

QoS optimization for an on-demand transportation system via a fractional linear objective function

Thierry Garaix, University of Avignon (France)

Column Generation 2008



- Outline
- Introduction
 - Load charge ratio
 - Experiments
 - Conclusion

Plan

- On-demand transportation system (DARP)
- Column generation for the DARP
- Maximization of the loading rate
- Computational results and observations

Outline

- Introduction
- Load charge ratio
- Experiments
- Conclusion

Master problem

$$\min \sum_{\omega \in \Omega} C_{\omega} \lambda_{\omega}$$

subject to

$$\sum_{\omega \in \Omega} \lambda_{\omega} \rho_{\omega r} = 1 \quad \forall r = 1, \dots, R$$

$$\sum_{\omega \in \Omega_k} \lambda_{\omega} \leq K_k \quad \forall k = 1, \dots, K$$

$$\lambda_{\omega} \text{ integer} \quad \forall \omega \in \Omega$$

- Ω_k : feasible routes with vehicle type k , $\Omega = \bigcup_{k=1}^K \Omega_k$
- Objective function minimizes total inconvenience
- $\rho_{\omega r} \in \{0, 1\}$: request r is in route ω

Outline

→ Introduction

- DARP
- ✓ *Classical method*
- QoS
- Load charge ratio
- Experiments
- Conclusion

Slave problems

$$C_\omega = \sum_{(i,j) \in \bar{\omega}} C_{ij}$$

$$RC_\omega = \sum_{(i,j) \in \bar{\omega}} (C_{ij} - \pi_{ij})$$

- $\bar{\omega}$: arcs of ω , π_{ij} : “well defined” dual cost
- We model slave problem as an ESPPRC solved by dynamic programming
 - Label L is a partial path ending in v^L at time T^L with open requests \mathcal{O}^L , closed requests that are still time reachable \mathcal{S}^L and a cost C^L
 - Dominance rule: $v^L = v^M$, $T^L \leq T^M$, $\mathcal{O}^L \subset \mathcal{O}^M$ and $\mathcal{S}^L \subset \mathcal{S}^M \cup \mathcal{O}^M \cup \mathcal{T}^M$ and $C^L \leq C^M$; \mathcal{T}^M are time reachable requests for M

Outline

→ Introduction

- DARP
- ✓ Classical method
- QoS
- Load charge ratio
- Experiments
- Conclusion

Quality of service

- We develop an algorithm using robust accelerations (LDS, heuristic. . .). We solve instances with 100 requests
- Adaptation to new QoS criteria?
 - *time lost* for *inbound* and *outbound* requests
 - Cost resource extension fonction is not non-decreasing
 - *distance*
 - Working network is a p-graph of non-dominated (distance/time) paths
 - *loading rate*
 - Fractional linear objective function

Outline

- **Introduction**
 - DARP
 - Classical method
 - ✓ **QoS**
- Load charge ratio
- Experiments
- Conclusion

Loading rate

- *Loading rate*: number of passengers times their travel time divided by the total travel time of vehicles

$$\max \frac{\sum_{\omega \in \Omega} N_{\omega} \lambda_{\omega}}{\sum_{\omega \in \Omega} D_{\omega} \lambda_{\omega}} = \frac{N(\lambda)}{D(\lambda)} \quad \left| \quad \begin{array}{l} N_{\omega} = \sum_{(i,j) \in \bar{\omega}} f_{ij} T_{ij} \\ D_{\omega} = \sum_{(i,j) \in \bar{\omega}} T_{ij} \end{array} \right.$$

- f_{ij} (variables): passenger flow
- T_{ij} (data): travel time. We exclude waiting times
- Practical interests:
 - Good indicator for local authorities, that prefer full-loaded vehicles on road
 - A social impact

Outline

- Introduction
- **Load charge ratio**
 - ✓ **Definition**
 - Direct method
 - Iterative method
- Experiments
- Conclusion

Direct algorithm – Charnes-Cooper (62)

