A New Approach to Probabilistic Knowledge-Based Decision Making

Thomas C. Henderson[1][a], Tessa Nishida[1] Amelia C. Lessen[1], Nicola Wernecke[1], Kutay Eken[1] and David Sacharny[2]
[1]*School of Computing, University of Utah, Salt Lake City, Utah, U.S.A.*
[2]*Blyncsy Inc, Salt Lake City, UT, U.S.A.*

Keywords: Probabilistic Logic Decision Making.

Abstract: Autonomous agents interact with the world by processing percepts and taking actions in order to achieve a goal. We consider agents which account for uncertainty when evaluating the state of the world, determine a high level goal based on this analysis, and then select an appropriate plan to achieve that goal. Such knowledge-based agents must take into account facts which are always true (e.g., laws of nature or rules) and facts which have some amount of uncertainty. This leads to probabilistic logic agents which maintain a knowledge base of facts each with an associated probability. We have previously described NILS, a nonlinear systems approach to solving atom probabilities, and compare it here to a hand-coded probability algorithm and a Monte Carlo method based on sampling possible worlds. We provide experimental data comparing the performance of these approaches in terms of successful outcomes in playing Wumpus World. The major contribution is the demonstration that the NILS method performs better than the human coded algorithm and is comparable to the Monte Carlo method. This advances the state-of-the-art in that NILS has been shown to have super-quadratic convergence rates.

1 INTRODUCTION

Knowledge-based agents generally exhibit brittle behavior when propositions can only be true or false. For example, Casado et al. (Casado et al., 2011) have used a knowledge-based approach to handle event recognition in multi-agent systems without consideration of uncertainty. More nuanced and informed decision making is possible when the uncertainty of a proposition can be characterized and included in the evaluation of the current state in order to select an action. Wang et al. (Wang et al., 2006) extend Hindrik's logic programming language for Belief, Desire, Intention (BDI) agents (Hindriks et al., 1997) by incorporating an interval-based uncertainty representation for the language. They define a probabilistic conjunction strategy to update the intervals based on the probabilities of random variables which satisfies the axioms of probability theory. However, to capture all necessary relations between the atoms requires an exponential number of constraints (i.e., $\sum_{k=1}^{n} \binom{n}{k} = 2^n$, where n is the number of logical atoms). Dix et al. (J. Dix and Subrahmanian, 2000) provide two broad classes of semantics for probabilistic agents. The drawback is that their analysis only applies to negation free programs, thus limiting their usefulness here. As another example, consider Milch and Koller (Milch and Kller, 2000) whose probabilistic epistemic logic (PEL) provides a formal semantics for probabilistic beliefs. However, PEL is based on Bayesian networks which require the definition of the full joint probability distribution. Other approaches to probabilistic logic have been proposed. Pearl (Pearl, 1988) developed Bayesian networks which structure the full joint probability distribution as conditional relations between the logical variables. Reiter (Reiter, 2001) extended the situation calculus of McCarthy to include probabilities, and Domingos and Lowd (Domingos and Lowd, 2009) applied Markov Logic Networks to relational problems in artificial intelligence. All these methods have high computational complexity (e.g., the expression of a Bayesian network requires representing the 2^n complete conjunctions in the network's conditional tables, and MLN inference is $\#P$-complete). Moreover, none of these methods exploit the probabilistic logical framework as advocated here wherein the agent's decision making processes are based on a novel probabilistic analysis of the world in terms of it laws (rules) and sensory data.

[a] https://orcid.org/0000-0002-0792-3882

Henderson, T., Nishida, T., Lessen, A., Wernecke, N., Eken, K. and Sacharny, D.
A New Approach to Probabilistic Knowledge-Based Decision Making.
DOI: 10.5220/0011606700003393
In *Proceedings of the 15th International Conference on Agents and Artificial Intelligence (ICAART 2023) - Volume 3*, pages 34-39
ISBN: 978-989-758-623-1; ISSN: 2184-433X

The approach proposed here (called NILS: Nonlinear Logic Solver) solves the probabilistic sentence satisfiability problem (see (Henderson et al., 2020) for details). This allows the estimation of the atom probabilities based on all a priori knowledge as well as that acquired from sensors during the execution of a task. The method is described in detail below as well as the results of its application to the Wumpus World problem (for more on Wumpus World, see (Russell and Norvig, 2009; Yob, 1975)).

