mathbf{I_{St}} \models \forall x^{s_i} \phi$ iff $\mathbf{I}_{\mathbf{St}, x^{s_i} \leftarrow v} \models \phi$ for each $v \in D_i$,
– $\mathbf{I_{St}} \models \phi \land \psi$ iff $\mathbf{I_{St}} \models \phi$ and $\mathbf{I_{St}} \models \psi$,
– $\mathbf{I_{St}} \models \phi \lor \psi$ iff $\mathbf{I_{St}} \models \phi$ or $\mathbf{I_{St}} \models \psi$,
– $\mathbf{I_{St}} \models \neg\phi$ iff $\mathbf{I_{St}} \not\models \phi$,
where $\mathbf{I}_{\mathbf{St}, x^{s_i} \leftarrow v}$ is a modified version of $\mathbf{I_{St}}$ s.t. the variable x^{s_i} is replaced by a value v in the domain D_i corresponding to the sort symbol s_i. Finally, if Φ is a set of formulae, then $\mathbf{I_{St}} \models \Phi$ iff $\mathbf{I_{St}} \models \phi$ for each $\phi \in \Phi$.

Definition 2.6 does not target the satisfaction of implications and equivalences, while they can be defined by: $(\phi \rightarrow \psi) \equiv (\neg\phi \lor \psi)$, and $(\phi \leftrightarrow \psi) \equiv (\phi \rightarrow \psi) \land (\psi \rightarrow \phi)$. For instance, the set of interpretations of the formula $P(a) \lor P(b)$ is equal to $\{\{P(a)\}, \{P(b)\}, \{P(a), P(b)\}\}$ and for $P(a) \land P(b)$ is $\{\{P(a), P(b)\}\}$.

Thanks to structures and interpretations on TMLNs, we define the consequence relations and logical consequences over MS-FOL.

Definition 2.7 (Consequence Relation). Let ϕ and ψ be two MS-FOL formulae. We say that ψ *is a consequence of* ϕ, denoted by $\phi \vdash \psi$, if for any structure $\mathbf{St}$, and any interpretation $\mathbf{I_{St}}$ over $\mathbf{St}$, $\mathbf{I_{St}} \models \phi$ implies $\mathbf{I_{St}} \models \psi$.

Definition 2.8 (Logical Consequences - Cn). Let $\phi \in$ MS-FOL. The function $\texttt{Cn}(\phi)$ is the set of all logical consequences of ϕ, *i.e.,* $\texttt{Cn}(\phi) = \{\psi \in \texttt{MS-FOL} \mid \phi \vdash \psi\}$.

The function Cn returns an infinite set of formulae, but for clarity we consider only one formula per equivalent class and only the predicates and constants appearing in the original formulae. Such as, $\texttt{Cn}(P(a) \lor P(b)) = \{P(a) \lor P(b)\}$ and
$\texttt{Cn}(P(a) \land P(b)) = \{P(a), P(b), P(a) \land P(b), P(a) \lor P(b)\}\}$.

Since we are working with TS-formulae, we extend inferences according to predicates' temporal interval such that for each formula, each predicate, we can also infer all possible temporal subsets. For example: $\texttt{Cn}(P(a, t_1, t_2) \land P(b, t_2, t_2)) = \{P(a, t_1, t_1),\ P(a, t_1, t_2),\ P(a, t_2, t_2),\ P(b, t_2, t_2),\ P(a, t_1, t_1) \land P(b, t_2, t_2), P(a, t_1, t_2) \land P(b, t_2, t_2), \dots, P(a, t_2, t_2) \lor P(b, t_2, t_2)\}\}$. In the rest of the article, if not specified we consider the temporally extended version of Cn.

F_1	$(Knowledge(OnsetIllPatient_1, t_1, t_1)$	$, \mathbf{0.9})$
F_2	$(Action(ReadBook, t_1, t_2)$	$, 10^{10})$
F_3	$(Knowledge(Hypo_1, t_2, t_7)$	$, \mathbf{0.4})$
R_1	$(\forall t_k^{s_1}, t_i^{s_1}, t_j^{s_1} \text{ s.t. } t_i^{s_1} \neq t_j^{s_1}, Knowledge(Hypo_1, t_i^{s_1}, t_j^{s_1}) \wedge Knowledge(OnsetIllPatient_1, t_k^{s_1}, t_k^{s_1}) \rightarrow Knowledge(IllPatient_1, t_k^{s_1}, t_{k+5}^{s_1})$	$, \mathbf{0.6})$
R_2	$(\forall t_i^{s_1}, Knowledge(OnsetIllPatient_1, t_i^{s_1}, t_i^{s_1}) \wedge \neg Knowledge(IllPatient_1, t_{i+5}^{s_1}, t_{i+5}^{s_1}) \rightarrow \neg Knowledge(Hypo_1, t_{i+5}^{s_1}, t_{i+5}^{s_1})$	$, \mathbf{0.7})$
F_4	$(Action(MoveToLab, t_3, t_3)$	$, 10^{10})$
F_5	$(Action(TalkScientifc, t_4, t_5)$	$, 10^{10})$
F_6	$(Knowledge(Hypo_1, t_5, t_{10})$	$, \mathbf{0.5})$
F_7	$(Action(TestingSample_1, t_6, t_6)$	$, 10^{10})$
F_8	$(\neg Knowledge(IllPatient_1, t_6, t_6)$	$, \mathbf{0.8})$
R_3	$(\forall t_i^{s_1}, Knowledge(Hypo_1, t_i^{s_1}, t_i^{s_1}) \rightarrow Goal(SubDiag, t_i^{s_1}, t_i^{s_1})$	$, \mathbf{1})$
R_4	$(\forall t_i^{s_1}, t_j^{s_1}, Action(MoveToLab, t_i^{s_1}, t_i^{s_1}) \vee Action(TestingSample, t_i^{s_1}, t_i^{s_1}) \vee Action(ReadBook, t_i^{s_1}, t_j^{s_1}) \rightarrow Goal(FindClue, t_i^{s_1}, t_i^{s_1})$	$, \mathbf{0.7})$
$F_{9(1,\dots,6)}$	$\big(Goal(SubDiag, t_{(1,\dots,6)}, t_{(1,\dots,6)}) \wedge \neg Goal(FindClue, t_{(1,\dots,6)}, t_{(1,\dots,6)})\big) \vee \big(\neg Goal(SubDiag, t_{(1,\dots,6)}, t_{(1,\dots,6)}) \wedge Goal(FindClue, t_{(1,\dots,6)}, t_{(1,\dots,6)})\big)$	$, 10^{10})$

Table 1: Example of TMLN for a `Crystal Island` game.

$GR_{11} = (Knowledge(Hypo_1, t_2, t_7) \wedge Knowledge(OnsetIllPatient_1, t_1, t_1) \rightarrow Knowledge(IllPatient_1, t_1, t_6)$	$, \mathbf{0.4})$
$GR_{12} = (Knowledge(Hypo_1, t_5, t_{10}) \wedge Knowledge(OnsetIllPatient_1, t_1, t_1) \rightarrow Knowledge(IllPatient_1, t_1, t_6)$	$, \mathbf{0.5})$
$GR_2 = (Knowledge(OnsetIllPatient_1, t_1, t_1) \wedge \neg Knowledge(IllPatient_1, t_6, t_6) \rightarrow \neg Knowledge(Hypo_1, t_6, t_6)$	$, \mathbf{0.7})$
$GR_{31(2,3,4,5,6,7)} = (Knowledge(Hypo_1, t_{(2,3,4,5,6,7)}, t_{(2,3,4,5,6,7)}) \rightarrow Goal(SubDiag, t_{(2,3,4,5,6,7)}, t_{(2,3,4,5,6,7)})$	$, \mathbf{0.4})$
$GR_{32(5,6,7,8,9,10)} = (Knowledge(Hypo_1, t_{(5,6,7,8,9,10)}, t_{(5,6,7,8,9,10)}) \rightarrow Goal(SubDiag, t_{(5,6,7,8,9,10)}, t_{(5,6,7,8,9,10)})$	$, \mathbf{0.5})$
$GR_{41} = (Action(ReadBook, t_1, t_2) \rightarrow Goal(FindClue, t_1, t_1)$	$, \mathbf{0.7})$
$GR_{43} = (Action(MoveToLab, t_3, t_3) \rightarrow Goal(FindClue, t_3, t_3)$	$, \mathbf{0.7})$
$GR_{46} = (Action(TestingSample, t_6, t_6) \rightarrow Goal(FindClue, t_6, t_6)$	$, \mathbf{0.7})$

Table 2: Ground Rules Instantiating R_1, R_2, R_3 and R_4 (from Table 1).

