Traffic Flow Optimization under Fairness Constraints

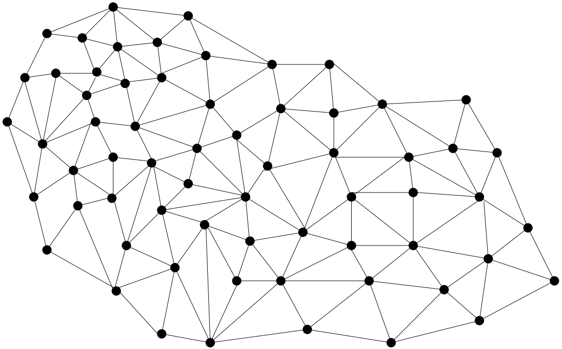
Combinatorial Optimization & Graph Algorithms (COGA)

Technische Universität Berlin

Outline

Route Guidance: Introduction

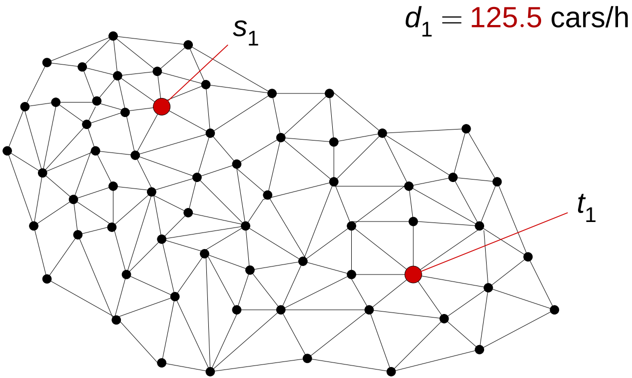
- given: sources s_k, targets t_k, demand rates d_k for traffic demands in a road network



- find "bestroutes from s_k to t_k for all demands $k \in K$

Route Guidance: Introduction

- given: sources s_k, targets t_k, demand rates d_k for traffic demands in a road network

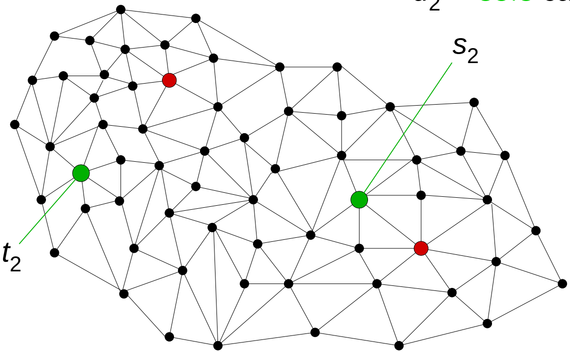


- find "bestroutes from s_k to t_k for all demands $k \in K$

Route Guidance: Introduction

- given: sources s_k, targets t_k, demand rates d_k for traffic demands in a road network

$d_2 = 85.3$ cars/h

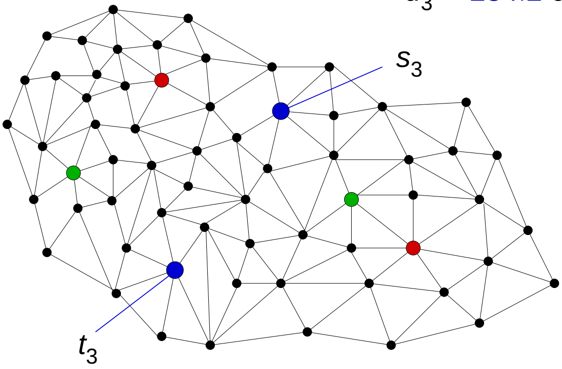


- find "bestroutes from s_k to t_k for all demands $k \in K$

Route Guidance: Introduction

- given: sources s_k, targets t_k, demand rates d_k for traffic demands in a road network

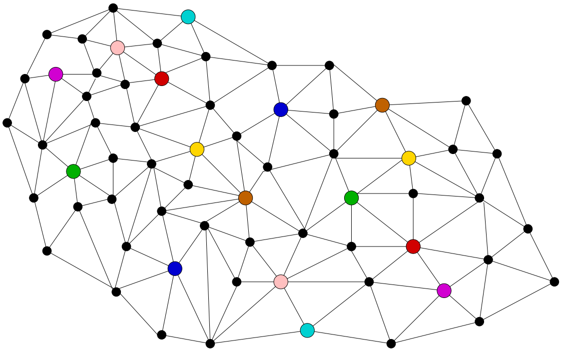
$d_3 = 234.2$ cars/h



- find "bestroutes from s_k to t_k for all demands $k \in K$

Route Guidance: Introduction

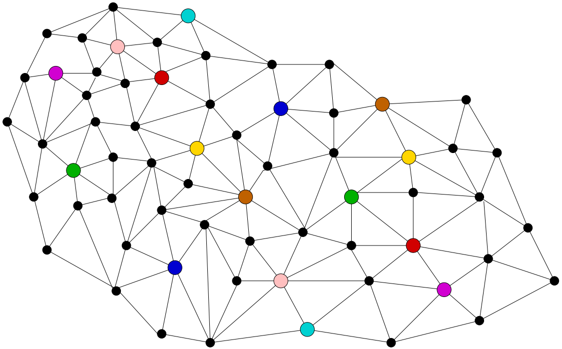
- given: sources s_k, targets t_k, demand rates d_k for traffic demands in a road network



- find "bestroutes from s_k to t_k for all demands $k \in K$

Route Guidance: Introduction

- given: sources s_k, targets t_k, demand rates d_k for traffic demands in a road network



- find ”bestroutes from s_k to t_k for all demands $k \in K$

Route Guidance: State of Technology

Route Guidance Systems...

... play an increasingly important role in today's traffic:

- in-car navigation systems
- urban road pricing schemes / centralized traffic routing

Today's systems use static data only:

- average travel times on road links
- locations / times of typical rush hour congestions
- locations of work zones

⇒ routes computed by static shortest path calculations

Route Guidance: State of Technology

Route Guidance Systems...

... play an increasingly important role in today's traffic:

- in-car navigation systems
- urban road pricing schemes / centralized traffic routing

Today's systems use static data only:

- average travel times on road links
- locations / times of typical rush hour congestions
- locations of work zones

⇒ routes computed by static shortest path calculations

Route Guidance: State of Technology

Route Guidance Systems...

... play an increasingly important role in today's traffic:

- in-car navigation systems
- urban road pricing schemes / centralized traffic routing

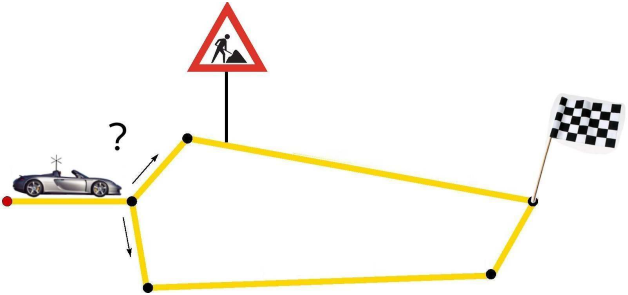
Today's systems use static data only:

- average travel times on road links
- locations / times of typical rush hour congestions
- locations of work zones

$\Rightarrow$ routes computed by static shortest path calculations

Results of Widespread Static Route Guidance

Travelers with the same origin and destination receive the same route suggestions:

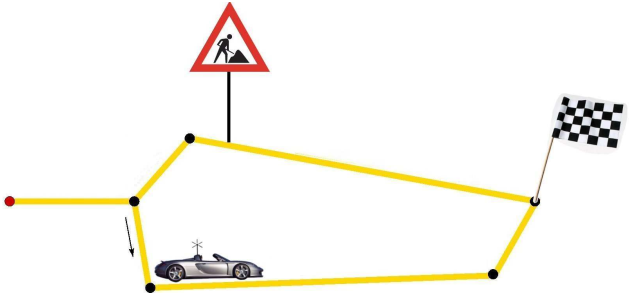


- suggested routes often not the quickest
- drivers will not accept route suggestions

