Probabilistic Logic Networks for Temporal and Procedural Reasoning

Nil Geisweiller and Hedra Yusuf

SingularityNET Foundation, The Netherlands
{nil,hedra}@singularitynet.io

Abstract. Probabilistic Logic Networks (PLN) offers an excellent theory to frame learning and planning as a form of reasoning. This paper offers a complement to the seminal PLN book [3], in particular to its Chapter 14 on temporal and procedural reasoning, by providing formal definitions of temporal constructs, as well as inference rules necessary to carry temporal and procedural reasoning.

Keywords: Temporal Reasoning · Procedural Reasoning · Probabilistic Logic Networks

1 Introduction

This paper builds upon the Chapter 14 of the Probabilistic Logic Networks book [3], adding and modifying definitions along the way to provide, we believe, a better foundation for carrying temporal and procedural reasoning with PLN. As we have found, even though the chapter is well written and conveys the conceptual ideas with clarity, it leaves some formal definitions out. In addition the Event Calculus [8] is intermingled with the definitions of sequential connectors in, what we consider to be, an arbitrary and inflexible manner. On the contrary, here we leave Event calculus aside, with the intention to re-introduce it in the future as a separate layer standing on top of the new definitions.

Although this paper is theoretical, the work presented here is motivated by practice, and has taken place in the context of developing a system for controlling an agent in uncertain environments while relying on temporal and procedural reasoning for both learning and planning [2].

2 Probabilistic Logic Networks Recall

PLN stands for *Probabilistic Logic Networks* [3]. It is a mixture of predicate and term logic that has been probabilitized to handle uncertainty. Inference rules can operate on direct evidence, or indirect evidence by combining existing relationships to introduce new ones. As such it is well suited for building a model of an environment, and planning in it. All it needs then is to be properly equipped with a vocabulary for representing and manipulating temporal and procedural knowledge.

2.1 Elementary Notions

Graphically speaking, PLN statements are sub-hypergraphs[1] made of *Links* and *Nodes*, called *Atoms*, decorated with *Truth Values*. Syntactically speaking, PLN statements are not very different from statements expressed in another logic, except that they are usually formatted in prefixed-operator with a tree-style indentation to emphasize their graphical nature and to leave room for their truth values. For instance

$$\begin{array}{l} \mathit{Implication}\ \langle TV \rangle \\ \quad P \\ \quad Q \end{array}$$

represents an implication link between P and Q with truth value TV. For the sake of conciseness we also introduce some notations. First, we adopt a flattened, as opposed to a tree-style, representation. For instance the implication link above is represented as

$$\mathit{Implication}(P, Q)\ \langle TV \rangle$$

Second, we introduce a more mathematically looking symbolic representation. For instance, that same implication can be represented as

$$P \rightarrow Q \stackrel{m}{=} TV$$

There is a large variety of constructs in PLN. Here, we will focus primarily on the ones for handling predicates. Let us recall that predicates are functions that output Boolean values. The domain of a predicate can be arbitrarily defined, but its range is always Boolean. In this paper, the letters a, b, c represent atoms of any type, x, y, z represent atoms that are variables, while the capital letter P, Q, R represent atoms that are predicates, thus typed as follows:

$$P, Q, R, \ldots : \mathit{Domain} \mapsto \{\mathit{True}, \mathit{False}\}$$

Note that in PLN, predicates are not necessarily crisp because their outputs can be totally or partially unknown, thus potentially measured by probabilities, or to be precised *Truth Values*.