3 TEMPORAL AND UNCERTAIN KNOWLEDGE REPRESENTATION

Markov Logic Networks (MLNs) combine Markov Networks and First-Order Logic (FOL) by attaching weights to first-order formulae and treating them as feature templates for Markov Networks [16]. We extend this framework to temporal information by resorting to Many-Sorted First-Order Logic (`MS-FOL`).

3.1 Temporal Markov Logic Networks

Let start with the Temporal Many-Sorted First-Order Logic `TS-FOL`.

Definition 3.1 (Temporal Many-Sorted FOL). A `TS-FOL` evaluated by a structure **St** is a constrained `MS-FOL` where $|\mathbf{So}| \geq 2$, for any interpretation $\mathbf{I_{St}}(s_1) = Time$, any predicate $P_i \in$ `TS-FOL` has $\mathtt{ar}(P_i) \geq 3$ with the sort of the last two parameters belonging to s_1 and t_{min} and t_{max} are time constants indicating the minimum and maximum time points for any pre-order between the time constants.

Using this constrained `MS-FOL` accompanied with a temporal domain (named $Time$) and temporal predicates (the last two parameters indicate the validity temporal bounds), we may represent temporal facts and rules. Finally, Temporal Markov Logic Networks (TMLN) extend `TS-FOL` (resp. MLN) by associating a degree of certainty to each formula (resp. by adding a temporal validity to the predicates).

Definition 3.2 (TMLN). A Temporal Markov Logic Network $\mathbf{M} = (\mathbf{F}, \mathbf{R})$, based on a `TS-FOL`, is a set of weighted temporal facts and rules where **F** and **R** are sets of pairs such that:
– $\mathbf{F} = \{(\phi_1, w_1), \dots, (\phi_n, w_n)\}$ with $\forall i \in \{1, \dots, n\}$, $\phi_i \in$ `TS-FOL` such that it is a ground formula and $w_i \in [0, \infty[$,
– $\mathbf{R} = \{(\phi'_1, w'_1), \dots, (\phi'_k, w'_k)\}$ with $\forall i \in \{1, \dots, k\}$, $\phi'_i \in$ `TS-FOL` such that it is not a ground formula and in the form (premises, conclusion), i.e., $(\psi_1 \wedge \dots \wedge \psi_l) \rightarrow \psi_{l+1}$ where $\forall j \in \{1, \dots, l+1\}$, $\psi_j \in$ `TS-FOL`, and $w_i \in [0, \infty[$.
The universe of all TMLNs is denoted by `TMLN`.

Note that, uncertain knowledge is described with a belief degree $w \in [0, 1[$ and for certain information/hard constraints we will use very high numbers, e.g., $w = 10^{10}$.

In the rest of the paper, we simplify the example by directly using the structure defined in Example 2.5 (*c.f.* Section 2.1).

Example 2.4 (Continued). The TMLN representation of our running example can be found in Table 1. We identify 6 independent facts and 2 rules, each one with temporal validity and certainty weights (arbitrary extracted from Example 2.4).

In order to select the most probable and consistent set of ground formulae with a MAP inference, we need first to have all data (facts and rules) represented in a TMLN. Then, we obtain the ground rules (if possible), by replacing the variables in the rules by constants (according to our TMLN). We call this second step the instantiation.

3.2 TMLN Instantiation

Let **M** be a TMLN, we denote by $\mathtt{MI}(\mathbf{M})$ the *Maximal TMLN Instantiation* of **M**. $\mathtt{MI}(\mathbf{M})$ contains the set of **M**'s facts and all ground rules that can be constructed by instantiating all its predicates containing variables by other deductible ground predicates (Reasoning with Def. 2.7 and 2.8). A ground rule's weight is the minimum of the weights of the formulae in **M** used to construct the instantiated rule.

Formally, to define the set of instantiations, we have to define two useful notions. Firstly, we denote by $\mathtt{TF}(\mathbf{M}) = \bigcup_{(\phi, w) \in \mathbf{M}} \phi$ the set of temporal formulae (without weight) of $\mathbf{M} \in$ `TMLN`. Secondly, we define the function $\mathtt{W} : \mathtt{TS\text{-}FOL} \times \mathtt{TMLN} \longrightarrow [0, \infty[$, returning the maximal weight of a temporal formula deductible from a TMLN: $\mathtt{W}(\phi, \mathbf{M}) = \mathtt{max}(\mathtt{min_w}(\mathbf{M}_1), \dots, \mathtt{min_w}(\mathbf{M}_m))$ s.t. $\{\mathbf{M}_1, \dots, \mathbf{M}_m\} = \{\mathbf{M}_i \subseteq \mathbf{M} \mid \mathtt{TF}(\mathbf{M}_i) \vdash \phi$ and $\nexists \mathbf{M}'_i \subset \mathbf{M}_i$ s.t. $\mathtt{TF}(\mathbf{M}'_i) \vdash \phi\}$ and $\mathtt{min_w}(\mathbf{M}_i = \{(\psi_1, w_1), \dots, (\psi_l, w_l)\}) = \mathtt{min}(w_1, \dots, w_l)$.

Definition 3.3 (TMLN Instantiation). Given $\mathbf{M} = (\mathbf{F}, \mathbf{R}) \in$ `TMLN`, the set of *instantiations* `MI` of **M** is defined as follows:

$\mathtt{MI}(\mathbf{M}) = \mathbf{F} \cup \{((\phi'_1 \wedge \ldots \wedge \phi'_k \rightarrow \phi_{l+1})_{V \leftarrow C}, w') \mid \exists(\phi_1 \wedge \ldots \wedge \phi_l \rightarrow \phi_{l+1}, w) \in \mathbf{R}$ s.t. $\phi'_1 \wedge \ldots \wedge \phi'_k \vdash \phi_1 \wedge \ldots \wedge \phi_l, V = \{v_1, \ldots, v_n\}$ is the set of variables in $\phi_1 \wedge \ldots \wedge \phi_k \rightarrow \phi_{k+1}, C = \langle c_1, \ldots, c_n \rangle$ is a vector of constants replacing each occurrence of the variables, $V'_i \subseteq V$ is the set of variables in $\phi_i, C'_i \subseteq C$ is the vector of constants replaced in ϕ_i and the instantiated rule satisfies the 2 following conditions:

1. $\forall \phi'_i \in \{\phi'_1, \ldots, \phi'_k\}, \phi'_{iV'_i \leftarrow C'_i} \in \mathtt{Cn}(\mathtt{TF}(\mathbf{M}))$
2. $w' = \mathtt{min}(w, \mathtt{W}(\phi_{1V'_1 \leftarrow C'_1}, \mathbf{M}), \ldots, \mathtt{W}(\phi_{kV'_k \leftarrow C'_k}, \mathbf{M})\}$

Where $\phi_{V \leftarrow C}$ is the formula ϕ s.t. all the occurrences of the variable $v_i \in V$ are replaced by the constant $c_i \in C$.

