Finding Objects through Stochastic Shortest Path Problems

Manuela Veloso

Thanks to Felipe Trevizan

School of Computer Science

Carnegie Mellon University

PEL, Fall 2016

CoBot: Autonomous Service Robot

Veloso, Biswas, Coltin, Rosenthal IJCAI'15

Motivation

- An autonomous agent moving in a **known environment** in order to **find an object** while **minimizing** the search **cost**, e.g.,
  - Taxi driver looking for passengers while minimizing the usage of gas
  - Software agent finding information about a product in the web while minimizing bandwidth
  - Service robot retrieving objects to users while minimizing the traveled distance

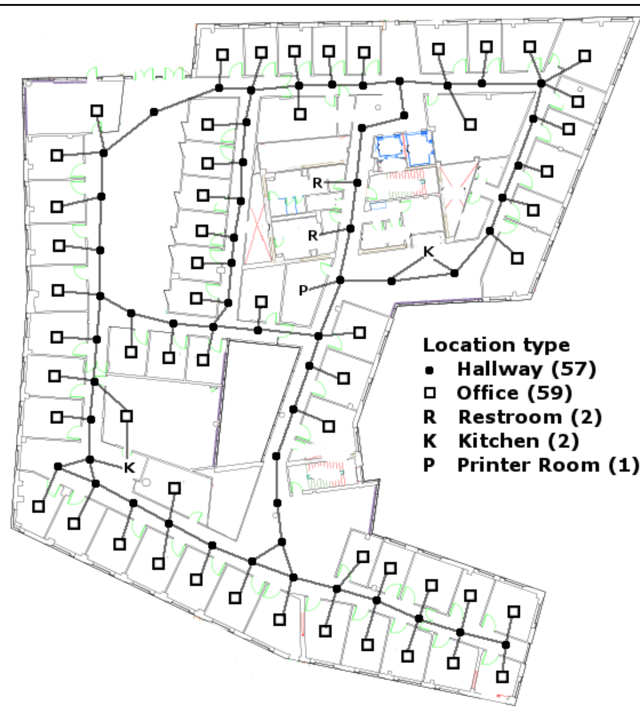
2

Querying and Learning from the Web

Search for Object Probabilistic Planning

- The probability of objects being in a given location type is obtained using **OpenEval** [Samadi et al, 2012].
- We compare 3 different probabilistic planners to find (also see paper):
  - **Coffee, Cup, Pen, Papers and Toner**

Map



Probabilistic Planners

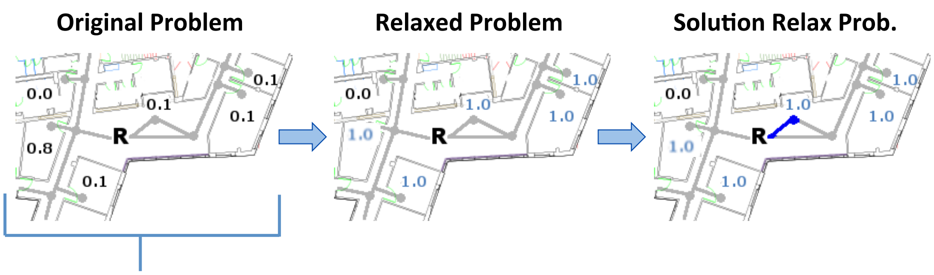
- We compared the following planners:
  - **FF-Replan** [Yoon et al, 2007]
  - **UCT** [Kocsis and Szepesvári, 2006]
  - **SSiPP** [Trevizan and Veloso, 2012]

18

FF-Replan

[Yoon et al, 2007]

- Main idea: simplify the problem by **removing the probabilities** from actions



Numbers represent the prob. of finding the object in that location

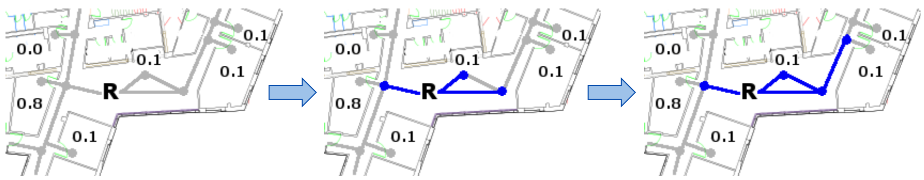
19

UCT

[Kocsis and Szepesvári, 2006]

