Ridesharing: Simulator, Benchmark, and Evaluation

James Jie Pan, Guoliang Li, Juntao Hu

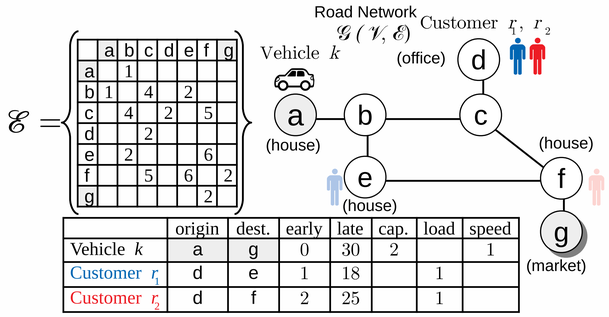
Motivation

Can ridesharing algorithms make high-quality assignments under real-time constraint?

How can we characterize assignment techniques?

How do we design a benchmark?

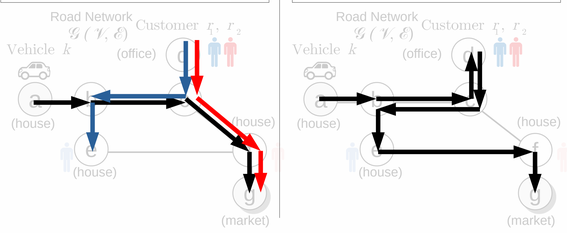
Reason for Ridesharing



	origin	dest.	early	late	cap.	load	speed
Vehicle k	a	g	0	30	2		1
Customer r_1	d	e	1	18		1	
Customer r_2	d	f	2	25		1	

Travel Separately (No Ridesharing)
Cost = 27

Travel Together (With Ridesharing)
Cost = 23



- Potential for saving total travel distances, thereby reducing overall fuel use and emissions
- Potential for reducing number of vehicles on the road

Customer-to-Vehicle Assignment Algorithms

An Exact Offline Method: Branch-and-Bound (BB) [Cor06]

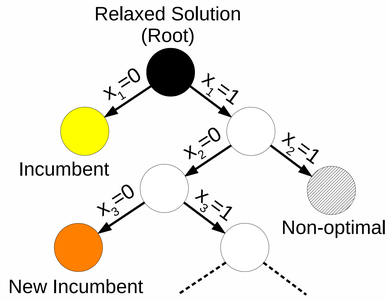
1. Label vehicles $M=\{1..m\}$, customers $N=\{1..n\}$
2. Design origin/destination labels:

For...	Origin is labeled...	Destination is labeled...
Vehicle $k \in M$	k	$k+m$
Customer $r \in N$	$2m+r$	$2m+n+r$

$\{1,\ldots,m\}, \{m+1,\ldots,2m\}, \{2m+1,\ldots,2m+n\}, \{2m+n+1,\ldots,2m+2n\}$
$M_o \quad M_d \quad N_o \quad N_d$

3. Min. $\sum_{k\in M}\sum_{i\in V}\sum_{j\in V} c_{ij}x_{ij}^k$ s.t. constraints, where

$V = N_o \cup N_d \cup \{k, k+m\}$ and c_{ij} is shortest-path cost from i to j.

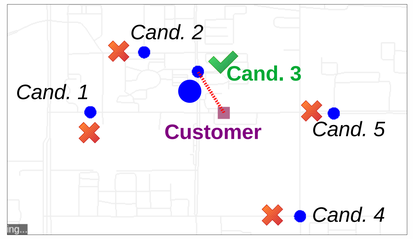


1. **Relax** the LP and solve with simplex
2. **Branch** on a decision variable
3. **Bound** against incumbent
4. **Repeat** until no more branches

Online Search-Based Algorithms: $\textbf{Assignment} = \sigma_{\textbf{Pred}}(\textbf{Candidates})$