Currently, we only deal with universal (*i.e.,* $\forall$) rules and no existential one (*i.e.,* $\exists$), to simplify the maximal TMLN instantiation. Indeed, with existential rules, we would have to deal with a set of sets of instantiations. Given that we would not know which set of instantiations would be true. We keep this question for future works.

From Example 2.4, the instantiations of R_3 are $GR_{31(2,\ldots,7)}$ where $(2, \ldots, 7)$ means that there are 6 ground rules $GR_{312}, \ldots, GR_{317}$ from the 6 possible temporal inferences of F_3: $Knowledge(Hypo_1, t_2, t_2)$, $\ldots, Knowledge(Hypo_1, t_7, t_7))$ and we do the same for $GR_{32(5,\ldots,10)}$ (from F_6). Hence $GR_{31(2,\ldots,7)}$ has a weight of 0.4 and $GR_{31(5,\ldots,10)}$ of 0.5. Ground rules instantiation is resumed in Table 2.

A TMLN instantiation $I \subseteq \mathtt{MI}(\mathbf{M})$ is a TMLN only composed of ground formulae, I is also called *a state of the TMLN* $\mathbf{M}$. The universe of all TMLN instantiations is denoted by $\mathtt{TMLN}^*$. Note that an instantiation can be inconsistent. In our example, GR_3, F_1, F_8 imply $\neg Knowledge(IllPatient_1, t_6, t_6)$ while GR_{21} (resp. GR_{22}) with F_3 (resp. F_6) imply $Knowledge(IllPatient_1, t_6, t_6)$, *i.e.*, they are inconsistent together. Thus, to obtain the most consistent and informative set of instantiations, we have to compute the *Maximum A-Posteriori* (MAP) inference [5, 17].

4 TEMPORAL AND UNCERTAIN KNOWLEDGE REASONING

Computing the MAP inference means "finding the most probable state of the world" [5]. Thus, we integrate semantics to TMLN, before examining principles on semantics and (in)consistency.

4.1 Temporal MAP Inference

Semantics compute the strength of a TMLN state. We denote the universe of all semantics by $\mathtt{Sem}$, such that for any $\mathcal{S} \in \mathtt{Sem}$, $\mathcal{S} : \mathtt{TMLN}^* \rightarrow [0, +\infty[$. We compute a strength above 0, instead of a probability between 0 and 1. One semantics may maximise the amount of information, while another may maximise the quality.

Temporal Maximum A-Posteriori (MAP) Inference in TMLN returns the most probable, temporally consistent, and expanded state *w.r.t.* a given semantics. Given $\mathbf{M} \in \mathtt{TMLN}$ and $\mathcal{S} \in \mathtt{Sem}$, a method solving a MAP problem is denoted by:

$\mathtt{map} : \mathtt{TMLN} \times \mathtt{Sem} \rightarrow \mathcal{P}(\mathtt{TMLN}^*)$, where $\mathcal{P}(X)$ denote the powerset of X, such that:

$$\mathtt{map}(\mathbf{M}, \mathcal{S}) = \{I \mid I \in \operatorname*{argmax}_{I \subseteq \mathtt{MI}(\mathbf{M})} \mathcal{S}(I) \text{ and } \nexists I' \in \operatorname*{argmax}_{I' \subseteq \mathtt{MI}(\mathbf{M})} \mathcal{S}(I') \text{ s.t. } I \subset I'\}.$$

To determine a MAP inference we need to define semantics, however, not all methods are desirable. Below we present some principles that semantics should satisfy.

4.2 Principles for semantics in Temporal MAP Inference

Our first principle states that adding new weightless information does not change the strength of the MAPs of a TMLN, whatever the temporality is. Given that the same set of predicates or formulae instantiated on different temporalities are not equivalent, we homogenise $\mathtt{TS\text{-}FOL}$ with the maximal time interval to define the information novelty.

For time homogenisation, we denote by $\tau : \mathcal{P}(\mathtt{TS\text{-}FOL}) \rightarrow \mathcal{P}(\mathtt{TS\text{-}FOL})$, the function transforming any temporal predicate into the maximal time interval (t_{min} and t_{max}).

Principle 1 (Temporal Neutrality). *Let* $\mathbf{M} \in \mathtt{TMLN}$, $\mathbf{M}' = \mathbf{M} \cup \{(\phi, w)\}$ *where* (ϕ, w) *is a weighted temporal formula* ($\phi \in \mathtt{TS\text{-}FOL}$ *and* $w \in [0, 1]$), *such that:*
- $\tau(\mathtt{TF}(\mathbf{M})) \nvdash \tau(\{\phi\})$, *and*
- $w = 0$.

A semantics $\mathcal{S} \in \mathtt{Sem}$ *satisfies temporal neutrality iff,* $\forall I \in \mathtt{map}(\mathbf{M}, \mathcal{S})$ *and* $\forall I' \in \mathtt{map}(\mathbf{M}', \mathcal{S})$, $\mathcal{S}(I) = \mathcal{S}(I')$.

The next principle ensures that one cannot decrease the strength of the MAPs of a TMLN by adding new and consistent information. The temporal consistency $\mathtt{Con} : \mathcal{P}(\mathtt{TS\text{-}FOL}) \rightarrow \{\top, \bot\}$ denotes a relation of consistence for a set of temporal formulae.

Principle 2 (Consistency Monotony). *Let a relation of consistence* $\mathtt{Con}$, $\mathbf{M} \in \mathtt{TMLN}$ *and* $\mathbf{M}' = \mathbf{M} \cup \{(\phi, w)\}$ *where* (ϕ, w) *is a weighted temporal formula such that:*
- $\tau(\mathtt{TF}(\mathbf{M})) \nvdash \tau(\{\phi\})$, *and*
- $\forall I \in \mathtt{map}(\mathbf{M}, \mathcal{S})$, $\mathtt{Con}(\{\phi\} \cup \mathtt{TF}(I))$ *is true and* $I \subset \mathtt{MI}(\{(\phi, w)\} \cup I)$.

A semantics $\mathcal{S} \in \mathtt{Sem}$ *satisfies consistency monotony iff:* $\forall I \in \mathtt{map}(\mathbf{M}, \mathcal{S})$ *and* $\forall I' \in \mathtt{map}(\mathbf{M}', \mathcal{S})$, $\mathcal{S}(I) \leq \mathcal{S}(I')$.

The last principle states that if we add a new temporal fact to a TMLN such that it is consistent with each instantiation, then the fact will be present in each new instantiation.

Principle 3 (Invariant Consistent Facts). *Let a relation of consistence* $\mathtt{Con}$, $\mathbf{M} \in \mathtt{TMLN}$ *and* $\mathbf{M}' = \mathbf{M} \cup \{(\phi, w)\}$ *where* (ϕ, w) *is a TMLN fact such that:*
- $\tau(\mathtt{TF}(\mathbf{M})) \nvdash \tau(\{\phi\})$, *and*
- $\forall I \in \mathtt{map}(\mathbf{M}, \mathcal{S})$, $\mathtt{Con}(\{\phi\} \cup \mathtt{TF}(I))$ *is true.*

A semantics $\mathcal{S} \in \mathtt{Sem}$ *satisfies invariant consistent facts iff,* $\forall I \in \mathtt{map}(\mathbf{M}, \mathcal{S})$, $I \cup \{(\phi, w)\} \in \mathtt{map}(\mathbf{M}', \mathcal{S})$.

4.3 Temporal Consistency and Inconsistency

We study here new temporal consistency interactions required to define our *Temporal MAP inference*. *Temporal Consistency* relations need to be refined according to the temporal validity of the predicates. For a predicate and its negation, no clear definition exists to express the temporal consistencies based on their time intervals. We propose a temporal consistency with a general case (*partial*) and a special case (*total*).