1.1 Probabilistic Logic

The agents considered here use a probabilistic logic representation of knowledge (Nilsson, 1986). The agent's knowledge base is a set of propositions (or beliefs) expressed in conjunctive normal form (CNF); i.e.:

$$KB \equiv C_1 \wedge C_2 \wedge \ldots \wedge C_m$$

where

$$C_i \equiv L_{i,1} \vee L_{i,2} \vee \ldots \vee L_{i,k_i}$$

where

$$L_{i,j} \equiv a_p \; or \; \neg a_p$$

where a_p is a logical atom. In addition, a probability, p_i, is associated with each clause (C_i). The agent uses the knowledge by selecting a goal (i.e., a belief which is to be made true) based on the current assessment of the situation. This involves assigning probabilities to the beliefs, and then making a rational decision based on these belief probabilities (e.g., to avoid danger or to achieve a reward).

In order to reason using probabilities it is necessary for the probabilities to be determined in a valid framework; for this we must solve the probabilistic satisfiability problem (Georgakopoulos et al., 1988) which is NP-hard. Probabilistic satisfiability means that there is a function, $\pi : \Omega \rightarrow [0,1]$, where Ω is the set of complete conjunctions over n variables such that:

$$\pi(\omega) \in [0,1], \forall \omega \in \Omega$$

$$\sum_{\omega \in \Omega} \pi(\omega) = 1$$

$$Pr(C_i) = \sum_{\omega \models C_i} \pi(\omega)$$

where the complete conjunctions are the set of all truth value assignments over n variables, and $\omega \models C_i$ means that the truth assignment ω makes C_i true. The probabilistic satisfiability problem is to determine if there is an appropriate function π.

We have provided an analysis of this problem and given the NILS method for its (approximate) solution (Henderson et al., 2020). This involves converting each clause to a nonlinear equation relating the probability of the clause to the probabilities of the atoms in the clause; a solution is then found for the atom probabilities which best satisfies the definition of the function π. The method finds the best (local) function and not necessarily an exact solution by using a nonlinear solver. NILS works as follows:

- Convert each CNF clause, C_i, with probability p_i, to an equation using the general addition rule of probability: $Pr(A \cup B) = Pr(A) + Pr(B) - Pr(A \wedge B)$
- Solve the system; note that this can be nonlinear if the variables are independent, or linear over new variables if not independent.
  - independent: $Pr(A \cup B) = Pr(A) + Pr(B) = -Pr(A)Pr(B)$, which leads to: $p_i = x_1 + x_2 - x_1 x_2$
  - not independent: $Pr(A \cup B) = Pr(A) + Pr(B) - Pr(A \wedge B)$, which leads to $p_i = x_1 + x_2 - x_3$

For a simple example, consider the CNF sentence S given by:

- $C_1 = a_1 [Pr(C_1) = 0.7]$
- $C_2 = \neg a_1 \vee a_2 [Pr(C_2) = 0.7]$

A solution for this is:

- $\pi(0,0) = 0.2$
- $\pi(0,1) = 0.1$
- $\pi(1,0) = 0.3$
- $\pi(1,1) = 0.4$

Note that $Pr(a_1) = Pr(1,0) + Pr(1,1) = 0.3 + 0.4 = 0.7$, and $Pr(\neg a_1 \vee a_2) = Pr(0,0) + Pr(0,1) + Pr(1,1) = 0.2 + 0.1 + 0.4 = 0.7$. Nilsson (Nilsson, 1986) shows that the solution for π is not unique, and that the $Pr(a_2) \in [0.4, 0.7]$ for the PSAT solutions.

Solving as a nonlinear system:

$$0.7 = Pr(a_1)$$

$$\begin{aligned} 0.7 = Pr(\neg a_1 \vee a_2) &= Prob(\neg a_1) + Pr(a_2) - Pr(\neg a_1 \wedge a_2) \\ &= (1 - Pr(a_1)) + Pr(a_2) - (1 - Pr(a_1)Pr(a_2) \\ &= 0.3 + Pr(a_2) - 0.3Pr(_2) \end{aligned}$$

So, $Pr(a_2) = 0.571$. Note that logical variables are assumed independent; that is, $Pr(A \wedge B) = Pr(A)Pr(B)$. We have also described a method for the case when they are not independent (see (Henderson et al., 2020)).

