Integrating Vehicle Routing and Motion Planning

Scott Kiesel, Ethan Burns, Christopher Wilt and Wheeler Ruml

UNIVERSITY of NEW HAMPSHIRE

Department of Computer Science

We are grateful for funding from the DARPA CSSG program (grant HR0011-09-1-0021) and NSF (grant IIS-0812141).

Contributions

1. New problem: Waypoint Allocation and Motion Planning
   (a) WAMP combines task planning and motion planning
   (b) vehicle routing but now with *real* routing!
2. Efficient solver
   (a) integrates tabu search, blind search, heuristic search, linear programming and simple temporal networks
3. Meets application requirements
   (a) 2.5x faster and more scalable than industrial partner's

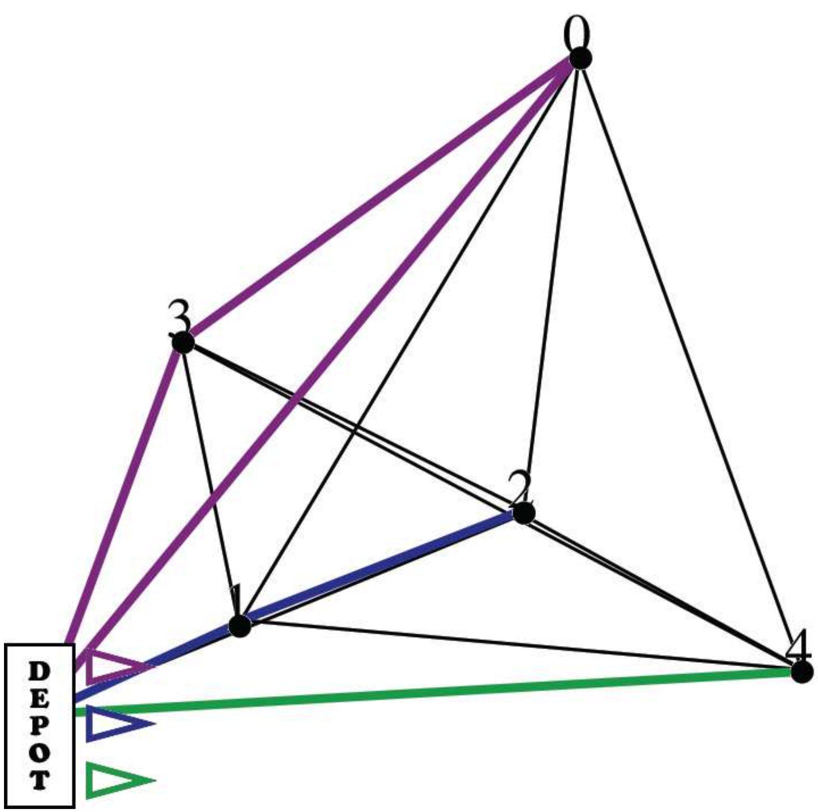
Vehicle Routing

allocate tasks to vehicles

routes given as a distance matrix