- Change of variable $x = 1 / \sum_{\omega \in \Omega} D_{\omega} \lambda_{\omega}$:

$$\max x \sum_{\omega \in \Omega} N_{\omega} \lambda_{\omega}$$

$$\text{s.t.} \quad x \sum_{\omega \in \Omega} \lambda_{\omega} \rho_{\omega r} = x \quad , \forall r = 1, \dots, R$$

$$x \sum_{\omega \in \Omega_k} \lambda_{\omega} \leq x K_k \quad , \forall k = 1, \dots, K$$

$$x \sum_{\omega \in \Omega} D_{\omega} \lambda_{\omega} = 1$$

$$x \lambda_{\omega} \geq 0 \quad , \forall \omega \in \Omega$$

$$x \geq 0$$

- $\lambda_{\omega} \leftarrow x \lambda_{\omega}$: $\max \sum_{\omega \in \Omega} N_{\omega} \lambda_{\omega}$

$$\text{s.t.} \quad \sum_{\omega \in \Omega} \lambda_{\omega} \rho_{\omega r} = x \quad , \forall r = 1, \dots, R$$

$$\sum_{\omega \in \Omega_k} \lambda_{\omega} \leq x K_k \quad , \forall k = 1, \dots, K$$

$$\sum_{\omega \in \Omega} D_{\omega} \lambda_{\omega} = 1$$

$$\lambda_{\omega} \geq 0 \quad , \forall \omega \in \Omega$$

$$x \geq 0$$

Outline

- Introduction
- **Load charge ratio**
 - Definition
 - ✓ **Direct method**
 - Iterative method
- Experiments
- Conclusion

Direct algorithm – slave problem

- ESPPRC: $\min - \sum_{\omega \in \Omega} N_{\omega} \lambda_{\omega}$
- $C_{\omega} = \sum_{(i,j) \in \bar{\omega}} -f_i T_{ij}$
- $RC_{\omega} = C_{\omega} - \sum_{(i,j) \in \bar{\omega}} (\pi_{ij} + \mu T_{ij})$
- New dominance rule:
 - $v^L = v^M, T^L \leq T^M, \mathcal{O}^L \subset \mathcal{O}^M$
 - $C^L \leq C^M$ and $\mathcal{S}^L \subset \mathcal{S}^M \cup \mathcal{O}^M \cup \mathcal{T}^L$ become

$$C^L \leq C^M + \sum_{r \in \mathcal{O}^M \setminus \mathcal{O}^L} \det(r^-, T^M) +$$

$$\sum_{r \in \mathcal{S}^L \setminus (\mathcal{S}^M \cup \mathcal{O}^M \cup \mathcal{T}^M)} \min\{\det(r^+, T^M) + \det(r^-, T^M), 0\}$$

Outline

- Introduction
- **Load charge ratio**
 - Definition
 - ✓ **Direct method**
 - Iterative method
- Experiments
- Conclusion

Direct algorithm – slave problem

- Lower bounds on detours by vertex v
 - Compare each feasible arc (u, w) with path (u, v, w)
 - $det(v, t) =$

$$\min_{u, w \in \mathcal{V}} (-\mu - f_u)(T_{uv} + T_{vu} - T_{uw}) - F_v T_{vw} - \pi_{uv}$$
 - Durations observe triangular inequality
 - Pre-preprocessing $\max f_u$ depending on u, v and w type, since other values are constant at each iteration

Outline

- Introduction
- **Load charge ratio**
 - Definition
 - ✓ **Direct method**
 - Iterative method
- Experiments
- Conclusion

Iterative algorithm – Dinkelbach(67)