1.2 Test Domain: Wumpus World

Wumpus world is given in the AI text of Russell and Norvig (Russell and Norvig, 2009); however, Wumpus World was originally developed by G. Yob (Yob, 1975). It is a game defined on a 4x4 board (see Figure 1). The cells are defined by their (x,y) centers

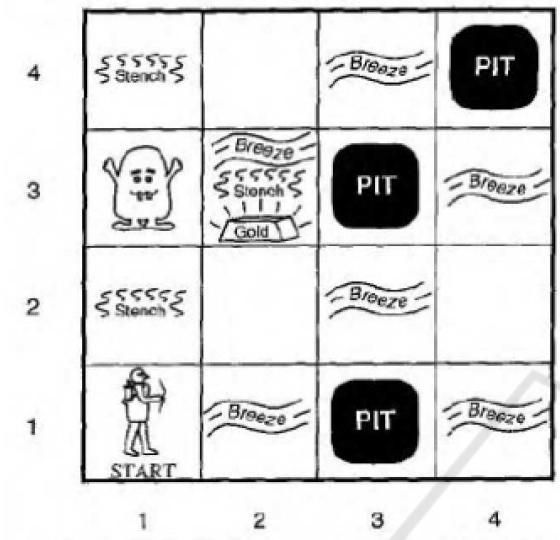

Figure 1: Wumpus World Layout (Russell and Norvig, 2009).

with the origin in the lower left, the x-axis is horizontal and the y-axis is vertical. The agent starts in cell $(1,1)$ and tries to find the gold (in this instance located in cell $(2,3)$) while avoiding pits (cells $(3,1)$, $(3,3)$ and $(4,4)$) and the Wumpus (cell $(1,3)$). The agent has the following percepts:

- Breeze: indicates there is a pit in a neighboring cell
- Stench: indicates there is a Wumpus in a neighboring cell
- Glitter: indicates gold in the current cell
- Bump: indicates running into a wall (after a Forward command and staying in start cell)
- Scream: indicates arrow (shot by agent) killed Wumpus

There are six actions available to the agent:

- Forward: move forward one cell (agent has a direction)
- Rotate Right: rotate direction 90 degrees to the right
- Rotate Left: rotate direction 90 degrees to the left
- Grab: grab gold (if in current cell)
- Shoot arrow: shoot arrow in direction facing (only one arrow)
- Climb: climb out of cave (only applies in cell $(1,1)$)

There are a number of rules in the game; for example:

- *Cells neighboring a pit have a breeze*
- *Cells neighboring the Wumpus have a stench*
- *There is one and only one Wumpus*
- *There is gold in one and only one cell*
- *If the agent moves into a cell with a pit or Wumpus, the agent dies.*
- *Pits occur in each cell (except* $(1,1)$*) with a fixed probability; here we use 0.2.*

Given the rules of Wumpus, it is necessary to formulate them as a CNF sentence. As a starting point, the set of atoms is defined as follows; for each cell (x,y):

- Bxy indicates a breeze in (x,y)
- Gxy indicates gold in (x,y)
- Pxy indicates a pit in (x,y)
- Sxy indicates a stench in (x,y)
- Wxy indicates the Wumpus in (x,y).

Since there is only one Wumpus, there are rules stating that if the Wumpus is in a given cell, then it is not in any other; e.g., $W23 \rightarrow \neg W22$; since implication is not a logical operator in CNF, this is written as $\neg W23 \vee \neg W22$. Since the Wumpus must be somewhere, there's a rule:

$$W21 \vee W31 \vee \ldots \vee W44$$

Note that the Wumpus is not allowed in cell $(1,1)$. Also, there are rules expressing that there may not be a pit and Wumpus in the same cell. The number of atoms is then 80 (i.e., 5*16), and the rules give rise to 402 clauses in the CNF KB. Note that the state of neighboring cells (e.g., pit or no pit) requires probabilistic reasoning since mutiple models can satisfy the percepts.

