Time Optimal Multi-agent Path Planning on Graphs*

Jingjin Yu and Steven M. LaValle
University of Illinois at Urbana-Champaign

Time Optimal Multi-agent Path Planning

Basic Multi-agent Path Planning Problem

Input:
$G \coloneqq (V, E), A \coloneqq \{a_1, \dots, a_n\}$, $x_I, x_G \colon A \to V$

Output:
$P \coloneqq \{p_1, \dots, p_n\}$, $p_i \colon \mathbb{Z}^+ \to V$, that takes the agents from $x_I(A)$ to $x_G(A)$, free of collision.

Types of Collisions

"Meet" collision
(collision on a vertex)

"Head-on" collision
(collision on an edge)

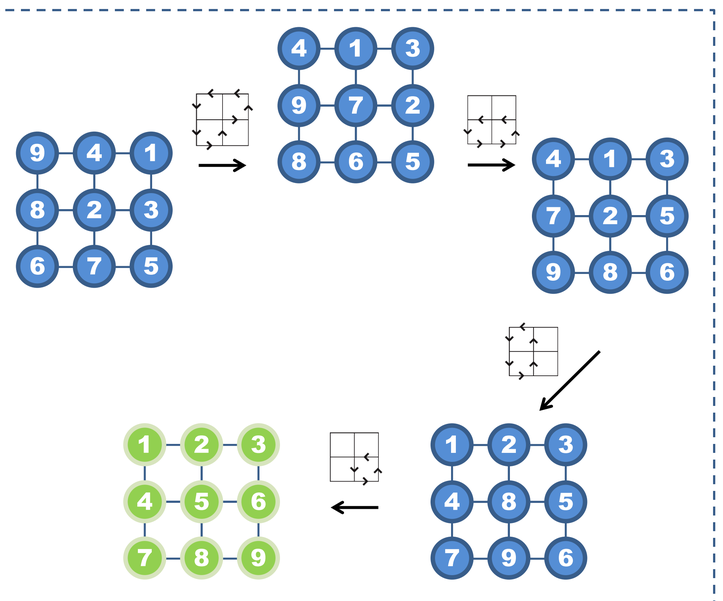
Time Optimal Multi-agent Path Planning

Given a path $p_i \colon \mathbb{Z}^+ \to V$, let t_i be the smallest such that $p_i(t_i) \equiv p_i(t)$ for all $t \geq t_i$. Recall that $P \coloneqq \{p_1, \dots, p_n\}$. We want to find

$$T_{min} \coloneqq \min_{P} \max_{1 \leq i \leq n} t_i \,,$$

and a path set P that yields T_{min}. The problem is NP-hard.

A Motivating Example



Main Results

1. An equivalence between multi-agent path planning and multi-commodity network flow.
2. An integer linear programming (ILP) model for finding time optimal solutions to the multi-agent path planning problem.
3. Computational applications of the ILP model
   - As a complete algorithm;
   - As a heuristic for solving larger problems.

Related Work

Discrete (grid/graph):

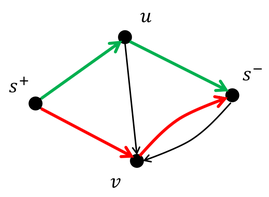
Wilson 1974, Goldreich 1984, Kornhauser, Miller, and Spirakis 1984, Ratner and Warmuth 1990, Silver 2006, Jansen and Sturtevant 2008, Ryan 2008, Standley 2010, Surynek 2010, Luna and Bekris 2011, Wang and Botea 2011

Continuous:

Erdmann and Lozano-Pérez 1986, Kant and Zucker 1986, O'Donnell and Lozano-Pérez 1989, Švestka and Overmars 1998, LaValle 1995, Peng and Akella 2002, Simeon, Leroy, and Laumond 2002, Guo and Parker 2003, Ghrist, O'Kane, and LaValle 2004, van den Berg and Overmars 2005, Peasgood, Clark and McPhee 2008, van den Berg et.al. 2009