objective: find cheapest ordering

temporal constraints



Combining Vehicle Routing and Motion Planning

Vehicle Routing

allocate tasks to vehicles

routes given as a distance matrix

objective: find cheapest ordering

temporal constraints

Motion Planning

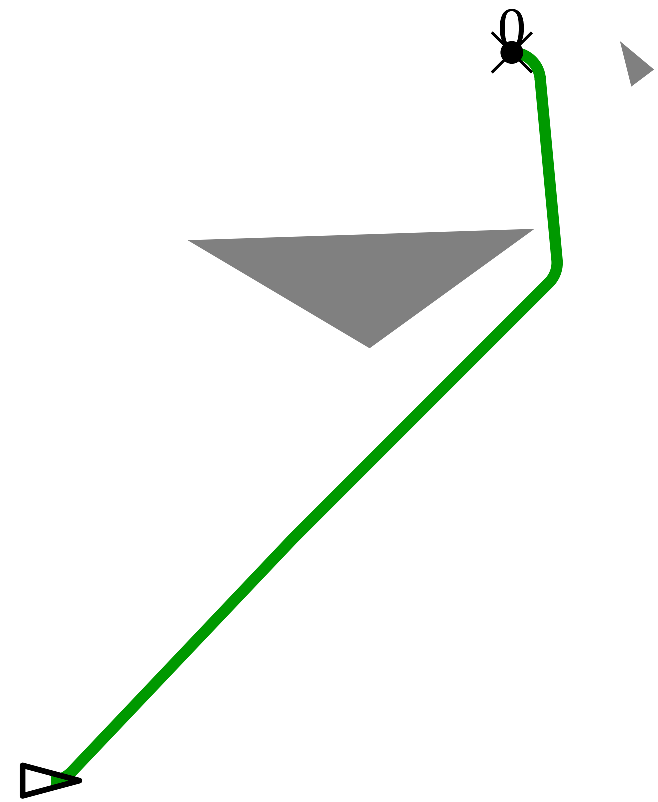
find feasible trajectory

continuous space

respect vehicle limitations

obstacles

objective: minimize time



WAMP

allocate tasks to vehicles

~~routes given as a distance matrix~~

~~objective: find cheapest ordering~~

temporal constraints

find feasible trajectory

continuous space

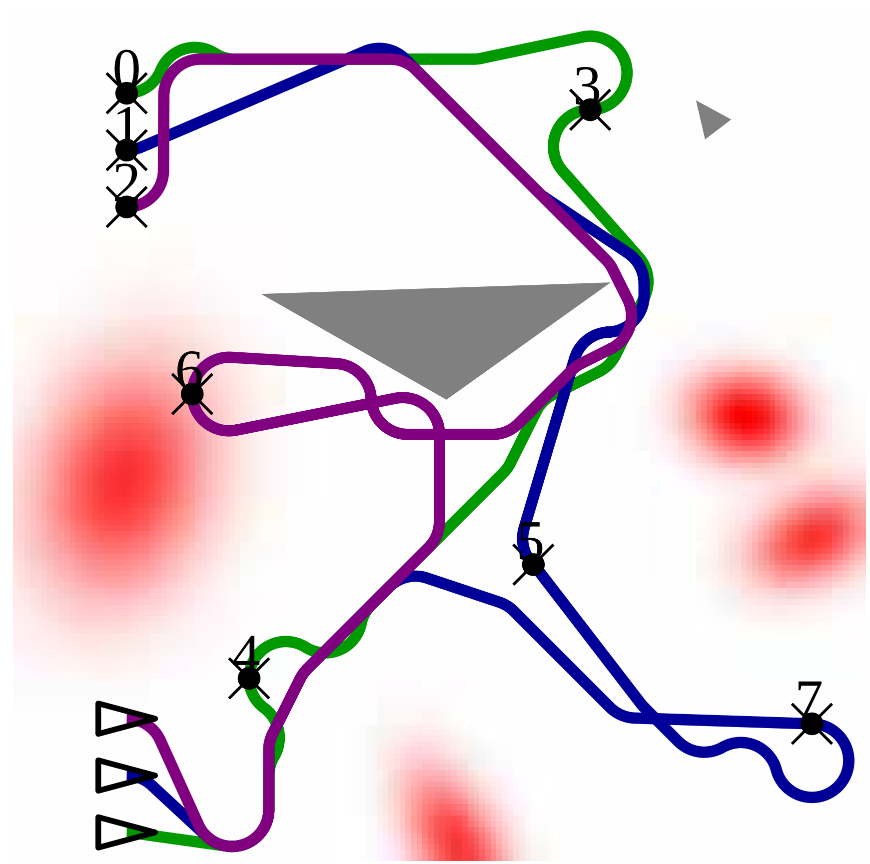
respect vehicle limitations

obstacles

~~objective: minimize time~~

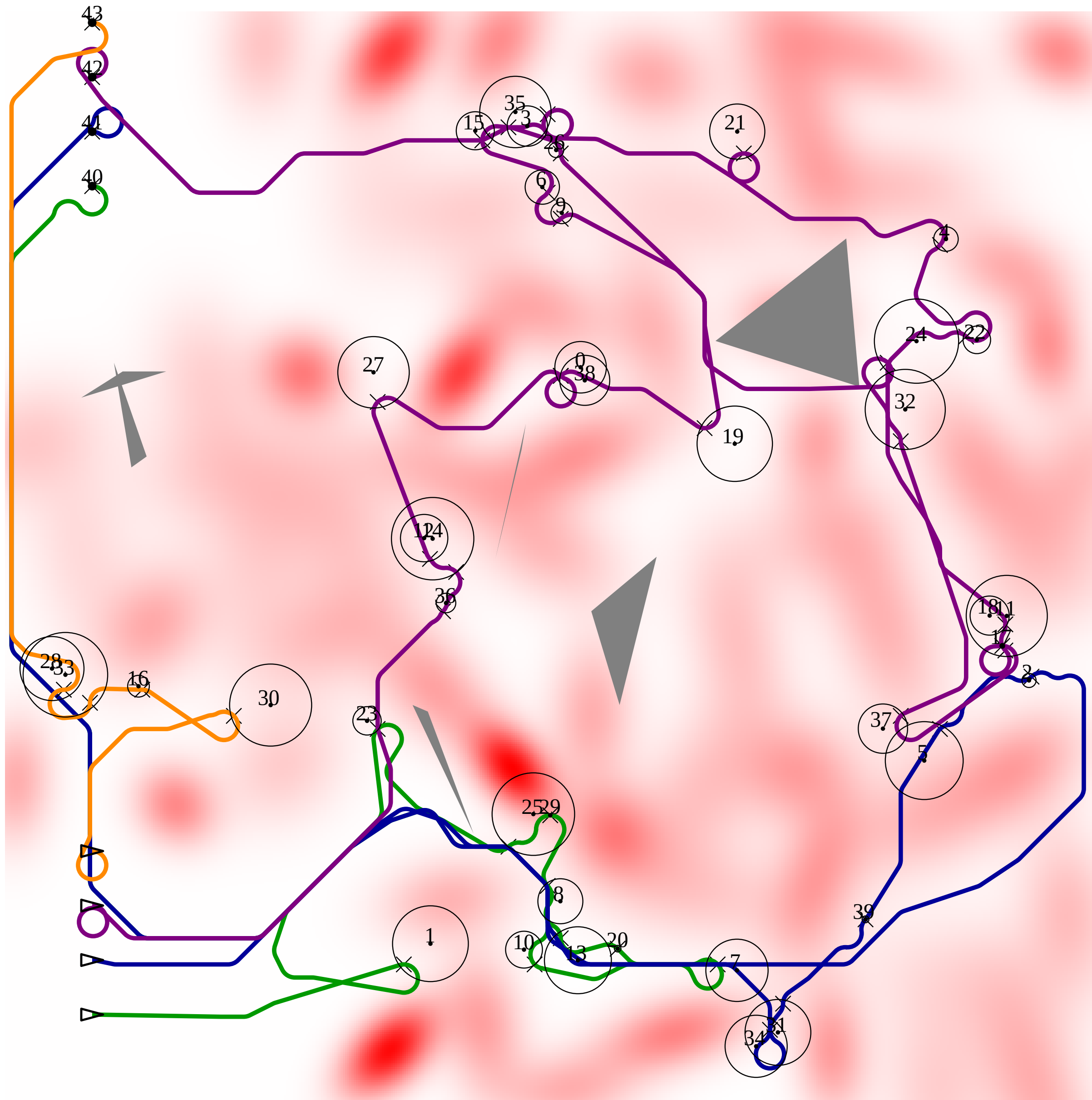
varying traversal costs

objective: minimize time and cost



