Energy Efficient Execution of POMDP Policies

Marek Grześ   Pascal Poupart, Xiao Yang, Jesse Hoey

University of Kent   UNIVERSITY OF WATERLOO

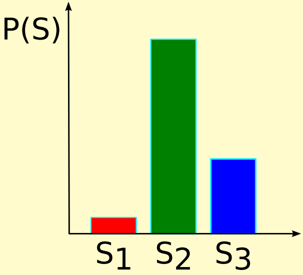
ICAPS 2015

Jerusalem, Israel, June 7–11

POMDP

Partially Observable Markov Decision Process

- an elegant framework for planning under uncertainty
- allows for uncertainty about transitions, sensors, and the state of the world



$$b = \left[P(s_1), P(s_2), P(s_3)\right]$$

POMDP—Policy Representation using α-vectors

$$\begin{bmatrix}\alpha_1(s_1)\\ \alpha_1(s_2)\\ \alpha_1(s_3)\end{bmatrix}, \quad \begin{bmatrix}\alpha_2(s_1)\\ \alpha_2(s_2)\\ \alpha_2(s_3)\end{bmatrix}, \quad \ldots, \quad \begin{bmatrix}\alpha_N(s_1)\\ \alpha_N(s_2)\\ \alpha_N(s_3)\end{bmatrix}$$

$$a_1 \qquad a_2 \qquad \ldots \qquad a_N$$

$$b = \Big[P(s_1), P(s_2), P(s_3)\Big]$$

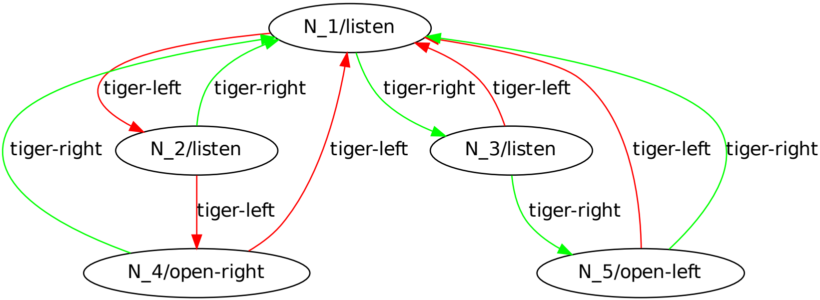
$$a(b) = a_k, \;\text{ s.t. }\; k = \arg\max_i \Big(b \cdot \alpha_i\Big)$$

Policy Execution

b ← an initial belief
while (true)
    action ← determine an action for b using alpha vectors
    execute the action
    read observations from sensors
    update b using observations and the action
end

Policy Execution in the Cloud

Finite State Controllers



NB: No need to do a belief update nor to consult alpha-vectors.

Battery Efficiency Experiment

- A POMDP, lacasa4.batt, with 2880 states, 72 observations, and 6 actions (designed and generated using WEB-SNAP[1]; Grzes *et al.* IJAR 2014)
- Installed on a smart phone to assist a cognitively disabled person; allows the patient to enjoy her walk, helps find way home when lost, or calls a care giver when necessary
- **When the patient is lost and the phone runs out of battery, the application is not available**
- Relying on cloud computing requires a reliable network connection and server infrastructure

[1] https://bitbucket.org/mgrzes/web-snap

Battery Efficiency on the Nexus 4 Phone

Experiment	1% battery depletion time (minutes)	standard error
OS only	6.74	0.13
OS with WIFI	6.69	0.08
Observation generator	6.40	0.26
Constant policy	5.82	0.21
FSC	5.71	0.18
Client/Server (cloud)	5.52	0.22
Flat policy	4.10	0.11
Symbolic Perseus	3.91	0.15

Policies for lacasa4.batt evaluated on the Nexus 4 phone. Every policy was queried once per second (time interval 1 second).

