Carpooling: the 2 Synchronization Points Shortest Paths Problem

Arthur Bit-Monnot
Christian Artigues
Marie-José Huguet
Marc-Olivier Killijian

September 5, 2013
Carpooling

Two users:
- a passenger
- a driver

With their own origin and destination

Goal
Find paths minimizing the travel time
Including:
- pick-up point
- drop-off point
Outline

Shortest Paths: pre-requisites

Solving our carpooling problem

Experiments

Conclusion
Multi-modal graph

- Mixed car, foot and public transportation edges
- FIFO
- Public transportation → time-dependent
Shortest Path Problem

Dijkstra Shortest Path from one node to all (One-to-All)

Properties

• Nodes are settled only once
• Nodes are settled with increasing cost
• Time-independent: backward search (All-to-One)
Best Origin Problem (BOP)

Given several origins with initial cost and arrival times, select the origin minimizing the cost at the arrival
Best Origin Problem (BOP)

Given several origins with initial cost and arrival times, select the origin minimizing the cost at the arrival

\((c = 2, t = 2)\)

\((c = 0, t = 0)\)
Best Origin Problem (BOP)

Given several origins with initial cost and arrival times, select the origin minimizing the cost at the arrival.

\( (c = 2, t = 2) \)

\( (c = 0, t = 0) \)
Best Origin Problem (BOP)

Given several origins with initial cost and arrival times, select the origin minimizing the cost at the arrival.

\[(c = 2, t = 2)\]

\[(c = 4, t = 4)\]

\[(c = 0, t = 0)\]
Best Origin Problem (BOP)

Given several origins with initial cost and arrival times, select the origin minimizing the cost at the arrival

\[(c = 2, t = 2)\]

\[(c = 0, t = 0)\]
Best Origin Problem (BOP)

Given several origins with initial cost and arrival times, select the origin minimizing the cost at the arrival.

\[(c = 2, t = 2)\]

\[(c = 0, t = 0)\]

\[(c = 2, t = 2)\]

\[(c = 12, t = 12)\]
Best Origin Problem (BOP)

Given several origins with initial cost and arrival times, select the origin minimizing the cost at the arrival

\[(c = 5, t = 2)\]

\[(c = 6, t = 0)\]
Best Origin Problem (BOP)

Given several origins with initial cost and arrival times, select the origin minimizing the cost at the arrival.

\((c = 5, t = 2)\)

\((c = 7, t = 4)\)

\((c = 6, t = 0)\)
Best Origin Problem (BOP)

Given several origins with initial cost and arrival times, select the origin minimizing the cost at the arrival

\[(c = 5, t = 2)\]

\[(c = 7, t = 4)\]  \(\Delta = 2\)

\[(c = 21, t = 18)\]

\[(c = 6, t = 0)\]  \(\Delta = 2\)

\[(c = 12, t = 4)\]  \(\Delta = 14\)

\[(c = 18, t = 12)\]  \(\Delta = 10\)
Best Origin Problem (BOP)

Given several origins with initial cost and arrival times, select the origin minimizing the cost at the arrival:

- $\text{(c = 5, t = 2)}$
- $\text{(c = 7, t = 4)}$
- $\text{(c = 8, t = 2)}$
- $\text{(c = 6, t = 0)}$
- $\text{(c = 21, t = 18)}$
Best Origin Problem (BOP)

Given several origins with initial cost and arrival times, select the origin minimizing the cost at the arrival

\[ (c = 5, t = 2) \]

\[ (c = 7, t = 4) \]

\[ (c = 8, t = 2) \]

\[ (c = 6, t = 0) \]

\[ (c = 21, t = 18) \]
Best Origin Problem (BOP)

**Consistency between cost and arrival time**

Given 2 labels \((c, t)\) et \((c', t')\),

Consistency if: \(c < c' \iff t < t'\)

- Need to take cost and arrival time into account
- Mono-objective variant of Martins’ algorithm

**Properties**

- Finds best origin for all nodes
- Labels settled by increasing cost
- Node’s best cost → first settled label
Best Origin Problem (BOP)

Consistency between cost and arrival time

Given 2 labels \((c, t)\) et \((c', t')\),
Consistency if : \(c < c' \iff t < t'\)

Need to take **cost** and **arrival time** into account
Mono-objective variant of Martins’ algorithm
Best Origin Problem (BOP)

Consistency between cost and arrival time
Given 2 labels $(c, t)$ et $(c', t')$, Consistency if : $c < c' \Leftrightarrow t < t'$

Need to take **cost** and **arrival time** into account
Mono-objective variant of Martins’ algorithm

Properties
- Finds best origin for all nodes
- Labels settled by increasing cost
- Node’s best cost $\rightarrow$ first settled label
Best Origin Problem (BOP)

Dominance rules

\((c_x, t_x)\) dominates \((c'_x, t'_x)\) if and only if:

\[ c_x - c'_x \leq t_x - t'_x \]