naturally combines task allocation and motion planning

A Realistic-Sized Instance



Application: Battlespace Management

A large instance:

- 16 aircraft
  - varying velocities and turning radii
- 40 waypoints
  - time windows
  - relative temporal constraints
- 40 radar sensitive (cost) zones
- strict no-fly zones
- 7 second time limit

Central Challenge

1. Can't do task allocation without routing costs

   need cost from one waypoint to next

2. Can't find routing costs without motion paths

   paths may intersect areas of high cost

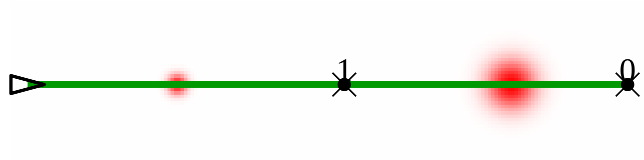
3. Can't find motion paths without leg durations

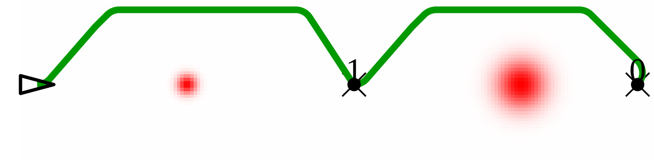
   is there time to navigate around high cost?

4. Can't assign leg durations without **time/cost tradeoff**

   which legs benefit most from additional time?

