Context-Aware Route Planning

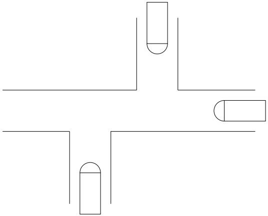
A. W. ter Mors, C. Witteveen, J. Zutt, F. A. Kuipers

Delft University of Technology

September 28, 2010

Problem description

- A set of agents, each with their own start and destination location
- An infrastructure of limited-capacity resources
- Find a set of conflict-free, minimum-cost route plans



Conflict-free routing

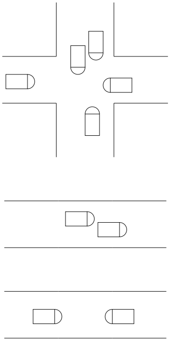
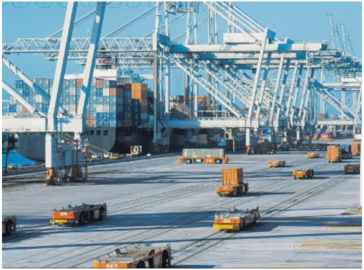


Figure: Resource capacity, overtaking, and head-on collision constraints

Application domains

(a) Airport taxi routing



(b) Automated Guided Vehicles at a container terminal

Approximations for MARP

- Finding an optimal set of route plans is NP-hard
- Sequential, single-agent approximations:
  - Context-Aware Route Planning (CARP): find an optimal single-agent route plan, given reservations of other agents
  - Fixed-Path Scheduling (FPS): find an optimal schedule, along a fixed sequence of resources

CARP and FPS characteristics

CARP:

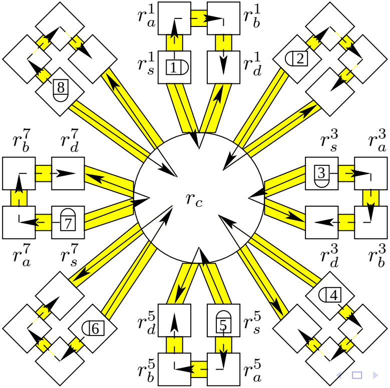
- Worst-case complexity: $O(|\mathcal{A}||R|\log(|\mathcal{A}||R|) + |\mathcal{A}||R|^2)$ ($\mathcal{A}$ the set of agents, R the set of resources.)
- The CARP algorithm utilizes less-congested (in space and time) areas of the infrastructure

FPS:

- Worst-case complexity: $O(|\mathcal{A}||R|\log(|\mathcal{A}||R|))$
- The FPS approach is limited in the way it can make use of traffic information in the route choice

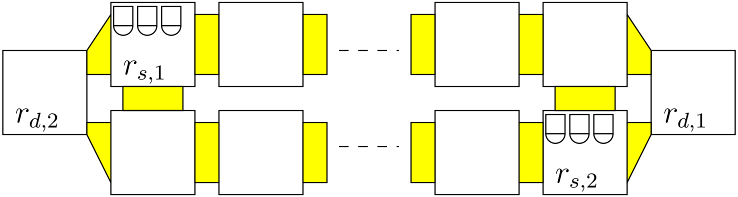
Example global plan quality 1

- CARP finds an optimal global plan
- FPS finds a sub-optimal global plan, in case each agent follows the shortest path



Example global plan quality 2

- FPS finds an optimal global plan, in case each agent follows the shortest path
- Depending on the order in which agents plan, CARP may find a sub-optimal global plan



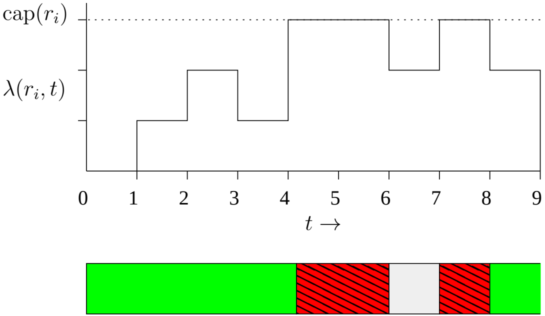
Route planning in space and time

- In sequential routing, each agent plans around the reservations of other agents
- For each infrastructure resource, find the set of free time windows: intervals during which a resource can be entered without creating a conflict
- Construct a graph of free time windows, and perform an adapted A*-search

Free time windows

Free time window

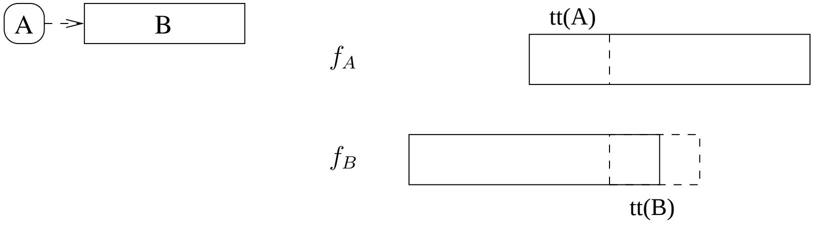
A time interval in which the resource load is less than the capacity. The interval should be at least as long as the minimum travel time.