Truth values are, in essence, second order probability distributions, or probabilities of probabilities. They are often described by two numbers: a strength, s, representing a probability, and a confidence, c, representing the confidence over that probability. Such truth values are called *Simple Truth Values* and are denoted as follows:

$$< s, c >$$

Alternatively, the strength and the confidence of a simple truth value TV can be denoted $TV.s$ and $TV.c$ respectively. Underneath, a simple truth value is a beta distribution [1], similarly to an *opinion* in Subjective Logic [5]. The parameters of the corresponding beta distribution can be obtained as follows:

$$\alpha(s, c) = \alpha_0 + \frac{s.c.k}{1 - c} \qquad \beta(s, c) = \beta_0 + \frac{(1 - s).c.k}{1 - c}$$

[1] because links can point to links, not just nodes

where k is a PLN parameter called the *Lookahead*, and α_0 and β_0 are usually set to 0.5 corresponding to Jeffreys prior. For truth values obtained from direct evidence, a simple truth value makes perfect theoretical sense. For truth values obtained from indirect evidence, not so much, even though they are often used in practice. When more precision is needed, to represent a multi-modal truth value for instance, a mixture of simple truth values can be used. Also, through out the paper, sometimes we may say *probability*, while what we really mean is *second order probability distribution*.

Below is a table of the constructs used in this paper with their flattened and symbolic representations, as well as precedence values to minimize parenthesis usage with the symbolic representation.

Flattened	Symbolic	Precedence
$\mathit{Evaluation}(P, a)$	$P(a)$	0
$\mathit{Not}(P)$	$\neg P$	1
$\mathit{And}(P, Q)$	$P \wedge Q$	2
$\mathit{Or}(P, Q)$	$P \vee Q$	2
$\mathit{Implication}(P, Q)$	$P \rightarrow Q$	4
$a\langle TV\rangle$	$a \stackrel{m}{=} TV$	5

For representing n-ary predicates evaluations we use $P(a_1, \ldots, a_n)$ which may be understood as a unary predicate evaluation applied to a tuple. Let us now explain their semantics and how their truth values are to be interpreted.

- $\neg P$ is the predicate resulting from the pointwise negation of P.
- $P \wedge Q$ is the predicate resulting from the pointwise conjunction of P and Q.
- $P \vee Q$ is the predicate resulting from the pointwise disjunction of P and Q.
- $P(a) \stackrel{m}{=} TV$ states that $P(a)$ outputs *True* with a second order probability measured by TV.
- $P \rightarrow Q \stackrel{m}{=} TV$ states that if $P(a)$ is *True* for some a in the domain of P, then $Q(a)$ is *True* with a second order probability measured by TV. In simple probability terms, it represents $\mathcal{P}r(Q|P)$, the conditional probability of Q knowing P [2]. We may also say that such implication is a *conditional predicate* where Q, the *implicand*, is conditioned by P, the *implicant*.
- $P \stackrel{m}{=} TV$ states that the prevalence of P being *True* is measured by TV.

2.2 Inference Rules

Inferences rules are used to construct PLN statements and calculate their truth values. They fall into two groups, direct evidence based or otherwise. Rules from

[2] To be precise, $\mathcal{P}r(Q|P)$ should be $\mathcal{P}r(\mathcal{S}at(Q)|\mathcal{S}at(P))$, where $\mathcal{S}at(P)$ and $\mathcal{S}at(Q)$ are the satisfying sets of P and Q respectively.

the former group infer abstract knowledge from direct evidence, while rules from the latter group infer knowledge by combining existing abstractions. In total there are dozens of inference rules. For now, we only recall two, *Implication Direct Introduction* and *Deduction*.

The Implication Direct Introduction Rule (IDI) takes evaluations as premises and produces an implication as conclusion. It can be understood as an inductive reasoning rule. It is formally depicted by the following proof tree.

$$\frac{P(a_1) \stackrel{m}{=} TV_1^P \qquad Q(a_1) \stackrel{m}{=} TV_1^Q \qquad \dots \qquad P(a_n) \stackrel{m}{=} TV_n^P \qquad Q(a_n) \stackrel{m}{=} TV_n^Q}{P \rightarrow Q \stackrel{m}{=} TV} \text{ (IDI)}$$

Assuming perfectly reliable direct evidence[3] then the resulting simple truth value can be calculated as follows:

$$TV.s = \frac{\sum_{i=1}^{n} f_{\wedge}(TV_i^P.s, TV_i^Q.s)}{\sum_{i=1}^{n} TV_i^P.s} \qquad\qquad TV.c = \frac{n}{n+k}$$

where $f_{\wedge}$ is a function embodying a probabilistic assumption about the conjunction of the events. Such function typically ranges from the product (perfect independence) to the min (perfect overlap). Note that this inference rule takes an arbitrary number of premises. In practice it is not a problem as it is decomposed into two rules covering the base and the recursive cases, while storing evidence to avoid double counting.