Multi-agent Path Planning and Network Flow

The Basic Network Flow Problem

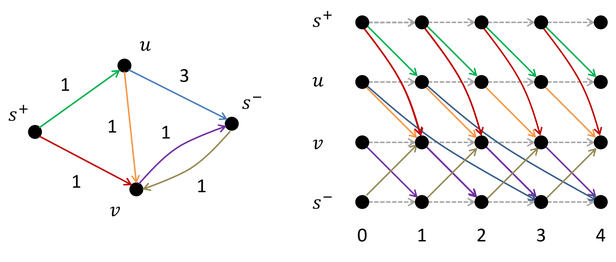


In a basic network flow problem, we want to find edge disjoint paths for transferring commodities. Maximum flow algorithms can efficiently solve this problem.

L. R. Ford and D. R. Fulkerson. Maximal flow through a network. *Research Memorandum RM-1400, The RAND Corporation*, 1954.

Network Flow Over Time

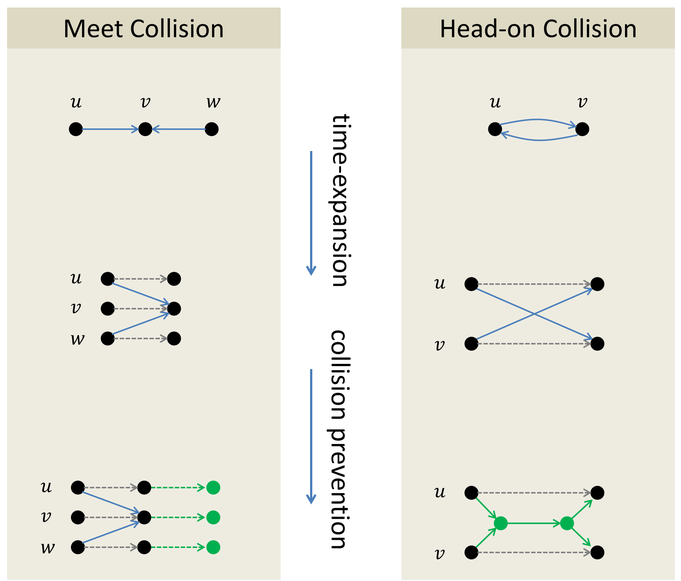
Basic network flow problems do not consider $time$. It is possible to introduce a set of costs on the edges to represent transfer time (left). These problems are called network flow over time and they can be solved using standard network flow problem over the $time\text{-}expanded\ network$ (right).



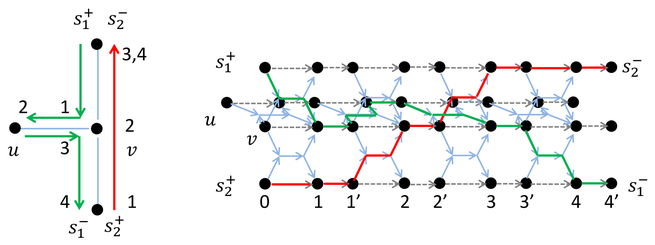
L. R. Ford. and D. R. Fulkerson. Constructing maximal dynamic flows from static flows. *Operations Research*, 6:419–433, 1958.

The time expansion technique, however, does not handle collisions by default. For the two types of collisions, extra efforts are required to prevent them from happening. These constructs are given below.

Collision Avoidance Constructs

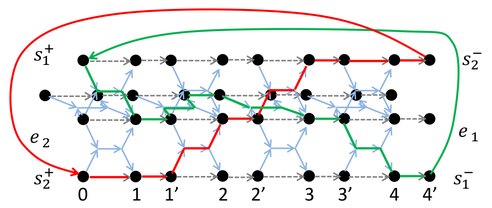


Equivalence Between Multi-agent Path Planning and Network Flow



Theorem: Fixing a natural number T, the multi-agent path planning problem admits a solution with at most T time steps if and only if the time-expanded network with T periods admits a max flow of n (the number of agents).

