Constraint Programming for Path Planning with Uncertainty

Solving the Optimal Search Path Problem

Michael Morin[1] **Anika-Pascale Papillon**[2] **Irène Abi-Zeid**[3]
François Laviolette[1] **Claude-Guy Quimper**[1]

[1]Department of Computer Science and Software Engineering
[2]Department of Mathematics and Statistics
[3]Department of Operations and Decision Systems
Université Laval, Québec, Qc, Canada
Michael.Morin.3@ulaval.ca

October 9, 2012

Outline

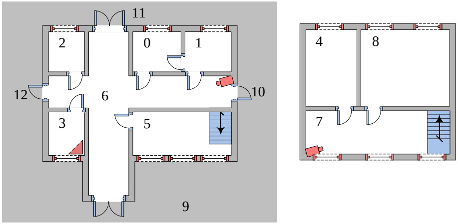
1. Introduction
2. The OSP Problem
3. A CP Model for the OSP
4. The Total Detection Heuristic
5. Experimentation
6. Results and Discussion
7. Conclusion

- Find a path that maximizes the probability of locating a survivor, a robber, an object, etc.
- Uncertain object detectability and location
- Markovian motion model
- Search theory (Stone [2004])
- $\mathcal{NP}$-hard problem (Trummel and Weisinger [1986])

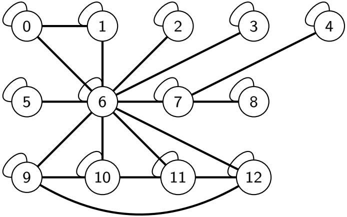
The OSP Problem

Definitions

- $G_A = (\mathcal{V}(G_A), \mathcal{E}(G_A))$ where $\mathcal{V}(G_A)$ is a set of discrete regions.



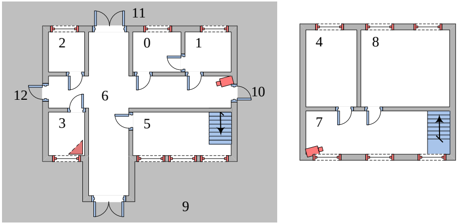
A fictive building



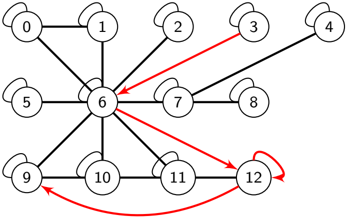
The OSP Problem

Definitions

- $\mathcal{T} = \{1, \ldots, T\}$ is the set of time steps available to search G_A.
- $y_t \in \mathcal{V}(G_A)$ is the searcher's location at time $t \in \mathcal{T}$.
  - When $y_t = r \in \mathcal{V}(G_A)$, the vertex r is searched at time t.
- $P = [y_0, y_1, \ldots, y_T]$ is the search path (plan).
  - $y_0 \in \mathcal{V}(G_A)$ is the searcher's starting location.
  - For all $t \in \mathcal{T}$, $(y_{t-1}, y_t) \in \mathcal{E}(G_A)$.



A fictive building

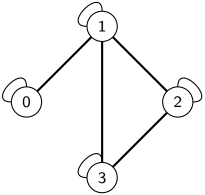


The OSP Problem

Definitions

- The object's movements are independent of the searcher's actions.
- **M** is the Markovian motion model matrix.

$$\mathbf{M} = \begin{pmatrix} \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ \frac{1}{5} & \frac{2}{5} & \frac{1}{5} & \frac{1}{5} \\ 0 & \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \\ 0 & \frac{1}{5} & \frac{2}{5} & \frac{1}{5} \end{pmatrix}.$$



Blue terms are a priori known probabilities.

- The initial probability of containment distribution: poc_1.
- The local probability of success ($\forall t \in \mathcal{T}$):

$$\overbrace{pos_t(r)}^{\text{Prob. of success}} = \overbrace{\underbrace{poc_t(r)}_{\text{Prob. of containment}} \times \underbrace{pod(r)}_{\text{Prob. of detection}}}^{\text{Prob. of success}}.$$

- The probability of detection (conditional to the presence of the object):

$$\begin{aligned} pod(r) &\in (0,1]\,, && \text{if } y_t = r; \\ pod(r) &= 0, && \text{otherwise.} \end{aligned}$$

- The local probability of containment ($\forall t \in \{2, \dots, T\}$):

$$poc_t(r) = \sum_{s \in \mathcal{V}(G_A)} \mathbf{M}(s,r)\left[poc_{t-1}(s) - pos_{t-1}(s)\right].$$