The Deduction Rule (D) takes two implications as premises and produces a third one. It can be understood as a deductive reasoning rule. Depending on the assumptions made there exists different variations of that rule. The simplest one is based on the Markov property

$$\mathcal{P}r(R|Q,P) = \mathcal{P}r(R|Q)$$

which gives rise to the rule depicted by the following proof tree.

$$\frac{P \rightarrow Q \stackrel{m}{=} TV^{PQ} \qquad Q \rightarrow R \stackrel{m}{=} TV^{QR} \qquad P \stackrel{m}{=} TV^P \qquad Q \stackrel{m}{=} TV^Q \qquad R \stackrel{m}{=} TV^R}{P \rightarrow R \stackrel{m}{=} TV} \text{ (D)}$$

The reader may notice that three additional premises have been added, corresponding to the probabilities $\mathcal{P}r(P)$, $\mathcal{P}r(Q)$ and $\mathcal{P}r(R)$. This is a consequence of the Markov property. The exact formula for that variation is not recalled here but it merely derives from

$$\mathcal{P}r(R|P) = \mathcal{P}r(R|Q,P) \times \mathcal{P}r(Q|P) + \mathcal{P}r(R|\neg Q,P) \times \mathcal{P}r(\neg Q|P)$$

[3] A perfectly reliable piece of evidence has a confidence of 1. Dealing with unreliable evidence involves using convolution products and is outside of the scope of this paper.

More information about this derivation can be found in Chapter 5, Section 5.3 of [3]. Finally, one may notice that the same conclusion may be inferred by different inference paths leading to different truth values. How to properly aggregate these truth values is not the subject of this paper and is discussed in Chapter 5, Section 5.10 of [3].

3 Temporal Probabilistic Logic Networks

A temporal extension of PLN is defined in Chapter 14 of [3]. However, we have found that some definitions are ambiguous, in particular the sequential connectors *SequentialAnd* and *SequentialOr* redefined further below. Let us begin by defining *Temporal Predicates*, or *Fluents*. Temporal predicates are regular predicates with a temporal dimension:

$$P, Q, R, \ldots : \mathit{Domain} \times \mathit{Time} \mapsto \{\mathit{True}, \mathit{False}\}$$

The type of the temporal dimension, *Time*, could in principle be any thing that has a minimum set of requirements, such as being an ordered semigroup or such. In practice so far, we have used integers, thus capturing a discrete notion of time. Not all temporal predicates need to have a non-temporal domain, *Domain*. In that case, we may simply assume that such domain is the unit type () and ignore it.

3.1 Temporal Operators

Let us define a set of temporal operators operating over temporal predicates.

Lag* and *Lead are temporal operators to shift the temporal dimension of a temporal predicate. They are similar to the metric variations, P_n and F_n, of the *Past* and *Future* operators of Tense Logic [7], with the distinction that they are applied over temporal predicates, as opposed to Boolean modal expressions. The *Lag* operator is formally defined as follows:

$$\mathit{Lag}(P, T) := \lambda x, t.P(x, t - T)$$

Meaning, given a temporal predicate P, it builds a temporal predicate shifted to the right by T time units. In order words, it allows to looks into the past, or one may say that it brings the past into the present. The *Lead* operator is the inverse of the *Lag* operator, thus

$$\mathit{Lead}(\mathit{Lag}(P, T), T) \equiv P$$

and is formally defined as follows:

$$\mathit{Lead}(P, T) := \lambda x, t.P(x, t + T)$$

It allows to look into the future, or one may say that it brings the future into the present.

