Policy Networks: A Framework for Scalable Integration of Multiple Decision-Making Models

Extended Abstract

Kyle Hollins Wray
University of Massachusetts Amherst
wray@cs.umass.edu

Shlomo Zilberstein
University of Massachusetts Amherst
shlomo@cs.umass.edu

ABSTRACT

Policy networks are graphical models that integrate decision-making models. They allow for multiple Markov decision processes (MDPs) that describe distinct focused aspects of a domain to work in harmony to solve a large-scale problem. This paper defines policy networks and shows how they are able to naturally generalize many previous models, such as options and constrained MDPs.

ACM Reference Format:
Kyle Hollins Wray and Shlomo Zilberstein. 2019. Policy Networks: A Framework for Scalable Integration of Multiple Decision-Making Models. In *Proc. of the 18th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2019), Montreal, Canada, May 13–17, 2019,* IFAAMAS, 3 pages.

1 INTRODUCTION

The Markov decision process (MDP) [4] and its many variants are sequential decision-making models that are increasingly deployed in large high-impact domains ranging from aircraft collision avoidance [10] to autonomous vehicles [19, 21]. Efforts to accurately represent these important real world problems has highlighted the inability for a single monolithic model—one state-action space for one objective—to scale, instead relying on a fragmented collection of techniques. Multi-objective approaches allow for more than one objective to be considered, either by combining them into a single objective via scalarization [14], defining a preference ordering with slack via a lexicographic MDP (LMDP) [22, 24], or stating multiple constraining objectives for a primary objective via a constrained MDP (CMDP) [1]. Hierarchical approaches allow for reasoning at varying levels of abstraction, either by hierarchical task networks (HTN) [8, 15], defining subtask hierarchies for MDPs via MAXQ-like techniques [7, 9], or leveraging another policy by temporarily transferring control to it via the options framework [3, 17].

Each one of these solutions introduces an important reasoning capability, but to support long-term autonomy [2] in the real world, we need to integrate these varied capabilities within one system [5]. As Marvin Minsky observed, "the power of intelligence stems from our vast diversity, not from any single, perfect principle" [11]. It is unlikely that any single MDP model will be sufficient. For scalability, we need new formal architectures that allow multiple models to be integrated within a single agent. To this end, we propose a novel framework called policy networks that unifies prior approaches and provides a solid foundation on which to build the next generation of large-scale models.

Proc. of the 18th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2019), N. Agmon, M. E. Taylor, E. Elkind, M. Veloso (eds.), May 13–17, 2019, Montreal, Canada. © 2019 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org). All rights reserved.

2 POLICY NETWORKS

In general, policy networks are graphs in which vertices denote sets of policies for a reward function and edges denote policy dependences among them. The objective is to capture the relations among distinct decision-making components to solve large multi-objective hierarchical problems. Thus, a **policy network** is a sequential decision-making model defined by a directed graph $\langle V, E\rangle$ [18]:

- V is a finite set of vertices such that each $v \in V$ denotes a set of policies Π_v for reward function $R_v : S_v \times A_v \rightarrow \mathbb{R}$; and
- E is a finite set of edges such that each $\langle v, w\rangle = e \in E$ forms a dependence of w on v, with optional properties:
  - policy constraint Π_e enforces that $\pi_w \in \Pi_e$, for the policy $\pi_w \in \Pi_w$ chosen for w; and/or
  - policy transition $T_e : S_v \times A_v \times S_w \rightarrow [0, 1]$ is a partial function for $Pr(w, s'_w | v, s_v, a_v)$.

The execution of a policy network operates over discrete time steps $t \in \mathbb{N}$ as a form of *Markov multi-reward process.* Each vertex v has a state space S_v and action space A_v for its policy and reward; it also has an initial state $s_v^0 \in S_v$. Each edge $e = \langle v, w\rangle$ makes w inherently depend on v such that when v performs an action or transitions its state it can affect w.

As in (PO)MDPs, to **perform** an action is simply the act of conditioning on the action so as to induce an update in the underlying vertex v's stochastic process following the distribution $Pr(w, s'_w | v, s_v, a_v)$; this is called a **state transition**. This probability distribution describes the state transition within the state space of v (i.e., $w = v$ and $s'_w = s'_v \in S_v$) as well as across other state spaces used by other vertices (i.e., $w \neq v$ and $s'_w \in S_w$). Policy networks require full specification of $Pr(w, s'_w | v, s_v, a_v)$ via the collection of functions T_e. In its simplest form, if v only transitions to itself by $T_e = T_{vv}$, then T_e is equivalent to a typical (PO)MDP state transition. Performing an action also induces a reward from R_v.