Battery Efficiency on the Nexus 4 Phone *(continued)*

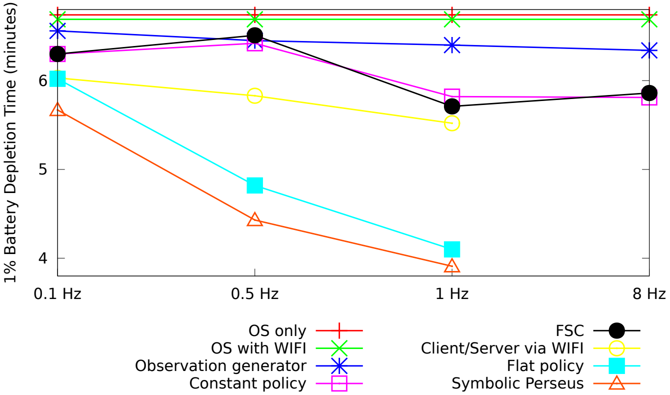
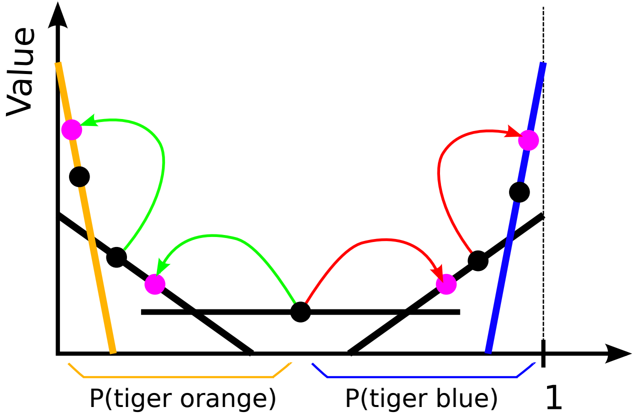


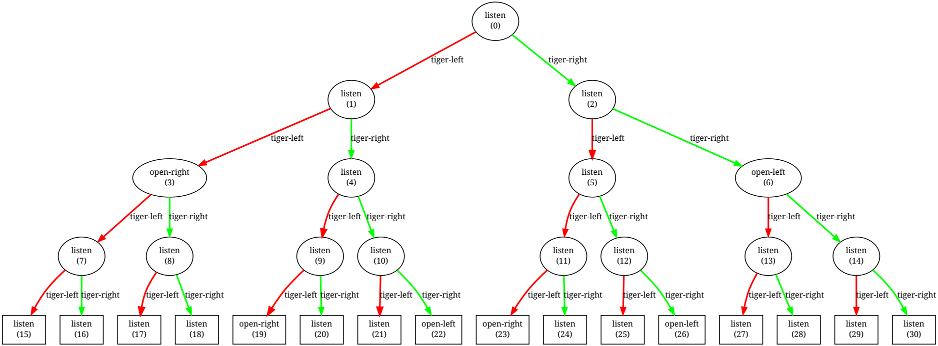
Figure : Battery consumption results on the Nexus 4 phone. Missing values indicate that the policy did not return fast enough for a given frequency.

Policy Compilation—Compile Existing Policies into Finite-state Controllers

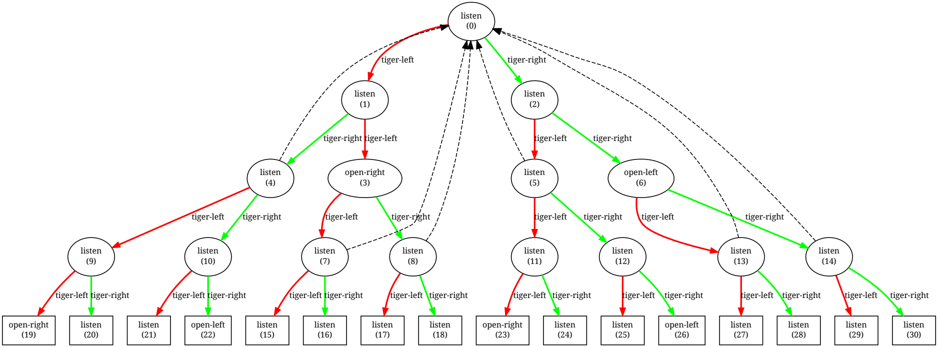
Controller Compilation Using Alpha Vectors