SequentialAnd is a temporal conjunction where one of the temporal predicate arguments have been temporally shifted. There are at least two variations that can be defined. A first where the past of the first predicate is brought into the present. A second where the future of the second predicate is brought into the present. In this paper we use the second one, formally defined as

$$SequentialAnd(T, P, Q) := And(P, Lead(Q, T))$$

which results into a temporal predicate that is $True$ at time t if and only if P is $True$ at time t and Q is $True$ at time $t + T$. Since we do not know at that point which one of the two variations is best, in practice we have implemented both, but in this paper we settle to one for the sake of simplicity.

SequentialOr is a temporal disjunction where one of the temporal predicate arguments have been temporally shifted. Like for *SequentialAnd* we settle to the variation where the future of the second predicate is brought into the present, defined as

$$SequentialOr(T, P, Q) := Or(P, Lead(Q, T))$$

which results into a temporal predicate that is $True$ at time t if and only if P is $True$ at time t or Q is $True$ at time $t + T$.

PredictiveImplication is an implication where the future of the implicand has been brought into the present, defined as

$$PredictiveImplication(T, P, Q) := Implication(P, Lead(Q, T))$$

resulting into a conditional predicate, that in order to be defined at time t requires that P is $True$ at time t, and if so, is $True$ at t if and only if Q is $True$ at time $t + T$.

Let us introduce a symbolic representation for these temporal constructs with precedence values to minimize parenthesis usage.

Flattened	Symbolic	Precedence
$Lag(P, T)$	$\overrightarrow{P}^T$	1
$Lead(P, T)$	$\overleftarrow{P}^T$	1
$SequentialAnd(T, P, Q)$	$P \mathbin{/\!\!\!\wedge}^T Q$	3
$SequentialOr(T, P, Q)$	$P \mathbin{/\!\!\!\vee}^T Q$	3
$PredictiveImplication(T, P, Q)$	$P \rightsquigarrow^T Q$	4

Additionally, we assume that $\mathbin{/\!\!\!\wedge}^T$ and $\mathbin{/\!\!\!\vee}^T$ are right-associative. The *Lag* (resp. *Lead*) operator is symbolized by an overlined arrow going to the right (resp. to the left) because it brings the past (resp. the future) into the present.

3.2 Temporal Rules

Given these operators we can now introduce a number of temporal inference rules.

The Predictive Implication to Implication Rule (PI) takes a predictive implication as premise and produces an equivalent implication, as depicted by the following proof tree.

$$\frac{P \leadsto^T Q \stackrel{m}{=} TV}{P \to \overleftarrow{Q}^T \stackrel{m}{=} TV} \text{ (PI)}$$

Note that because the conclusion is equivalent to the premise, the truth values may optionally be stripped out the rule.

$$\frac{P \leadsto^T Q}{P \to \overleftarrow{Q}^T} \text{ (PI)}$$

The Implication to Predictive Implication Rule (IP) takes an implication as premise and produces an equivalent predictive implication, as depicted, here without truth value, by the following proof tree.

$$\frac{P \to \overleftarrow{Q}^T}{P \leadsto^T Q} \text{ (IP)}$$

The Temporal Shifting Rule (S) takes a temporal predicate and shits its temporal dimension to the left or the right. An example of such rule is depicted by the following proof tree.

$$\frac{P \stackrel{m}{=} TV}{\overleftarrow{P}^T \stackrel{m}{=} TV} \text{ (S)}$$

Shifting does not change the truth value of the predicate. Indeed, the prevalence of being *True* remains the same, only the origin of the temporal dimension changes. Note however that the predicate itself changes, it is shifted. Therefore, unlike for the IP and PI inference rules that produce equivalent predicates, the truth values must be included in the rule definition, otherwise the rule of replacement would incorrectly apply. There are a number of variations of that rule. For the sake of conciseness we will not enumerate them all, and instead show one more variation over conditional predicates.

$$\frac{P \to Q \stackrel{m}{=} TV}{\overleftarrow{P}^T \to \overleftarrow{Q}^T \stackrel{m}{=} TV} \text{ (S)}$$

The Predictive Implication Direct Introduction Rule (PIDI) is similar to the implication direct introduction rule of Section 2 but accounts for temporal delays between evaluations. It is formalized by the following proof tree.

$$\frac{\left(P(a_i, t_i) \overset{m}{\equiv} TV_i^P\right)_{i=1,\dots,n} \qquad \left(Q(a_i, t_i + T) \overset{m}{\equiv} TV_i^Q\right)_{i=1,\dots,n}}{P \rightsquigarrow^T Q \overset{m}{\equiv} TV} \text{(PIDI)}$$

The truth value formula is identical to that of the implication direct introduction rule. In fact, such rule can be trivially derived by combining the implication direct introduction rule, the implication to predictive implication rule and the definition of the *Lead* operator.