At each time step, any **controller** vertex v performs the action $\pi_v(s_v) \in A_v$ at their current state $s_v \in S_v$ from a policy $\pi_v \in \Pi_v \cap (\bigcap_w \Pi_{wv})$ chosen for v. Each policy network has an initial controller $v^0 \in V$. The actions performed by v may result in a stochastic state transition to a different vertex w's state space. We call this a **transfer of control**, with the controller changing from v to w who now performs actions following its policy. Obviously, only a controller vertex can transfer control. If a non-controller vertex performs a state update, transfer of control attempts instead result in a self-loop state transition.

For any edge $\langle v, w\rangle = e \in E$, there is an inherent dependence that w has on v. First, if v is the controller and the state spaces are shared $S_w = S_v$, then w also follows the state transition result of v (i.e., $s_w^{t+1} = s_v^{t+1}$). Second, if v is the controller and the action spaces

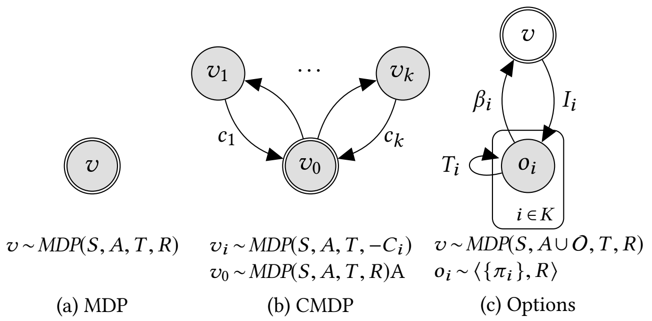
v
$v \sim MDP(S, A, T, R)$
(a) MDP
v_1
$\cdots$
v_k
c_1
c_k
v_0
$v_i \sim MDP(S, A, T, -C_i)$
$v_0 \sim MDP(S, A, T, R)A$
(b) CMDP
v
β_i
I_i
T_i
o_i
$i \in K$
$v \sim MDP(S, A \cup O, T, R)$
$o_i \sim \langle \{\pi_i\}, R \rangle$
(c) Options

Figure 1: Three policy network graphical representations.

are shared $A_w = A_v$, then w also performs the action that v performs (i.e., $a_w^t = a_v^t$); performing an action in this way also emits a reward R_w for w. Thus, if $S_w = S_v$ and $A_w = A_v$ then w performs action $a_w^t = a_v^t$ and the successor state is $s_w^{t+1} = s_v^{t+1}$. However, if $S_w \neq S_v$ and $A_w = A_v$ then w still performs action $a_w^t = a_v^t$ and induces a state transition as normal, with a caveat that any transfer of control attempt self-loops instead. Any additional dependences can be optionally added to an edge as well, as described above.

For each vertex v, we can define its relative Markov reward process for a set of policies. This defines a hierarchy of relative constrained semi-MDPs (CSMDPs) [1, 12] as an optimality criterion.

3 THEORETICAL ANALYSIS

We can show the generality of policy networks by proving that they can encapsulate various models such as CMDPs and the options framework. First, Proposition 3.1 states that they generalize MDPs; partially observable MDPs (POMDPs) [16] are continuous state MDPs, *belief MDPs*, and are similarly generalized.

Proposition 3.1. *Policy networks generalize (PO)MDPs.*

Next, we consider two cases of *policy constraint edges*: constrained MDPs (CMDPs) [1] in Proposition 3.2 and MODIA [21] in Proposition 3.3. Constraints limit the space of policies from parent vertices to a child controller vertex. CMDPs represent a policy network with a shared both state and action space. Figure 1(b) provides a graphical representation. MODIA represents a policy network with a different state space but a shared action space, illustrating how performing action can simultaneously affect many models.

Proposition 3.2. *Policy networks generalize CMDPs.*

Proposition 3.3. *Policy networks generalize MODIA.*

Lastly, we consider two related cases of *policy transition edges*: the options framework [17] (both Markov and semi-Markov) in Proposition 3.4 and semi-autonomous systems (SAS) [19] in Proposition 3.5. Here, transfer of control happens between parent and child vertices, both online (options) and offline (SAS). Options represent a policy network with a shared state space and shared action space. Figure 1(c) provides a graphical representation. SAS represents a policy network with different state space and different action space, illustrating how different models can interact.