The OSP Problem

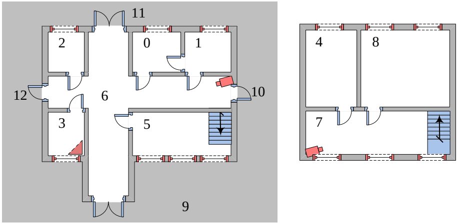
Problem Statement

Find an optimal search plan $P = [y_0, y_1, \ldots, y_T]$ maximizing the cumulative overall probability of success (COS) defined as:

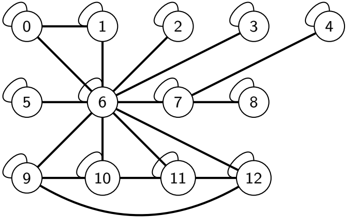
$$COS(P) = \sum_{t \in \mathcal{T}} \sum_{r \in \mathcal{V}(G_A)} pos_t(r).$$

The OSP Problem

Example

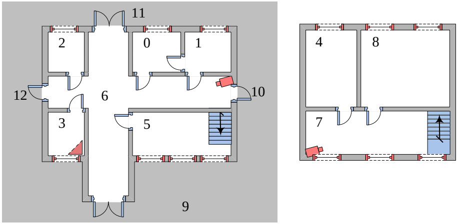


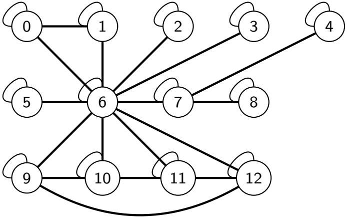
A fictive building



The OSP Problem

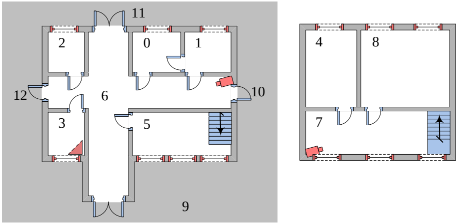
Example



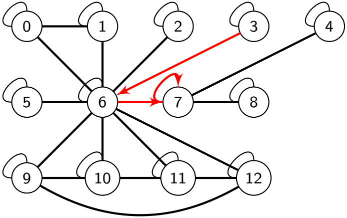


A fictive building

- Let $T = 5$, $y_0 = 3$, $poc_1(4) = 1.0$, $pod(y_t) = 0.9$ $(\forall t \in \mathcal{T})$, and assume a uniform Markovian motion model between accessible vertices.



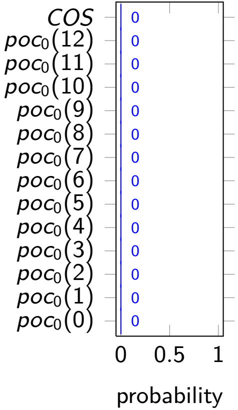
A fictive building



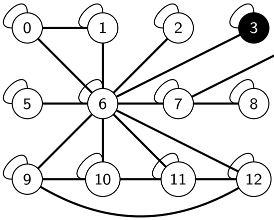
- Let $T = 5$, $y_0 = 3$, $poc_1(4) = 1.0$, $pod(y_t) = 0.9$ $(\forall t \in \mathcal{T})$, and assume a uniform Markovian motion model between accessible vertices.
- P^* is the optimal search plan:

$$P^* = [y_0, y_1, \ldots, y_5] = [3, 6, 7, 7, 7, 7] .$$

Probability distribution at $t = 0$



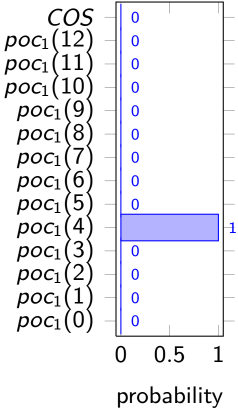
$$P^* = [y_0, y_1, \ldots, y_5] = [\mathbf{3}, 6, 7, 7, 7, 7] .$$



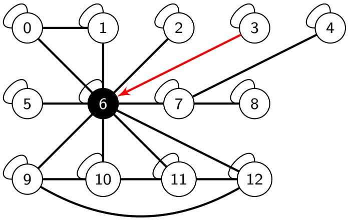
The OSP Problem

Example

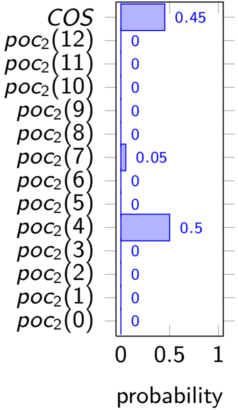
Probability distribution at $t = 1$



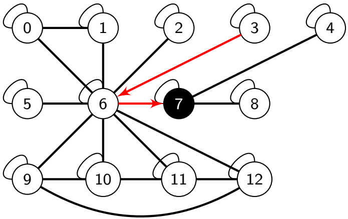
$$P^* = [y_0, y_1, \ldots, y_5] = [3, \mathbf{6}, 7, 7, 7, 7] \,.$$