The Temporal Deduction Rule (TD) is similar to the deduction rule of Section 2 but operates on predictive implications. It is formally depicted by the following proof tree.

$$\frac{P \rightsquigarrow^{T_1} Q \overset{m}{\equiv} TV^{PQ} \qquad Q \rightsquigarrow^{T_2} R \overset{m}{\equiv} TV^{QR} \qquad P \overset{m}{\equiv} TV^P \qquad Q \overset{m}{\equiv} TV^Q \qquad R \overset{m}{\equiv} TV^R}{P \rightsquigarrow^{T_1+T_2} R \overset{m}{\equiv} TV} \text{(TD)}$$

As it turns out, the truth value formula is also identical to that of the deduction rule, but the proof is not so trivial. In order to convince us that it is the case, let us construct a proof tree that can perform the same inference without requiring the temporal deduction rule. The result is depicted below

$$\dfrac{\dfrac{\dfrac{P \rightsquigarrow^{T_1} Q \overset{m}{\equiv} TV^{PQ}}{P \rightarrow \overleftarrow{Q}^{T_1} \overset{m}{\equiv} TV^{PQ}}\text{(PI)} \quad \dfrac{\dfrac{Q \rightsquigarrow^{T_2} R \overset{m}{\equiv} TV^{QR}}{Q \rightarrow \overleftarrow{R}^{T_2} \overset{m}{\equiv} TV^{QR}}\text{(PI)}}{\overleftarrow{Q}^{T_1} \rightarrow \overleftarrow{R}^{T_1+T_2} \overset{m}{\equiv} TV^{QR}}\text{(S)} \quad P \overset{m}{\equiv} TV^P \quad \dfrac{Q \overset{m}{\equiv} TV^Q}{\overleftarrow{Q}^{T_1} \overset{m}{\equiv} TV^Q}\text{(S)} \quad \dfrac{R \overset{m}{\equiv} TV^R}{\overleftarrow{R}^{T_1+T_2} \overset{m}{\equiv} TV^R}\text{(S)}}{\dfrac{P \rightarrow \overleftarrow{R}^{T_1+T_2} \overset{m}{\equiv} TV}{P \rightsquigarrow^{T_1+T_2} R \overset{m}{\equiv} TV}\text{(IP)}}\text{(D)}$$

As you may see, the premises and the conclusion of that inference tree match exactly the premises and the conclusion of the temporal deduction rule. Since none of the intermediary formulae, beside the deduction formula, alter the truth values, we may conclude that the formula of the temporal deduction rule is identical to that of the deduction rule.

3.3 Example

In this section we show how to carry an inference combining direct and indirect evidence. To illustrate this process, we consider the temporal predicates P, Q and R, with two datapoints as direct evidence of $P \rightsquigarrow^1 Q$, combined with another predictive implication, $Q \rightsquigarrow^2 P$, given as background knowledge, to produce a third predictive implication, $P \rightsquigarrow^3 R$, based on indirect evidence. The whole inference tree is given below (using $k = 100$ as *Lookahead* in the truth value formulea).