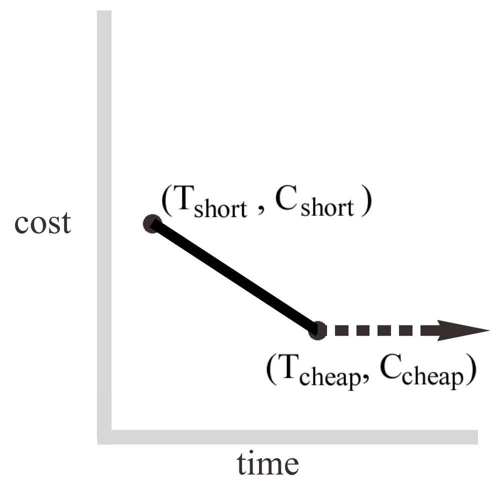
Surrogate Objective

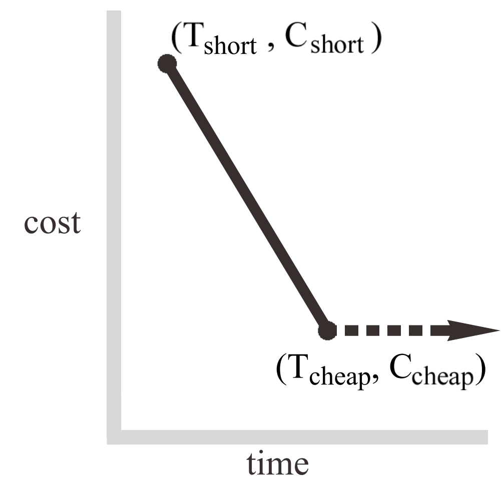


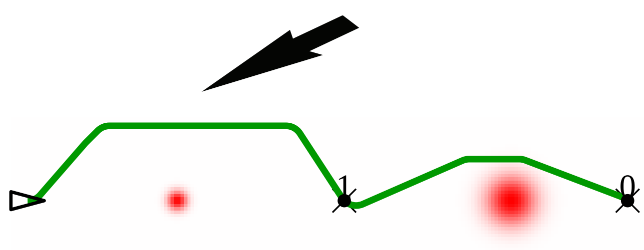


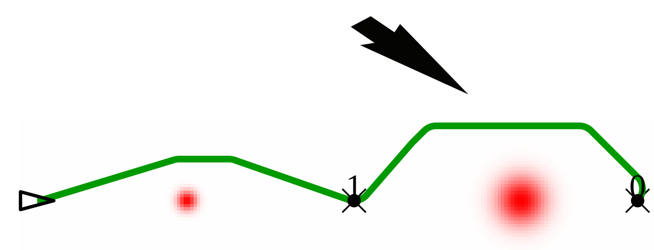
fastest route is too expensive

cheapest route is too long







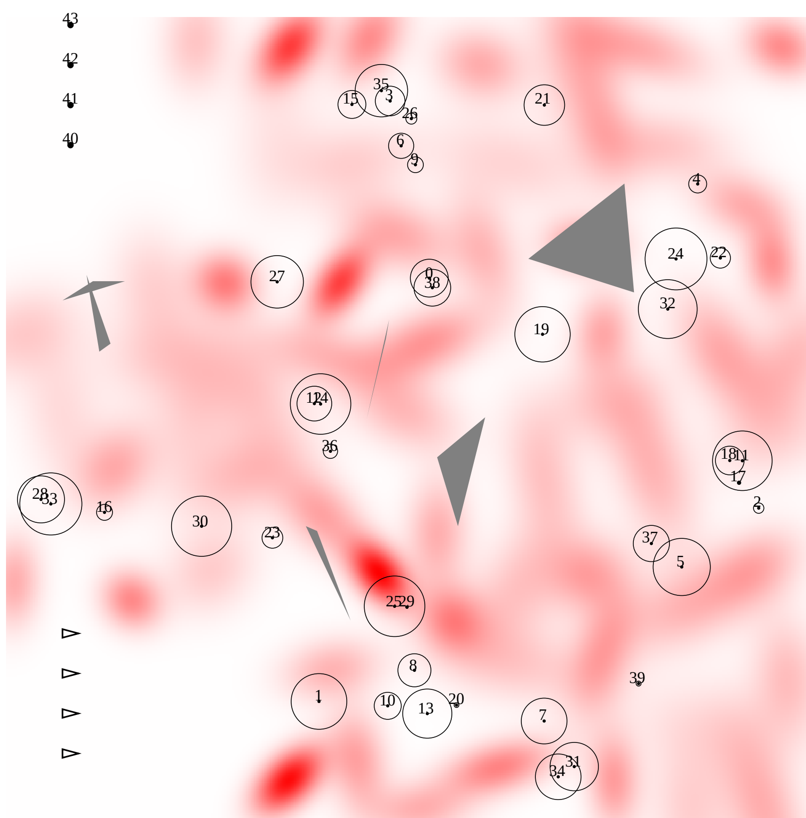


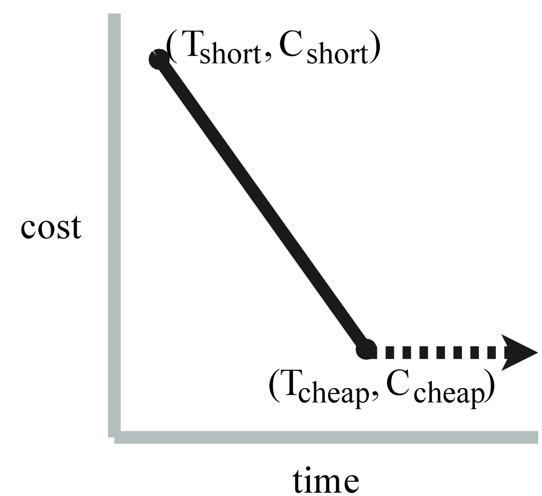
Our Approach

Overview

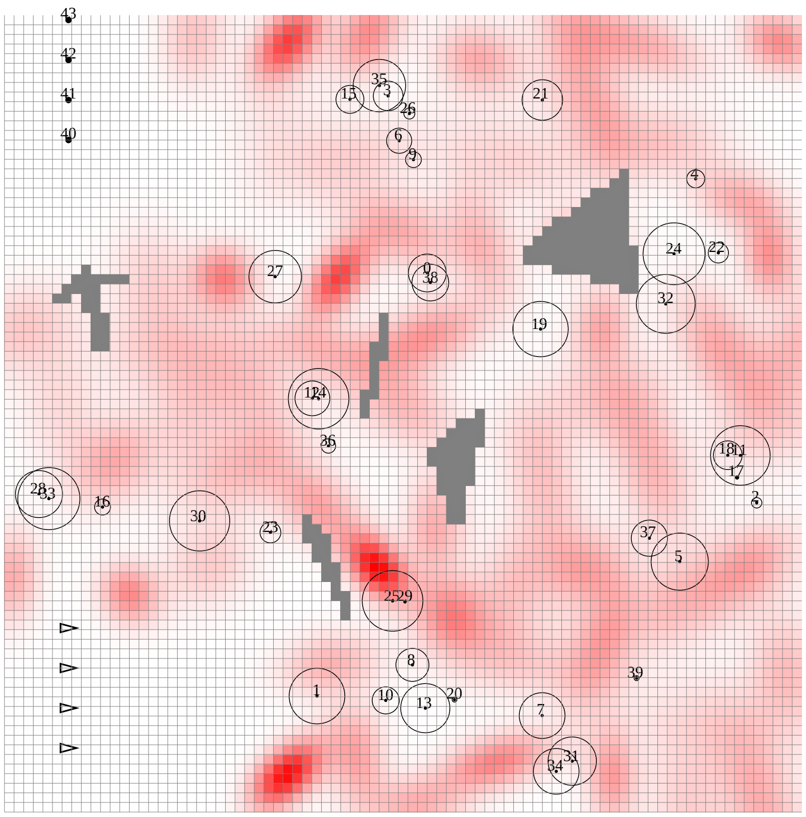
1. Precomputation: find surrogate objective endpoints
   ↓ surrogate objective
2. Sequencer: assign and order waypoints to vehicles
   ↓ ordering