To establish these different temporal consistency relations, we introduce a function $\mathtt{TI}$ to create pre-orders between the temporal constants in the domain $Time$ of a $\mathtt{TS\text{-}FOL}$ and which extracts the interval of time points from two constants.

Definition 4.1 (Temporal (in)consistency). Let a set of formulae $\Phi \subseteq \mathtt{TS\text{-}FOL}$. In all the following notions of temporal consistency, we will exceptionally use the classical (non-temporal) logical consequence of $\mathtt{Cn}$ in order to work on the original maximal predicate interval (and not all its subsets).

Temporal consistency:

– Φ has a partial temporal consistency denoted by $\mathtt{pCon}(\Phi)$ iff: $\forall \phi, \psi \in \mathtt{Cn}(\Phi)$ s.t. $\phi = P(x_1, \ldots, x_k, t_1, t_1')$ and $\psi = \neg P(x_1, \ldots, x_k, t_2, t_2')$, $(\mathtt{TI}(t_1, t_1') \setminus \mathtt{TI}(t_2, t_2') \neq \emptyset) \wedge (\mathtt{TI}(t_2, t_2') \setminus \mathtt{TI}(t_1, t_1') \neq \emptyset)$.
Otherwise $\neg\mathtt{pCon}(\Phi)$ is true.

– Φ has a total temporal consistency denoted by $\mathtt{tCon}(\Phi)$ iff: $\forall \phi, \psi \in \mathtt{Cn}(\Phi)$ s.t. $\phi = P(x_1, \ldots, x_k, t_1, t_1')$ and $\psi = \neg P(x_1, \ldots, x_k, t_2, t_2')$, $(\mathtt{TI}(t_1, t_1') \cap \mathtt{TI}(t_2, t_2') = \emptyset)$.
Otherwise $\neg\mathtt{tCon}(\Phi)$ is true.

Temporal inconsistency:

– Φ has a partial temporal inconsistency denoted by $\mathtt{pInc}(\Phi)$ iff: $\exists \phi, \psi \in \mathtt{Cn}(\Phi)$ s.t. $\phi = P(x_1, \ldots, x_k, t_1, t_1')$, $\psi = \neg P(x_1, \ldots, x_k, t_2, t_2')$ and $\mathtt{TI}(t_1, t_1') \cap \mathtt{TI}(t_2, t_2') \neq \emptyset$.
Otherwise $\neg\mathtt{pInc}(\Phi)$ is true.

– Φ has a total temporal inconsistency denoted by $\mathtt{tInc}(\Phi)$ iff: $\exists \phi, \psi \in \mathtt{Cn}(\Phi)$ s.t. $\phi = P(x_1, \ldots, x_k, t_1, t_1'), \psi = \neg P(x_1, \ldots, x_k, t_2, t_2')$ and $(\mathtt{TI}(t_1, t_1') = \mathtt{TI}(t_2, t_2'))$.
Otherwise $\neg\mathtt{tInc}(\Phi)$ is true.

We now examine the interaction properties between $\mathtt{pCon}$, $\mathtt{tCon}$, $\mathtt{pInc}$ and $\mathtt{tInc}$, such as complementarity, subsumption and inclusion.

Definition 4.2 (Complementarity & Subsumption). $\forall \Phi \subseteq \mathtt{TS\text{-}FOL}$, $\forall$ relation r_1, r_2 if:

– $r_1(\Phi) \leftrightarrow \neg r_2(\Phi)$ then r_1 and r_2 are complementary.

– $r_1(\Phi) \rightarrow r_2(\Phi)$ then r_1 subsume r_2.

The next two propositions show that firstly $\mathtt{tCon}$ and $\mathtt{pInc}$ are complementary; secondly different subsumption relations exist between the temporal consistencies.

Proposition 4.3. *(Complementarity: temporal consistencies) For any $\Phi \subseteq \mathtt{TS\text{-}FOL}$:*
$\neg\mathtt{tCon}(\Phi) \leftrightarrow \mathtt{pInc}(\Phi)$ *and* $\mathtt{tCon}(\Phi) \leftrightarrow \neg\mathtt{pInc}(\Phi)$.

Proposition 4.4. *(Subsumption: temporal consisencies) For any $\Phi \subseteq \mathtt{TS\text{-}FOL}$:*
$\mathtt{pCon}(\Phi) \rightarrow \neg\mathtt{tInc}(\Phi)$, $\mathtt{tInc}(\Phi) \rightarrow \neg\mathtt{pCon}(\Phi)$,
$\neg\mathtt{pCon}(\Phi) \rightarrow \mathtt{pInc}(\Phi)$, *and* $\neg\mathtt{pInc}(\Phi) \rightarrow \mathtt{pCon}(\Phi)$.

In the following, for temporal consistency and inconsistency relations, we denote by $\{r\} = \{\Phi \subseteq \mathtt{TS\text{-}FOL} \mid r(\Phi)\}$ their set of sets of formulae respecting their condition, where $r \in \{\mathtt{pCon}, \mathtt{tCon}, \mathtt{pInc}, \mathtt{tInc}, \neg\mathtt{pCon}, \neg\mathtt{tCon}, \neg\mathtt{pInc}, \neg\mathtt{tInc}\}$.

Definition 4.5 (Inclusion). Let two relations of temporal consistency $r_1, r_2 \in \{\mathtt{pCon}, \mathtt{tCon}, \mathtt{pInc}, \mathtt{tInc}, \neg\mathtt{pCon}, \neg\mathtt{tCon}, \neg\mathtt{pInc}, \neg\mathtt{tInc}\}$, r_1 is considered included in r_2 if: $\{r_1\} \subseteq \{r_2\}$ iff $\forall \Phi \subseteq \mathtt{TS\text{-}FOL}$, $r_1(\Phi) \rightarrow r_2(\Phi)$.

Some inclusions of temporal consistency relations may be defined between the sets of sets of formulae respecting them.

Proposition 4.6. *(Inclusion: temporal consistencies)*
$\{\mathtt{tCon}\} = \{\neg\mathtt{pInc}\} \subseteq \{\mathtt{pCon}\} \subseteq \{\neg\mathtt{tInc}\}$
$\{\mathtt{tInc}\} \subseteq \{\neg\mathtt{pCon}\} \subseteq \{\mathtt{pInc}\} = \{\neg\mathtt{tCon}\}$

Links between different relations of temporal consistency are illustrated on Figure 1.

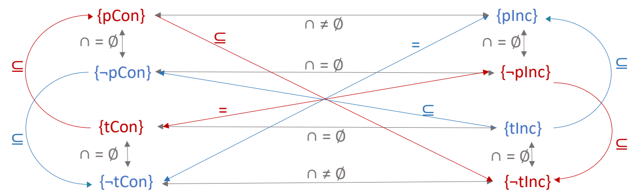

Figure 1: Links between Consistency and Inconsistency

4.4 Temporal Parametric Semantics

To avoid defining several different semantics, we decompose the construction of semantics and identify three steps. Then, we propose the definition of *Temporal Parametric Semantics*, relying on the combination of three functions: i) a *validation* function Δ of instantiations integrating various consistency relations, ii) a *selecting* function σ able to modify the weight of the formulae of an instantiation and iii) an *aggregate* function Θ returning the final strength.