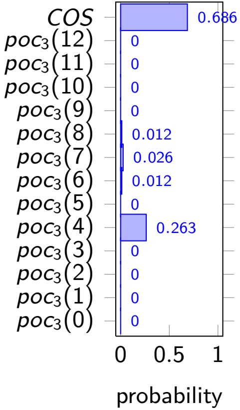
Probability distribution at $t = 2$



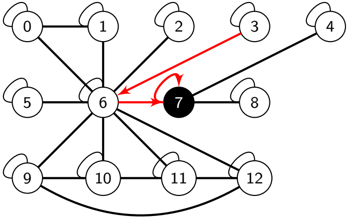
$$P^* = [y_0, y_1, \ldots, y_5] = [3, 6, \mathbf{7}, 7, 7, 7] .$$



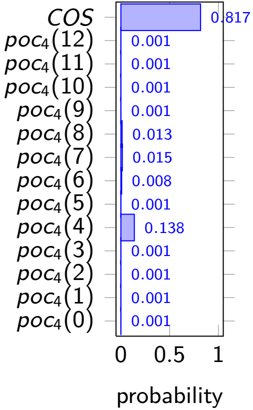
Probability distribution at $t = 3$



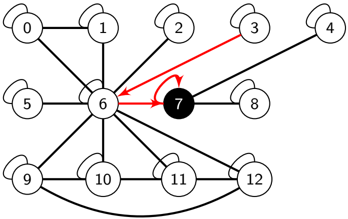
$$P^* = [y_0, y_1, \ldots, y_5] = [3, 6, 7, \mathbf{7}, 7, 7] \,.$$



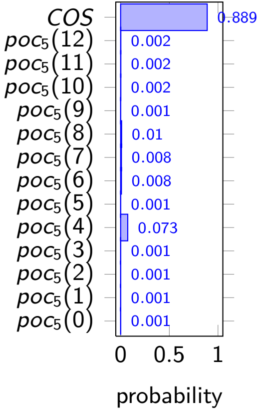
Probability distribution at $t = 4$



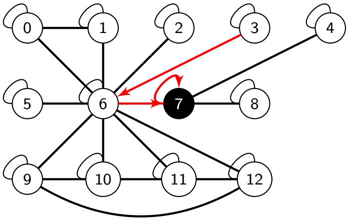
$$P^* = [y_0, y_1, \ldots, y_5] = [3, 6, 7, 7, \mathbf{7}, 7] .$$



Probability distribution at $t = 5$



$$P^* = [y_0, y_1, \ldots, y_5] = [3, 6, 7, 7, 7, \mathbf{7}] .$$



A CP Model for the OSP

- The *variables* and the *constraints* are given by the problem definition.
- Two *equivalent* objective functions with a *different* performance:
  - First choice: The double sum definition

$$\max COS,$$
$$COS = \sum_{t \in \mathcal{T}} \sum_{r \in \mathcal{V}(G_A)} POS_t(r).$$

  - Second choice: The sum and max definition

$$\max COS,$$
$$COS = \sum_{t \in \mathcal{T}} \max_{r \in \mathcal{V}(G_A)} POS_t(r).$$

VARIABLES are displayed in UPPER case and *constants* are displayed in lower case.

- The searcher searches one vertex per time step.
- Thus, there is only one vertex r such that $POS_t(r) \neq 0$.
- Consequently,

$$\max \sum_{t \in \mathcal{T}} \sum_{r \in \mathcal{V}(G_A)} POS_t(r) \equiv \max \sum_{t \in \mathcal{T}} \max_{r \in \mathcal{V}(G_A)} POS_t(r).$$

- First choice: Poor filtering = poor bound:

$$\max COS = \sum_{t \in \mathcal{T}} \sum_{r \in \mathcal{V}(G_A)} POS_t(r),$$

$$\lceil COS \rceil = \sum_{t \in \mathcal{T}} \sum_{r \in \mathcal{V}(G_A)} \lceil POS_t(r) \rceil .$$

- Second choice: Better filtering = better bound:

$$\max COS = \sum_{t \in \mathcal{T}} \max_{r \in \mathcal{V}(G_A)} POS_t(r),$$

$$\lceil COS \rceil = \sum_{t \in \mathcal{T}} \max_{r \in \mathcal{V}(G_A)} \lceil POS_t(r) \rceil .$$

The Total Detection Heuristic

- Ignore negative information when searching.
- What is the most promising vertex?
  - The one with the highest *total probability* of detecting the object in the remaining time.