- Main ideas:
  - **Limit** the maximum **number of actions** that can be applied to reach the goal
  - Use sparse **sampling** to search for a solution

Original Problem **Sampled Traj. (1 action)** **Sampled Traj. (2 actions)**



SSiPP

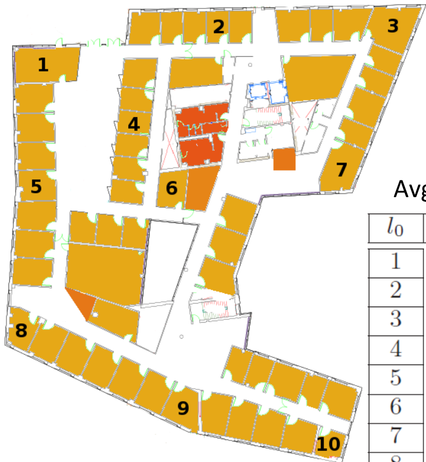
[Trevizan and Veloso, 2012]

- Main idea: **prune** states reachable only in the **far future**

Original Problem **Relaxed Problem** **Solution Relax Prob.**

Everything beyond this point is **pruned** and **heuristically estimated**

Results: Finding a Pen



Probability of finding the a pen at:

Restroom	Kitchen	Office	Printer R.
0.15	0.23	0.35	0.27

Avg and 95% conf. int. of the cost to find a pen:

l_0	FF-Replan	UCT ($c = 8$)	SSiPP ($t = 20$)
1			
2			
3			
4			
5			
6			
7			
8			
9			
10			

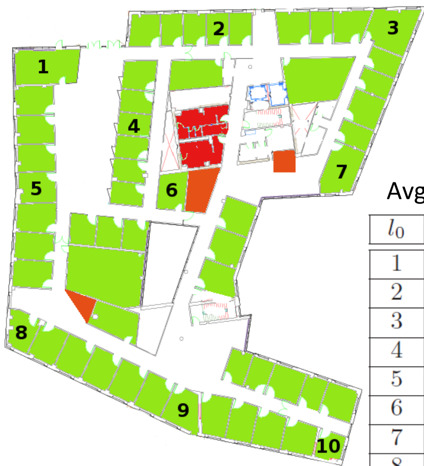
22

Results: Finding a Pen (All parameters)

			UCT $w = 1000$			SSiPP					
	l_0	FF-Replan	$c = 2$	$c = 4$	$c = 8$	$t = 10$	$t = 12$	$t = 14$	$t = 16$	$t = 18$	$t = 20$
pen	1	9.4 ±2	9.1 ±3	8.7 ±3	9.3 ±4	9.0 ±2	10.2 ±2	8.7 ±2	8.5 ±2	9.1 ±2	8.4 ±1
	2	8.8 ±2	8.9 ±4	9.0 ±2	8.7 ±3	9.8 ±2	9.2 ±2	9.8 ±2	8.5 ±1	8.9 ±2	8.9 ±2
	3	8.5 ±1	10.8 ±3	10.8 ±3	12.0 ±3	9.5 ±2	8.2 ±2	9.5 ±2	8.9 ±2	8.7 ±2	7.8 ±1
	4	8.2 ±2	9.6 ±3	10.4 ±3	9.1 ±3	9.2 ±2	8.3 ±2	9.0 ±2	8.7 ±3	9.0 ±2	8.5 ±2
	5	8.7 ±2	9.6 ±3	8.6 ±2	9.7 ±5	9.6 ±1	9.9 ±2	8.8 ±2	9.0 ±2	9.4 ±2	9.1 ±2
	6	11.1 ±3	11.0 ±3	11.7 ±2	10.8 ±3	11.0 ±2	10.7 ±1	10.6 ±2	10.0 ±2	10.1 ±2	10.0 ±2
	7	10.9 ±2	11.7 ±3	11.9 ±3	11.4 ±4	11.4 ±2	11.1 ±2	11.2 ±2	11.3 ±2	11.2 ±2	11.5 ±2
	8	10.7 ±2	10.4 ±3	10.9 ±2	10.5 ±3	10.1 ±2	11.8 ±2	8.6 ±2	10.8 ±2	10.4 ±2	10.2 ±2
	9	11.3 ±2	10.4 ±3	10.6 ±3	10.9 ±4	10.2 ±2	10.9 ±2	10.8 ±2	10.9 ±2	10.0 ±2	10.9 ±2
	10	9.7 ±2	9.3 ±2	9.9 ±2	9.7 ±2	9.4 ±2	9.8 ±2	9.5 ±2	9.6 ±2	9.9 ±2	9.5 ±2