- $P = \{\lambda \in \mathbb{R}^+ : \sum_{\omega \in \Omega} \lambda_{\omega} \rho_{\omega r} = 1 \forall r = 1, \dots, R;$
 $\sum_{\omega \in \Omega_k} \lambda_{\omega} \leq K_k \forall k = 1, \dots, K\}$
- MP: $\max_{\lambda \in P} q(\lambda) = N(\lambda)/D(\lambda)$
- Introduce $F(x) = \max_{\lambda \in P} N(\lambda) - xD(\lambda)$
- Propertie: λ^* is optimal for MP $\Leftrightarrow F(q(\lambda^*)) = 0$
 - Find λ^n s.t. $q(\lambda^*) - q(\lambda^n) < \delta, \delta > 0$
 \Leftrightarrow Find λ^n s.t. $F(q(\lambda^n)) < \varepsilon, \varepsilon > 0$
 - Since $F(x)$ is decreasing and $F(q(\lambda^*)) = 0$
- Proof:
 - $q(\lambda^0) = N(\lambda^0)/D(\lambda^0) \Rightarrow F(q(\lambda^0)) \geq 0$
 - $F(q(\lambda^0)) = 0 \Rightarrow 0 > N(\lambda) - q(\lambda^0)D(\lambda), \forall \lambda$
 $\Rightarrow q(\lambda^0) = \max_{\lambda \in P} N(\lambda)/D(\lambda)$

Outline

- Introduction

→ Load charge ratio

- Definition
- Direct method
- ✓ Iterative method

- Experiments

- Conclusion

Iterative algorithm

- Sequence (λ^n) : λ^{n+1} is defined as the optimal solution of $F(q(\lambda^n)) = \max_{\lambda \in P} N(\lambda) - q(\lambda^n)D(\lambda)$
- Sequence $(q^n) = q(\lambda^n) = N(\lambda^{n-1})/D(\lambda^{n-1})$ increases and converges
- Algorithm builds (λ^n) and stops when $F(q^n) < \varepsilon$
- Two schemes integrate column generation (CG):
 - Apply Dinkelbach algorithm on MP
 - Apply Dinkelbach algorithm on each RMP
- Branch-and-Bound scheme is based on q^n values