The Total Detection Heuristic

Variables and Values Ordering

- Decision variables order: $Y_0, Y_1, \ldots, Y_T$.
- Values order:

$$\underset{y' \in \text{dom}(Y_t)}{\text{argmax}} \sum_{o \in \mathcal{V}(G_A)} w_t(y', o) POC_t(o), \qquad \forall t \in \mathcal{T}.$$

- $w_t(y', o)$ is the conditional probability that the searcher detects the object before the end of the search given that, at time t, the searcher is in y' and the object in o.
- $w_t(y', o)$ is computed using dynamic programming and the following data:
  - the Markovian motion model matrix $\mathbf{M}$;
  - the probability of detection pod.

- Let $w_t(y, o)$ be the conditional probability that the searcher detects the object before the end of the search given that, at time t, the searcher is in y and the object in o:

$$w_t(y, o) \stackrel{\text{def}}{=} \begin{cases} pod(o), & \text{if } o = y \text{ and } t = T, \\ 0, & \text{if } o \neq y \text{ and } t = T, \\ p_t(y, o), & \text{if } o \neq y \text{ and } t < T, \\ pod(o) + (1 - pod(o))p_t(y, o), & \text{if } o = y \text{ and } t < T. \end{cases}$$

  where

$$p_t(y, o) = \sum_{o' \in \mathcal{N}(o)} \mathbf{M}(o, o') \max_{y' \in \mathcal{N}(y)} w_{t+1}(y', o').$$

- Decision variables order: $Y_0, Y_1, \ldots, Y_T$
- Values order:

$$\underset{y' \in \text{dom}\,(Y_t)}{\text{argmax}} \sum_{o \in \mathcal{V}(G_A)} w_t(y', o) POC_t(o), \qquad \forall t \in \mathcal{T}.$$

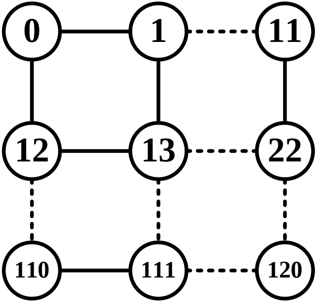
Experimentation

- Three different probabilities of detection: $pod(r) \in \{0.3, 0.6, 0.9\}$ $(\forall r \in \mathcal{V}(G_A))$.
- Three different motion models:

$$\mathbf{M}(s,r) = \begin{cases} \frac{1-\rho}{\deg(s)-1}, & \text{if } (s,r) \in \mathcal{E}(G_A), \\ \rho, & \text{if } s = r, \end{cases}$$

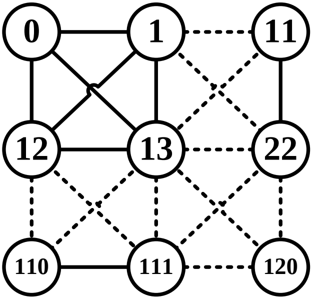
  where $\deg(s)$ is the degree of s and $\rho \in \{0.3, 0.6, 0.9\}$ is the probability that the object stays in its current location.
- Six different allowed time values: $T \in \{9, 11, 13, 15, 17, 19\}$.
- Three different graph structures...

- The 11×11 grid G^+



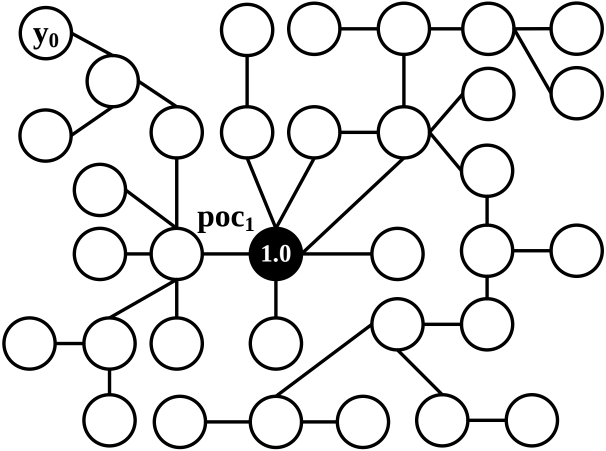
$$poc_1(60) = 1$$
$$y_0 = 0$$

- The 11×11 grid G^*



$$poc_1(60) = 1$$
$$y_0 = 0$$

- The graph G^L (the Université Laval tunnels map)



Experimentation

- Java implementation:
  - Choco solver (Laburthe and Jussien [2012])
  - Java Universal Network/Graph (JUNG) 2.0.1 framework (O'Madadhain et al. [2010])
- 20 minutes time limit
- A maximum of 5,000,000 backtracks

- The CpMax model uses the max objective function.
- The CpSum model uses the $\sum$ objective function.