Exact

\[ t_x \leq t'_x \text{ and } c_x \leq c'_x \]

Heuristic

Complexity

\[ O(|E| \cdot |V|^2) \]
Best Origin Problem (BOP)

Dominance rules

$(c_x, t_x)$ dominates $(c'_x, t'_x)$ if and only if:

- **Exact**
  
  $t_x \leq t'_x$ and $c_x - c'_x \leq t_x - t'_x$

- **Heuristic**
Best Origin Problem (BOP)

**Dominance rules**

\((c_x, t_x)\) dominates \((c'_x, t'_x)\) if and only if:

- **Exact** \(t_x \leq t'_x\) and \(c_x - c'_x \leq t_x - t'_x\)
- **Heuristic** \(t_x \leq t'_x\) and \(c_x \leq c'_x\)
Best Origin Problem (BOP)

**Dominance rules**

\((c_x, t_x)\) dominates \((c'_x, t'_x)\) if and only if:

- *Exact* \(t_x \leq t'_x\) and \(c_x - c'_x \leq t_x - t'_x\)
- *Heuristic* \(t_x \leq t'_x\) and \(c_x \leq c'_x\)

**Complexity**

\(O(|E| \cdot |V|^2)\)
Problem definition

Cost to be minimized
arrival\text{(pedestrian)} - departure\text{(pedestrian)}
+ arrival\text{(driver)} - departure\text{(driver)}

\[ cost = c(P_1) + c(P_2) + \text{waiting time} + 2 \times c(P_3) + c(P_4) + c(P_5) \]
Solving principle

\[ \text{cost} = c(P_1) + c(P_2) + \text{wait} + 2 \times c(P_3) + c(P_4) + c(P_5) \]
Solving principle

\[
\text{cost} = c(P_1) + c(P_2) + \text{wait} + 2 \times c(P_3) + c(P_4) + c(P_5)
\]
Solving principle

\[ \text{cost} = c(P_1) + c(P_2) + \text{wait} + 2 \times c(P_3) + c(P_4) + c(P_5) \]
Solving principle

\[ \text{cost} = c(P_1) + c(P_2) + \text{wait} + 2 \times c(P_3) + c(P_4) + c(P_5) \]
Solving principle

\[\text{cost} = c(P_1) + c(P_2) + \text{wait} + 2 \times c(P_3) + c(P_4) + c(P_5)\]
Solving principle

\[
\text{cost} = c(P_1) + c(P_2) + \text{wait} + 2 \times c(P_3) + c(P_4) + c(P_5)
\]
Solving principle

\[ \text{cost} = c(P_1) + c(P_2) + \text{wait} + 2 \times c(P_3) + c(P_4) + c(P_5) \]
Solving principle

\[ \text{cost} = c(P_1) + c(P_2) + \text{wait} + 2 \times c(P_3) + c(P_4) + c(P_5) \]
Solving principle

\[ \text{cost} = c(P_1) + c(P_2) + \text{wait} + 2 \times c(P_3) + c(P_4) + c(P_5) \]
Solving principle

<table>
<thead>
<tr>
<th>Algo</th>
<th>Source</th>
<th>Dest.</th>
<th>Settled Nodes</th>
<th>Problem</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A_1$</td>
<td>$o_p$</td>
<td>All</td>
<td>$N_1$</td>
<td>SPP</td>
</tr>
<tr>
<td>$A_2$</td>
<td>$o_c$</td>
<td>All</td>
<td>$N_2$</td>
<td>SPP</td>
</tr>
<tr>
<td>$A_3$</td>
<td>$X_{in} = N_1 \cap N_2$</td>
<td>All</td>
<td>$N_3$</td>
<td>Best Orig. (Dijkstra)</td>
</tr>
<tr>
<td>$A_4$</td>
<td>$d_c$</td>
<td>All</td>
<td>$N_4$</td>
<td>SPP (backward)</td>
</tr>
<tr>
<td>$A_5$</td>
<td>$X_{off} = N_3 \cap N_4$</td>
<td>$d_p$</td>
<td>$N_5$</td>
<td>Best Orig. (Martins)</td>
</tr>
</tbody>
</table>
# Solving principle

<table>
<thead>
<tr>
<th>Algo</th>
<th>Source</th>
<th>Dest.</th>
<th>Settled Nodes</th>
<th>Problem</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A_1$</td>
<td>$o_p$</td>
<td>All</td>
<td>$N_1$</td>
<td>SPP</td>
</tr>
<tr>
<td>$A_2$</td>
<td>$o_c$</td>
<td>All</td>
<td>$N_2$</td>
<td>SPP</td>
</tr>
<tr>
<td>$A_3$</td>
<td>$X_{in} = N_1 \cap N_2$</td>
<td>All</td>
<td>$N_3$</td>
<td>Best Orig. (Dijkstra)</td>
</tr>
<tr>
<td>$A_4$</td>
<td>$d_c$</td>
<td>All</td>
<td>$N_4$</td>
<td>SPP (backward)</td>
</tr>
<tr>
<td>$A_5$</td>
<td>$X_{off} = N_3 \cap N_4$</td>
<td>$d_p$</td>
<td>$N_5$</td>
<td>Best Orig. (Martins)</td>
</tr>
</tbody>
</table>