3. Linear Program: assign timepoints to waypoints
   ↓ timetable
4. Routing: find motion plans between waypoints
5. (Feedback: provide new information to previous layers)
   ↓↑ new constraints

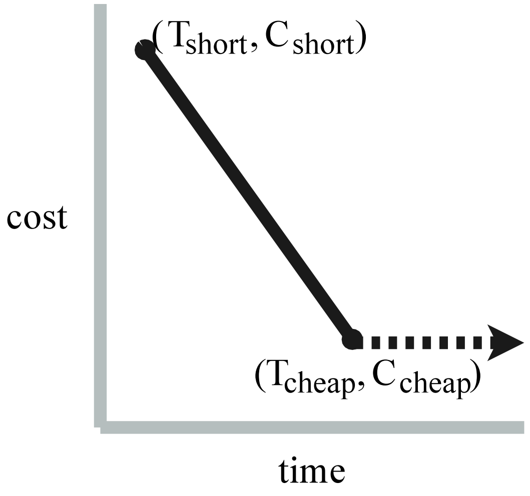
Input: problem
Output: surrogate objective
Techniques: Dijkstra





Input: problem
Output: surrogate objective
Techniques: Dijkstra



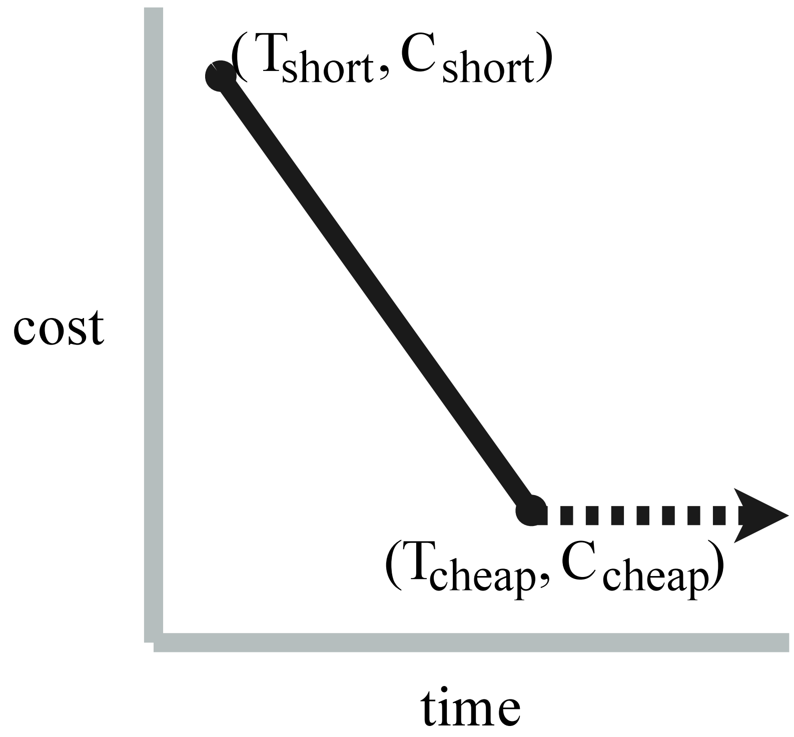


Step 2: Sequencer

Input: surrogate objective
Output: feasible ordering : $\{w_2, w_3, w_1\}$
Techniques: tabu search (based on *Lau, Sim and Teo, 2003*)
simple temporal network (STN)