Definition 4.7 (Temporal Parametric Semantics). A *temporal parametric semantics* is a tuple $\mathtt{TPS} = \langle \Delta, \sigma, \Theta \rangle \in \mathtt{Sem}$, *s.t.*:

– $\Delta : \mathtt{TMLN}^* \rightarrow \{0, 1\}$,

– $\sigma : \mathtt{TMLN}^* \rightarrow \bigcup_{k=0}^{+\infty} [0, 1]^k$,

– $\Theta : \bigcup_{k=0}^{+\infty} [0, 1]^k \rightarrow [0, +\infty[$,

For any $\mathbf{M} \in \mathtt{TMLN}$, $I \subseteq \mathtt{MI}(\mathbf{M})$, the strength of a temporal parametric semantics $\mathtt{TPS} = \langle \Delta, \sigma, \Theta \rangle$ is computed by:
$\mathtt{TPS}(I) = \Delta(I) \cdot \Theta\Big(\sigma(I)\Big)$.

We propose below key properties that must be satisfied by each of the three functions Δ, σ, Θ of a $\mathtt{TPS}$. Those properties constrain the range of functions to be considered, and discard those exhibiting undesired behaviours.

Definition 4.8. A temporal parametric semantics $\mathtt{TPS} = \langle \Delta, \sigma, \Theta \rangle$ is *well-behaved* according to a temporal consistency relation $\mathtt{Con}$ iff the following conditions hold:

Δ- **(a)** $\Delta(I) = 1$ if $\mathtt{Con}(\mathtt{TF}(I))$ is true (*i.e.,* I is temporally consistent w.r.t. $\mathtt{Con}$).

Θ- **(a)** $\Theta() = 0$.
(b) $\Theta(w) = w$.
(c) Θ is symmetric.
(d) $\Theta(w_1, \ldots, w_k) = \Theta(w_1, \ldots, w_k, 0)$.
(e) $\Theta(w_1, \ldots, w_k, y) \leq \Theta(w_1, \ldots, w_k, z)$ if $y \leq z$.

σ- **(a)** $\sigma() = ()$.
(b) $\sigma(\{(\phi_1, w_1), \ldots, (\phi_k, w_k)\}) = (w_1', \ldots, w_n')$ such that if $k \geq 1$ then $n \geq 1$.
(c) $\sigma(\{(\phi_1, w_1), \ldots, (\phi_k, w_k), (\phi_{k+1}, 0)\}) = (\sigma(\{(\phi_1, w_1), \ldots, (\phi_k, w_k)\}), 0)$, if $\tau(\mathtt{TF}(\{\phi_1, \ldots, \phi_k\})) \nvDash \tau(\{\phi\})$.
(d) $\sigma(\{(\phi_1, w_1), \ldots, (\phi_k, w_k)\}) \subset \sigma(\{(\phi_1, w_1), \ldots, (\phi_k, w_k), (\phi_{k+1}, w_{k+1})\})$ if ϕ_{k+1} is a ground temporal formula, $\tau(\mathtt{TF}(\{\phi_1, \ldots, \phi_k\})) \nvDash \tau(\{\phi_{k+1}\})$ and $\mathtt{Con}(\mathtt{TF}(\{(\phi_1, w_1), \ldots, (\phi_k, w_k), (\phi_{k+1}, w_{k+1})\}))$.
(e) $\Theta\Big(\sigma(\{(\phi_1, w_1), \ldots, (\phi_k, w_k)\})\Big) \leq \Theta\Big(\sigma(\{(\phi_1, w_1), \ldots, (\phi_k, w_k), (\phi_{k+1}, w_{k+1})\})\Big)$ if $\tau(\mathtt{TF}(\{\phi_1, \ldots, \phi_k\})) \nvDash \tau(\{\phi_{k+1}\})$ and $\mathtt{Con}(\mathtt{TF}(\{(\phi_1, w_1), \ldots, (\phi_k, w_k), (\phi_{k+1}, w_{k+1})\}))$.

We also say that Δ is *well-behaved* according to $\mathtt{Con}$, and Θ, σ are *well-behaved*.

THEOREM 4.9. *Any temporal parametric semantics well-behaved wrt. a temporal consistency relation* $\mathtt{Con}$ *satisfies the principles from Section 4.2: Temporal Neutrality, Consistency Monotony and Invariant Consistent Facts (the last two according to* $\mathtt{Con}$*).*

Once temporal consistency relations are defined, we may enhance semantics for MAP inference with temporal validation functions. One TMLN instantiation can be valid or not according to different criteria (*i.e.,* accept an instantiation).

Definition 4.10 (Temporal Consistency Validation Function). Let $\mathbf{M} \in \mathtt{TMLN}$, an instantiation $I \subseteq \mathtt{MI}(\mathbf{M})$ and $x \in \{\mathtt{pCon}, \mathtt{tCon}, \mathtt{pInc}, \mathtt{tInc}\}$. We define $\Delta_x : \mathtt{TMLN}^* \longrightarrow \{0, 1\}$, a temporal consistency validation function according to x, s.t.:

$$\Delta_{\mathtt{pCon}}(I) = \begin{cases} 1 & \text{if } \mathtt{pCon}(\mathtt{TF}(I)) \\ 0 & \text{if } \neg\mathtt{pCon}(\mathtt{TF}(I)) \end{cases} \quad \Delta_{\mathtt{tCon}}(I) = \begin{cases} 1 & \text{if } \mathtt{tCon}(\mathtt{TF}(I)) \\ 0 & \text{if } \neg\mathtt{tCon}(\mathtt{TF}(I)) \end{cases}$$

$$\Delta_{\mathtt{pInc}}(I) = \begin{cases} 0 & \text{if } \mathtt{pInc}(\mathtt{TF}(I)) \\ 1 & \text{if } \neg\mathtt{pInc}(\mathtt{TF}(I)) \end{cases} \quad \Delta_{\mathtt{tInc}}(I) = \begin{cases} 0 & \text{if } \mathtt{tInc}(\mathtt{TF}(I)) \\ 1 & \text{if } \neg\mathtt{tInc}(\mathtt{TF}(I)) \end{cases}$$

COROLLARY 4.11. *For any* $I \subseteq \mathtt{TMLN}^*$, $\Delta_{\mathtt{tCon}}(I) = \Delta_{\mathtt{pInc}}(I)$.

COROLLARY 4.12. *Let* $x \in \{\mathtt{pCon}, \mathtt{tCon}, \mathtt{pInc}, \mathtt{tInc}\}$*, each* Δ_x *is well-behaved.*

Then, we can order the value of the Δ_x for any instantiation.

PROPOSITION 4.13. *Let* $\mathbf{M} \in \mathtt{TMLN}$ *and* Δ_x *a temporal consistency validation function such that* $x \in \{\mathtt{pCon}, \mathtt{tCon}, \mathtt{pInc}, \mathtt{tInc}\}$. *For any instantiation* $I \subseteq \mathtt{MI}(\mathbf{M})$*:*
$\Delta_{\mathtt{tCon}}(I) = \Delta_{\mathtt{pInc}}(I) \leq \Delta_{\mathtt{pCon}}(I) \leq \Delta_{\mathtt{tInc}}(I)$

Theorem 4.14 shows that the strength of the temporal MAP inferences with σ and Θ on any TMLN is ranked according to the temporal consistency validation functions Δ_x.