Table: CpMax vs CpSum on a 11×11 G^{+} grid with $T = 17$.

		CpMax		CpSum	
$pod(r)$	ρ	Time to last incumbent (s)	*COS* value	Time to last incumbent (s)	*COS* value
0.3	0.6	**1199**	**0.128**	991	0.127
	0.9	**1026**	**0.338**	1166	0.338
0.6	0.6	**1169**	**0.220**	1016	0.217
	0.9	**1166**	**0.512**	942	0.501
0.9	0.6	**692**	**0.315**	728	0.315
	0.9	**1170**	**0.628**	880	0.625

- The CpMax model uses the max objective function.
- The TDValSel+CpMax model uses the Total Detection value selection heuristic.

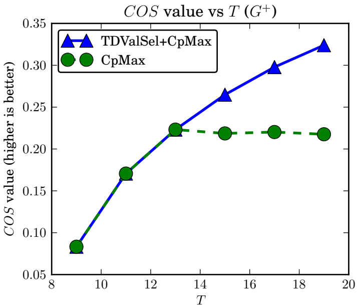


Figure: CpMax vs Total Detection on a 11×11 G^+ instance where $pod(y_t) = 0.6$ $(\forall t \in \mathcal{T})$, and $\rho = 0.6$.

- The CpMax model uses the max objective function.
- The TDValSel+CpMax model uses the Total Detection value selection heuristic.

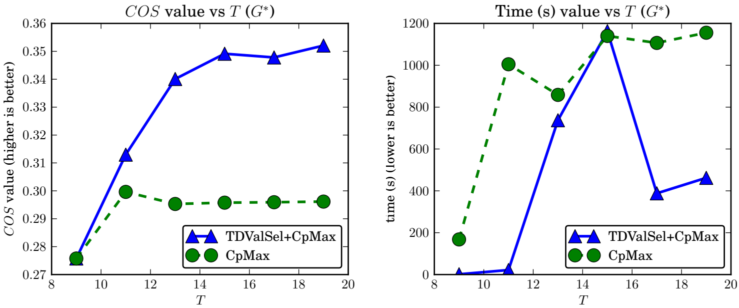


Figure: CpMax vs Total Detection on a 11×11 G^* instance where $pod(y_t) = 0.6$ $(\forall t \in \mathcal{T})$, and $\rho = 0.6$.

- The CpMax model uses the max objective function.
- The TDValSel+CpMax model uses the Total Detection value selection heuristic.

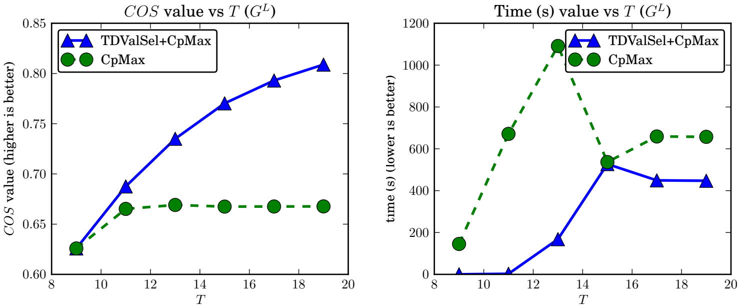


Figure: CpMax vs Total Detection on a G^L instance where $pod(y_t) = 0.6$ $(\forall t \in \mathcal{T})$, and $\rho = 0.6$.

Conclusion

- Contributions and novelties:
  - A new CP model to solve the OSP problem
  - A tighter bound using the max objective function encoding
  - The Total Detection heuristic
- Future work:
  - Use the concept of the Total Detection heuristic to develop a better bounding technique for the objective function.

Thank you!

Photography by Yann Arthus-Bertrand

Stay tuned! :)
http://www.agora.ulaval.ca/ mimor225/

References I

F. Laburthe and N. Jussien. *Choco Solver Documentation*, 2012. http://www.emn.fr/z-info/choco-solver/.

J. O'Madadhain, D. Fisher, T. Nelson, S. White, and Y. Boey. Jung: Java universal network/graph framework. http://jung.sourceforge.net, 2010.

L. Stone. *Theory of Optimal Search*. Academic Press, New York, 2004.

K. Trummel and J. Weisinger. The complexity of the optimal searcher path problem. *Operations Research*, 34(2):324–327, 1986.