Step 3: Linear Program

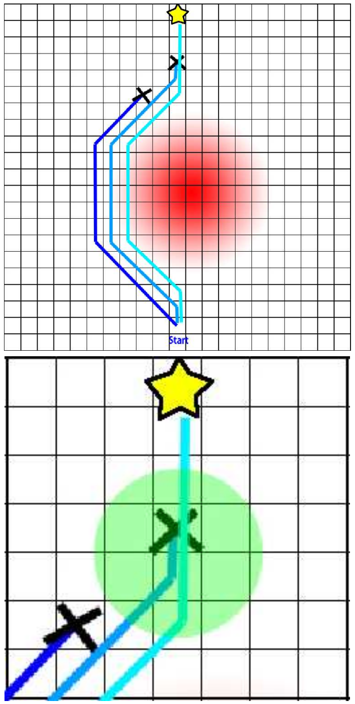
Input: feasible ordering : $\{w_2, w_3, w_1\}$, surrogate objective
Output: timetable : $\{w_2 = 2, w_3 = 3, w_1 = 5\}$
Techniques: linear programming



Step 4: Router

Input: timetable : $\{w_2 = 2, w_3 = 3, w_1 = 5\}$
Output: solution
Techniques: discretized A* search $\langle location, time \rangle$, smoothing

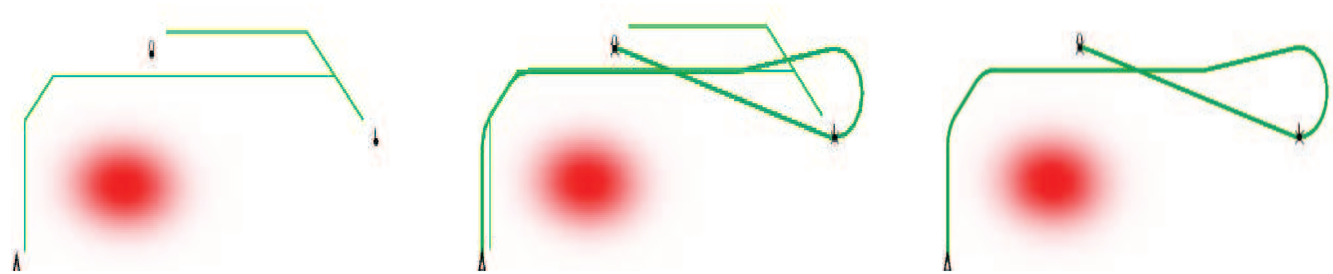
- discretized A* search
- temporal pruning: $t(s,n) + \hat{t}(n,g) > TT(g)$
- re-expansions: $g(n) < g(n')$ but $t(n) > t(n')$
- resumable



Step 4: Router

Input: timetable : $\{w_2 = 2, w_3 = 3, w_1 = 5\}$
Output: solution
Techniques: discretized A* search $\langle location, time \rangle$, smoothing

- smoothing



Feedback

Router : leg can not be routed to achieve timetable

↓ new constraints

Linear Program : LP can not be solved with new constraint

↓ new constraints

- Sequencer

Experiments

Experiment: Small Instance Scaling

Test scalability against a unified A* Search

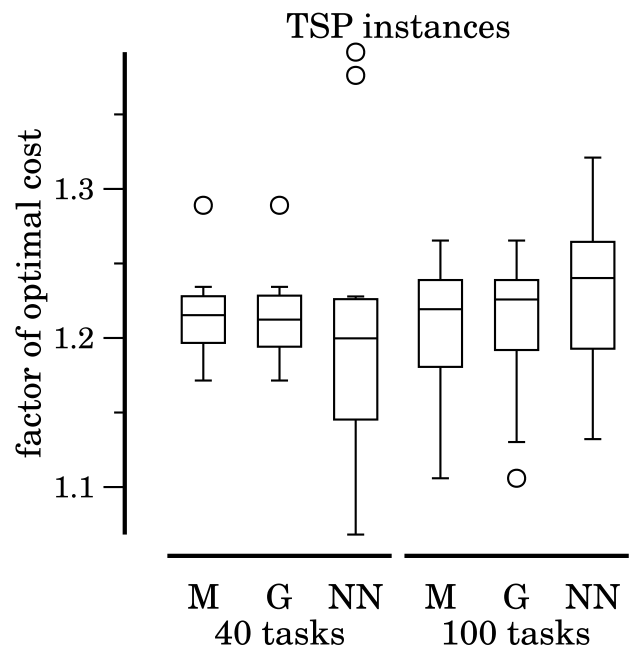
- single vehicle $\langle x, y, \theta, t \rangle$
- no temporal constraints
- spanning tree heuristic
- infinite time w/bounded memory (7.5GB)