THEOREM 4.14. *Let* $\mathbf{M} \in \mathtt{TMLN}$*, for any* σ *and* Θ*, as:*
– $\mathtt{TPS}_{\mathtt{tCon}} = \langle \Delta_{\mathtt{tCon}}, \sigma, \Theta \rangle$, $\mathtt{TPS}_{\mathtt{pInc}} = \langle \Delta_{\mathtt{pInc}}, \sigma, \Theta \rangle$,
– $\mathtt{TPS}_{\mathtt{pCon}} = \langle \Delta_{\mathtt{pCon}}, \sigma, \Theta \rangle$, $\mathtt{TPS}_{\mathtt{tInc}} = \langle \Delta_{\mathtt{tInc}}, \sigma, \Theta \rangle$.
Hence: $\forall I_{\mathtt{tCon}} \in \mathtt{map}(\mathbf{M}, \mathtt{TPS}_{\mathtt{tCon}})$, $\forall I_{\mathtt{pInc}} \in \mathtt{map}(\mathbf{M}, \mathtt{TPS}_{\mathtt{pInc}})$, $\forall I_{\mathtt{pCon}} \in \mathtt{map}(\mathbf{M}, \mathtt{TPS}_{\mathtt{pCon}})$, $\forall I_{\mathtt{tInc}} \in \mathtt{map}(\mathbf{M}, \mathtt{TPS}_{\mathtt{tInc}})$,
$\mathtt{TPS}_{\mathtt{tCon}}(I_{\mathtt{tCon}}) = \mathtt{TPS}_{\mathtt{pInc}}(I_{\mathtt{pInc}}) \leq \mathtt{TPS}_{\mathtt{pCon}}(I_{\mathtt{pCon}}) \leq \mathtt{TPS}_{\mathtt{tInc}}(I_{\mathtt{tInc}})$.

We study different instances of the aggregate function, using different sums. This type of parameters will determine the strength of an instantiation, in various ways.

Definition 4.15 (Aggregate Functions). Let $\{w_1, \ldots, w_n\}$ such that $n \in [0, +\infty[$ and $\forall i \in [0, n]$, $w_i \in [0, +\infty[$.

– $\Theta_{sum}(w_1, \ldots, w_n) = \sum_{i=1}^{n} w_i$, if $n = 0$ then $\Theta_{sum}() = 0$.

– $\Theta_{sum,\alpha}(w_1, \ldots, w_n) = \left(\sum_{i=1}^{n} (w_i)^{\alpha} \right)^{\frac{1}{\alpha}}$ s.t. $\alpha \geq 1$, if $n = 0$ then $\Theta_{sum,\alpha}() = 0$.

Those aggregate functions target different kinds of semantics. For instance, $\Theta_{sum,\alpha}$ emphasises strong weights for inference, while Θ_{sum} considers each weight without appriori.

PROPOSITION 4.16. *The two functions* Θ_{sum} *and* $\Theta_{sum,\alpha}$ *are well-behaved.*

We propose below a selective function σ_{id} which returns all the weights and another function selecting weights with a threshold ($\sigma_{thresh,\alpha}$).

Definition 4.17 (Selective Functions). Let $\mathbf{M} \in \mathtt{TMLN}$, $\{(\phi_1, w_1), \ldots, (\phi_n, w_n)\} \subseteq \mathtt{MI}(\mathbf{M})$:

- $\sigma_{id}(\{(\phi_1, w_1), \ldots, (\phi_n, w_n)\}) = (w_1, \ldots, w_n)$
- $\sigma_{thresh,\alpha}(\{(\phi_1, w_1), \ldots, (\phi_n, w_n)\}) = (\mathtt{max}(w_1 - \alpha, 0), \ldots, \mathtt{max}(w_n - \alpha, 0))$ s.t. $\alpha \in [0, +\infty[$

PROPOSITION 4.18. *The two functions* σ_{id} *and* $\sigma_{thresh,\alpha}$ *are well-behaved.*

In [5], the MAP inference uses a semantics working on $\mathtt{Herbrand}$ models (containing temporal formulae, included in $\mathtt{TS\text{-}FOL}$) and built from uncertain ($w < 1$) and certain ($w = 10^{10}$) temporal facts and with a set of TMLN certain rules. This semantics also determines the temporal inconsistency by a classical consistency (if there is no formula φ such that $\Phi \vdash \varphi$ and $\Phi \vdash \neg\varphi$) and by summing all the weights of facts in the instantiation. Therefore, for TMLNs without any uncertain rule, the MAP inference will return the same instantiations as ours, using the temporal parametric semantics $\langle \Delta_{\mathtt{tInc}}, \sigma_{id}, \Theta_{sum} \rangle$. Our Temporal MAP inference generalises their work.

5 COMPUTING THE TEMPORAL MAP INFERENCE

In literature, computing the MAP inference relies on a Data Mining inference process which checks if rules can be applied on possible grounded facts [7] (*i.e.*, finding models). This heavy process is optimised by aggregating formulae sharing common predicates, but also by parallelising mining. Those approaches' complexity is highly dependent on the number of possibilities. We propose a totally different approach based on conflicts produced by rules.

Our approach relies on building compatible worlds instead of mining valid worlds. To achieve this, we extract a conflict graph between facts based on rules as in [4] but with weighted nodes. Thus, the MAP inference tries combinations of non-conflict graphs, however, it requires to maximise node weights and not to minimise conflicts pruning [12].

5.1 Graph of Conflicts

For conflicts extraction, we propose an approach based on a property graph which represents the TMLN instantiation I. The representation is a mapping[1] between TMLN and property graphs where: Constants $\mathbf{C}$ and predicates $\mathbf{P}$ are *Concept nodes* linked by formulae ϕ as *TS nodes*; Temporal predicates and weights correspond to properties of *TS nodes*; relationships give the links with *concepts*.

Figure 2 illustrates an instantiation of a TMLN where concept (blue) and TS (red) nodes are represented. The subject "*knowledge*" (subject *:s*) is a "*is*" (predicate *:p*) linked to "*Hypo1*" (object *:o*). The three TS-formulae (in red) correspond to information grounded from Ex. 2.4 with corresponding timeframes and polarities.

Applying semantics corresponds to a pattern matching on the graph. It searches for conflicts between TS nodes. Each rule (*i.e.,* temporal consistency) is a pattern which starts with a same concept node to focus only on conflictual nodes. For our example, we

[1] For space reason, the mapping is not formalised but remains direct.

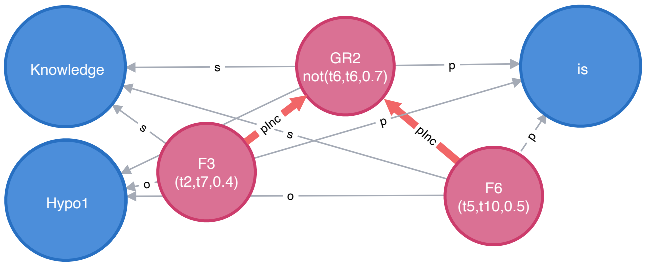

Figure 2: TMLN's property graph representation

start with the object node "*Hypo1*" for whom we can find three TS with opposite polarities while sharing the same concepts (subject and predicate). Since their timeframes ($[t_2, t_7]$ and $[t_5, t_{10}]$ against $[t_6, t_6]$) are not disjoint we can infer `pInc` and `tCon` or because one is included in the other we can infer `pCon` as conflict. The type of conflict becomes a property on the extracted conflict link.

Thus, after applying all rules (pattern matching) all conflicts are identified. We provide for each TS node the list of conflictual nodes according to a given parametric semantics.

5.2 Searching 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) we process the MAP inference with the `MaPy` algorithm (Alg. 1).

`MaPy` works as follows: for each node in a dictionary of conflictual nodes (i.e., a connected component), we create step by step the list of possible solutions. Note that a solution has three elements: *1)* the list of "solution" nodes, *2)* the list of conflict nodes with the solution and *3)* the solution's weight.

For each node (line 2), we search for each an existing solution when the node is compatible (line 4). In that case, the node is merged to the solution (line 6). Otherwise, we extract the maximum sub-solution that is compatible with this node and merge them together as a new solution (line 8).