The Integer Linear Programming Model



$x_{i,j}$: agent i → edge e_j

$x_{i,j} = 1 \to e_j$ used by a_i

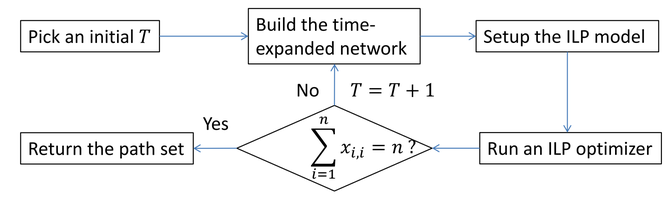
The Model

$$maximize \sum_{i=1}^{n} x_{i,i}$$

$$subject\ to \begin{cases} \forall e_j, & \sum_{i=1}^{n} x_{i,j} \leq 1 \\ \forall v, & \sum_{e_j \in \delta^+(v)} x_{i,j} = \sum_{e_j \in \delta^-(v)} x_{i,j} \end{cases}$$

Computational Applications

As a Complete Algorithm



The algorithm is complete since T can be upper bounded.

n^2-puzzles

n^2	Average running time	Average # of steps	Number of states	Branching factor
9	10 seconds	4	10^5	13
16	2 minutes	6	10^{13}	> 500
25	3 hours	8	10^{25}	$> 10^4$
36	> 24 hours	?	10^{41}	$> 10^6$

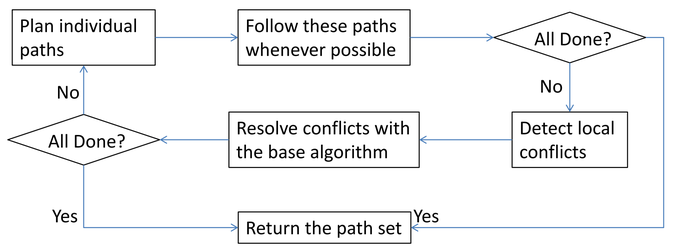
General Path Planning

20 x 15 grid, 5 runs each. Vertices are removed to simulate obstacles.

% obs	Number of agents			
	10	20	30	40
0	36.2 (20.6)	64.8 (23.6)	229 (26.8)	259 (24.0)
10	31.3 (26.0)	81.9 (26.6)	157 (26.0)	246 (25.6)
20	25.9 (26.4)	57.5 (24.8)	151 (27.0)	539 (29.6)
30	20.2 (28.6)	141 (34.2)	368 (33.0)	1368 (32.7)
40	27.4 (35.0)	572 (42.4)	N/A	N/A

All computation were carried out on an Intel Q6600 PC with 8GB memory using the Gurobi optimizer.

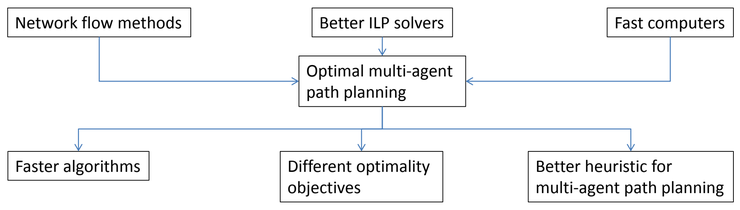
As a Heuristic for Large Problem Instances



32 x 32 grid, 20% obstacles, 100 runs each, 10 seconds cutoff. Local problem size $\leq$ 8 x 8. Program coded in Java.

	Number of agents				
	25	50	75	100	125
Running time (s)	0.038	0.225	0.732	1.944	4.935
% goals reached	100.0	99.95	99.78	98.84	98.47
Ave. path length	24.61	24.84	25.03	25.52	26.26
Heu. path length	24.6	24.67	24.56	24.60	24.51
% length difference	0.05	0.69	1.91	3.74	7.14

Conclusion and Future Directions



* This research was supported in part by NSF grant 0904501 (IIS Robotics), NSF grant 1035345 (Cyberphysical Systems), DARPA SToMP grant HR0011-05-1-0008, and MURI/ONR grant N00014-09-1-1052.