⇝ benefits of route guidance strongly compromised

Results of Widespread Static Route Guidance

Travelers with the same origin and destination receive the same route suggestions:

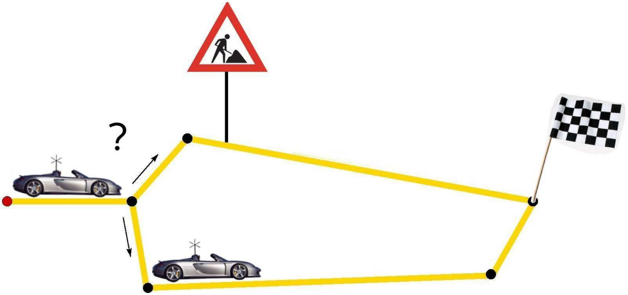


- suggested routes often not the quickest
- drivers will not accept route suggestions

⇝ benefits of route guidance strongly compromised

Results of Widespread Static Route Guidance

Travelers with the same origin and destination receive the same route suggestions:

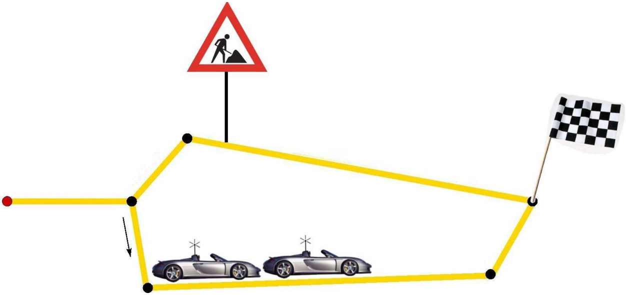


- suggested routes often not the quickest
- drivers will not accept route suggestions

⇝ benefits of route guidance strongly compromised

Results of Widespread Static Route Guidance

Travelers with the same origin and destination receive the same route suggestions:

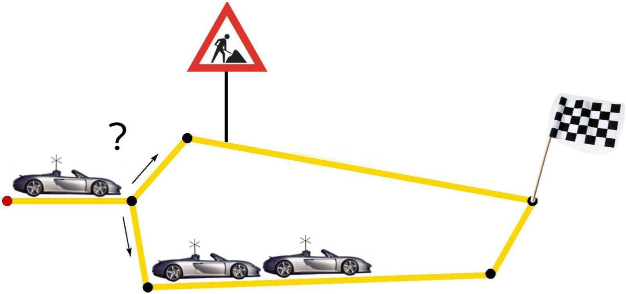


- suggested routes often not the quickest
- drivers will not accept route suggestions

⇝ benefits of route guidance strongly compromised

Results of Widespread Static Route Guidance

Travelers with the same origin and destination receive the same route suggestions:

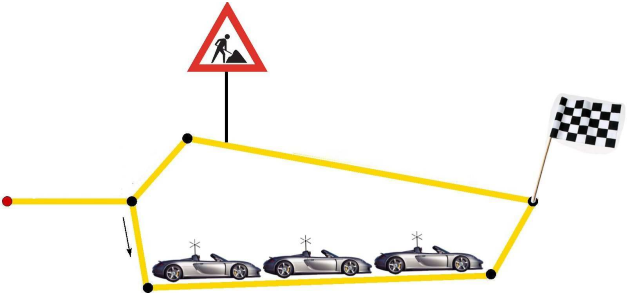


- suggested routes often not the quickest
- drivers will not accept route suggestions

⇝ benefits of route guidance strongly compromised

Results of Widespread Static Route Guidance

Travelers with the same origin and destination receive the same route suggestions:

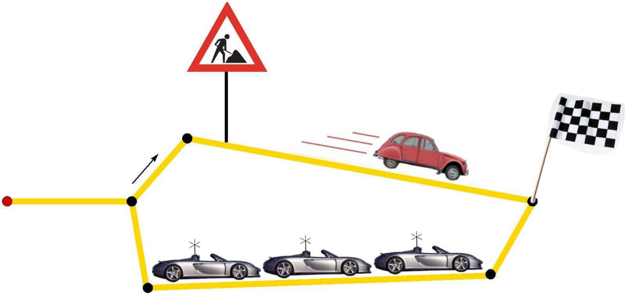


- suggested routes often not the quickest
- drivers will not accept route suggestions

⇝ benefits of route guidance strongly compromised

Results of Widespread Static Route Guidance

Travelers with the same origin and destination receive the same route suggestions:

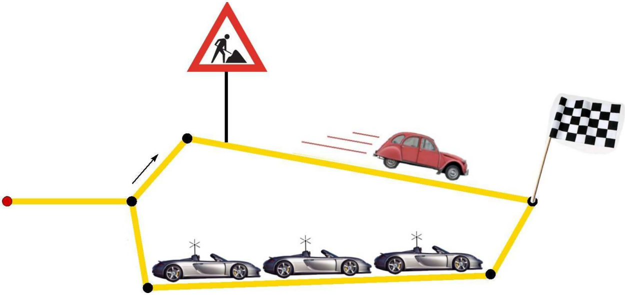


- suggested routes often not the quickest
- drivers will not accept route suggestions

⇝ benefits of route guidance strongly compromised

Results of Widespread Static Route Guidance

Travelers with the same origin and destination receive the same route suggestions:

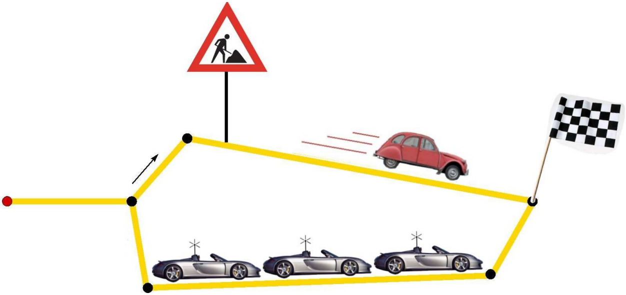


- suggested routes often not the quickest
- drivers will not accept route suggestions

⇝ benefits of route guidance strongly compromised

Results of Widespread Static Route Guidance

Travelers with the same origin and destination receive the same route suggestions:



- suggested routes often not the quickest
- drivers will not accept route suggestions

⇝ benefits of route guidance strongly compromised

The Need for Intelligent Traffic Routing

Fact

Intelligent Route Guidance Systems need to take into account the effects on travel times of their own route suggestions.

⇝ Some global optimization scheme is needed!

The Need for Intelligent Traffic Routing

Fact

Intelligent Route Guidance Systems need to take into account the effects on travel times of their own route suggestions.

$\leadsto$ Some global optimization scheme is needed!

Two Definitions of Optimality

System Optimum

Sum of all travel times is minimal.

Problems (e.g. [Mahmassani and Peeta 1993]):

- "unfair": drivers with same origin and destination may have vastly different travel times

⇝ drivers will not accept these route suggestions!

Two Definitions of Optimality

System Optimum

Sum of all travel times is minimal.

Problems (e.g. [Mahmassani and Peeta 1993]):

- “unfair”: drivers with same origin and destination may have vastly different travel times

⇝ drivers will not accept these route suggestions!

Two Definitions of Optimality

System Optimum

Sum of all travel times is minimal.

Problems (e.g. [Mahmassani and Peeta 1993]):

- “unfair”: drivers with same origin and destination may have vastly different travel times

$\leadsto$ drivers will not accept these route suggestions!

Two Definitions of Optimality

User Equilibrium

No user can improve his travel time by individually changing his route.

$\Rightarrow$ “natural” flow pattern of unguided traffic

Result:

- “fair”: drivers with same origin and destination have same travel times

Problems:

- sum of all travel times possibly a multiple of the one in system optimum (“price of anarchy”, e.g. [Roughgarden and Tardos 2002])
- no indication about network performance (Braess paradox)

Two Definitions of Optimality

User Equilibrium

No user can improve his travel time by individually changing his route.