# Waypoints	Failure Rate
1	24%
2	64%
3	88%
4	98%
5	98%
6	100%

Generic A* does not scale to meet our requirements

Experiment: Sequencer Stressing

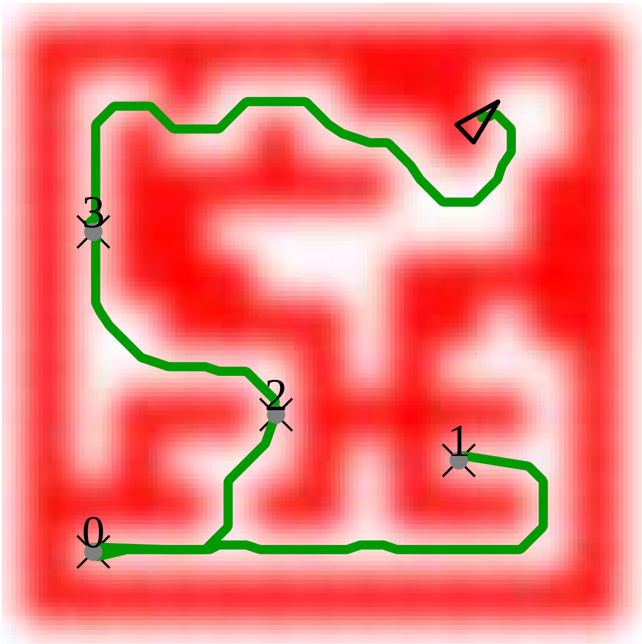
Test ability to find quality orderings
- single vehicle with ϵ turn radius
- no temporal constraints
- uniform cost and no keep-out zones



The sequencer produces near optimal waypoint orderings for TSP instances

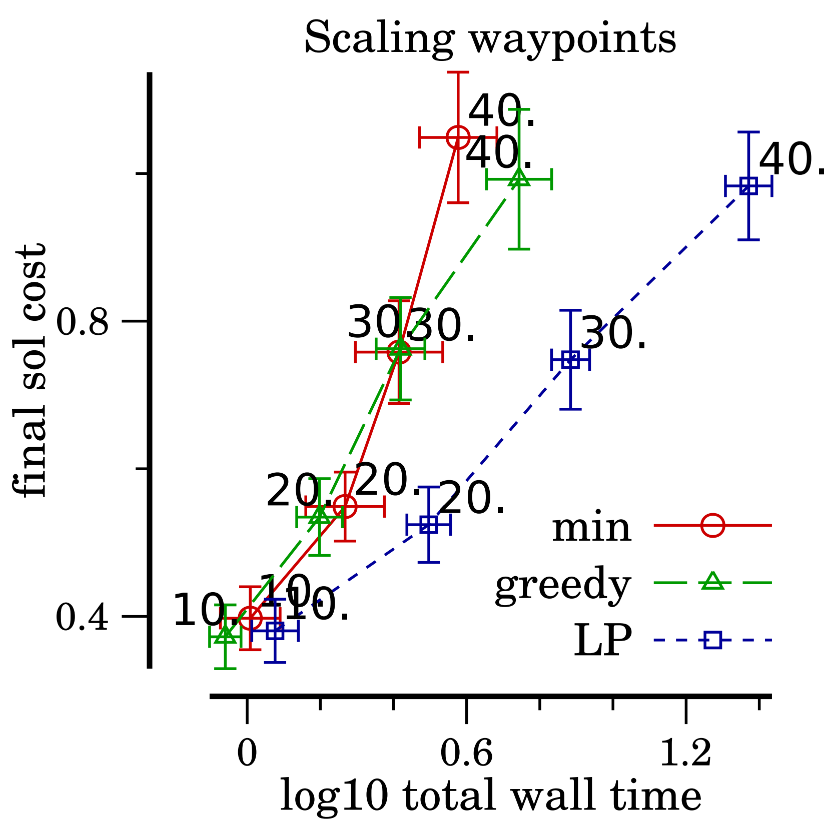
Experiment: Router Stressing

Test ability to find solution paths that minimize cost
- single vehicle
- no temporal constraints



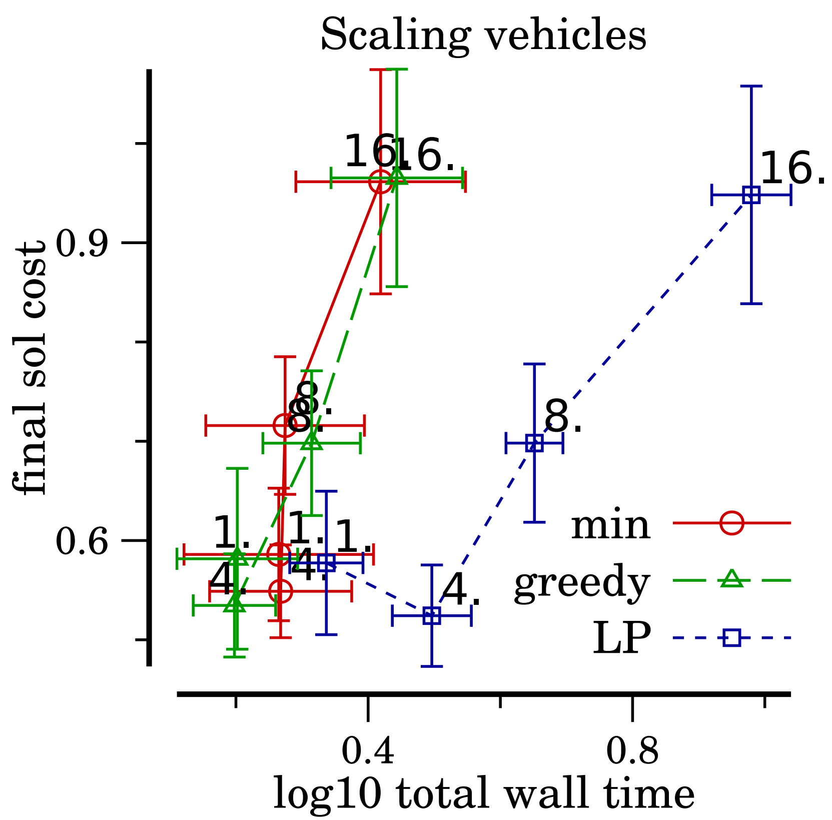
The router produces high quality low level plans for complex cost instances

