Temporal and Procedural Reasoning for Rational Agent Control in OpenCog

Nil Geisweiller and Hedra Yusuf

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

Abstract. TODO

Keywords: Temporal · Procedural · Reasoning · Probabilistic Logic Networks · OpenCog

1 Introduction

The goal of this project is to make an agent as rational as possible, not necessarily as efficient as possible. This stems from the concern that in order to autonomously gain efficiency the agent must first be able to make the best possible decisions, starting first in the outer world, and then in the inner world.

The paper presents

The agent starts in a completely unknown environment

The idea is that reasoning is used at all levels, discovering patterns from raw observations, building plans and making decisions.

2 Contributions

The contributions of that paper are:

1. Build upon existing temporal reasoning framework defined in Chap.14 [TODO: cite PLN book].
2. Design an architecture for controlling an agent based on that temporal reasoning extension.

3 Outline

1. Temporal reasoning
2. ROCCA
3. Minecraft experiment

4 Recall: Probabilistic Logic Networks

PLN, which stands for Probabilistic Logic Networks, is a mixture of predicate and term logic that has been probabilitized to properly handle uncertainty. It has two types of rules

1. one type for introducing relationships from direct observations,
2. the other for introducing relationships from existing relationships.

As such it is especially suited for building an ongoing understanding of an unknown environment (using direct introduction rules), and then planning in that environment (using indirect introduction rules).

4.1 Elementary Notions

Graphically speaking, PLN statements are sub-hypergraphs[1] made of links and nodes, called *Atoms*, decorated with *Truth Values* that can be understood as probabilities incorporating uncertainties [**?**]. Syntactically speaking however PLN statements are not very different from statements expressed in another logic, except that they are usually formatted in prefixed-operator indented-argument style to emphasize their graphical nature and leave room for truth values. There is a large variety of constructs for PLN, here we will focus primarily on constructs for manipulating predicates. Let us recall that predicates are functions that take tuples of Atoms and output boolean values

$$P, Q, R, \ldots : Atom^n \mapsto \{True, False\}$$

Within predicate constructs there are two classes of operators

1. one for defining predicate from instances, such as *Evaluation* and *Lambda*,
2. and another for combining existing predicates, such as *And*, *Not* and *Implication*.

Let us present these operators below, corresponding to the minimum subset we will need in the rest of the paper.

- Evaluation:

$$\begin{array}{l} Evaluation\ \langle TV \rangle \\ \quad P \\ \quad e \end{array}$$

  states that $P(e)$ outputs *True* to a degree set by the truth value TV.
- Lambda:

$$\begin{array}{l} Lambda\ \langle TV \rangle \\ \quad x \\ \quad P(x) \end{array}$$

  is a predicate constructor with variable x and predicate body $P(x)$, where the true value TV corresponds to the probability $\mathbf{Pr}(P)$ of $P(x)$ to output *True* for a random input.

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

– Conjunction:

$$\begin{array}{c} And\ \langle TV \rangle \\ P \\ Q \end{array}$$

represents the predicate obtained by taking the conjunction of P and Q, or equivalently the indicator function corresponding to the intersection of the *satisfying sets* of P and Q. The truth value TV then represents an estimate of the probability $\mathbf{Pr}(P, Q)$ of the conjunction of P and Q.

– Negation:

$$\begin{array}{c} Not\ \langle TV \rangle \\ P \end{array}$$

represents the negation of P, or equivalently the indicator function corresponding to the complement of the satisfying set of P. The truth value TV then represents an estimate of the probability $\mathbf{Pr}(\neg P)$ of the negation of P.

– Implication:

$$\begin{array}{c} Implication\ \langle TV \rangle \\ P \\ Q \end{array}$$

represents the predicate Q conditioned on P, that is only defined for instances x for which $P(x)$ is $True$. The truth value TV then represents an estimate of the conditional probability $\mathbf{Pr}(Q|P)$. There is some subtleties to take into account due to the fact $P(x)$ can actually be partially true (stated by the truth value of *Evaluation* as explained above), but this all resolves nicely by assuming degrees of truth are probabilistic. More is explained about that below.

Truth values are fundamentally second order probability distributions. However in practice they are usually represented by two numbers, a strength and a confidence, both ranging from 0 to 1. The strength represents a probability while the confidence represents a precision over that probability. Underneath, strength and confidence can be mapped into a second order distribution such as a Beta distribution [TODO: add figure].

4.2 Inference Rules

Beside operators, inferences rules are used to construct PLN statements and calculate their truth values. They mainly fall into two categories, direct and indirect. Direct rules infer abstract knowledge from direct evidence, while indirect rules infer knowledge by combining existing abstractions, themselves inferred directly or indirectly. There are dozens of inference rules but for now we will only recall two which are needed for the paper:

1. *Implication Direct Introduction Rule*
2. *Deduction Rule*

The Implication Direct Introduction Rule (IDI) takes *Evaluation* links as premises and produces an *Implication* link as conclusion, formally depicted as the proof tree

$$\frac{\begin{array}{ccc} \textit{Evaluation } \langle TV_i^P \rangle & & \textit{Evaluation } \langle TV_i^Q \rangle \\ P & \cdots & Q \\ E_i & & E_i \end{array}}{\begin{array}{c} \textit{Implication } \langle TV \rangle \\ P \\ Q \end{array}} \text{(IDI)}$$

Assuming perfectly reliable direct evidence[2] then the resulting truth value is calculated as follows

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

$$TV.c = \frac{n}{n+k}$$

where $TV.s$ and $TV.c$ represents the strength and the confidence of TV respectively, k is a system parameter, and $f_\wedge$ is a function embodying a probabilistic assumption about the intersection of the events corresponding to the *probabilitized degrees of truth* of $P(E_i)$ and $Q(E_i)$. Such function typically ranges from the product function (perfect independence) to the min function (perfect overlap).

The Deduction Rule

5 Temporal Logic

The temporal logic used here builds upon what is described in the Chapter 14 of the PLN book [TODO: cite]. Let us define that

5.1 Mapping Temporal into Atemporal Statements

6 Rational OpenCog Controlled Agent (ROCCA)

7 Experiment with Simple Minecraft Environment

8 Conclusion

References

[2] dealing with unreliable direct evidence involves expensive convolution products and is outside of the scope of this paper.