⇒ “natural” flow pattern of unguided traffic

Result:

- “fair”: drivers with same origin and destination have same travel times

Problems:

- sum of all travel times possibly a multiple of the one in system optimum (“price of anarchy”, e.g. [Roughgarden and Tardos 2002])
- no indication about network performance (Braess paradox)

Two Definitions of Optimality

User Equilibrium

No user can improve his travel time by individually changing his route.

⇒ “natural” flow pattern of unguided traffic

Result:

- “fair”: drivers with same origin and destination have same travel times

Problems:

- sum of all travel times possibly a multiple of the one in system optimum (“price of anarchy”, e.g. [Roughgarden and Tardos 2002])
- no indication about network performance (Braess paradox)

Two Definitions of Optimality

User Equilibrium

No user can improve his travel time by individually changing his route.

⇒ "natural" flow pattern of unguided traffic

Result:

- "fair": drivers with same origin and destination have same travel times

Problems:

- sum of all travel times possibly a multiple of the one in system optimum ("price of anarchy", e.g. [Roughgarden and Tardos 2002])
- no indication about network performance (Braess paradox)

Two Definitions of Optimality

User Equilibrium

No user can improve his travel time by individually changing his route.

$\Rightarrow$ “natural” flow pattern of unguided traffic

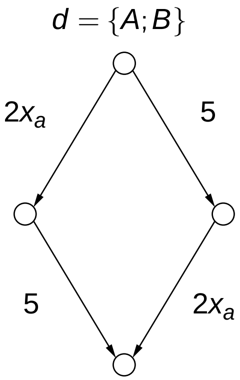
Result:

- “fair”: drivers with same origin and destination have same travel times

Problems:

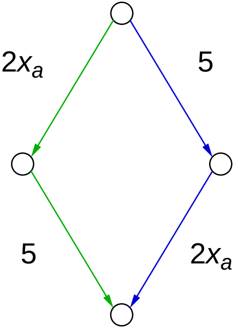
- sum of all travel times possibly a multiple of the one in system optimum (“price of anarchy”, e.g. [Roughgarden and Tardos 2002])
- no indication about network performance (Braess paradox)

The Braess Paradox



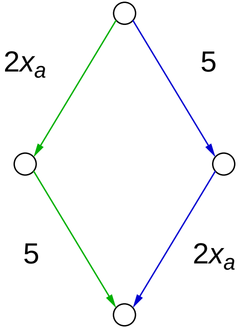
The Braess Paradox

$d = \{A; B\}$



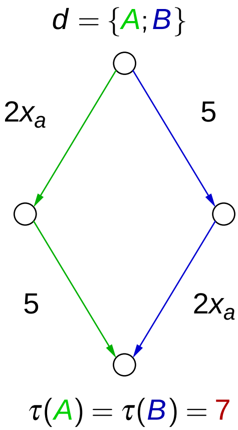
The Braess Paradox

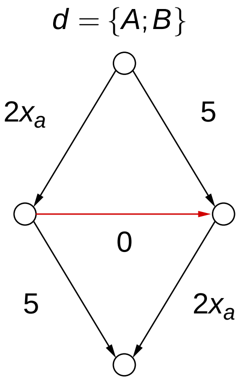
$d = \{A; B\}$



$\tau(A) = \tau(B) = 7$

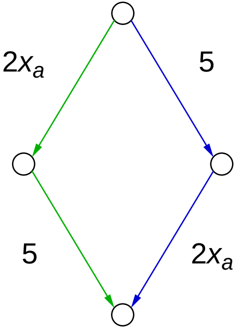
The Braess Paradox





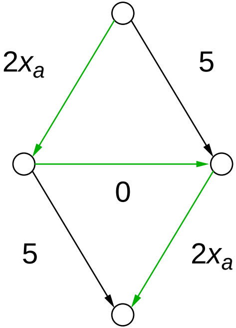
The Braess Paradox

$d = \{A; B\}$



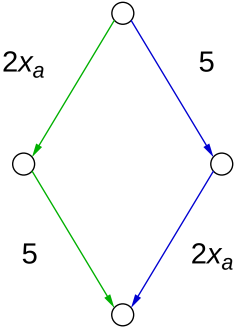
$\tau(A) = \tau(B) = 7$

$d = \{A; B\}$



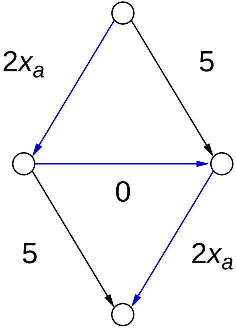
The Braess Paradox

$d = \{A; B\}$



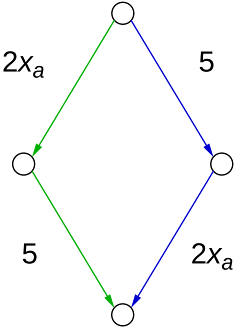
$\tau(A) = \tau(B) = 7$

$d = \{A; B\}$



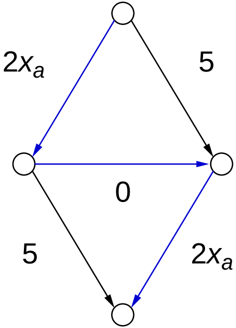
The Braess Paradox

$d = \{A; B\}$



$\tau(A) = \tau(B) = 7$

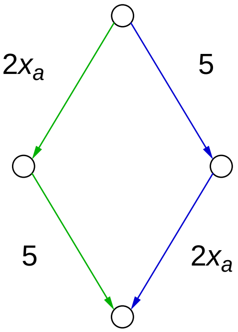
$d = \{A; B\}$



$\tau(A) = \tau(B) = 8$

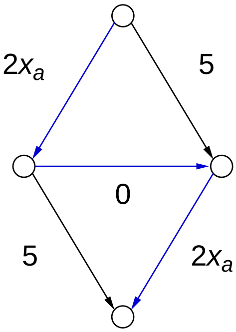
The Braess Paradox

$d = \{A; B\}$



$\tau(A) = \tau(B) = 7$

$d = \{A; B\}$



$\tau(A) = \tau(B) = 8 > 7$

Outline

System Optimum with Fairness Constraints

Idea [Jahn, Möhring, Schulz, Stier-Moses 2005]

Minimize sum of all travel times, but restrict usage of paths drivers would not accept:

- $\tau_p :=$ travel time on path p in UE
- $T_k :=$ travel time on paths chosen by commodity k in UE

$\Rightarrow$ only use paths p with

$$\tau_p \leq \varphi \cdot T_k$$

- suggestion: $\varphi = 1.02$

$\Rightarrow$ drivers are suggested paths which they think are fair!

System Optimum with Fairness Constraints

Idea [Jahn, Möhring, Schulz, Stier-Moses 2005]

Minimize sum of all travel times, but restrict usage of paths drivers would not accept:

- $\tau_p :=$ travel time on path p in UE
- $T_k :=$ travel time on paths chosen by commodity k in UE

⇒ only use paths p with

$$\tau_p \leq \varphi \cdot T_k$$

- suggestion: $\varphi = 1.02$

⇒ drivers are suggested paths which they think are fair!

System Optimum with Fairness Constraints

Idea [Jahn, Möhring, Schulz, Stier-Moses 2005]

Minimize sum of all travel times, but restrict usage of paths drivers would not accept:

- $\tau_p :=$ travel time on path p in UE
- $T_k :=$ travel time on paths chosen by commodity k in UE

$\Rightarrow$ only use paths p with

$$\tau_p \leq \varphi \cdot T_k$$

- suggestion: $\varphi = 1.02$

$\Rightarrow$ drivers are suggested paths which they think are fair!