Free time window graph

Which arcs between free time windows?

1. resources must be connected
2. free time windows must overlap
3. there must be sufficient time to traverse the second resource



CARP algorithm

while $\text{open} \neq \emptyset$ **do**
  $w \leftarrow \text{argmin}_{w' \in \text{open}} f(w')$
  $\text{mark}(w, \text{closed})$
  $r \leftarrow \text{resource}(w)$
  if $r = r_2$ **then**
    return $\text{followBackPointers}(w)$
  $t_{\text{exit}} \leftarrow g(w) = \text{entryTime}(w) + d(\text{resource}(w))$
  for all $w' \in \{\rho(r, t_{\text{exit}}) \setminus \text{closed}\}$ **do**
    $t_{\text{entry}} \leftarrow \max(t_{\text{exit}}, \text{start}(w'))$
    if $t_{\text{entry}} < \text{entryTime}(w')$ **then**
      $\text{entryTime}(w') \leftarrow t_{\text{entry}}$
      $\text{mark}(w', \text{open})$
      $\text{backpointer}(w') \leftarrow w$

CARP algorithm

while open $\neq \emptyset$ **do**
  $w \leftarrow \text{argmin}_{w' \in \text{open}} f(w')$ $\triangleright f = g + h$
  mark(w, closed)
  $r \leftarrow \text{resource}(w)$
  if $r = r_2$ **then**
    return followBackPointers(w)
  $t_{\text{exit}} \leftarrow g(w) = \text{entryTime}(w) + d(\text{resource}(w))$
  for all $w' \in \{\rho(r, t_{\text{exit}}) \setminus \text{closed}\}$ **do**
    $t_{\text{entry}} \leftarrow \max(t_{\text{exit}}, \text{start}(w'))$
    if $t_{\text{entry}} < \text{entryTime}(w')$ **then**
      $\text{entryTime}(w') \leftarrow t_{\text{entry}}$
      mark(w', open)
return null

CARP algorithm

while open $\neq \emptyset$ **do**
  $w \leftarrow \text{argmin}_{w' \in \text{open}} f(w')$ ▷ $f = g + h$
  $\text{mark}(w, \text{closed})$
  $r \leftarrow \text{resource}(w)$
  if $r = r_2$ **then**
    return $\text{followBackPointers}(w)$
  $t_{\text{exit}} \leftarrow g(w) = \text{entryTime}(w) + d(\text{resource}(w))$
  for all $w' \in \{\rho(r, t_{\text{exit}}) \setminus \text{closed}\}$ **do**
    $t_{\text{entry}} \leftarrow \max(t_{\text{exit}}, \text{start}(w'))$
    if $t_{\text{entry}} < \text{entryTime}(w')$ **then**
      $\text{entryTime}(w') \leftarrow t_{\text{entry}}$
      $\text{mark}(w', \text{open})$
return null

CARP algorithm

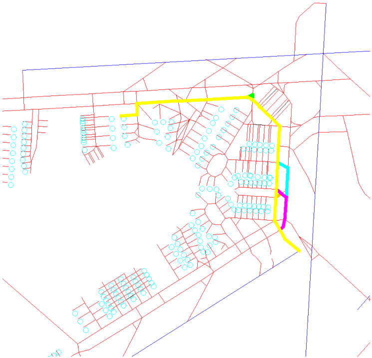
while open $\neq \emptyset$ **do**
    $w \leftarrow \mathrm{argmin}_{w' \in \mathrm{open}} f(w')$ ▷ $f = g + h$
    mark(w, closed)
    $r \leftarrow$ resource(w)
    if $r = r_2$ **then**
        return followBackPointers(w)
    $t_{\mathrm{exit}} \leftarrow g(w) = \mathrm{entryTime}(w) + d(\mathrm{resource}(w))$
    for all $w' \in \{\rho(r, t_{\mathrm{exit}}) \setminus \mathrm{closed}\}$ **do**
        $t_{\mathrm{entry}} \leftarrow \max(t_{\mathrm{exit}}, \mathrm{start}(w'))$
        if $t_{\mathrm{entry}} < \mathrm{entryTime}(w')$ **then**
            $\mathrm{entryTime}(w') \leftarrow t_{\mathrm{entry}}$
            mark(w', open)
return null