To show the power of the NILS method, consider estimating the a priori atom probabilities given just the rules of the game. These probabilities can be estimated using Monte Carlo by sampling a large number of boards and finding the likelihood of each atom. Similarly, NILS can provide an estimate. Figure 2 shows the two sets of probabilities, and it can be seen that NILS provides a very good estimate; moreover, the important issue is that safer cells be distinguished from less safe ones, and even with the differences in exact values, the order relations of the probabilities are preserved.

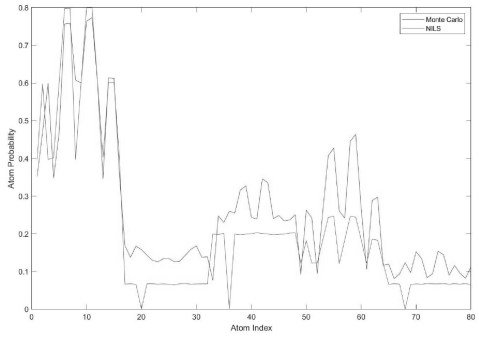

Figure 2: The A Priori Atom Probabilties found by Monte Carlo sampling and the NILS method.

Performance in Wumpus world is measured by a point system wherein:

- Take an action: costs -1 (except shoot the arrow which costs -10)
- Die: costs -1000
- escape with gold: reward of 1000

The higher the score the better the performance. A score greater than zero is deemed a success.

1.3 Problem Statement and Experimental Methodology

The hypothesis is that it is possible to develop autonomous agents that:

- represent knowledge of the world as logical propositions, both universal laws and temporal variables (fluents),
- assign probabilities to those propositions,
- use a consistent formal framework to make inferences which allow informed rational actions to be taken, and
- achieve a strong level of performance.

Cognitive-level knowledge forms the core of the knowledge base, and goals are formulated as beliefs to be made true. The agent's task is to organize the goals in a reasonable manner and select appropriate plans which when executed will make the goal belief true.

In order to restrict the study to compare only the way in which probabilities are produced, a common agent algorithm was developed; its logic is shown in Figure 3. This allows alternative methods to be used to provide the atom probabilities used by the agent in its decision making process. That is, difference in behavior is only possible due to differences in atom probabilities. Note that the most important probabilities concern whether a Wumpus or pit are present in a cell.

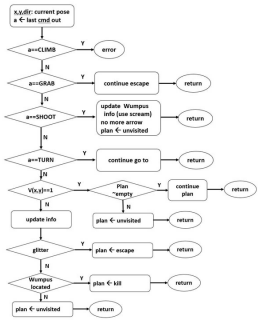

Figure 3: The Agent Behavior Algorithm.

Three mechanisms for atom probability are considered: (1) a human hand-coded method based on an understanding of the game, (2) a Monte Carlo method which samples a set of boards which satisfy the known conditions and computes probabilities based on those boards, and (3) the NILS method. The Monte Carlo method serves as an approximation to the ground truth (and would produce the exact probabilities if the samples included all satisfying boards; since there 1,105,920 possible boards to filter at each move, this option is not exploited). Therefore, the comparison allows determination of how well the NILS method performs compared to a human-based method as well as with respect to the best possible result.

2 EXPERIMENTS

Ten sets of 1000 random solvable boards were produced; that is, for each board there exists a path from the start cell to the gold with no pit. The agent overall strategy is to move to the closest cell with lowest probability of danger; the agent then goes there either dies, finds the gold and escapes, or continues searching. The number of successful games is used as the measure of success. Table 1 gives the number of boards solved for each probability method for each of the ten sets of 1000 boards.

2.1 Discussion

The question posed here is whether probabilities produced by NILS lead to a higher rate of success compared to the human defined probability algorithm, and also to determine how well NILS performs compared to the Monte Carlo approximation. As can be seen in Table 1, NILS averaged about twelve more successes per 1000 boards as the human algorithm, and was only outperformed (by three successes) in one of the test sets. Moreover, the 95% confidence intervals

Table 1: Results of Performance Test of Agents. There are 10 trial sets consisting of 1000 solvable boards each. The mean success rate for these 10 sets is given as well as the variance. The 95% confidence intervals are [599.7,610.2], [612.0,622.4], and [612.0,628.8], respectively. [Note that the Monte Carlo success rates are the result of 10 independent trials on each board test set.]