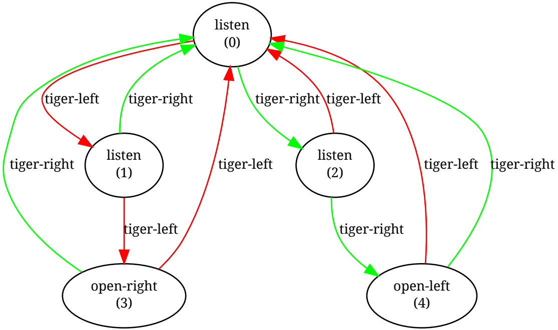
Controller Compilation Using Policy Trees: (1) Policy Tree



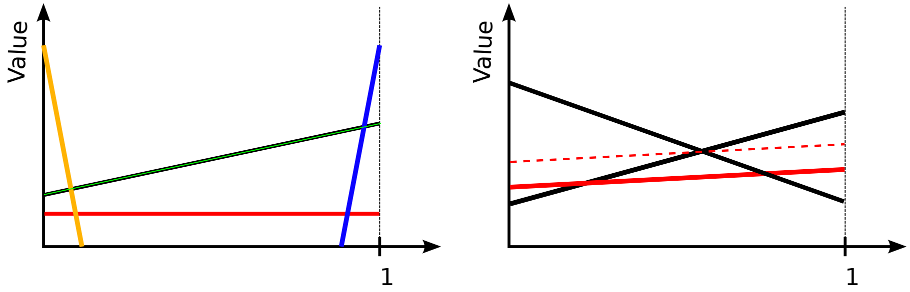
Controller Compilation Using Policy Trees: (2) Identical Conditional Plans



Controller Compilation Using Policy Trees: (3) Removing Repeated Nodes



Controller Compilation: More Pruning of Redundant Nodes



Controller Compilation Results (1)

POMDP	GapMin	method	depth	tree size	nodes	value	time	c
chainOfChains3	GM-lb=157	**alpha2fsc**	10	10	10(10)	**157**(157)	0.26	0
$\vert S\vert=10$, $\vert A\vert=4$	GM-ub=157	**GM-LB**	11	11	10(10)	**157**(157)	0.42	0
$\vert O\vert=1$, $\gamma = 0.95$	time=0.86s	**GM-UB**	11	11	10(10)	**157**(157)	0.26	0
	$\vert lb\vert=10$	**B&B**			10	**157**	1.69	
	$\vert ub\vert=1$	EM			10	0.17 ± 0.06	6.9	
		QCLP			10	0 ± 0	0.16	
		BPI			10	25.7 ± 0.77	4.25	
cheese-taxi	GM-lb=2.481	**alpha2fsc**			17(22)	**2.476**(2.476)	0.29	1
$\vert S\vert=34$, $\vert A\vert=7$	GM-ub=2.481	**GM-LB**	15	167	17(24)	**2.476**(2.476)	0.56	1
$\vert O\vert=10$, $\gamma = 0.95$	time=1.88s	**GM-UB**	15	167	17(24)	**2.476**(2.476)	0.55	1
	$\vert lb\vert=22$	B&B			10	-19.9^*	24h	
	$\vert ub\vert=13$	EM			17	-12.16 ± 2.08	337.9	
		QCLP			17	-18.22 ± 1.77	227.4	
		BPI			16	-18.1 ± 0.39	7.18	
lacasa4.batt	GM-lb=291.1	alpha2fsc			10(10)	285.5(285.5)	302	0
$\vert S\vert=2880$, $\vert A\vert=6$	GM-ub=292.6	GM-LB	3	745	19(22)	287.3(287.1)	3652	1
$\vert O\vert=72$, $\gamma = 0.95$	time=8454s	**GM-UB**	4	23209	87(94)	**290.8**(290.8)	3681	1
	$\vert lb\vert=10$	B&B			10	285.0^*	24h	
	$\vert ub\vert=23$	EM			3	290.2 ± 0.0	19920	
		BPI			6	290.6 ± 0.2	4124	
machine	GM-lb=62.38	alpha2fsc			5(39)	54.61(54.09)	5.53	1
$\vert S\vert=256$, $\vert A\vert=4$	GM-ub=66.32	GM-LB	9	376	26(41)	62.92(62.84)	18.5	1
$\vert O\vert=16$, $\gamma = 0.99$	time=3784s	**GM-UB**	12	2864	11(159)	**63.02**(60.29)	86.8	2
	$\vert lb\vert=39$	B&B			6	62.6	52100	
	$\vert ub\vert=243$	EM			11	62.93 ± 0.03	1757	
		QCLP			11	62.45 ± 0.22	4636	
		BPI			10	35.7 ± 0.52	2.14	