23

Results: Finding Papers



Probability of finding papers at:

Restroom	Kitchen	Office	Printer R.
0.00	0.13	0.70	0.17

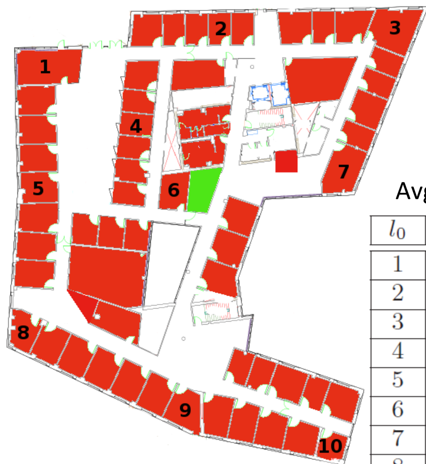
Avg and 95% conf. int. of the cost to find papers:

l_0	FF-Replan	UCT ($c = 8$)	SSiPP ($t = 20$)
1			
2			
3			
4			
5			
6			
7			
8			
9			
10			

Results: Finding Papers (All parameters)

	l_0	FF-Replan	UCT $w = 1000$			SSiPP					
			$c = 2$	$c = 4$	$c = 8$	$t = 10$	$t = 12$	$t = 14$	$t = 16$	$t = 18$	$t = 20$
papers	1	3.3 ±1	3.2 ±1	3.9 ±1	3.9 ±2	3.2 ±0	3.6 ±1	3.2 ±0	3.8 ±1	3.3 ±0	3.6 ±1
	2	3.7 ±1	3.7 ±1	3.1 ±1	4.4 ±1	4.0 ±1	3.7 ±1	4.2 ±1	3.5 ±1	3.8 ±1	3.4 ±1
	3	4.4 ±1	4.9 ±1	4.4 ±1	4.8 ±1	3.7 ±1	3.5 ±1	3.8 ±1	3.8 ±1	3.5 ±1	3.6 ±1
	4	4.4 ±1	4.3 ±1	4.7 ±1	4.9 ±3	3.6 ±1	3.7 ±1	3.5 ±1	3.5 ±1	3.6 ±1	3.7 ±1
	5	3.5 ±1	3.4 ±1	3.9 ±1	3.3 ±1	3.7 ±1	3.9 ±1	3.4 ±1	3.9 ±1	3.5 ±1	3.4 ±1
	6	3.6 ±1	3.7 ±1	3.9 ±1	3.8 ±1	3.5 ±1	3.5 ±1	3.9 ±1	3.6 ±1	3.4 ±1	3.6 ±1
	7	5.9 ±1	6.4 ±1	6.2 ±1	6.0 ±1	6.0 ±1	6.1 ±1	6.0 ±1	5.8 ±1	6.2 ±1	5.8 ±1
	8	4.7 ±1	3.9 ±1	3.5 ±1	3.8 ±1	4.4 ±1	3.5 ±1	3.9 ±1	3.6 ±1	3.6 ±1	3.7 ±1
	9	4.8 ±1	3.5 ±1	3.7 ±1	4.0 ±1	4.0 ±1	3.5 ±1	3.9 ±1	3.8 ±1	3.8 ±1	3.8 ±1
	10	3.4 ±0	3.3 ±1	4.1 ±2	3.5 ±1	3.2 ±1	3.3 ±0	3.5 ±1	3.4 ±1	3.7 ±1	3.5 ±1

Results: Finding a Toner



Probability of finding a toner at:

Restroom	Kitchen	Office	Printer R.
0.05	0.02	0.06	0.87

Avg and 95% conf. int. of the cost to find a pen:

l_0	FF-Replan	UCT ($c = 8$)	SSiPP ($t = 20$)
1			
2			
3			
4			
5			
6			
7			
8			
9			
10			

Results: Finding a Toner (All parameters)