System Optimum with Fairness Constraints

Idea [Jahn, Möhring, Schulz, Stier-Moses 2005]

Minimize sum of all travel times, but restrict usage of paths drivers would not accept:

- $\tau_p :=$ travel time on path p in UE
- $T_k :=$ travel time on paths chosen by commodity k in UE

$\Rightarrow$ only use paths p with

$$\tau_p \leq \varphi \cdot T_k$$

- suggestion: $\varphi = 1.02$

$\Rightarrow$ drivers are suggested paths which they think are fair!

Properties of the Constrained System Optimum

Results [Jahn, Möhring, Schulz, Stier-Moses 2005]

With appropriate φ, τ, solutions to CSO yield

- a lot more fairness than System Optimum
  - travel time of 99% of all users at most 30% higher than on fastest route.
  - in SO: 50%
- much better system performance than User Equilibrium
  - total travel time only $\frac{1}{3}$ as far away from SO as UE
- better routes for most drivers
  - 75% spend less travel time than in UE
  - only 0.4% spend 10% more (SO: 5%)

Properties of the Constrained System Optimum

Results [Jahn, Möhring, Schulz, Stier-Moses 2005]

With appropriate φ, τ, solutions to CSO yield

- a lot more fairness than System Optimum
  - travel time of 99% of all users at most 30% higher than on fastest route.
  - in SO: 50%
- much better system performance than User Equilibrium
  - total travel time only $\frac{1}{3}$ as far away from SO as UE
- better routes for most drivers
  - 75% spend less travel time than in UE
  - only 0.4% spend 10% more (SO: 5%)

Properties of the Constrained System Optimum

Results [Jahn, Möhring, Schulz, Stier-Moses 2005]

With appropriate φ, τ, solutions to CSO yield

- a lot more fairness than System Optimum
  - travel time of 99% of all users at most 30% higher than on fastest route.
  - in SO: 50%
- much better system performance than User Equilibrium
  - total travel time only $\frac{1}{3}$ as far away from SO as UE
- better routes for most drivers
  - 75% spend less travel time than in UE
  - only 0.4% spend 10% more (SO: 5%)

Properties of the Constrained System Optimum

Results [Jahn, Möhring, Schulz, Stier-Moses 2005]

With appropriate φ, τ, solutions to CSO yield

- a lot more fairness than System Optimum
  - travel time of 99% of all users at most 30% higher than on fastest route.
  - in SO: 50%
- much better system performance than User Equilibrium
  - total travel time only $\frac{1}{3}$ as far away from SO as UE
- better routes for most drivers
  - 75% spend less travel time than in UE
  - only 0.4% spend 10% more (SO: 5%)

The CSO Problem

min-cost multi-commodity flow problem with convex objective function and path constraints:

$$
\begin{aligned}
\textit{Minimize} \quad & \sum_{a \in A} l_a(x_a) x_a & \\
\textit{subject to} \quad & \sum_{k \in K} z_a^k = x_a & a \in A \\
& \sum_{p \in P_k : a \in p} x_p = z_a^k & a \in A \\
& \sum_{p \in P_k} x_p = d_k & k \in K \\
& \tau_p \leq \varphi T_k & p \in P_k : x_p > 0;\ k \in K \\
& x_p \geq 0 & p \in P
\end{aligned}
$$

The CSO Problem

min-cost multi-commodity flow problem with convex objective function and path constraints:

$$
\begin{aligned}
&\textit{Minimize} && \sum_{a\in A} l_a(x_a)x_a && \\
&\textit{subject to} && \sum_{k\in K} z_a^k = x_a && a \in A \\
& && \sum_{p\in P_k : a\in p} x_p = z_a^k && a \in A \\
& && \sum_{p\in P_k} x_p = d_k && k \in K \\
& && \tau_p \leq \varphi T_k && p \in P_k : x_p > 0;\ k \in K \\
& && x_p \geq 0 && p \in P
\end{aligned}
$$

The CSO Problem

min-cost multi-commodity flow problem with convex objective function and path constraints:

$$
\begin{array}{lll}
\textit{Minimize} & \sum_{a \in A} l_a(x_a) x_a & \\
\textit{subject to} & \sum_{k \in K} z_a^k = x_a & a \in A \\
& \sum_{p \in P_k : a \in p} x_p = z_a^k & a \in A \\
& \sum_{p \in P_k} x_p = d_k & k \in K \\
& \tau_p \leq \varphi T_k & p \in P_k : x_p > 0;\ k \in K \\
& x_p \geq 0 & p \in P
\end{array}
$$

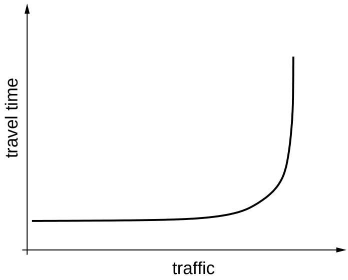
The CSO Problem

min-cost multi-commodity flow problem with convex objective function and path constraints:

$$
\begin{array}{lll}
\textit{Minimize} & \sum_{a \in A} l_a(x_a)x_a & \\
\textit{subject to} & \sum_{k \in K} z_a^k = x_a & a \in A \\
& \sum_{p \in P_k : a \in p} x_p = z_a^k & a \in A \\
& \sum_{p \in P_k} x_p = d_k & k \in K \\
& \tau_p \leq \varphi T_k & p \in P_k : x_p > 0;\ k \in K \\
& x_p \geq 0 & p \in P
\end{array}
$$

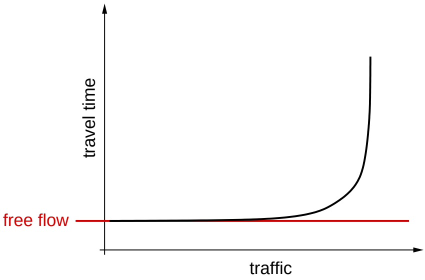
Mathematical Challenges

- CSO is non-linear: travel times vary with flow rate



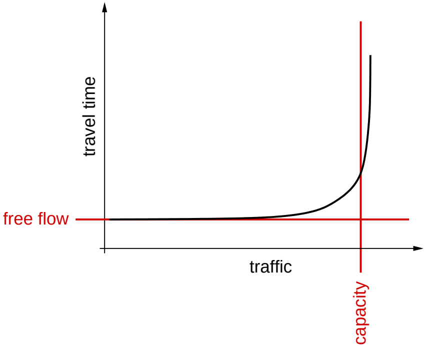
Mathematical Challenges

- CSO is non-linear: travel times vary with flow rate



Mathematical Challenges

- CSO is non-linear: travel times vary with flow rate



Mathematical Challenges

- CSO is non-linear: travel times vary with flow rate
- exponentially many paths in G
  - $\Rightarrow$ cannot deal with variables x_p explicitly

Previous work [Jahn, Möhring, Schulz, Stier-Moses 2004]:

- solve CSO by variant of Frank-Wolfe convex combinations algorithm and constrained shortest path calculations

$\Rightarrow$ runtime acceptable: instances with a few thousand nodes / arcs / commodities take some minutes

- improvement needed for practical use

Mathematical Challenges

- CSO is non-linear: travel times vary with flow rate
- exponentially many paths in G
  ⇒ cannot deal with variables x_p explicitly

Previous work [Jahn, Möhring, Schulz, Stier-Moses 2004]:

- solve CSO by variant of Frank-Wolfe convex combinations algorithm and constrained shortest path calculations

⇒ runtime acceptable: instances with a few thousand nodes / arcs / commodities take some minutes

- improvement needed for practical use

Mathematical Challenges

- CSO is non-linear: travel times vary with flow rate
- exponentially many paths in G
  - ⇒ cannot deal with variables x_p explicitly

Previous work [Jahn, Möhring, Schulz, Stier-Moses 2004]:

- solve CSO by variant of Frank-Wolfe convex combinations algorithm and constrained shortest path calculations

⇒ runtime acceptable: instances with a few thousand nodes / arcs / commodities take some minutes

- improvement needed for practical use

A Different Approach

Idea

- define appropriate Lagrangian relaxation
- use cutting plane method to solve dual problem

