# Bayesian-Markov Feedback in Constraint-based Planning*

Matthew A. Schack† Neil T. Dantam†

***Abstract*— Planning under uncertainty presents numerous challenges in expressivity and scalability. We propose a new algorithm that extends the constraint-based planning framework to address state uncertainty. The approach has produced optimal plans in simulation, and we are implementing it on a physical robot.**

## I. Introduction

Constraint-based planning has proven to be a powerful framework [1]. Modern constraint algorithms [2], [3] and solvers [4], [5] enable practical solutions to many computationally hard (e.g., NP-complete) problems, and various planners have exploited constraint-solving to achieve scalability [6], [7], [8]. However, constraint-based planners have previously focused on deterministic cases. When the robot faces uncertainty, such as sensor noise or classification error, we must also incorporate probabilistic data into the planning process.

We extend the constraint based planning framework to enable planning in uncertain conditions. Our method alternatives between symbolic and probabilistic reasoning. First, we construct and solve a symbolic planning problem using an iteratively relaxed threshold on uncertain observations. Then, we compute the current plan's probability of success. We continue this process of repeatedly generating candidate plans and ranking them by success probability, terminating when no different candidates exist or we exceed a timeout. At any point, our planner can return the currently-found plan that is most likely to succeed.

*a) Related work:* A common approach for planning under uncertainty is to construct and solve a Partially Observable Markov Decision Process (POMDP), producing a policy over a probability distribution across all states. While POMDPs provide general capabilities to reason about uncertainty, the large dimensionality of belief space computation presents challenges for scalability [9]. Our approach address the scalability challenge by leveraging highly-engineered constraint solvers to compute an anytime solution for the most likely to succeed plan.

## II. Approach

Our algorithm computes the most-likely-to-succeed plan given uncertain information about the start state. Figure 1 outlines the algorithm's steps. First, we convert the uncertain information about the start state to symbolic expressions for a state based on probability thresholds. Then, we use a constraint solver for Satisfiability Modulo Theories (SMT) to find a candidate plan. If the SMT solver identifies a valid plan for any start state, we compute the plan's probability of sucess by (1) propagating probabilistic states based on the candidate plan for each timestep through a Dynamic Bayesian Network (DBN) and (2) using a Markov Logic Network (MLN) to find the probability that we reach the goal. The algorithm records the plan with the highest probability of success. We continue to generate canidate plans until there are no more candidate plans left (i.e., the solver returns UNSAT). Once there are no more plans, the algorithm relaxes the thresholding bounds, calculates a new symbolic start state, and then repeats the process.

*a) Symbolic Planning:* Symbolic planning finds a sequence of actions from an initial state to a set of goal states. We consider problems of the following form.

1) State space $Q$, expressed as the product of Boolean variables: $Q = f_0 \times f_1 \times \ldots \times f_n$
2) Actions $U$, each with preconditions and effects on the state
3) Transition relation $T = Q \times U \times Q \mapsto \mathbb{B}$, which holds when the current state, current action, and next state are allowable based on the action's preconditions and effects.
4) a start state $q_s \in Q$
5) a set of goal states $q_g \subseteq Q$

We extend constraint-based planning, which plans solving a set of constraints (e.g., a Boolean formula).

*b) Bayesian Network Representation:* We compute the probability of fluents over successive steps using a probabilistic graphical model in the form of a Dynamic Bayesian Network (DBN) [10] to define dependencies between fluents and actions. We construct the DBN relating the current and next state by showing that the probability of any fluent is dependent on any action that could modify the fluent— i.e., the action's effect sets the fluent to true or false in a Boolean representation. We calculate the probability of an action $u_j$ as:

$$P(u_j) = P(i_j)P(p_j) \tag{1}$$

where $P(p_j)$ is the probability of the precondition of $u_j$ holding and $P(i_j)$ is the intent to perform the action, i.e., $P(i_j) = 1$ when the action is taken in the candidate plan and $P(i_j) = 0$ otherwise. The probability of any fluent remains unchanged if no actions are performed to modify the fluent (eg. $p(f_i^{[k+1]}) = p(f_i^{[k]})$ if no actions are performed that modify the value of $p(f_i)$). If an action is performed that modifies the value of $p(f_i)$ at a timestep then the value of $p(f_i^{[k+1]}|u_j^{[k]})$ the next timestep is either 1 or 0 depending on if the action modifies the value of the fluent to be true or false respectively.

* This work supported by ARL DCIST and NSF CNS-1823245.

† Department of Computer Science, Colorado School of Mines, USA. mschack@mymail.mines.edu, ndantam@mines.edu.

We iteratively expand this DBN over each step of the plan to compute the probability for each fluent at each step given the previous state and action.

*c) Markov Logic Network Conversion:* To address state uncertainty, we construct a probabilistic representation for transition relation $T$ in the form of a Markov Logic Network (MLN) [11]. We construct the MLN for $T$ through following conversion: [11]:

1) Convert $T$ to conjunctive normal form.
2) Replace ANDs with mutiplication of the arguments

$$a \wedge b \quad \rightsquigarrow \quad p(a) * p(b) \tag{2}$$

3) Replace ORs with the following expression

$$a \vee b \quad \rightsquigarrow \quad (1 - (1 - p(a)) * (1 - p(b))) \tag{3}$$

4) Replace NOTs with 1 - the arguments

$$\neg a \quad \rightsquigarrow \quad (1 - p(a)) \tag{4}$$

When these 3 steps are applied the transition function, rather than returning whether a transition is valid, now returns the probability that a transition is valid. Similarly, fluents and actions, rather than being Booleans, now are real numbers representing the probability the fluent is true or action is taken.

