# Guiding Probabilistic Logical Inference with Nonlinear Dynamical Attention Allocation

Cosmo Harrigan[1,2,6], Ben Goertzel[2], Matthew Iklé[3], Amen Belayneh[4,5], and Gino Yu[5]

[1] OpenCog Foundation
[2] Novamente LLC
[3] Adams State University
[4] iCog Labs
[5] School of Design, Hong Kong Poly U
[6] University of Washington

**Abstract.** In order to explore the practical manifestations of the "cognitive synergy" between the PLN (Probabilistic Logic Networks) and ECAN (Economic Attention Network) components of the OpenCog AGI architecture, we explore the behavior of PLN and ECAN operating together on two standard test problems commonly used with Markov Logic Networks (MLN). Our preliminary results suggest that, while PLN can address these problems adequately, ECAN offers little added value for the problems in their standard form. However, we outline modified versions of the problem that we hypothesize would demonstrate the value of ECAN more effectively, via inclusion of confounding information that needs to be heuristically sifted through.

## 1 Introduction

One approach to creating AGI systems is the "integrative" strategy, involving combining multiple components embodying different structures or algorithms, and relying on synergistic dynamics between components. One kind of integrative system involves various highly independent software components, each solving a specialized set of problems in a mostly standalone manner, with occasional communication between each other in order to exchange problems and solutions. On the other end of the scale, are systems designed as tightly interconnected components that give rise to complex non-linear dynamical phenomena. Here, we are specifically focused on the latter approach. We will discuss the particulars of one form of cognitive synergy – between probabilistic inference and nonlinear-dynamical attention allocation – within the context of one particular integrative AGI architecture, OpenCogPrime [2].

## 2 OpenCogPrime

Our work here is based upon specific details of the AGI architecture called **OpenCogPrime** (OCP), based on the open-source OpenCog project at

B. Goertzel et al. (Eds.): AGI 2014, LNAI 8598, pp. 238–241, 2014.

http://opencog.org. OCP is a large and complex system whose detailed description occupies two volumes [4].

The concept of cognitive synergy is at the core of the design, with highly interdependent subsystems responsible for inference regarding patterns obtained from visual, auditory and abstract domains, uncertain reasoning, language comprehension and generation, concept formation, and action planning.

The medium-term goal of the OCP project is to create systems that can function broadly comparably to young human children in virtual and robotic preschool contexts [3]. In the longer-term, the aim of the project is to engineer systems that exhibit general intelligence equivalent to a human adult, and ultimately beyond.

The dynamics of interaction between processes in OCP is designed in such a way that knowledge can be converted between different types of memory; and when a learning process that is largely concerned with a particular type of memory encounters a situation where the rate of learning is very slow, it can proceed to convert some of the relevant knowledge into a representation for a different type of memory to overcome the issue, demonstrating **cognitive synergy**. The simple case of synergy between ECAN and PLN explored here is an instance of this broad concept; PLN being concerned mainly with declarative memory and ECAN mainly with attentional memory.

## 3 Probabilistic Logic Networks

PLN serves as the probabilistic reasoning system within OpenCog's more general artificial general intelligence framework. PLN logical inferences take the form of syllogistic rules, which give patterns for combining statements with matching terms. Related to each rule is a truth-value formula which calculates the truth value resulting from application of the rule. PLN uses forward-chaining and backward-chaining processes to combine the various rules and create inferences.

## 4 Economic Attention Networks

The attention allocation system within OpenCog is handled by the Economic Attention Network (ECAN). ECAN is a graph of untyped nodes and links that may be typed either HebbianLink or InverseHebbianLink. Each Atom in an ECAN is weighted with two numbers, called STI (short-term importance) and LTI (long-term importance), while each Hebbian or InverseHebbian link is weighted with a probability value. A system of equations, based upon an economic metaphor of STI and LTI values as artificial currencies, governs importance value updating. These equations serve to spread importance to and from various atoms within the system, based upon the importance of their roles in performing actions related to the system's goals.

An important concept with ECAN is the attentional focus, consisting of those atoms deemed most important for the system to achieve its goals at a particular instant. Through the attentional focus, one key role of ECAN is to guide the

forward and backward chaining processes of PLN inference. Quite simply, when PLN's chaining processes need to choose logical terms or relations to include in their inferences, they can show priority to those occurring in the system's attentional focus (due to having been placed there by ECAN). Conversely, when terms or relations have proved useful to PLN, they can have their importance boosted, which will affect ECAN's dynamics. This is a specific example of the cognitive synergy principle at the heart of the OpenCog design.