- similar approach successfully applied to other multi-commodity flow problems [Babonneau and Vial 2005]

A Different Approach

Idea

- define appropriate Lagrangian relaxation
- use cutting plane method to solve dual problem

- similar approach successfully applied to other multi-commodity flow problems [Babonneau and Vial 2005]

A Different Approach

Idea

- define appropriate Lagrangian relaxation
- use cutting plane method to solve dual problem

- similar approach successfully applied to other multi-commodity flow problems [Babonneau and Vial 2005]

Outline

Lagrangian Relaxation for CSO

$$
\begin{aligned}
&\textit{Minimize} && L(x,u) := \sum_{a\in A} l_a(x_a)x_a && \\
&\textit{subject to} && \sum_{p\in P_k : a\in p} x_p = z_a^k && a \in A \\
& && \sum_{p\in P_k} x_p = d_k && k \in K \\
& && \tau_p \le \varphi T_k && p \in P^k : x_p > 0;\ k \in K \\
& && x_p \ge 0 && p \in P \\
& && \sum_{k\in K} z_a^k = x_a && a \in A
\end{aligned}
$$

Lagrangian Relaxation for CSO

- drop constraints coupling total and commodity flows

$$
\begin{array}{lll}
\textit{Minimize} & L(x,u) := \sum_{a \in A} l_a(x_a)x_a & \\
\textit{subject to} & \sum_{p \in P_k : a \in p} x_p = z_a^k & a \in A \\
& \sum_{p \in P_k} x_p = d_k & k \in K \\
& \tau_p \leq \varphi T_k & p \in P^k : x_p > 0;\ k \in K \\
& x_p \geq 0 & p \in P \\
& \sum_{k \in K} z_a^k = x_a & a \in A
\end{array}
$$

Lagrangian Relaxation for CSO

- drop constraints coupling total and commodity flows

$$
\begin{array}{lll}
\textit{Minimize} & L(x,u) := \sum_{a \in A} l_a(x_a) x_a & \\
\textit{subject to} & \sum_{p \in P_k : a \in p} x_p = z_a^k & a \in A \\
& \sum_{p \in P_k} x_p = d_k & k \in K \\
& \tau_p \leq \varphi T_k & p \in P^k : x_p > 0;\ k \in K \\
& x_p \geq 0 & p \in P
\end{array}
$$

Lagrangian Relaxation for CSO

- add penalty terms with multipliers u_j to objective

$$
\begin{array}{llr}
\textit{Minimize} & L(x,u) := \sum_{a\in A} l_a(x_a)x_a & \\
\textit{subject to} & \sum_{p\in P_k : a\in p} x_p = z_a^k & a \in A \\
& \sum_{p\in P_k} x_p = d_k & k \in K \\
& \tau_p \leq \varphi T_k & p \in P^k : x_p > 0;\ k \in K \\
& x_p \geq 0 & p \in P
\end{array}
$$

Lagrangian Relaxation for CSO

- add penalty terms with multipliers u_j to objective

$$\text{Minimize} \quad L(x,u) := \sum_{a\in A} l_a(x_a)x_a + \sum_{a\in A}\left(u_a \cdot \left(\sum_{k\in K} z_a^k - x_a\right)\right)$$

$$\text{subject to} \quad \sum_{p\in P_k : a\in p} x_p = z_a^k \qquad a\in A$$

$$\sum_{p\in P_k} x_p = d_k \qquad k\in K$$

$$\tau_p \le \varphi T_k \qquad p\in P^k : x_p > 0;\ k\in K$$

$$x_p \ge 0 \qquad p\in P$$

Lagrangian Relaxation for CSO

- remaining constraints resemble those of $|K|$ constrained shortest path problems in z_a^k

$$\text{Minimize} \quad L(x,u) := \sum_{a\in A} l_a(x_a)x_a + \sum_{a\in A}\left(u_a \cdot \left(\sum_{k\in K} z_a^k - x_a\right)\right)$$

$$\text{subject to} \quad \sum_{p\in P_k : a\in p} x_p = z_a^k \qquad a \in A$$

$$\sum_{p\in P_k} x_p = d_k \qquad k \in K$$

$$\tau_p \le \varphi T_k \qquad p \in P^k : x_p > 0;\ k \in K$$

$$x_p \ge 0 \qquad p \in P$$

Lagrangian Relaxation for CSO

- Lagrangian separable in x and z?

$$
\begin{aligned}
&\textit{Minimize} && L(x,u) := \sum_{a\in A} l_a(x_a)x_a + \sum_{a\in A}\left(u_a\cdot\left(\sum_{k\in K} z_a^k - x_a\right)\right) && \\
&\textit{subject to} && \sum_{p\in P_k : a\in p} x_p = z_a^k && a\in A\\
& && \sum_{p\in P_k} x_p = d_k && k\in K\\
& && \tau_p \le \varphi T_k && p\in P^k : x_p > 0;\ k\in K\\
& && x_p \ge 0 && p\in P
\end{aligned}
$$

Lagrangian Relaxation for CSO

- Lagrangian separable in x and z?

$$\begin{aligned}
&\textit{Minimize} && L(x,u) := \overbrace{\sum_{a\in A}(l_a(x_a)-u_a)\cdot x_a}^{L_1(x,u)} + \overbrace{\sum_{k\in K}\sum_{a\in A} u_a\cdot z_a^k}^{L_2(z,u)} \\
&\textit{subject to} && \sum_{p\in P_k: a\in p} x_p = z_a^k && a\in A \\
& && \sum_{p\in P_k} x_p = d_k && k\in K \\
& && \tau_p \le \varphi T_k && p\in P^k: x_p>0;\ k\in K \\
& && x_p \ge 0 && p\in P
\end{aligned}$$

Lagrangian Relaxation for CSO

⇝ Yes!

$$\textit{Minimize} \quad L(x,u) := \overbrace{\sum_{a\in A} (l_a(x_a) - u_a)\cdot x_a}^{L_1(x,u)} + \overbrace{\sum_{k\in K}\sum_{a\in A} u_a \cdot z_a^k}^{L_2(z,u)}$$

$$\textit{subject to} \quad \sum_{p\in P_k : a\in p} x_p = z_a^k \qquad a \in A$$

$$\sum_{p\in P_k} x_p = d_k \qquad k \in K$$

$$\tau_p \leq \varphi T_k \qquad p \in P^k : x_p > 0;\ k \in K$$

$$x_p \geq 0 \qquad p \in P$$

Lagrangian Relaxation for CSO

- easier problem: analytical minimization in x...

$$\text{Minimize} \quad L(x,u) := \overbrace{\sum_{a\in A} (l_a(x_a) - u_a)\cdot x_a}^{L_1(x,u)} + \overbrace{\sum_{k\in K}\sum_{a\in A} u_a \cdot z_a^k}^{L_2(z,u)}$$

$$\begin{aligned}
\text{subject to} \quad & \sum_{p\in P_k : a\in p} x_p = z_a^k & a \in A \\
& \sum_{p\in P_k} x_p = d_k & k \in K \\
& \tau_p \leq \varphi T_k & p \in P^k : x_p > 0;\ k \in K \\
& x_p \geq 0 & p \in P
\end{aligned}$$

Lagrangian Relaxation for CSO

- ...and $|K|$ constrained shortest path problems in z^k

$$\text{Minimize} \quad L(x,u) := \overbrace{\sum_{a \in A} (l_a(x_a) - u_a) \cdot x_a}^{L_1(x,u)} + \overbrace{\sum_{k \in K} \sum_{a \in A} u_a \cdot z_a^k}^{L_2(z,u)}$$

$$\begin{aligned}
\text{subject to} \quad & \sum_{p \in P_k : a \in p} x_p = z_a^k && a \in A \\
& \sum_{p \in P_k} x_p = d_k && k \in K \\
& \tau_p \leq \varphi T_k && p \in P^k : x_p > 0;\ k \in K \\
& x_p \geq 0 && p \in P
\end{aligned}$$