			UCT $w = 1000$			SSiPP					
	l_0	FF-Replan	$c = 2$	$c = 4$	$c = 8$	$t = 10$	$t = 12$	$t = 14$	$t = 16$	$t = 18$	$t = 20$
toner	1	54.1 ±9	43.2 ±10	41.9 ±11	41.3 ±11	42.8 ±7	29.5 ±7	27.2 ±5	37.9 ±7	27.1 ±6	27.9 ±6
	2	56.8 ±9	41.9 ±10	45.7 ±12	40.3 ±11	41.5 ±5	19.0 ±5	18.3 ±5	18.7 ±5	18.5 ±6	18.3 ±6
	3	50.1 ±9	56.6 ±12	55.3 ±11	53.1 ±13	38.5 ±5	33.1 ±6	25.3 ±6	22.4 ±4	23.4 ±9	21.2 ±5
	4	61.3 ±9	59.3 ±10	58.0 ±12	42.2 ±11	30.2 ±9	20.7 ±6	20.5 ±6	19.1 ±7	21.3 ±7	19.3 ±7
	5	39.3 ±6	38.9 ±10	31.5 ±10	36.5 ±12	30.2 ±7	31.8 ±8	23.9 ±5	23.2 ±6	25.0 ±7	23.6 ±7
	6	53.3 ±6	37.5 ±11	29.8 ±7	23.1 ±6	18.6 ±6	19.6 ±4	19.0 ±5	18.9 ±6	18.4 ±4	18.6 ±6
	7	45.5 ±7	26.4 ±10	20.7 ±8	21.2 ±7	18.3 ±5	17.9 ±5	18.0 ±6	18.4 ±7	17.6 ±7	17.9 ±5
	8	33.9 ±8	21.5 ±10	19.8 ±12	18.7 ±9	23.4 ±10	19.7 ±9	18.8 ±6	16.7 ±8	16.2 ±8	17.1 ±7
	9	36.8 ±8	29.9 ±10	25.9 ±10	23.6 ±9	18.5 ±8	17.6 ±6	18.8 ±7	18.3 ±9	16.6 ±6	16.2 ±5
	10	54.5 ±8	31.5 ±9	29.5 ±7	27.6 ±10	27.8 ±6	25.1 ±6	23.0 ±6	24.1 ±7	22.6 ±7	22.1 ±6

Search for Object

- We showed
  - how to model object finding as **probabilistic planning** problems
  - that **domain-independent probabilistic planners** offer framework which:
    - is **extremely flexible** (e.g., *makeCoffee*, *buyCoffee*)
    - represents a **well defined** optimization problem
- We empirically compared FF-Replan, UCT and **SSiPP** for the obtained class of problems

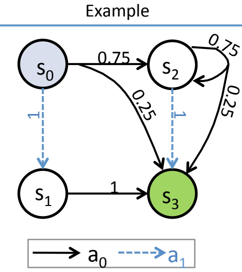
28

Stochastic Shortest Path Problems

An SSP is the tuple $\langle S,s_0,G,A,P,C \rangle$:

- Set of states S
- Initial state s_0
- Set of goal states $G \subseteq S$
- Set of actions A
- Transition probability $P(s'|s,a)$
- Cost $C(s,a,s') > 0$
  - defined when $P(s'|s,a) > 0$

Example



$G = \{s_3\}$

$C(,a_0,)$:

s'/s	s_0	s_1	s_2
s_2	1	--	1
s_3	2	2	2

$C(,a_1,)$:

s'/s	s_0	s_2
s_1	2	--
s_3	--	7

29

Solutions for SSPs

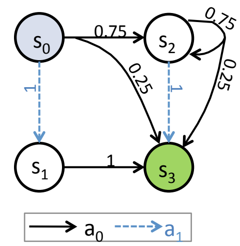
- The **solution** for an SSP is a policy, i.e., **a mapping from states to actions**.
- An **optimal** policy π^* minimizes $V^*(s_0)$, i.e., the expected cost to reach a goal state from s_0.

$$V^*(s) = \begin{cases} 0 & \text{if } s \in \mathsf{G} \\ \min_{a \in \mathsf{A}} \sum_{s' \in \mathsf{S}} P(s'|s,a)[C(s,a,s') + V^*(s')] & \text{otherwise} \end{cases}$$