## 5 Evaluating PLN on Standard MLN Test Problems

In order to more fully understand the nature of PLN/ECAN synergy, we chose to explore it in the context of two test problems standardly used in the context of MLNs (Markov Logic Networks) [7]. These problems are relatively easy for both PLN and MLN, and do not stress either system's capabilities.

The first test case considered is a very small-scale logical inference called the *smokes* problem, discussed in its MLN form at [1]. The PLN format of the *smokes* problem used for our experiments is given at `https://github.com/opencog/test-datasets/blob/master/pln/tuffy/smokes/smokes.scm` The conclusions obtained from PLN backward chaining on the *smokes* test case are

```
cancer(Edward) <.62, 1>
cancer(Anna)   <.50, 1>
cancer(Bob)    <.45, 1>
cancer(Frank)  <.45, 1>
```

which is reasonably similar to the output of MLN as reported in [1],

```
0.75 Cancer(Edward)
0.65 Cancer(Anna)
0.50 Cancer(Bob)
0.45 Cancer(Frank)
```

The second test case is a larger problem referred to as *RC* involving the placement of research papers in categories based on information about their authors and citations. [5] The full RC problem contains 4 relations, 15 rules, 51K entities, 430K evidence tuples, 10K query atoms, 489 components [6]. The RC1000 problem is scaled-down with only 1000 pieces of evidence. Human raters display 72% agreement on mapping papers into categories. Straightforward statistical methods can get up to 66%, and MLN does roughly the same.

The full set of rules and evidence used for feeding the RC problem to PLN is at `https://github.com/opencog/test-datasets/tree/master/pln/tuffy/class`. The corresponding information for the RC1000 problem is at `https://github.com/opencog/test-datasets/tree/master/pln/tuffy/rc1000`.

## 6 Exploring PLN/ECAN Synergy with Standard MLN Test Problems

We explored the possibility of utilizing ECAN to assist PLN on these test problems, so far achieving results more educational than successful. Based on our work so far, it seems that ECAN's guidance is not of much use to PLN on these problems as formulated. However, exploring ways to modify the test problems so as to enable to them to better showcase ECAN, led us to conceptually interesting conclusions regarding the sorts of circumstances in which ECAN is most likely to help PLN.

We hypothesize that if one modified the *smokes* example via adding a substantial amount of irrelevant evidence about other aspects of the people involved, then one would have a case where ECAN could help PLN, because it could help focus attention on the relevant relationships. We also hypothesize that, if one had information about the words occurring in the papers in the RC test problem, then ECAN could help, because *some* of these words would be useful for guessing the categories papers belong to, and others would not; ECAN could help via spreading importance from words found to be important, to other related words, saving PLN the trouble of attempting inferences involving all the words. One of our threads of current research focuses on the substantiation of these hypotheses via running PLN and ECAN together on test problems of this nature.

In sum, the exploration of some standard MLN test problems in a PLN/ECAN context has led us to interesting hypotheses regarding where ECAN can, and cannot, prove useful to PLN. Preliminarily, it appears that this particular cognitive synergy is going to be most useful in cases where, unlike these MLN test problems in their standard form, there is a considerable amount of detailed information and part of the problem involves heuristically sifting through this information to find the useful bits.

## References

1. Project tuffy, `http://hazy.cs.wisc.edu/hazy/tuffy/doc/`
2. Goertzel, B.: Opencogprime: A cognitive synergy based architecture for artificial general intelligence. In: 8th IEEE International Conference on Cognitive Informatics, ICCI 2009, pp. 60–68. IEEE (2009)
3. Goertzel, B., Bugaj, S.V.: Agi preschool: A framework for evaluating early-stage human-like agis. In: Proceedings of the Second International Conference on Artificial General Intelligence (AGI 2009), pp. 31–36 (2009)
4. Goertzel, B., Pennachin, C., Geisweiller, N.: Engineering General Intelligence, Part 1. Atlantis Press (2014)
5. McCallum, A., Nigam, K., Rennie, J., Seymore, K.: Automating the construction of internet portals with machine learning. Information Retrieval 3(2), 127–163 (2000)
6. Niu, F., Ré, C., Doan, A., Shavlik, J.: Tuffy: Scaling up statistical inference in markov logic networks using an rdbms. Proceedings of the VLDB Endowment 4(6), 373–384 (2011)
7. Richardson, M., Domingos, P.: Markov logic networks. Machine Learning 62(1-2), 107–136 (2006)