Proposition 3.4. *Policy networks generalize options.*

Proposition 3.5. *Policy networks generalize SAS.*

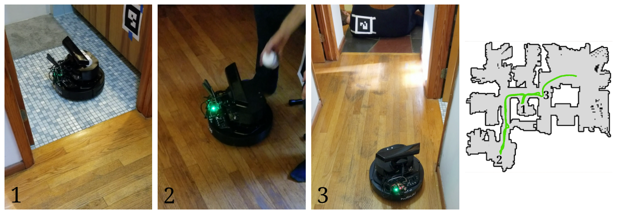

Figure 2: Experiments with the home healthcare robot using this policy network in the real household shown above. Three highlights are shown: (1) medicine retrieval for task t_1, (2) medicine delivery completion with transfer $t_1 \to h \to t_2$, and (3) interruption of cleaning task t_2 by detecting a fall with task f_i and calling for assistance.

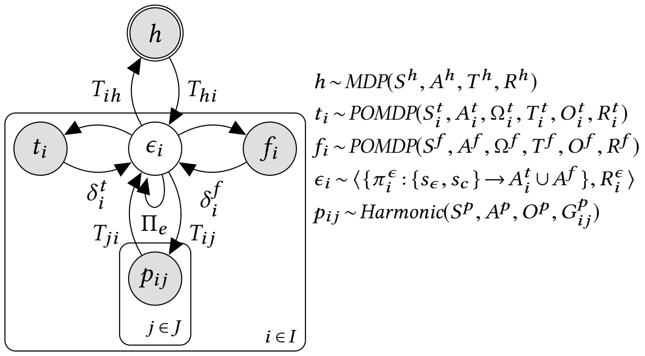
h
T_{ih}
T_{hi}
t_i
ϵ_i
f_i
δ_i^t
δ_i^f
Π_e
T_{ji}
T_{ij}
p_{ij}
$j \in J$
$i \in I$
$h \sim MDP(S^h, A^h, T^h, R^h)$
$t_i \sim POMDP(S_i^t, A_i^t, \Omega_i^t, T_i^t, O_i^t, R_i^t)$
$f_i \sim POMDP(S^f, A^f, \Omega^f, T^f, O^f, R^f)$
$\epsilon_i \sim \langle \{\pi_i^\epsilon : \{s_\epsilon, s_c\} \to A_i^t \cup A^f\}, R_i^\epsilon \rangle$
$p_{ij} \sim Harmonic(S^p, A^p, O^p, G_{ij}^p)$

Figure 3: The policy network for the home healthcare robot.

4 EVALUATION

Home healthcare robots serve in household and eldercare scenarios, providing solutions to a wide array of helpful tasks ranging from cleaning to medicine delivery [13]. We focus on a robot solution that captures the three top-ranked needs [6]: (1) medicine notification and delivery; (2) cleaning; and (3) monitoring and helping with falls. Figure 3 shows the policy network for our healthcare robot. The vertices are a task selector h, tasks t_i (medicine and cleaning), fall monitor f_i, executor ϵ_i, and path planner p_{ij}. Tasks are solved using *nova* [23] and path planning uses harmonic function solver *epic* [20]. Figure 2 shows this policy network's successful execution on a real robot acting in a home environment.

5 CONCLUSION

Policy networks represent a principled mathematical model that enables the integrated design of multiple models. They provide a formal approach to generalize select state-of-the-art models and use them within a single reasoning system. The implementation of policy networks demonstrates that they can be used effectively to model and solve a challenging home healthcare robot problem. This work lays the foundation for scalable integration of multiple models in support of reasoning about complex real-world domains and long-term autonomy. The full description, proofs, and formalization is provided in Wray's thesis [18].

Acknowledgements. This research was supported by the National Science Foundation grant number IIS-1724101.

REFERENCES

[1] Eitan Altman. 1999. *Constrained Markov decision processes*. Chapman & Hall/CRC Press, England.

[2] Tim Barfoot, Jonathan Kelley, and Gabe Sibley. 2013. Special Issue on Long-Term Autonomy. *The International Journal of Robotics Research* 32, 14 (2013), 1609–1610.