**Integrated approach:** select the algorithm with lowest cost in heap.
Restrictions on pick-up and drop-off points

Integrating knowledge of the problem
Goal: restrain the considered pick-up and drop-off point

Stop conditions:
- $Z_{up}$ explored $\rightarrow$ stop $A_1$ and $A_2$
- $Z_{off}$ explored $\rightarrow$ stop $A_3$ and $A_4$
A*: guided search

Guiding towards a node $d$

**Principle**
Explore nodes close to the destination first

Heuristic $h_d(n)$: lower bound of the distance $n \rightarrow d$
A*: guided search

Guiding towards a node $d$

**Principle**

Explore nodes close to the destination first

Heuristic $h_d(n)$: lower bound of the distance $n \rightarrow d$

Guiding towards an area $Z$

$$H_Z(n) = \min_{z \in Z} h_z(n)$$

Guiding towards the closest node in the area
A*: guided search

Guiding towards a node $d$

**Principle**
Explore nodes close to the destination first

Heuristic $h_d(n):$ lower bound of the distance $n \rightarrow d$

Guiding towards an area $Z$

$$H_Z(n) = \min_{z \in Z} h_z(n)$$

Guiding towards the closest node in the area

**Landmarks:** $H_Z(n)$ computed only once.
Outline

Shortest Paths: pre-requisites

Solving our carpooling problem

Experiments

Conclusion
Experiments: Data

**Instances** Carpooling Bordeaux-Toulouse-Albi

**Graph** South West of France
- 639,765 nodes
- 21,439 public transportation nodes
- 5,000,000 edges

**Restrictions** Pick-up/Drop-off areas
- Entire cities
- Nodes accessible in 10 minutes
### Experimental Results

<table>
<thead>
<tr>
<th>Restrictions</th>
<th>Runtime (ms)</th>
<th>Settled labels</th>
<th>Labels/node in $A_5$</th>
<th>Cost (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>-</td>
<td>48 377</td>
<td>5 610 354</td>
<td>21.52</td>
<td>24 607</td>
</tr>
<tr>
<td>-</td>
<td>4 316</td>
<td>1 793 205</td>
<td>1.17</td>
<td>24 621</td>
</tr>
</tbody>
</table>
## Experimental Results

<table>
<thead>
<tr>
<th>Restrictions</th>
<th>Runtime (ms)</th>
<th>Settled labels</th>
<th>Labels/node in $A_5$</th>
<th>Cost (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>48 377</td>
<td>5 610 354</td>
<td>21.52</td>
<td>24 607</td>
</tr>
<tr>
<td></td>
<td>4 316</td>
<td>1 793 205</td>
<td>1.17</td>
<td>24 621</td>
</tr>
<tr>
<td>cities-guided</td>
<td>5 910</td>
<td>928 487</td>
<td>13.99</td>
<td>24 610</td>
</tr>
<tr>
<td>cities-guided</td>
<td>853</td>
<td>378 404</td>
<td>1.26</td>
<td>24 623</td>
</tr>
</tbody>
</table>
## Experimental Results

<table>
<thead>
<tr>
<th>Restrictions</th>
<th>Runtime (ms)</th>
<th>Settled labels</th>
<th>Labels/node in $A_5$</th>
<th>Cost (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-</td>
<td>48 377</td>
<td>5 610 354</td>
<td>21.52</td>
<td>24 607</td>
</tr>
<tr>
<td>-</td>
<td>4 316</td>
<td>1 793 205</td>
<td>1.17</td>
<td>24 621</td>
</tr>
<tr>
<td>cities-guided</td>
<td>5 910</td>
<td>928 487</td>
<td>13.99</td>
<td>24 610</td>
</tr>
<tr>
<td>cities-guided</td>
<td>853</td>
<td>378 404</td>
<td>1.26</td>
<td>24 623</td>
</tr>
<tr>
<td>10-min-guided</td>
<td>220</td>
<td>122 706</td>
<td>4.54</td>
<td>24 881</td>
</tr>
<tr>
<td>10-min-guided</td>
<td>195</td>
<td>120 126</td>
<td>1.15</td>
<td>24 881</td>
</tr>
</tbody>
</table>
Outline

Shortest Paths: pre-requisites

Solving our carpooling problem

Experiments

Conclusion
Conclusion

• Study of the Best Origin Problem

• Efficient method to solve the carpooling problem

• Integration of user preferences to provide a further speed up
Questions?