- In the example:

	s_0	s_1	s_2
π^*	a_1	a_0	a_0

Example



$G = \{s_3\}$

$C(,a_0,)$:

s'/s	s_0	s_1	s_2
s_2	1	--	1
s_3	2	2	2

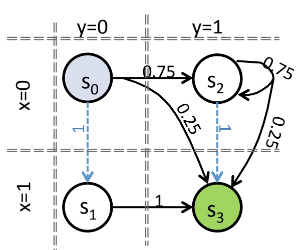
$C(,a_1,)$:

s'/s	s_0	s_2
s_1	2	--
s_3	--	7

30

Factored Representation

- Use **state variables** to represent the state space S:
  - F: $\{f_1,\ldots,f_k\}$, f_i in {0,1}
  - $S = \{0,1\}^k$
- **Benefit**: compact representation
- In the example, two state variables: x and y



Representation of $P(s'|s,a_0)$

Explicit:

s/s'	s_0	s_1	s_2	s_3
s_0	0	0	.75	.25
s_1	0	0	0	1
s_2	0	0	.75	.25
s_3	0	0	0	1

Factored:

x/x'	0	1
0	.75	.25
1	0	1

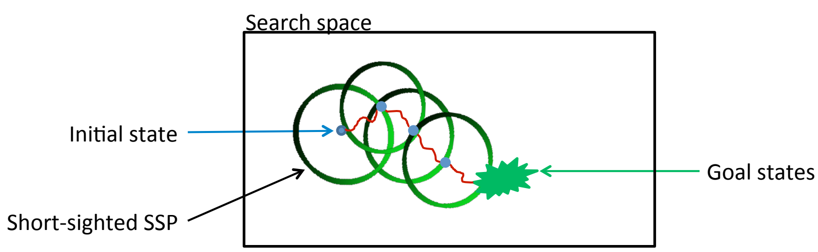
y/y'	0	1
0	0	1
1	0	1

31

Planner: SSiPP

[Trevizan and Veloso, 2012]

- Generate a *short-sighted SSP*:
  - **Prune** the state space
  - Heuristically **estimate the cost** of pruned states
- Solve the subproblems and execute this solution
- Repeat until goal is reached



33

Short-Sighted SSPs: Definition

Given:
- an SSP <S,s_0,G,A,P,C>,
- $s \in$ S
- **t** > 0 and
- a heuristic function H

$\delta(s, s')$: minimum number of actions to reach s' from s

the (s,t)-short-sighted SSP is <S',s,G',A,P,C'>:

States reachable using **up to t** actions

- $\mathsf{S}' = \{s' \in \mathsf{S} | \delta(s, s') \leq t\}$

Artificial goal: states reachable using **exactly t** actions

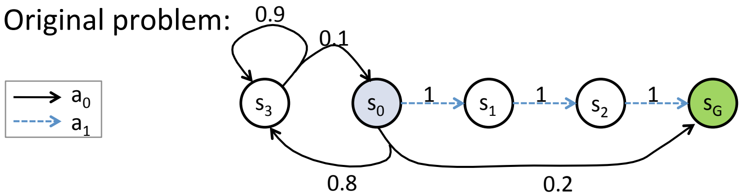
- $\mathsf{G}' = \{s' \in \mathsf{S} | \delta(s, s') = t\} \cup (\mathsf{G} \cap \mathsf{S}')$
- $C'(s, a, s') = \begin{cases} C(s, a, s') + H(s') & \text{if } s' \in \mathsf{G}' \\ C(s, a, s') & \text{otherwise} \end{cases}$

If s' is an artificial goal, then its cost is incremented by its heuristic value

34

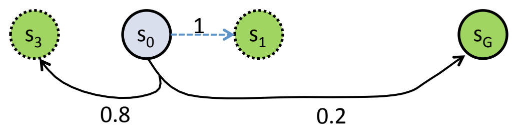
Short-Sighted SSPs: Example

- Original problem:

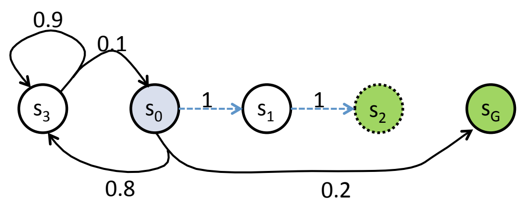


	$\delta(s_0, s)$
s_0	0
s_1	1
s_2	2
s_3	1
s_G	1

- (s_0,1)-short-sighted:

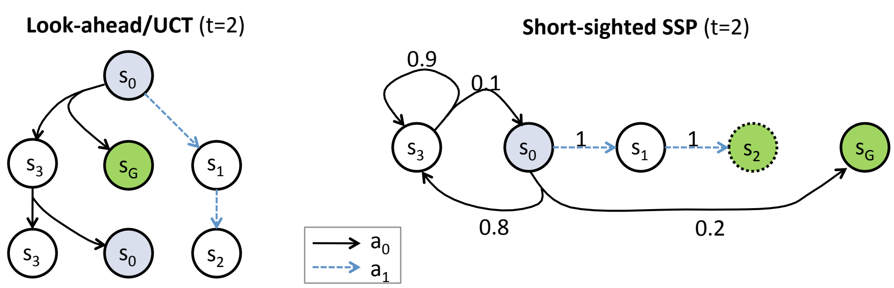


- (s_0,2)-short-sighted



Short-Sighted SSPs and Look-ahead

Theorem: the optimal value-function for an (s,t)-short-sighted SSP is at least as good as the t-look-ahead value of s.



- Short-sighted SSPs preserve the action structure, e.g., self-loop actions and loops of actions

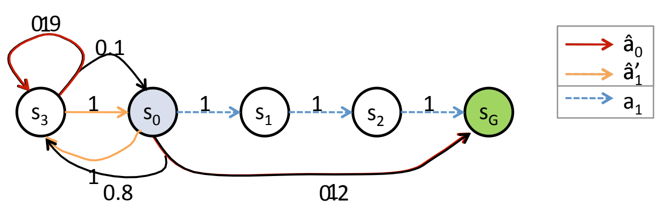
Short-Sighted Probabilistic Planner (SSiPP)

begin
  $s \leftarrow s_0$
  while $s \notin \mathbf{G}$ **do**
    $\langle \mathbf{S}', s, \mathbf{G}', \mathbf{A}, P, C' \rangle \leftarrow$ GENERATE-SHORT-SIGHTED-SSP$(\mathbb{S}, s, H)$
    $\hat{\pi}^* \leftarrow$ OPTIMAL-SSP-SOLVER$(\langle \mathbf{S}', s, \mathbf{G}', \mathbf{A}, P, C' \rangle, H)$
    while $s \notin \mathbf{G}'$ **do**
      $s \leftarrow$ execute-action$(\hat{\pi}^*(s))$
end

Since short-sighted SSPs are much smaller than the original problem. we can compute a complete policy for them.

Planner: FF-Replan

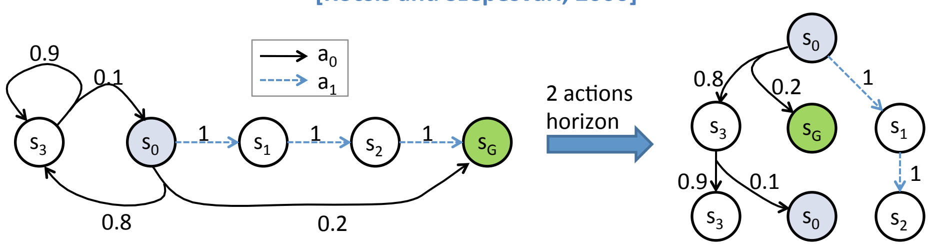
[Yoon et al, 2007]



- Relax probabilistic actions into deterministic actions (**determinization**)
- Pro: scales up
- Cons: oblivious to probabilities

Planner: UCT

[Kocsis and Szepesvári, 2006]



- Relaxes a given SSP by a sequence of **finite-horizon** problems
- Uses sparse sampling to efficiently explore the search tree
- Pro: scales up
- Con: can't represent loops

39

Conclusion – Related Work

- **[Aydemir et al, 2011]**
  - MDP in the belief space of relational descriptions
  - Solved using greedy search over finite horizon
- **[Velez et al, 2011]**
  - Maps objects while moving
  - Minimizes traveled distance and false positives
  - Solved using sampling over finite horizon
- **[Kollar and Roy, 2009]**
  - Finds objects using co-location data (label comparison on Flickr)
  - Minimizes expected plan size over a posteriori prob. of finding the object.
  - Solved using breath-first search with additional constraints
- **[Samadi et al, 2012]**
  - Finds object by querying Google to obtain prior probability
  - Maximizes utility function based on probability of finding object, cost of obtaining object and feedback about the object
  - Solved using beam search

40