To optimise the previous step: *1)* we remove any solution that can be included in another one (line 9); *2)* we look for the maximum potential solution starting from the best current solution and naively add a maximum of remaining nodes to create a maximum potential solution (MPS - line 10); *3)* then, we augment each solution with all the nodes that do not conflict with the base solution (worst case) and if it is lower than the MPS, we delete wrong solutions (line 11); *4)* Finally, we limit the result set to a list of the k best solutions (k=100 in our experiments), to avoid useless solutions and earn time (line 12-13). After processing all the nodes, each solution is checked for potential missed compatible nodes (line 14). Then best solution is kept among the top-k (line 15).

6 EXPERIMENTS

Our approach has been implemented in two distinct settings. The first part builds and extracts the graph of conflicts in `Neo4j`[2], and the second one process the MAP inference in `MaPy` developed in Python. The source code and datasets are available on GitHub[3].

Algorithm 1 MaPy(Dico, k)

Input: Dico = {IdNode: [W_Node, [ConfNode, ...]], ...}, $k \in \mathbb{N}$
Output: Best_Solution = [{Node, ...}, {ConfNode, ...}, W_Sol]

```
1:  List_Sol = [set(NodeDico[0]), set(ConfDico[0]), W_Dico[0]];
2:  for node ∈ Dico do
3:      for sol ∈ List_Sol do
4:          (new_sol,compatible) = Compatible_Merge(node,sol);
5:          if compatible then
6:              sol.update(node);
7:          else
8:              List_Sol.add(new_sol);
9:      List_Sol.delete_Include();
10:     Max_Potential_Sol = search_MPS(List_Sol);
11:     List_Sol.delete_Wrong_Solutions(Max_Potential_Sol);
12:     if len(List_Sol) > k then
13:         List_Sol = Top_MAP(List_Sol, k);
14: List_Sol.add_Missing_Nodes(Dico);
15: Best_Solution = Top_MAP(List_Sol, 1);
```

6.1 NeoMaPy: a MAP Inference Extractor

Conflicts extraction from TS nodes has been implemented over `Neo4j`. Facts are imported from CSV files. The graph is composed of `Concept` and `TS` nodes (see Section 5.1) with (s,o,p) relationships.

Rules are applied to instantiate ground rules as *Cypher* queries. The rule R_2 (Table 1) is illustrated below. It searches for a "*Knowledge*" predicate (cp) linked to "`OnsetIllPatient1`" and "`IllPatient1`" subjects by *ts1* and *ts2* formulae. If the polarity of "*ts2*" is false and occurred at *ts1* time plus 5 (ts1.start=ts2.start+5), then when can infer a new TS formula "*new_ts*" which *knowledge* (cp) is the object "*Hypo1*". Interestingly, we can track of the inference with a "`rule`" relationship between *ts1/ts2* (the premise) and *new_ts*.

```
MATCH p1=(cp:Concept{ID:"Knowledge"}),
  (cp) <-[:p]-(ts1:TS)-[:s]-> (cs1:Concept{ID:"OnsetIllPatient1"}),
  (cp) <-[:p]-(ts2:TS)-[:s]-> (cs2:Concept{ID:"IllPatient1"})
WHERE ts2.polarity = false and ts1.start = ts2.start+5
MERGE (cp) <-[:p]- (new_ts:TS{start:ts2.start, end:ts2.start,
  polarity:false,weight:0.7}) -[:s]-> (cs2)
MERGE (new_ts) -[:o]-> (:Concept{ID:"Hypo1"})
MERGE (ts1) -[:rule{ID:"R_2"}]-> (new_ts)
MERGE (ts2) -[:rule{ID:"R_2"}]-> (new_ts);
```

Conflicts are then instantiated as "*conflict*" relationships on the graph by searching for `TS` with corresponding rules. The *Cypher* query below illustrates the generation of conflicts for the `pCon` rule. If two *TS* ts1 and ts2 share same concepts s, o and p with opposite polarities and an intersection of timeframes, it produces a conflict of type "*pCon*" between ts1 and ts2. For optimisation purpose, concept IDs are repeated in TS nodes (*e.g.,* ts1.p=ts2.p).

```
MATCH (ts1:TS) -[:s]-> (:Concept) <-[:s]- (ts2:TS)
WHERE ts1.p=ts2.p AND ts1.o=ts2.o AND
  ts1.polarity <> ts2.polarity AND  (ts1.start < ts2.start AND
  ts2.start < ts1.end AND ts1.end < ts2.end)
MERGE (ts1)-[c:conflict]-(ts2) SET c.type="TC", c.pCon=true;
```

We must notice that each conflict and inference rules relationship are typed. The resulting graph can be used for explainability of the

[2] https://neo4j.com

[3] ANONYMIZED

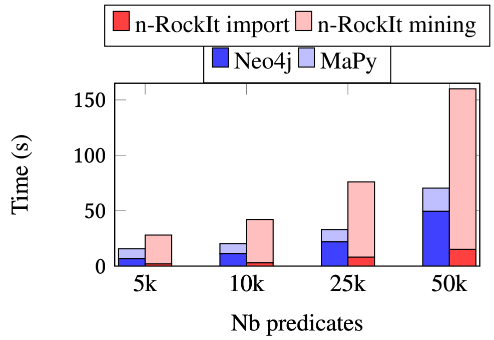

Figure 3: MAP Inference wrt. Nb of predicates

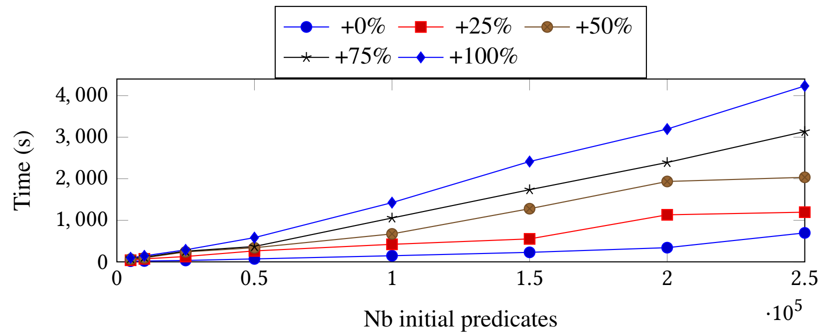

Figure 4: MAP Inference Computation wrt. Injected Conflicts

MAP inference or traceability purposes. Moreover, for inference a parametric semantics corresponds to a Cypher query that extracts corresponding conflicts (temporal consistency), rules (if a premise is ignored, so does the inferred TS), thresholds (filter on weights), etc.

6.2 Performances

Experiments have been computed on an Intel Xeon-E 2136 - 6c/12t - 3.3 GHz/4.5 GHz with 64GB RAM. For the setting, *Neo4j* has been put in a *Docker* container with 4 cores and 60GB RAM.

As presented previously, Chekol et al. [7] have proposed an approach to compute the MAP inference with a parallelised data mining process with *TeCoRe* over *n-RockIt*. We compare our approach with them using the dataset they provided in [5]. It is composed of different datasets from 5k to 250k predicates (TS) and by adding false predicates (from +25% to +100%). For the comparison, we only focus on a TMLN without polarities and a MAP inference without parametric semantics while our implementation allows it.

Figure 3 shows the performance of *n-RockIt* vs *NeoMaPy*. We differentiate the preprocessing (import / neo4j) from the MAP inference computation (mining / MaPy). We can see that *n-RockIt* spends most of the time to search for the MAP inference while *NeoMaPy* spend more time creating the conflict graph. Our solution outperforms *n-RockIt*. Moreover, we must notice that processing graphs bigger than 50k predicates on *n-RockIt* led to a time-out (after 1 hour delay).

Figure 4 shows the computation time of *NeoMaPy* on graphs from 5k to 250k predicates and by adding false predicates (from +0% to +100%). Those false predicates produce each time at least one conflict with existing nodes. For example, the graph with 250k predicates with +100% is a graph with 500k predicates and 39,497,681 conflicts. It is worth noting that our approach keeps an almost linear computation time. The difference of time between datasets with various injected conflicts is due to the fact that resulting graphs are bigger and thus leads to proportional computation growth.