Lagrangian Relaxation for CSO

- up next: dual problem (maximize this minimum over u)

$$\text{Minimize} \quad L(x,u) := \overbrace{\sum_{a \in A} (l_a(x_a) - u_a) \cdot x_a}^{L_1(x,u)} + \overbrace{\sum_{k \in K} \sum_{a \in A} u_a \cdot z_a^k}^{L_2(z,u)}$$

$$\text{subject to} \quad \sum_{p \in P_k : a \in p} x_p = z_a^k \qquad a \in A$$

$$\sum_{p \in P_k} x_p = d_k \qquad k \in K$$

$$\tau_p \leq \varphi T_k \qquad p \in P^k : x_p > 0;\ k \in K$$

$$x_p \geq 0 \qquad p \in P$$

Outline

Analytic Center Cutting Plane Method

- approximation scheme for maximization of a concave function over a convex set
- implementation by Babonneau, Vial et. al. at LogiLab, University of Geneva

Two components:

- query point generator
  - manages a localization set containing all optimal points
  - selects query points which are tried for optimality
- oracle
  - generates cutting planes to further bound the localization set
  - problem dependent!

Analytic Center Cutting Plane Method

- approximation scheme for maximization of a concave function over a convex set
- implementation by Babonneau, Vial et. al. at LogiLab, University of Geneva

Two components:

- query point generator
  - manages a localization set containing all optimal points
  - selects query points which are tried for optimality
- oracle
  - generates cutting planes to further bound the localization set
  - problem dependent!

Analytic Center Cutting Plane Method

- approximation scheme for maximization of a concave function over a convex set
- implementation by Babonneau, Vial et. al. at LogiLab, University of Geneva

Two components:

- query point generator
  - manages a localization set containing all optimal points
  - selects query points which are tried for optimality
- oracle
  - generates cutting planes to further bound the localization set
  - problem dependent!

Analytic Center Cutting Plane Method

- approximation scheme for maximization of a concave function over a convex set
- implementation by Babonneau, Vial et. al. at LogiLab, University of Geneva

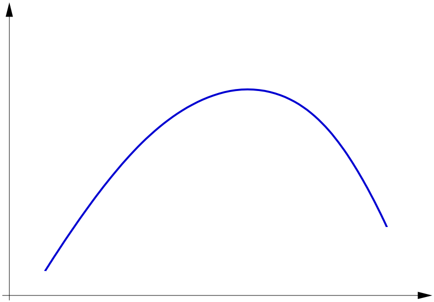
Two components:

- query point generator
  - manages a localization set containing all optimal points
  - selects query points which are tried for optimality
- oracle
  - generates cutting planes to further bound the localization set
  - problem dependent!

Oracle for CSO

- evaluate objective function ⇝ CSP calculations
- calculate subgradient at query point ⇝ easy

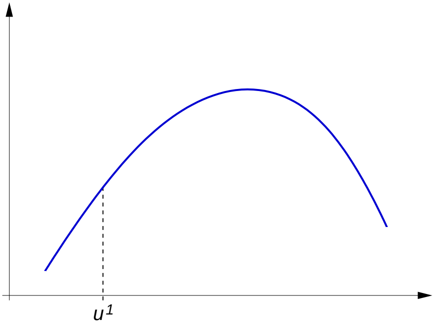
⇒ subgradients and best objective value define cutting planes bounding the localization set



Oracle for CSO

- evaluate objective function ⇝ CSP calculations
- calculate subgradient at query point ⇝ easy

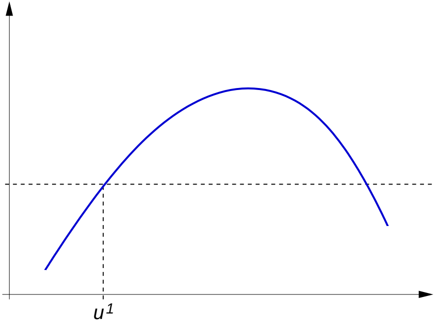
⇒ subgradients and best objective value define cutting planes bounding the localization set



Oracle for CSO

- evaluate objective function ⇝ CSP calculations
- calculate subgradient at query point ⇝ easy

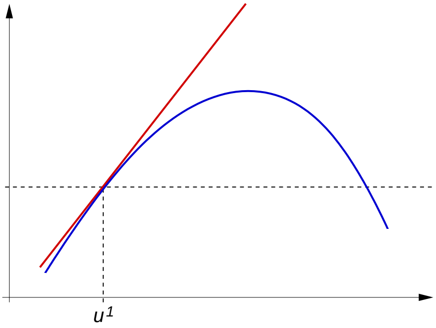
⇒ subgradients and best objective value define cutting planes bounding the localization set



Oracle for CSO

- evaluate objective function ⇝ CSP calculations
- calculate subgradient at query point ⇝ easy

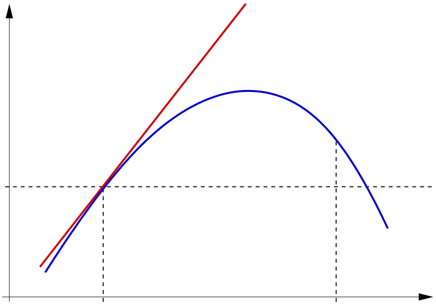
⇒ subgradients and best objective value define cutting planes bounding the localization set



Oracle for CSO

- evaluate objective function ⇝ CSP calculations
- calculate subgradient at query point ⇝ easy

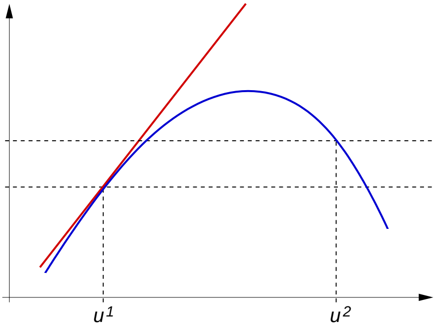
⇒ subgradients and best objective value define cutting planes bounding the localization set



Oracle for CSO

- evaluate objective function ⇝ CSP calculations
- calculate subgradient at query point ⇝ easy

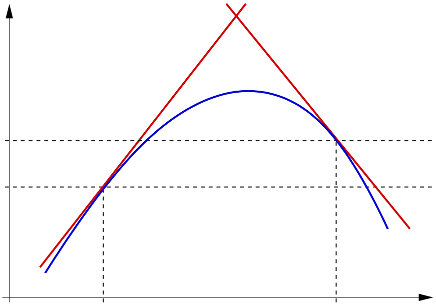
⇒ subgradients and best objective value define cutting planes bounding the localization set



Oracle for CSO

- evaluate objective function ⇝ CSP calculations
- calculate subgradient at query point ⇝ easy

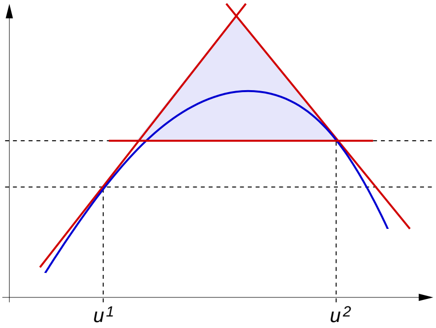
⇒ subgradients and best objective value define cutting planes bounding the localization set



Oracle for CSO

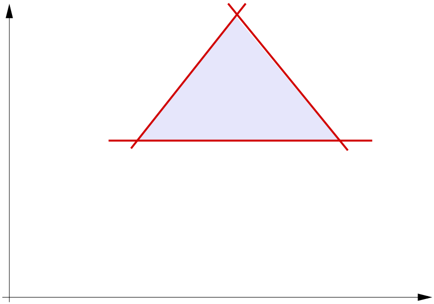
- evaluate objective function ⇝ CSP calculations
- calculate subgradient at query point ⇝ easy