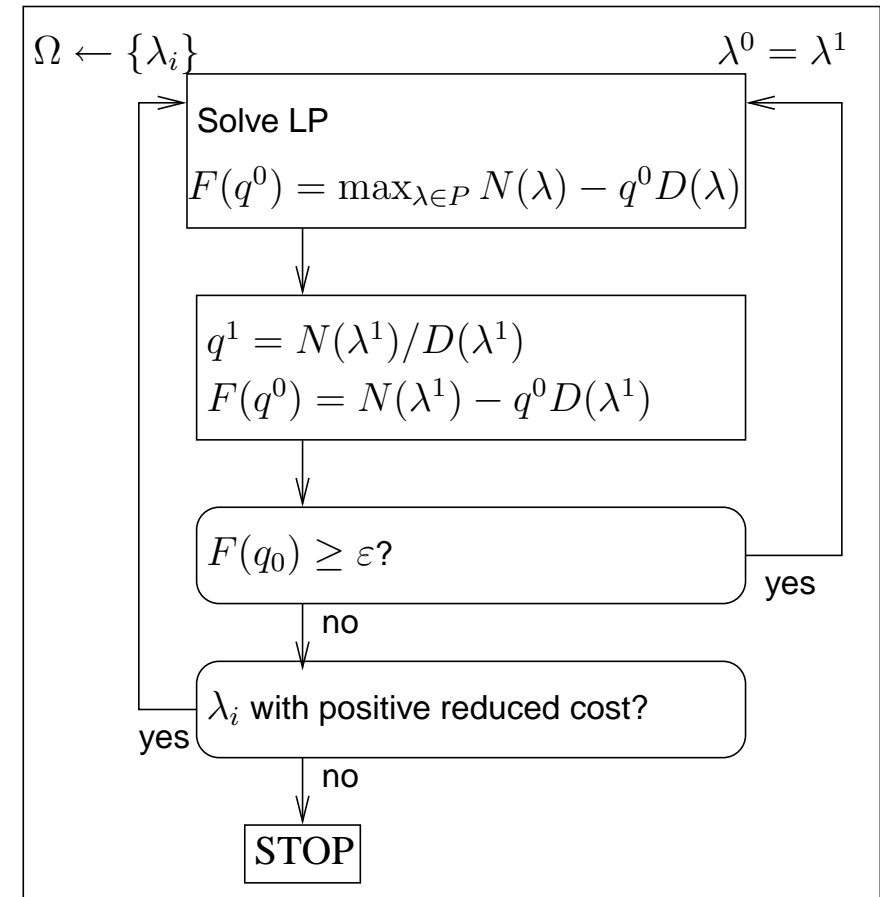
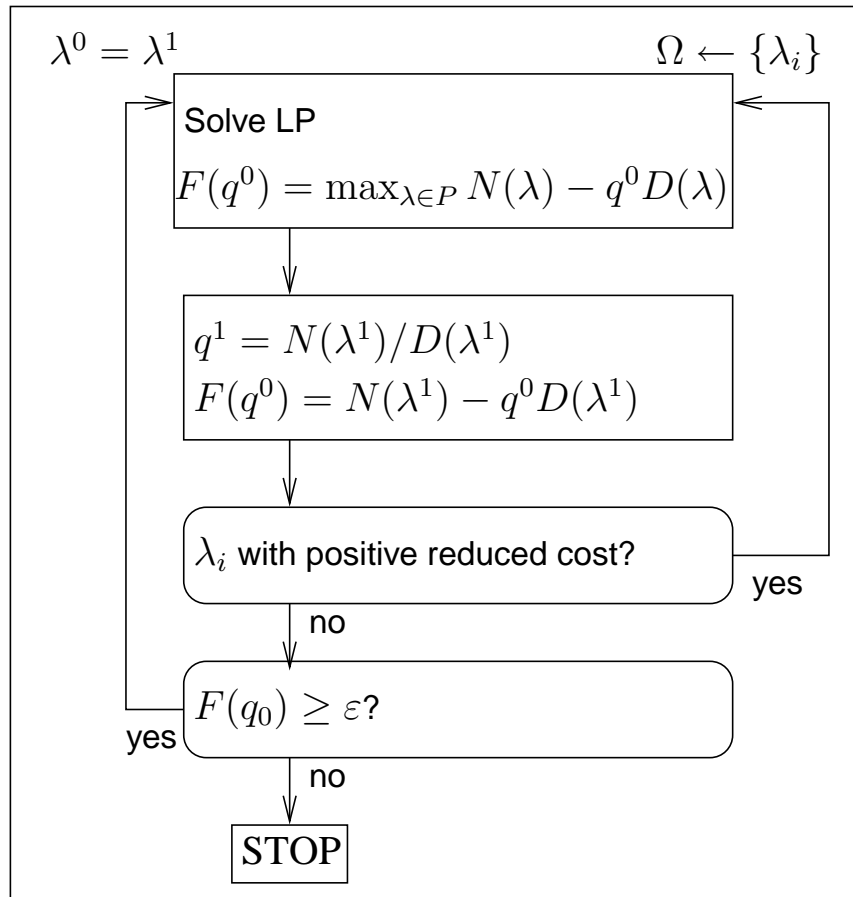
Outline

- Introduction
- **Load charge ratio**
 - Definition
 - Direct method
 - ✓ **Iterative method**
- Experiments
- Conclusion

Iterative algorithm

Outline

- Introduction
- **Load charge ratio**
 - Definition
 - Direct method
 - ✓ **Iterative method**
- Experiments
- Conclusion



Iterative algorithm – slave problem

- ESPPRC: $\min -F(q^0) = q^0 D(\lambda) - N(\lambda)$
- $C_\omega = \sum_{(i,j) \in \bar{\omega}} q^0 T_{ij} - f_i T_{ij}$
- $RC_\omega = C_\omega - \sum_{(i,j) \in \bar{\omega}} \pi_{ij}$
- Same remarks as for the direct algorithm and lower bounds on detours:
 - $det(v, t) = \min_{u,w \in \mathcal{V}} (q^0 - f_u)(T_{uv} + T_{vu} + T_{uw}) - F_v T_{vw} - \pi_{uv}$
 - Pre-preprocessing $\max f_u$

Outline

- Introduction
- **Load charge ratio**
 - Definition
 - Direct method
 - ✓ **Iterative method**
- Experiments
- Conclusion