$$\dfrac{\dfrac{P(1) \overset{m}{\equiv} <1,1> \quad P(2) \overset{m}{\equiv} <1,1> \quad Q(1+1) \overset{m}{\equiv} <0,1> \quad Q(2+1) \overset{m}{\equiv} <1,1>}{P \rightsquigarrow^1 Q \overset{m}{\equiv} <0.5, 0.02>}\text{(PIDI)} \quad Q \rightsquigarrow^2 R \overset{m}{\equiv} <0.3, 0.1> \quad P \overset{m}{\equiv} <1, 0.02> \quad Q \overset{m}{\equiv} <0.5, 0.02> \quad R \overset{m}{\equiv} <0.2, 0.5>}{P \rightsquigarrow^3 R \overset{m}{\equiv} <0.2, 0.018>}\text{(TD)}$$

4 Procedural Reasoning

Let us now examine how to use temporal deduction to perform a special type of procedural reasoning, to build larger plans made of smaller plans by chaining their actions. Given plans, also called *Cognitive Schematics* [4], of the form

$$C_1 \wedge A_1 \rightsquigarrow^{T_1} C_2 \stackrel{m}{=} TV_1$$
$$\vdots$$
$$C_n \wedge A_n \rightsquigarrow^{T_n} G \stackrel{m}{=} TV_n$$

expressing that in context C_i, executing action A_i may lead to subgoal C_{i+1} or goal G, after T_i time units, with a likelihood of success measured by TV_i, we show how to infer the composite plan

$$C_1 \wedge A_1 \mathbin{/\!\!\!\wedge}^{T_1} \dots \mathbin{/\!\!\!\wedge}^{T_{n-1}} A_n \rightsquigarrow^{T_1+\dots+T_n} G \stackrel{m}{=} TV$$

alongside its truth value TV. The inferred plan expresses that in context C_1, executing actions A_i to A_n in sequence, waiting T_i time units between A_i and A_{i+1}, leads to goal G after $T_1 + \dots + T_n$ time units, with a likelihood of success measured by TV. Note that strictly speaking, A_i is not an action, it is a predicate that captures the temporal activation of an action. This can be formalized in PLN as well but is not where the difficulty lies. Thus here we directly work with action activation predicates and refer to them as actions for the sake of convenience.

Let us show how to do that with two action plans by building a proof tree like we did for the temporal deduction rule. The final inference rule we are trying to build should look like

$$\frac{C_1 \wedge A_1 \rightsquigarrow^{T_1} C_2 \stackrel{m}{=} TV^{12} \qquad C_2 \wedge A_2 \rightsquigarrow^{T_2} C_3 \stackrel{m}{=} TV^{23} \qquad \dots}{C_1 \wedge A_1 \mathbin{/\!\!\!\wedge}^{T_1} A_2 \rightsquigarrow^{T_1+T_2} C_2 \stackrel{m}{=} TV}$$

where the dots are premises to be filled once we know what they are. Indeed, we cannot directly apply temporal deduction because the implicand of the first premise, C_2, does not match the implicant of the second premise, $C_2 \wedge A_2$. For that reason it is unclear what the remaining premises are. However, we can build an equivalent proof tree using regular deduction, as well as other temporal inferences rules defined in Section 3. The resulting tree (without truth values so that it can fit within the width of the page) is given below.

$$\dfrac{\dfrac{\dfrac{C_1{\wedge}A_1 \rightsquigarrow^{T_1} C_2}{C_1{\wedge}A_1 \rightarrow^{T_1} \overleftarrow{C_2}^{T_1}}\,\text{(PI)}}{C_1{\wedge}A_1{\wedge}\overleftarrow{A_2}^{T_1} \rightarrow \overleftarrow{C_2}^{T_1}{\wedge}\overleftarrow{A_2}^{T_1}}\,\text{(I)} \quad \dfrac{\dfrac{C_2{\wedge}A_2 \rightsquigarrow^{T_2} C_3}{C_2{\wedge}A_2 \rightarrow \overleftarrow{C_3}^{T_2}}\,\text{(PI)}}{\overleftarrow{C_2}^{T_1}{\wedge}\overleftarrow{A_2}^{T_1} \rightarrow \overleftarrow{C_3}^{T_1+T_2}}\,\text{(S)} \quad C_1{\wedge}A_1{\wedge}\overleftarrow{A_2}^{T_1} \quad \dfrac{C_2{\wedge}A_2}{\overleftarrow{C_2}^{T_1}{\wedge}\overleftarrow{A_2}^{T_1}}\,\text{(S)} \quad \dfrac{C_3}{\overleftarrow{C_3}^{T_1+T_2}}\,\text{(S)}}{\dfrac{C_1{\wedge}A_1{\wedge}\overleftarrow{A_2}^{T_1} \rightarrow \overleftarrow{C_3}^{T_1+T_2}}{C_1{\wedge}A_1 \mathbin{/\!\!\!\wedge}^{T_1} A_2 \rightsquigarrow^{T_1+T_2} C_3}\,\text{(IP)}}\,\text{(D)}$$