We compute the probability that a plan succeeds by calculating the probability that all the transitions hold and that the final state is the goal state.

$$p(s) = T(q_s, u^{[0]}, q^{[1]}) * ... T(q^{[n-1]}, u^{[n-1]}, q^{[n]}) * p(q_g) \tag{5}$$

Equation (5) is the metric by which we compare plans.

*d) Feedback planning with Bayesian and Markov Logic networks:* Our feedback planning algorithm takes the same inputs as a symbolic planning problem, except that the start state is the probability of a set of fluents being true rather than a deterministicly known set of fluents. To construct the symbolic constraints from these probabilities, we use two thresholds: a true threshold and a false threshold. Any probability above the true threshold sets the fluent to (deterministicly) true, and any probability below the false threshold sets the fluent to (deterministicly) false. If the probability is between thresholds, the fluent is unconstrained (and can thus take any value) resulting in a symbolic expression for the start state. As the algorithm iterates through candidate plans, we relax the threshold constraints (true theshold approaches 1 and false threshold approaches 0) so that more fluents fall between thresholds and are thus unconstrained. By iteratively relaxing constraints, we first consider candidate plans more likely to be applicable at the start before considering other plans that are less likely to hold.

We pass the symbolic start state to the constraint solver to generate a candidate plan. Then, we evaluate success probability using the DBN to propagate probabilities of fluents holding given the actions and the MLN to determine the probability that all the transitions hold and that the final state is the goal state. When there are no more plans to evaluate, the theshold constraints are relaxed, and we consider and evaluate more candidate plans.

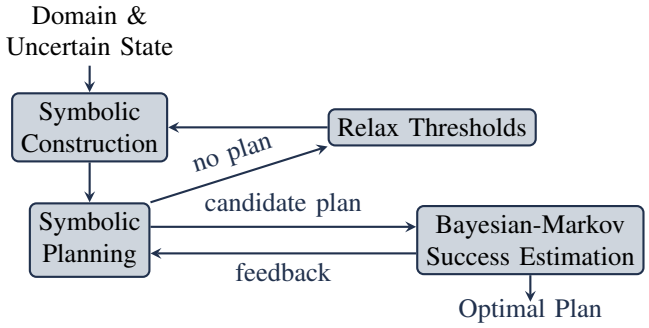


Fig. 1: Feedback Planning with Bayesian and Markov Logic Networks.

## III. Simulated Test Case

We tested this algorithm on a probabilistic variation of the blocksworld domain [12] using simulated probabilities. The algorithm produced optimal (most-likely-to-succeed) plans even when all true fluents in the start state were asserted with a probability of 0.6 and all false fluents were asserted with a probability of 0.4. This result demonstrates that the algorithm can produce optimal plans under state uncertainty.

## IV. Conclusion and Future Work

We have presented an algorithm using iterative threshold relaxation with Markov logic and Dynamic Bayesian networks to account for uncertainty and leverage scalable constraint solving techniques. In the future, we will evaluate this method on more complicated domains, using a physical robot, and for a human-robot collaboration scenario.

## References


[1] H. A. Kautz and B. Selman, "Planning as satisfiability." *Eu. Conf. on Artificial Intelligence (ECAI)*, vol. 92, pp. 359–363, 1992.

[2] J. P. Marques-Silva and K. A. Sakallah, "Grasp: A search algorithm for propositional satisfiability," *IEEE Transactions on Computers*, vol. 48, no. 5, pp. 506–521, 1999.

[3] M. W. Moskewicz, C. F. Madigan, Y. Zhao, L. Zhang, and S. Malik, "Chaff: Engineering an efficient sat solver," in *Proceedings of the 38th annual Design Automation Conference*. ACM, 2001, pp. 530–535.

[4] L. De Moura and N. Bjørner, "Z3: An efficient SMT solver," in *Tools and Algorithms for the Construction and Analysis of Systems*. Springer, 2008, pp. 337–340.

[5] C. Barrett, C. L. Conway, M. Deters, L. Hadarean, D. Jovanović, T. King, A. Reynolds, and C. Tinelli, "CVC4," in *Int. Conf. on Computer-Aided Verification (CAV)*. Springer, 2011, pp. 171–177.

[6] H. Kautz and B. Selman, "Blackbox: A new approach to the application of theorem proving to problem solving," in *AIPS98 Workshop on Planning as Combinatorial Search*, vol. 58260, 1998, pp. 58–60.

[7] J. Rintanen, "Madagascar: Scalable planning with sat," in *8th International Planning Competition (IPC-2014)*, 2014, pp. 66–70.

[8] N. T. Dantam, Z. K. Kingston, S. Chaudhuri, and L. E. Kavraki, "An incremental constraint-based framework for task and motion planning," *International Journal of Robotics Research, Special Issue on the 2016 Robotics: Science and Systems Conference*, vol. 37, no. 10, pp. 1134–1151, 2018.

[9] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, 2005.

[10] D. Koller and N. Friedman, *Probabilistic Graphical Models: Principles and Techniques - Adaptive Computation and Machine Learning*. The MIT Press, 2009.

[11] M. Richardson and P. Domingos, "Markov logic networks," *Machine Learning*, vol. 62, no. 1, pp. 107–136, Feb 2006. [Online]. Available: https://doi.org/10.1007/s10994-006-5833-1

[12] *AIPS'00: Proceedings of the Fifth International Conference on Artificial Intelligence Planning Systems*. AAAI Press, 2000.