Instances

- Two benchmarks
 - Li and Lim (≈ 50 requests) from Solomon's
 - Cordeau (from 16 to 96 requests)
 - One vehicle type
- Modifications
 - We compute new tight time windows
 - $T_{r^-}^{sup} - T_{r^+}^{inf} = 1.5T_{r^+r^-}$

Outline

- Introduction
- **Load charge ratio**
 - Definition
 - Direct method
 - Iterative method
- Experiments
- Conclusion

Results on random Li & Lim's

instance name	loading rate		cpu sec.	
	$\min D(\lambda)$	$\max N(\lambda)/D(\lambda)$	direct	iterative
r101	0.5721	0.5692	0	0
r102	0.7078		0	6
r103	0.7642		0	6
r104	0.6359		0	1
r105	0.6035		0	1
r106	0.7392		0	4
r107	0.7727		0	3
r108	0.6266	0.6274	0	1
r109	0.7957		0	0
r110	0.6013	0.6070	0	0
r111	0.6134		0	2
r112	0.5987	0.6024	0	0

Outline

- Introduction
- **Load charge ratio**
 - Definition
 - Direct method
 - Iterative method
- Experiments
- Conclusion

Results on cluster Li & Lim's

instance name	loading rate		cpu sec.	
	$\min D(\lambda)$	$\max N(\lambda)/D(\lambda)$	direct	iterative
c101	0.8163		4	2
c102	0.8870		6	3
c103	0.6882		3	2
c104	0.4509		1	1
c105	0.7310		4	2
c106	0.6626		4	3
c107	0.6967		4	1
c108	0.7987		3	2
c109	0.7957		3	1

Outline

- Introduction
- **Load charge ratio**
 - Definition
 - Direct method
 - Iterative method
- Experiments
- Conclusion

Results on random/cluster Li & Lim's

instance name	loading rate		cpu sec.	
	$\min D(\lambda)$	$\max N(\lambda)/D(\lambda)$	direct	iterative
rc101	0.6708	0.6968	0	0
rc102	0.8639	0.8712	0	2
rc103	0.6984		0	2
rc104	0.5303	0.5356	0	1
rc105	0.6184	0.6207	0	0
rc106	0.7919	0.7968	0	0
rc107	0.5890	0.5942	0	0
rc108	0.5889	0.6032	0	0

Outline

- Introduction
- **Load charge ratio**
 - Definition
 - Direct method
 - Iterative method
- Experiments
- Conclusion

Results on Cordeau's

instance name	loading rate		cpu sec.	
	min $D(\lambda)$	max $N(\lambda)/D(\lambda)$	direct	iterative
b2-16	0.6540		0	0
b2-20	0.7142	0.7190	1	0
b2-24	0.6796		1	1
b3-18	0.7051		0	0
b3-24	0.6888		1	0
b3-30	0.7152		2	0
b3-36	0.7072		5	2
b4-16	0.7315	0.7326	0	0
b4-24	0.6648		1	0
b4-32	0.7034		3	1
b4-40	0.7137		8	2
b4-48	0.6858		19	10
b5-40	0.6860		6	3
b5-50	0.7004		16	4
b5-60	0.6872		42	5
b6-48	0.7333		15	2
b6-60	0.7294	0.7297	79	5
b6-72	0.7088		115	15
b7-56	0.7123	0.7127	26	30
b7-70	0.7023	0.7033	75	15
b7-84	0.7102	0.7111	199	35
b8-64	0.7033	0.7044	53	50
b8-80	0.7287	0.7297	181	60
b8-96	0.6991	0.6992	638	150

Outline

- Introduction
- **Load charge ratio**
 - Definition
 - Direct method
 - Iterative method
- Experiments
- Conclusion

Conclusions

- Generic schemes managing fractional linear objective function with column generation
 - They succeed in solving our problem
 - Travel time minimization also (almost) optimizes loading rate
- We could implement many of “classical” acceleration methods in the proposed algorithm
- Experiments have to be carried out with less bounded numerator and denominator in order to test the convergence speed

Outline

- Introduction
- **Load charge ratio**
 - Definition
 - Direct method
 - Iterative method
- Experiments
- Conclusion