Note that we have used of a new rule labeled (I) at the left of the proof tree. This rule eliminates independent predicates from an implication without modifying the truth value of its conclusion. Its use is justified by the fact that A_2 is executed immediately *after* reaching C_2, thus cannot have an effect on it.

After retaining the premises and the conclusion only, and adding back the truth values, we obtain the following procedural deduction rule:

$$\frac{C_1 \wedge A_1 \rightsquigarrow^{T_1} C_2 \stackrel{m}{=} TV^{12} \quad C_2 \wedge A_2 \rightsquigarrow^{T_2} C_3 \stackrel{m}{=} TV^{23} \quad C_1 \wedge A_1 \wedge \overleftarrow{A_2}^{T_1} \stackrel{m}{=} TV^{1} \quad C_2 \wedge A_2 \stackrel{m}{=} TV^{2} \quad C_3 \stackrel{m}{=} TV^{3}}{C_1 \wedge A_1 \mathbin{\measuredangle}^{T_1} A_2 \rightsquigarrow^{T_1+T_2} C_3 \stackrel{m}{=} TV} \text{(PD)}$$

with a formula identical to that of the deduction rule, once again. The premises filling the dots are therefore

$$C_1 \wedge A_1 \wedge \overleftarrow{A_2}^{T_1} \stackrel{m}{=} TV^{1} \qquad C_2 \wedge A_2 \stackrel{m}{=} TV^{2} \qquad C_3 \stackrel{m}{=} TV^{3}$$

There is no doubt these premises could be further decomposed into sub-inferences as it was done with the (I) rule. Indeed, likely more simplifications can be made by assuming that the agent has a form of freewill and thus that its actions are independent of the rest of the universe, outside of its decision policy influenced by its very procedural reasoning. This is reminiscent of the do-calculus [6] and will be explored in more depth in the future. In the meantime, these are left as they are, as it introduces no additional assumption, and their truth values can always be calculated using inference rules based on direct evidence, if anything else. Future directions may also include adding inference rules to support behavior trees; introducing Event Calculus operators as predicate transformers (similar to how *Lag* and *Lead* are defined); as well as supporting temporal intervals and continuous time.

References

1. Abourizk, S., Halpin, D., Wilson, J.: Fitting beta distributions based on sample data. Journal of Construction Engineering and Management **120** (1994)
2. Geisweiller, N., Yusuf, H.: Rational opencog controlled agent. Artificial General Intelligence: Sixteenth International Conference (2023)
3. Goertzel, B., Ikle, M., Goertzel, I.F., Heljakka, A.: Probabilistic Logic Networks. Springer US (2009)
4. Goertzel, B., Pitt, J., Wigmore, J., Geisweiller, N., Cai, Z., Lian, R., Huang, D., Yu, G.: Cognitive synergy between procedural and declarative learning in the control of animated and robotic agents using the opencogprime agi architecture. Proceedings of the AAAI Conference on Artificial Intelligence (2011)
5. Jøsang, A.: Subjective Logic: A Formalism for Reasoning Under Uncertainty. Springer Publishing Company, Incorporated, 1st edn. (2016)
6. Pearl, J.: Causal diagrams for empirical research. Biometrika **82**, 669–688 (1995)
7. Prior, A.N.: Past, Present and Future. Oxford, England: Clarendon Press (1967)
8. Shanahan, M.: The Event Calculus Explained. In Artificial Intelligence LNAI **1600** (06 2000)