6.3 Example of Reasoning on TMLN

Example 2.4 (Continued).

Let us study the goal recognition problem between *Submit Diagnosis* (SD) and *Find Clue* (FC) at time t_6 as it shows the value of reasoning on the data. Notice that SD goal inference depends on the validity of the *Hypothesis1* while FC relies on a certain fact. Thus, we have to solve a MAP inference problem which can be simplified to the acceptance of $F_3 + F_6$ (valid Hypo1) or GR_2 (invalid). We propose different semantics capturing different strategies.

In this example we take the identity selection function σ_{id} (threshold σ_{thresh} is generally more useful for TS with low degrees - which is not the case here). As for the consistency functions: in the case of `tInc`, which does not have an identical temporality, $F_3 + F_6$ are always consistent with GR_2 and *Hypo1* is valid. According to `pInc` (resp. `pCon` and `tCon`) they are inconsistent because the temporality of GR_2 is included in those of F_3 and F_6. Thus, besides the case of classical inconsistency `tInc`, other temporal semantics requires to study different aggregation function strategies. For instance, with Θ_{sum} $Hypo_1$ is valid because $0.4 + 0.5 > 0.7$, while with $\Theta_{sum,2}$ $Hypo_1$ is not valid because $\sqrt{(0.4^2 + 0.5^2)} \approx 0.64 < \sqrt{0.7^2}$, *i.e.,* higher weights have more impact. Therefore, SD is the most likely goal given by $\langle \Delta_{\texttt{pInc}}, \sigma_{id}, \Theta_{sum} \rangle$ and FC for $\langle \Delta_{\texttt{pInc}}, \sigma_{id}, \Theta_{sum,2} \rangle$. As a result, users choose the TPS for reasoning in various ways (strong probabilities, more incertain clues, etc.)

7 CONCLUSION

Reasoning on Knowledge Graphs has recently been approached with MLNs, to find the most probable state of the world. However, they were limited to a strict temporal inconsistency. We propose 1) to extend MLN by TMLN with sorted information which is capable of combining temporal facts and rules, 2) to extend MAP inference semantics to temporal data with principles to be respected, 3) we introduce temporal parametric semantics (TPS) which offer flexibility and explainability to tailor semantics reasoning to one's needs, 4) finally, we propose a new `NeoMaPy` tool that is faster than the best existing one (n-RockIt) with different TPS.

As a perspective, we first wish to extend our reasoning model with a learning system to automatically obtain weights on TMLN information and then test them in practice for goal recognition problems. Secondly, we want to extend this work to existential rules in order to capture more real data. Thirdly, some relationships between formulae in a TMLN are not yet exploited and could be represented with argumentation graphs (*e.g.,* the notion of support [8] or similarity between pieces of information [1, 2]), to enhance the weight of inferred knowledge. Fourthly, it will be useful to investigate some properties to describe the strategies and specific behaviours of the parametrisation functions.

REFERENCES

[1] Leila Amgoud and Victor David. 2018. Measuring Similarity between Logical Arguments. In *Proceedings of the Sixteenth International Conference on Principles of Knowledge Representation and Reasoning KR*. 98–107.

[2] Leila Amgoud and Victor David. 2021. A General Setting for Gradual Semantics Dealing with Similarity. In *35th AAAI Conference en Artificial Intelligence (AAAI 2021)*. AAAI : Association for the Advancement of Artificial Intelligence, AAAI Press, Virtual Conference, United States. https://hal.archives-ouvertes.fr/hal-03141729

[3] Alok Baikadi, Jonathan Rowe, Bradford Mott, and James Lester. 2014. Generalizability of goal recognition models in narrative-centered learning environments. In *International Conference on User Modeling, Adaptation, and Personalization*. Springer, 278–289.

[4] L. Bertossi. 2011. *Database Repairs and Consistent Query Answering*. Morgan & Claypool Publishers. https://books.google.fr/books?id=leVcAQAAQBAJ

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

[6] Melisachew Wudage Chekol, Jakob Huber, Christian Meilicke, and Heiner Stuckenschmidt. 2016. Markov Logic Networks with Numerical Constraints. In *Proceedings of the Twenty-Second European Conference on Artificial Intelligence* (The Hague, The Netherlands) *(ECAI'16)*. IOS Press, NLD, 1017–1025.

[7] Melisachew Wudage Chekol, Giuseppe Pirro, Joerg Schoenfisch, and Heiner Stuckenschmidt. 2017. TeCoRe: temporal conflict resolution in Knowledge Graphs. *Proceedings of the VLDB Endowment* 10 (2017), Iss–12.

[8] Andrea Cohen, Sebastian Gottifredi, Alejandro J García, and Guillermo R Simari. 2014. A survey of different approaches to support in argumentation systems. *The Knowledge Engineering Review* 29, 5 (2014), 513–550.

[9] Jean H Gallier. 2015. *Logic for computer science: foundations of automatic theorem proving*. Courier Dover Publications.

[10] Eun Young Ha, Jonathan P Rowe, Bradford W Mott, and James C Lester. 2011. Goal recognition with Markov logic networks for player-adaptive games. In *Seventh Artificial Intelligence and Interactive Digital Entertainment Conference*.

[11] Eun Y Ha, Jonathan P Rowe, Bradford W Mott, James C Lester, G Sukthankar, R Goldman, C Geib, D Pynadath, and H Bui. 2014. Recognizing player goals in open-ended digital games with markov logic networks. *Plan, Activity and Intent Recognition: Theory and Practice* (2014), 289–311.

[12] Keith W. Hipel, Liping Fang, and D. Marc Kilgour. 2020. The Graph Model for Conflict Resolution: Reflections on Three Decades of Development. *Group Decision and Negotiation* 29, 1 (2020), 11–60. https://doi.org/10.1007/s10726-019-09648-z

[13] James C Lester, Eun Y Ha, Seung Y Lee, Bradford W Mott, Jonathan P Rowe, and Jennifer L Sabourin. 2013. Serious games get smart: Intelligent game-based learning environments. *AI Magazine* 34, 4 (2013), 31–45.

[14] Wookhee Min, Eun Young Ha, Jonathan Rowe, Bradford Mott, and James Lester. 2014. Deep learning-based goal recognition in open-ended digital games. In *Tenth artificial intelligence and interactive digital entertainment conference*.

[15] Wookhee Min, Bradford W Mott, Jonathan P Rowe, Barry Liu, and James C Lester. 2016. Player Goal Recognition in Open-World Digital Games with Long Short-Term Memory Networks.. In *IJCAI*. 2590–2596.

[16] Matthew Richardson and Pedro Domingos. 2006. Markov Logic Networks. *Machine Learning* 62, 1 (01 Feb 2006), 107–136. https://doi.org/10.1007/s10994-006-5833-1

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

[18] Romain Rincé, Romain Kervarc, and Philippe Leray. 2018. Complex Event Processing Under Uncertainty Using Markov Chains, Constraints, and Sampling. In *Rules and Reasoning*, Christoph Benzmüller, Francesco Ricca, Xavier Parent, and Dumitru Roman (Eds.). Springer, Cham, 147–163.

[19] Jonathan Rowe, Bradford Mott, Scott McQuiggan, Jennifer Robison, Sunyoung Lee, and James Lester. 2009. Crystal island: A narrative-centered learning environment for eighth grade microbiology. In *workshop on intelligent educational games at the 14th international conference on artificial intelligence in education, Brighton, UK*. 11–20.

[20] Hao Wang. 1952. Logic of many-sorted theories. *The Journal of Symbolic Logic* 17, 2 (1952), 105–116.