Experimental setup

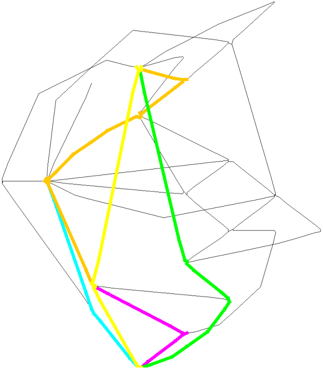
- Compare global plan quality between CARP and k-shortest path scheduling
- Infrastructures: model of Amsterdam Airport Schiphol, and randomly generated instances
- Global plan cost: makespan or sum of agent plan costs

k-shortest paths

Standard algorithm by Yen [1] used to find the 5 shortest paths



(c) Schiphol airport



(d) Random graph on 21 nodes and 39 edges

CPU time comparison

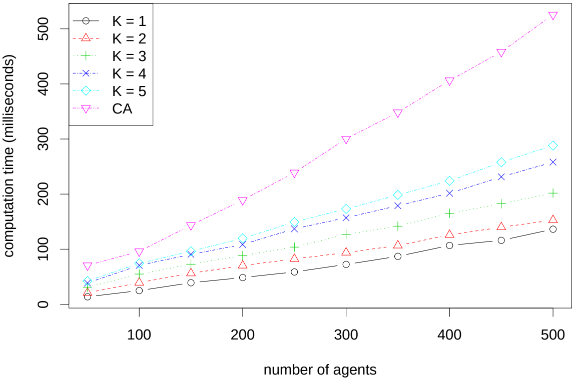
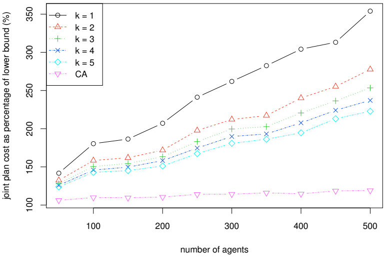


Figure: CPU times for CARP and FPS (k = 1, 2,...,5) on random infrastructures of 180 nodes and 300 edges.

Plan quality comparison: Schiphol

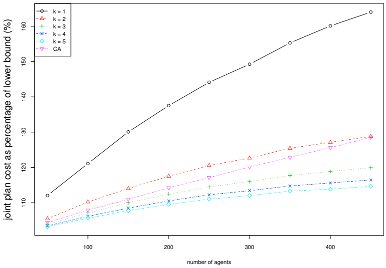


(a) Joint plan cost

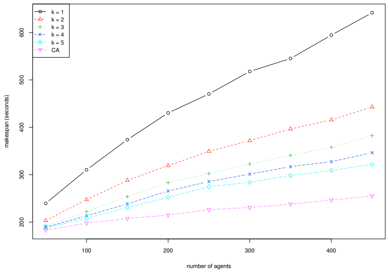
(b) Makespan

Figure: Schiphol airport, an infrastructure of around 1000 resources

Plan quality comparison: random graphs



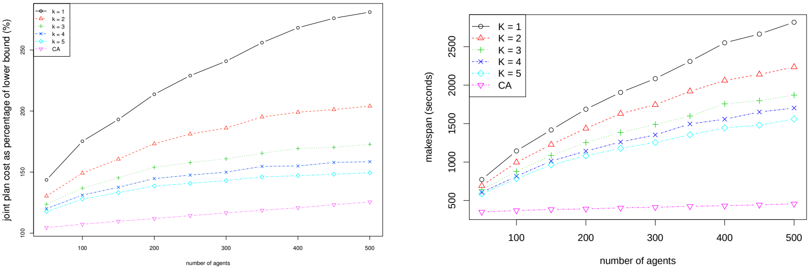
(a) Joint plan cost



(b) Makespan

Figure: Random graphs on 180 nodes and 300 edges

Plan quality comparison: lattice graphs

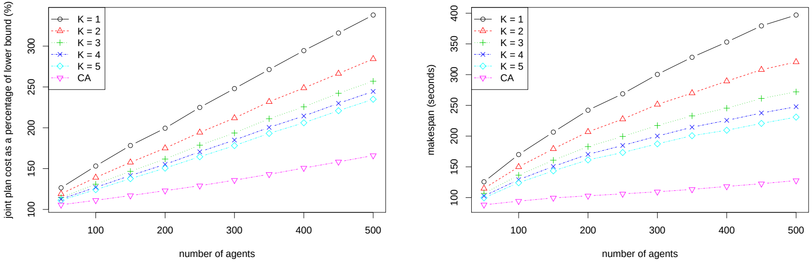


(a) Joint plan cost

(b) Makespan

Figure: Lattice graphs of around 450 resources

Plan quality comparison: small-world networks



(a) Joint plan cost (b) Makespan

Figure: Small-world networks of around 450 resources

Conclusions

- Context-aware route planning is fast and performs well on a wide range of infrastructures
- The success of fixed-path scheduling depends on finding sufficiently different paths between every pair of locations
- Future work: can infrastructure agents improve global plan quality of self-interested route planners?

J. Y. Yen.
Finding the K shortest loopless paths in a network.
Management Science, 17(11):712–716, July 1971.