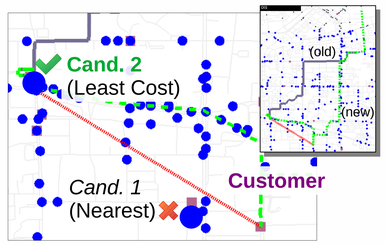
= assignment = candidate vehicle = customer

Nearest-Neighbor (NN)



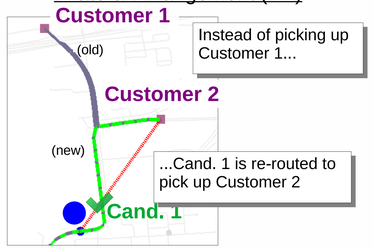
For $k \in$ Candidates,
Pred(k) = (k is Nearest) $\wedge$ (k is valid)

Greedy (GR), Kinetic Tree (KT)



For $k \in$ Candidates,
Pred(k) = (k minimizes Cost) $\wedge$ (k is valid)

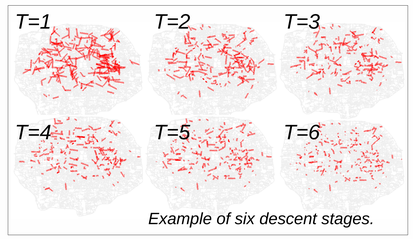
Bilateral Arrangement (BA)



For $k \in$ Candidates,
Pred(k) = (k minimizes Cost) $\wedge$ (k is valid, with Replace if necessary)

Online Join-Based Algorithms: $\textbf{Assignments} = \textbf{Customers} \bowtie_{\textbf{Pred}} \textbf{Candidates}$

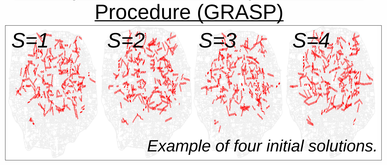
Simulated Annealing (SA)



Example of six descent stages.

Descend using random reassigns, with chance of *hill-climbing*

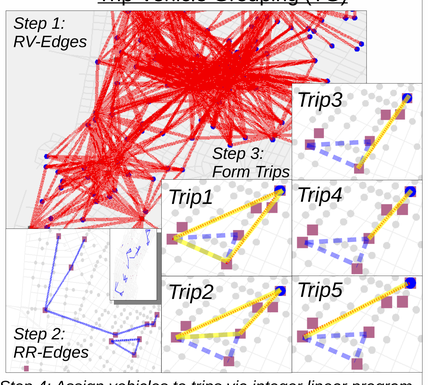
Greedy Randomized Adaptive Search Procedure (GRASP)



Example of four initial solutions.

Descend through n randomized solutions, then select the best one.
Descent procedures:
- *Replace*
- *Swap*
- *Rearrange*

Trip-Vehicle Grouping (TG)



Step 4: Assign vehicles to trips via integer linear program

Our Ridesharing Simulator

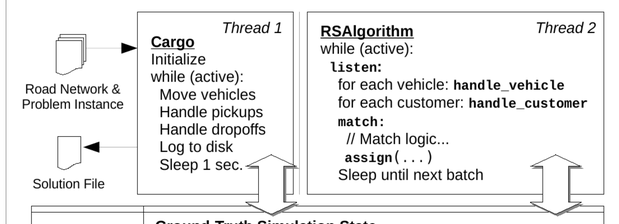


TABLE **nodes**(id, lng, lat)
TABLE **vehicles**(id, origin, destination, early, late, load, queued, status, route, last_visited_node, next_node_distance, schedule)
TABLE **customers**(id, origin, destination, early, late, load, status, assignedTo)
TABLE **stops**(owner, location, type, early, late, visitedAt)

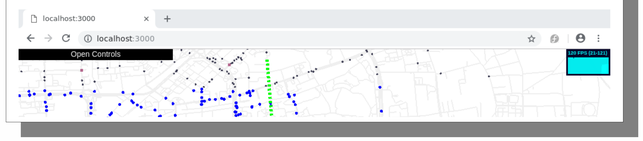
Modularized Ridesharing Algorithms:

```
class YourAlgorithm : public RSAlgorithm {
 public:
  YourAlgorithm();

  /* Your code in these methods */
  virtual void handle_customer(const Customer &);
  virtual void handle_vehicle(const Vehicle &);
  virtual void end();
  virtual void listen();
}
```