⇒ subgradients and best objective value define cutting planes bounding the localization set



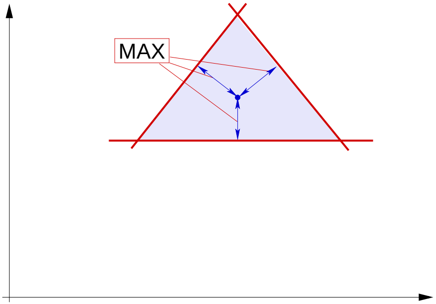
Query Points

- analytic center: maximum distances from cutting planes
  - calculation by damped Newton method
- *u*-component is next query point



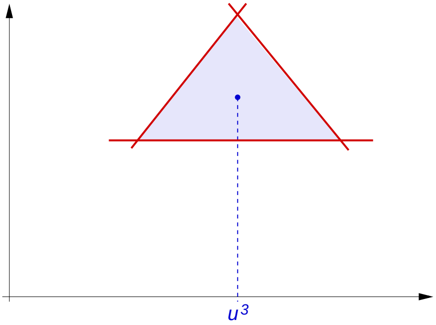
Query Points

- analytic center: maximum distances from cutting planes
  - calculation by damped Newton method
- u-component is next query point



Query Points

- analytic center: maximum distances from cutting planes
  - calculation by damped Newton method
- u-component is next query point



Query Points

- analytic center: maximum distances from cutting planes
  - calculation by damped Newton method
- u-component is next query point

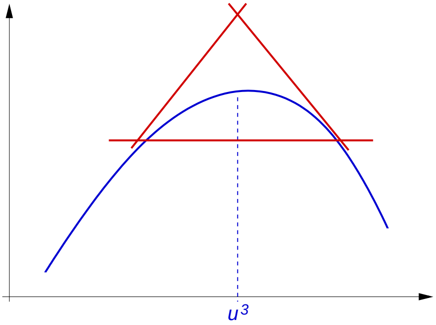
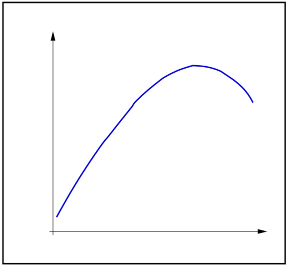
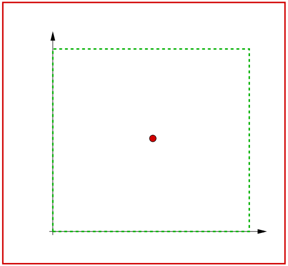


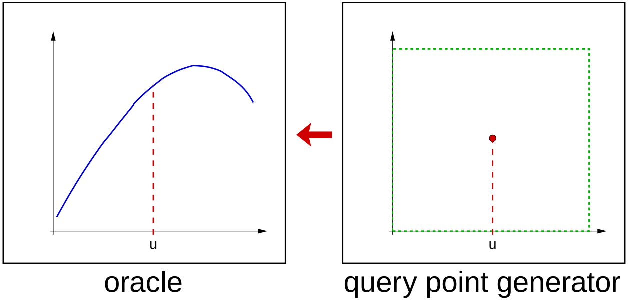
Illustration of an ACCPM Run


oracle


query point generator

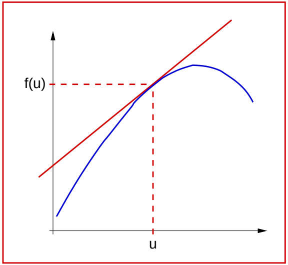
- localization set artificially bounded $\Rightarrow$ compact

Illustration of an ACCPM Run

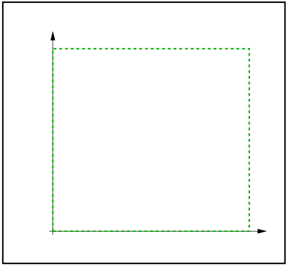


In each iteration, a query point is sent to the oracle,...

Illustration of an ACCPM Run



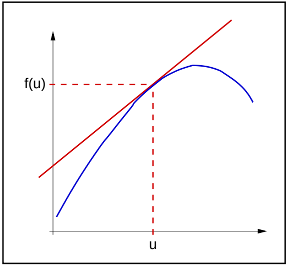
oracle



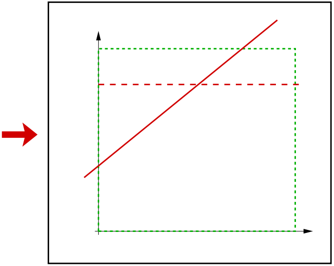
query point generator

... the value and subgradient of θ are calculated...

Illustration of an ACCPM Run



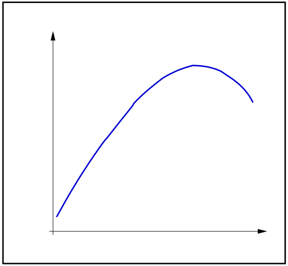
oracle



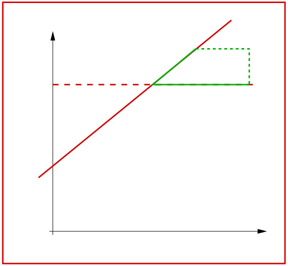
query point generator

... which define cutting planes...

Illustration of an ACCPM Run



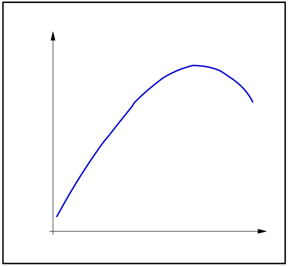
oracle



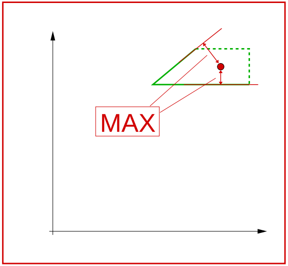
query point generator

... to further bound the localization set.

Illustration of an ACCPM Run



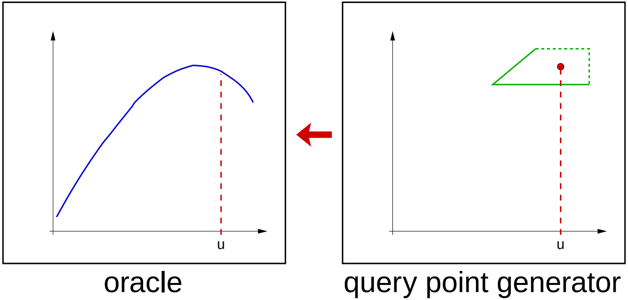
oracle



query point generator

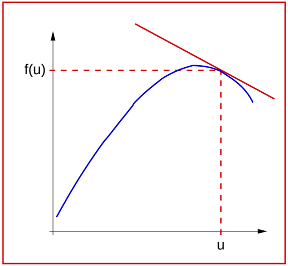
Then, the proximal analytic center is calculated...

Illustration of an ACCPM Run

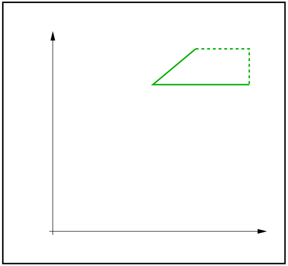


... which defines the next query point.

Illustration of an ACCPM Run



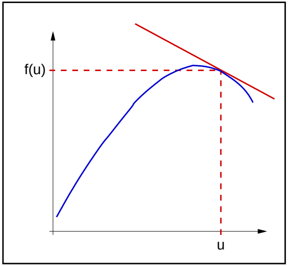
oracle



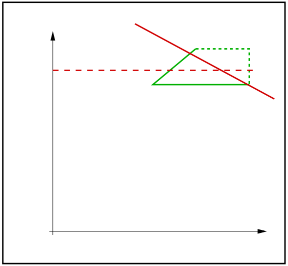
query point generator

Process is repeated...