Test effects of scaling the number of waypoints
4 vehicles



The system scales with an increasing number of waypoints

Test effects of scaling the number of vehicles
20 waypoints



The system scales with an increasing number of vehicles

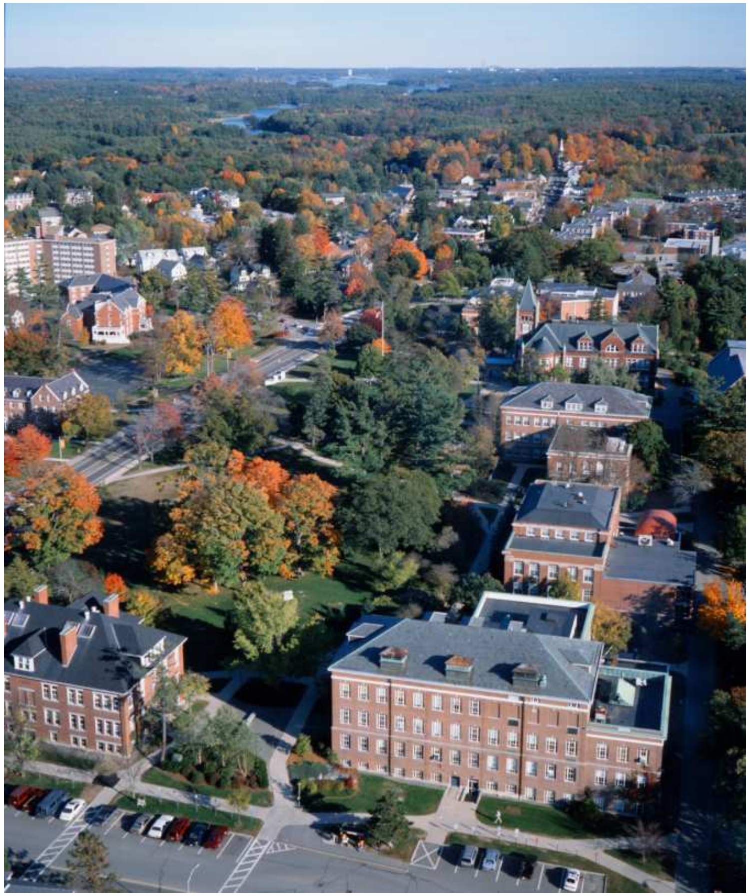
Conclusion

Summary

1. New problem: Waypoint Allocation and Motion Planning
   (a) combines task planning and motion planning
   (b) vehicle routing but now with *real* routing
2. Efficient Solver
   (a) integrates tabu search, blind search, heuristic search, linear programming and simple temporal networks
3. Meets Application Requirements
   (a) 2.5x faster and more scalable than industrial partner's

The University of New Hampshire

Tell your students to apply to grad school in CS at UNH!



- friendly faculty
- funding
- individual attention
- beautiful campus
- low cost of living
- easy access to Boston, White Mountains
- strong in AI, infoviz, networking, bioinformatics

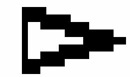
Back-up Slides

Problem Formulation

WAMP : $\langle Size, V, W, K, C, R \rangle$

- V: Vehicles

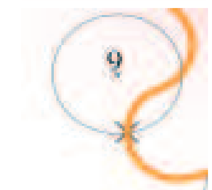
  $v_i = \langle x_0, y_0, \theta_0, v, r \rangle$



- W: Waypoints

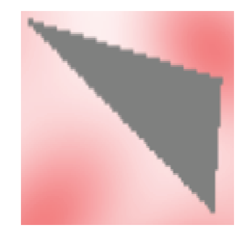
  $w_i = \langle x, y, r, \theta_0, \theta_1, t_s, t_e, A \rangle$

  $A \subseteq V$



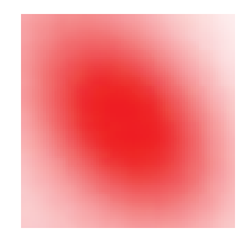
- K: Keep-Out Zones

  $k_i = \langle x_0, y_0, x_1, y_1, x_2, y_2 \rangle$



- C: Cost Zones

  $c_i = \langle x, y, h, \sigma_x, \sigma_y, c \rangle$



- R: Relative Constraints

  $r_i = \langle w_i, w_j, min, max \rangle$