Controller Compilation Results (2)

POMDP	SARSOP	method	depth	tree size	nodes	value	time	c
baseball	time 122.7s	**policy2fsc**	7	175985	10(47)	**0.641**(0.641)	78.22	1
$\lvert S\rvert$=7681, $\lvert A\rvert$=6	$\lvert\alpha\rvert$ =1415	B&B			5	0.636*	24h	
$\lvert O\rvert$=9, $\gamma = 0.999$	UB=0.642	EM			2	0.636 $\pm$ 0.0	48656	
	LB=0.641	BPI			9	0.636 $\pm$ 0.0	445	
elevators_inst_pomdp_1	time 11,228s	**policy2fsc**	11	419	20(24)	**-44.41**(-44.41)	1357	1
$\lvert S\rvert$=8192, $\lvert A\rvert$=5	$\lvert\alpha\rvert$ =78035	B&B			10	-149.0*	24h	
$\lvert O\rvert$=32, $\gamma = 0.99$	UB=-44.31							
	LB=-44.32							
tagAvoid	time 10,073s	**policy2fsc**	28	7678	91(712)	**-6.04**(-6.04)	582.2	1
$\lvert S\rvert$=870, $\lvert A\rvert$=5	$\lvert\alpha\rvert$ =20326	B&B			10	-19.9*	24h	
$\lvert O\rvert$=30, $\gamma = 0.95$	UB=-3.42	EM			9	-6.81 $\pm$ 0.12	19295	
	LB=-6.09	QCLP			2	-19.99 $\pm$ 0.0	12.9	
		BPI			88	-12.42 $\pm$ 0.13	1808	
underwaterNav	time 10,222s	policy2fsc	51	1242	52(146)	745.3(745.3)	5308	1
$\lvert S\rvert$=2653, $\lvert A\rvert$=6	$\lvert\alpha\rvert$ =26331	B&B			10	747.0*	24h	
$\lvert O\rvert$=103, $\gamma = 0.95$	UB=753.8	**EM**			5	**749.9** $\pm$ 0.02	31611	
	LB=742.7	BPI			49	748.6 $\pm$ 0.24	14758	
rockSample-7_8	time 10,629s	**policy2fsc**	31	2237	204(224)	**21.58**(21.58)	1291	1
$\lvert S\rvert$=12545, $\lvert A\rvert$=13	$\lvert\alpha\rvert$ =12561	B&B			10	11.9*	24h	
$\lvert O\rvert$=2, $\gamma = 0.95$	UB=24.22	BPI			5	7.35 $\pm$ 0.0	78.8	
	LB=21.50							

Table : Compilation and compression of SARSOP policies.

Conclusion

1. Energy consumption can limit deployment of POMDPs on mobile devices
2. **Finite-state controllers are the cheapest POMDP policies to execute**
3. Two methods to compile POMDP policies into finite-state controllers were shown
4. Compilation is more robust against local optima than direct optimization with local search
5. Compilation is feasible on large problems where exhaustive search with branch-and-bound can be challenging
6. **More time spent during off-line POMDP planning may lead to huge battery/energy savings when the policy is deployed on thousands of mobile phones**
7. **For recent advancements on direct optimisation see Grzes & Poupart AAMAS 2015**