Board Set	Human	NILS	Monte Carlo
1	590	609	613.3
2	610	620	626.1
3	590	608	617.9
4	610	619	628.5
5	607	604	619.3
6	597	620	619.7
7	611	630	627.6
8	614	626	631.0
9	611	626	638.2
10	609	610	623.4
Mean	604.9	617.2	624.9
Var	81.9	79.5	44.8

of the two methods do not overlap. With respect to the Monte Carlo method, NILS averaged six fewer successes per thousand, but outperformed it in two of the trial sets. The confidence intervals of these two methods do overlap.

The results support the claim that NILS is better than the human probability algorithm and comparable to the Monte Carlo method. In examining specific cases, it was determined that the success of NILS over the human algorithm mainly related to the fact that the encoding of the Wumpus World rules into the knowledge base provided implicit influence on probabilities (i.e., implicit conditional probabilities) which the human failed to capture. The success of NILS over Monte Carlo when it occurred was seen to be related to the result of the selection of sample boards by the Monte Carlo method. To control for this, Monte Carlo performance is given in terms of statistical measurements (mean and variance) over a set of ten independent trials per board set test case. It may be possible to improve Monte Carlo performance by increasing the number of samples, but computational costs go up rapidly since each sample board must fit the current sensed data constraints, and a larger set of random boards must be examined to get the desired appropriate sample set.

3 CONCLUSIONS

We have demonstrated the viability of the nonlinear logic solver (NILS) system as the basis for probabilistic logic agents. Moreover, the method is superior to hand coded probability functions for the same application domain, and comparable to the Monte Carlo agent which operates with more detailed information about the game.

In future work, we intend to investigate the application of probabilistic decision making in terms of:

- deeper cognitive representations for the agent using a Belief, Desire, Intention (BDI) architecture.
- larger problem domains with multiple agents,
- knowledge compilation for individual agents cooperating in a team effort in order to provide them with just the information they need, and
- application to large-scale unmanned aircraft systems traffic management (UTM).

ACKNOWLEDGEMENTS

This work was supported in part by National Science Foundation award 2152454.

REFERENCES

Casado, A., Martinez-Tomas, R., and Fernandez-Caballero, A. (2011). Multi-agent System for Knowledge-Based Event Recognition and Composition. *Expert Systems*, 28(5):488–501.

Domingos, P. and Lowd, D. (2009). *Markov Logic Networks*. Morgan and Claypool Publishers, Williston, VT.

Georgakopoulos, G., Kavvadias, D., and Papadimitriou, C. (1988). Probabilistic Satisfiability. *Journal of Complexity*, 4:1–11.

Henderson, T., Simmons, R., Serbinowski, B., Cline, M., Sacharny, D., Fan, X., and Mitiche, A. (2020). Probabilistic Sentence Satisfiability: An Approach to PSAT. *Artificial Intelligence*, 278:71–87.

Hindriks, K., de Boer, F., van der Hoek, W., and Meyer, J. (1997). Formal Semantics of an Abstract Agent Programming Language. In *Proceedings of the International Workshop on Agent Theories, Architectures and Languages*, pages 215–229. Springer Verlag.

J. Dix, M. N. and Subrahmanian, V. (2000). Probabilistic Agent Programs. *ACM Transactions on Computational Logic*, 1(2):208–246.

Milch, B. and Kller, D. (2000). Probabilistic Models for Agents' Beliefs and Decisions. In *Uncertainty in Artificial Intelligence Proceedings*, pages 389–396. Association for Computing Machinery.

Nilsson, N. (1986). Probabilistic Logic. *Artificial Intelligence Journal*, 28:71–87.

Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufmann, San Mateo, CA.

Reiter, R. (2001). *Knowledge in Action: Logical Foundations for Specifying and Implementing Dynamical Systems*. MIT Press, Cambridge, MA.

Russell, S. and Norvig, P. (2009). *Artificial Intelligence: A Modern Approach*. Prentice Hall Press, Upper Saddle River, NJ, 3rd edition.

Wang, J., Ju, S.-E., and Liu, C.-H. (2006). Agent-Oriented Probabilistic Logic Programming. *Journal of Computational Science and Technology*, 21(3):412–417.

Yob, G. (1975). Hunt the Wumpus? *Creative Computing*.