[3] Andrew G. Barto and Sridhar Mahadevan. 2003. Recent advances in hierarchical reinforcement learning. *Discrete Event Dynamic Systems* 13, 4 (2003), 341–379.

[4] Richard E. Bellman. 1957. *Dynamic Programming*. Princeton University Press, Princeton, NJ.

[5] Joydeep Biswas and Manuela M. Veloso. 2013. Localization and navigation of the CoBots over long-term deployments. *International Journal of Robotics Research* 32, 14 (2013), 1679–1694.

[6] Elizabeth Broadbent, Rie Tamagawa, Ngaire Kerse, Brett Knock, Anna Patience, and Bruce MacDonald. 2009. Retirement home staff and residents' preferences for healthcare robots. In *Proceedings of the 18th IEEE International Symposium on Robot and Human Interactive Communication*. 645–650.

[7] Thomas G. Dietterich. 2000. Hierarchical reinforcement learning with the MAXQ value function decomposition. *Journal of Artificial Intelligence Research* 13 (2000), 227–303.

[8] Kutluhan Erol. 1996. *Hierarchical Task Network Planning: Formalization, Analysis, and Implementation*. Ph.D. Dissertation. University of Maryland, College Park, MD.

[9] Nakul Gopalan, Marie desJardins, Michael L. Littman, James MacGlashan, Shawn Squire, Stefanie Tellex, John Winder, and Lawson L.S. Wong. 2017. Planning with Abstract Markov Decision Processes. In *Proceedings of the International Conference on Automated Planning and Scheduling*. 480–488.

[10] Mykel J. Kochenderfer. 2015. *Decision Making Under Uncertainty: Theory and Application*. MIT Press.

[11] Marvin Minsky. 1986. *The Society of Mind*. Simon and Schuster, New York, NY.

[12] Martin L. Puterman. 1994. *Markov decision processes: Discrete stochastic dynamic programming*. John Wiley & Sons, New York, NY.

[13] Hayley Robinson, Bruce MacDonald, and Elizabeth Broadbent. 2014. The Role of Healthcare Robots for Older People at Home: A Review. *International Journal of Social Robotics* 6, 4 (November 2014), 575–591.

[14] Diederik M. Roijers, Peter Vamplew, Shimon Whiteson, and Richard Dazeley. 2013. A survey of multi-objective sequential decision-making. *Journal of Artificial Intelligence Research* 48 (2013), 67–113.

[15] Earl D. Sacerdoti. 1974. Planning in a hierarchy of abstraction spaces. *Artificial Intelligence* 5, 2 (1974), 115–135.

[16] Richard D. Smallwood and Edward J. Sondik. 1973. The optimal control of partially observable Markov processes over a finite horizon. *Operations Research* 21, 5 (1973), 1071–1088.

[17] Richard S. Sutton, Doina Precup, and Satinder Singh. 1999. Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence* 112, 1-2 (1999), 181–211.

[18] Kyle Hollins Wray. 2019. *Abstractions in Reasoning for Long-Term Autonomy*. Ph.D. Dissertation. University of Massachusetts, Amherst, MA.

[19] Kyle Hollins Wray, Luis Pineda, and Shlomo Zilberstein. 2016. Hierarchical Approach to Transfer of Control in Semi-Autonomous Systems. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence*. 517–523.

[20] Kyle Hollins Wray, Dirk Ruiken, Roderic A. Grupen, and Shlomo Zilberstein. 2016. Log-Space Harmonic Function Path Planning. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 1511–1516.

[21] Kyle Hollins Wray, Stefan J. Witwicki, and Shlomo Zilberstein. 2017. Online Decision-Making for Scalable Autonomous Systems. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*. 4768–4774.

[22] Kyle Hollins Wray and Shlomo Zilberstein. 2015. Multi-Objective POMDPs with Lexicographic Reward Preferences. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence*. 1719–1725.

[23] Kyle Hollins Wray and Shlomo Zilberstein. 2015. A Parallel Point-Based POMDP Algorithm Leveraging GPUs. In *Proceedings of the 2015 AAAI Fall Symposium on Sequential Decision Making for Intelligent Agents*. 95–96.

[24] Kyle Hollins Wray, Shlomo Zilberstein, and Abdel-Illah Mouaddib. 2015. Multi-Objective MDPs with Conditional Lexicographic Reward Preferences. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence*. 3418–3424.