Addendum: Relaxation of the Optimal Search Path Problem with the Cop and Robber Game

Frédéric Simard ⋆, Michael Morin ⋆⋆,
Claude-Guy Quimper, François Laviolette, and Josée Desharnais

Department of Computer Science and Software Engineering
Université Laval, Québec Qc G1V 0A6, Canada,
{frederic.simard.10,Michael.Morin.3}@ulaval.ca,
{Claude-Guy.Quimper,Francois.Laviolette,Josee.Desharnais}@ift.ulaval.ca

Abstract. In the Optimal Search Path problem from search theory, the objective is to find a finite length searcher's path that maximizes the probability of detecting a lost wanderer on a graph. We introduce a novel bound on the probability of finding the wanderer in the remaining search time and discuss how this bound is derived from a relaxation of the problem into a game of cop and robber from graph theory. We demonstrate the efficiency of this bound on a constraint programming model of the problem.

Keywords: Optimal Search Path, Cop and Robber, constraint relaxation, pursuit games

1 Complete results

This document includes supplementary results for the experiments described in the corresponding article by Simard et al. [1]. Please refer to the original paper for the full article. We now recall the experimental context and show the complete results.

All the experiments were run on an Intel(R) Core(TM) i7 2.6 GHz with 8 GB of RAM using the Choco 2.1.5 solver [2] along with the Java Universal Network/Graph (JUNG) 2.0.1 framework [3]. We implemented the bound on two known OSP CP models : a basic model (*max*), and the *max* model with the total detection heuristic implemented as a value selection heuristic (TD) [4]. The TD heuristic is on many account similar to the bound derived in this paper which gives its theoretical justification. We generated 27 OSP instances using the graph G^+, G^* and G^L [4] with $T = 17$. G^+ is a 11×11 grid with North, South, West and East accessibility, G^* is a 11×11 grid plus diagonals, and the University Laval tunnels map G^L. For each instance, we chose $\text{pod}(r) \in \{0.3, 0.6, 0.9\}$ (for all vertex r) and generated a transition matrix as follows:

⋆ MSc student, main author. Co-supervised by F. Laviolette and J. Desharnais.
⋆⋆ PhD candidate, co-author. Co-supervised by C.G. Quimper and I. Abi-Zeid.

1. the wanderer has a probability ρ of staying in place;
2. the remaining probability mass $1-\rho$ was uniformly distributed on the neighbouring vertices.

We chose $\rho \in \{0.3, 0.6, 0.9\}$. The total allowed CPU time is 5 minutes (or 5 million backtracks). [1] The results are shown in Table 1.

Table 1. Results on OSP problem instances without and with a bound (+B)

		Max model						**TD model**					
$pod(r)$	ρ	**Objective**		**Time**†		**Backtracks**		**Objective**		**Time**†		**Backtracks**	
			+B		+B		+B		+B		+B		+B
G^L *with* $T = 17$													
0.3	0.3	0.318	**0.346**	291	164	21649	14548	0.519	0.519	4	3	282	181
	0.6	0.359	**0.392**	299	215	21706	18639	0.585	**0.586**	130	188	10391	16055
	0.9	0.480	**0.520**	294	209	21707	17955	0.749	0.749	147	78	12160	6715
0.6	0.3	0.514	**0.551**	294	214	21936	18737	0.741	**0.743**	107	157	8605	13574
	0.6	0.566	**0.607**	300	269	22702	22726	0.785	**0.789**	157	207	12954	17956
	0.9	0.675	**0.720**	272	241	20264	20201	0.883	**0.888**	182	297	15321	25944
0.9	0.3	0.651	**0.683**	300	247	22855	21410	0.831	**0.838**	188	128	15000	11076
	0.6	0.704	**0.740**	299	267	22796	23074	0.870	**0.881**	177	204	14628	17604
	0.9	0.829	**0.858**	276	271	20263	23360	0.939	**0.945**	179	262	15212	22999
G^+ *with* $T = 17$													
0.3	0.3	0.014	**0.028**	47	185	360	1458	0.105	0.105	8	9	5	5
	0.6	0.013	**0.036**	46	170	360	1230	0.162	0.162	101	121	918	918
	0.9	0.003	**0.015**	47	71	355	489	0.442	0.442	75	67	682	478
0.6	0.3	0.027	**0.054**	147	192	1220	1521	0.187	0.187	99	121	918	918
	0.6	0.025	**0.070**	47	170	360	1227	0.281	0.281	165	188	1579	1452
	0.9	0.005	**0.038**	15	178	78	1234	0.641	**0.650**	99	274	918	2136
0.9	0.3	0.039	**0.077**	195	177	1223	1394	0.252	0.252	99	121	918	918
	0.6	0.037	**0.103**	47	168	360	1229	0.376	0.376	164	185	1574	1420
	0.9	0.008	**0.047**	46	171	355	1174	0.784	**0.807**	230	256	2235	1988
G^* *with* $T = 17$													
0.3	0.3	0.071	**0.075**	233	280	3448	4042	0.130	0.130	47	56	748	748
	0.6	0.111	**0.121**	233	292	3446	4346	0.213	0.213	51	60	821	821
	0.9	0.241	**0.329**	232	208	3438	2945	0.601	0.601	50	62	830	830
0.6	0.3	0.137	**0.146**	233	275	3450	4032	0.222	0.222	48	56	748	748
	0.6	0.215	**0.232**	232	228	3446	3343	0.329	0.329	52	63	830	830
	0.9	0.469	**0.560**	232	191	3437	2705	0.728	0.728	53	62	830	830
0.9	0.3	0.198	**0.211**	231	276	3450	4024	0.292	0.292	51	63	821	821
	0.6	0.313	**0.335**	233	279	3446	4106	0.406	0.406	51	63	830	830
	0.9	0.687	**0.746**	228	253	3436	3533	0.791	0.791	50	62	830	830

† The time, in seconds, to the last incumbent solution.
‡ Bold values are better.

References

1. Simard, F., Morin, M., Quimper, C.G., Laviolette, F., Desharnais, J.: Relaxation of the optimal search path problem with the cop and robber game. In: CP Doctoral Program. (2014)

[1] In all cases, the solver failed to prove the optimality within 5 minutes.

2. Laburthe, F., Jussien, N.: Choco solver documentation. Ecole de Mines de Nantes (2012)
3. O'Madadhain, J., Fisher, D., Nelson, T., White, S., Boey, Y.B.: The jung (java universal network/graph) framework. (2010)
4. Morin, M., Papillon, A.P., Abi-Zeid, I., Laviolette, F., Quimper, C.G.: Constraint programming for path planning with uncertainty. In: Principles and Practice of Constraint Programming. (2012) 988–1003