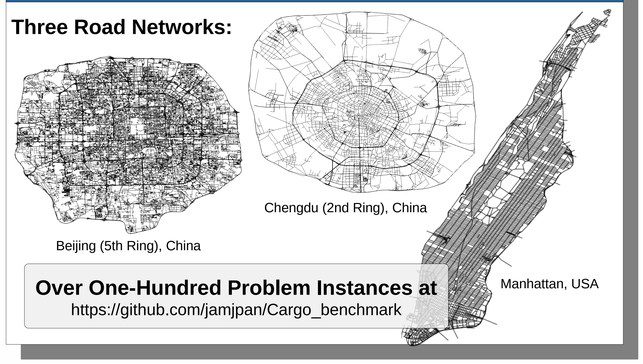
Visit https://github.com/jamjpan/Cargo for sourcecode and examples

WebGL Visualization with Live Updates:



Our Benchmark Problem Instances

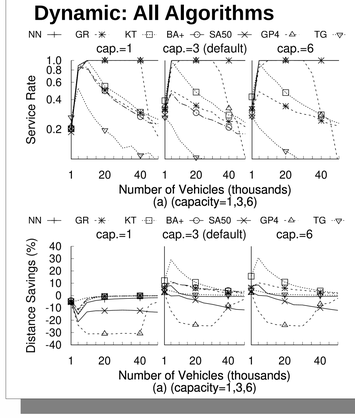
Three Road Networks:



Over One-Hundred Problem Instances at
https://github.com/jamjpan/Cargo_benchmark

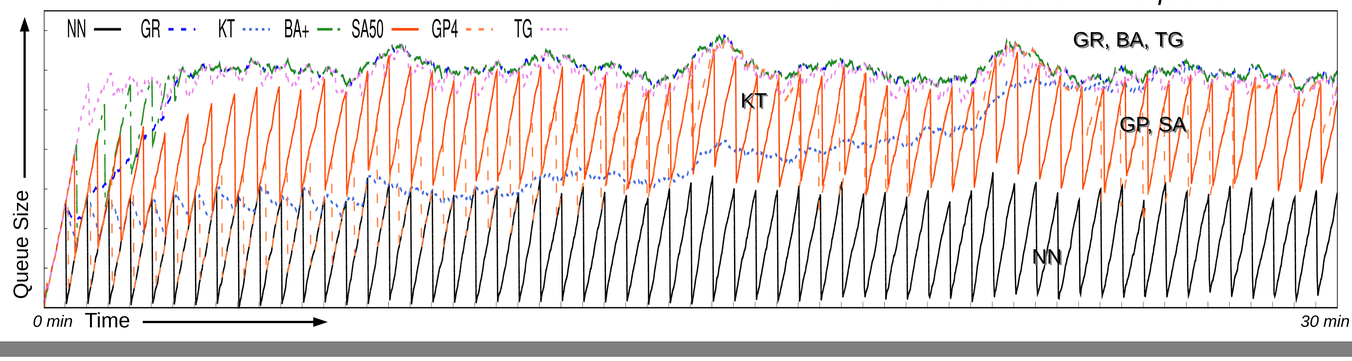
Our Simulation Results

Dynamic: All Algorithms



Number of Queued Customers Over Time, per Algorithm

For high service rate, matching rate should meet or exceed the rate of requests.



- The algorithms that achieved the highest service rates were those that cleared the queue (Nearest Neighbor, Simulated Annealing, and GRASP).
- Kinetic Tree saved more distance, achieved better service rate, and cleared the queue faster than Greedy.
- Despite high service rate, Simulated Annealing and GRASP did not perform enough descents to result in distance savings.
- Trip-Vehicle Grouping struggled to clear the queue and had the lowest service rate.