Illustration of an ACCPM Run



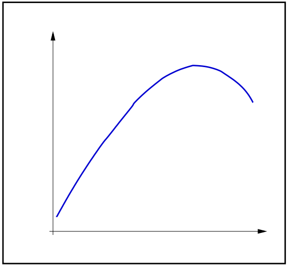
oracle



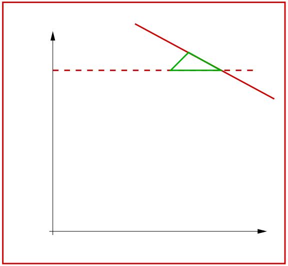
query point generator

Process is repeated...

Illustration of an ACCPM Run



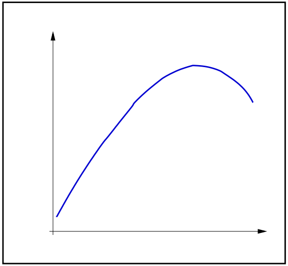
oracle



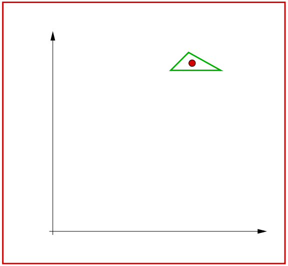
query point generator

Process is repeated...

Illustration of an ACCPM Run



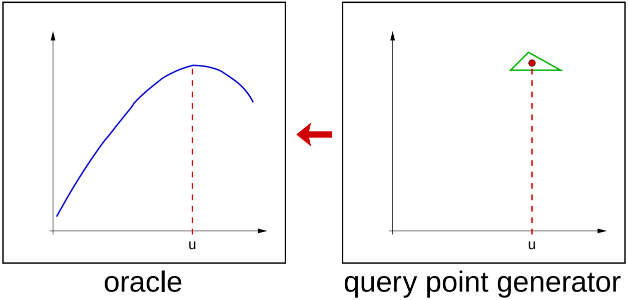
oracle



query point generator

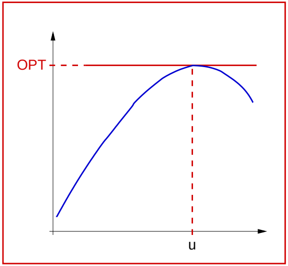
Process is repeated...

Illustration of an ACCPM Run

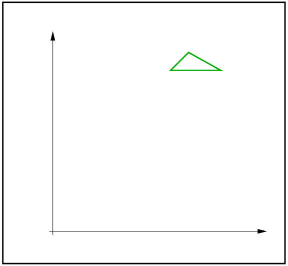


Process is repeated...

Illustration of an ACCPM Run



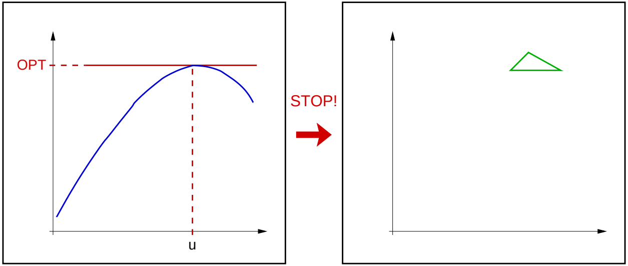
oracle



query point generator

... until desired precision is achieved.

Illustration of an ACCPM Run



oracle

query point generator

... until desired precision is achieved.

There is much more inside ACCPM

Accelerating convergence:

- sophisticated parameters for dynamic weighting of cuts
- cut elimination techniques
- rules for updating the proximal reference point

Problem specific functionalities:

- multiple cuts per iteration
- active set strategies

...

There is much more inside ACCPM

Accelerating convergence:

- sophisticated parameters for dynamic weighting of cuts
- cut elimination techniques
- rules for updating the proximal reference point

Problem specific functionalities:

- multiple cuts per iteration
- active set strategies

...

There is much more inside ACCPM

Accelerating convergence:

- sophisticated parameters for dynamic weighting of cuts
- cut elimination techniques
- rules for updating the proximal reference point

Problem specific functionalities:

- multiple cuts per iteration
- active set strategies

...

There is much more inside ACCPM

Accelerating convergence:

- sophisticated parameters for dynamic weighting of cuts
- cut elimination techniques
- rules for updating the proximal reference point

Problem specific functionalities:

- multiple cuts per iteration
- active set strategies

...

There is much more inside ACCPM

Accelerating convergence:

- sophisticated parameters for dynamic weighting of cuts
- cut elimination techniques
- rules for updating the proximal reference point

Problem specific functionalities:

- multiple cuts per iteration
- active set strategies

...

Outline

The Algorithm

Algorithm

- define Lagrangian Relaxation of CSO
- use proximal ACCPM to solve Lagrangian dual problem
  - oracle: solve CSP problems
    $\Rightarrow$ primal lower bound
  - query point generator: damped Newton method
- primal solution / upper bound through heuristic: convex combination of paths from oracle
- stop when desired precision guaranteed

- different variants of labeling algorithm for CSP: basic, bidirectional, goal-oriented

The Algorithm

Algorithm

- define Lagrangian Relaxation of CSO
- use proximal ACCPM to solve Lagrangian dual problem
  - oracle: solve CSP problems
    $\Rightarrow$ primal lower bound
  - query point generator: damped Newton method
- primal solution / upper bound through heuristic: convex combination of paths from oracle
- stop when desired precision guaranteed

- different variants of labeling algorithm for CSP: basic, bidirectional, goal-oriented

The Algorithm

Algorithm

- define Lagrangian Relaxation of CSO
- use proximal ACCPM to solve Lagrangian dual problem
  - oracle: solve CSP problems
    $\Rightarrow$ primal lower bound
  - query point generator: damped Newton method
- primal solution / upper bound through heuristic: convex combination of paths from oracle
- stop when desired precision guaranteed

- different variants of labeling algorithm for CSP: basic, bidirectional, goal-oriented

The Algorithm

Algorithm

- define Lagrangian Relaxation of CSO
- use proximal ACCPM to solve Lagrangian dual problem
  - oracle: solve CSP problems
    $\Rightarrow$ primal lower bound
  - query point generator: damped Newton method
- primal solution / upper bound through heuristic: convex combination of paths from oracle
- stop when desired precision guaranteed

- different variants of labeling algorithm for CSP: basic, bidirectional, goal-oriented

The Algorithm

Algorithm

- define Lagrangian Relaxation of CSO
- use proximal ACCPM to solve Lagrangian dual problem
  - oracle: solve CSP problems
    $\Rightarrow$ primal lower bound
  - query point generator: damped Newton method
- primal solution / upper bound through heuristic: convex combination of paths from oracle
- stop when desired precision guaranteed

- different variants of labeling algorithm for CSP: basic, bidirectional, goal-oriented

The Algorithm

Algorithm

- define Lagrangian Relaxation of CSO
- use proximal ACCPM to solve Lagrangian dual problem
  - oracle: solve CSP problems
    $\Rightarrow$ primal lower bound
  - query point generator: damped Newton method
- primal solution / upper bound through heuristic: convex combination of paths from oracle
- stop when desired precision guaranteed

- different variants of labeling algorithm for CSP: basic, bidirectional, goal-oriented

The Algorithm

Algorithm

- define Lagrangian Relaxation of CSO
- use proximal ACCPM to solve Lagrangian dual problem
  - oracle: solve CSP problems
    $\Rightarrow$ primal lower bound
  - query point generator: damped Newton method
- primal solution / upper bound through heuristic: convex combination of paths from oracle
- stop when desired precision guaranteed

- different variants of labeling algorithm for CSP: basic, bidirectional, goal-oriented

Test Instances

Name	$\|V\|$	$\|A\|$	$\|K\|$
`Sioux Falls`	24	76	528
`Winnipeg`	1052	2836	4344
`Neukoelln`	1890	4040	3166
`Chicago Sketch`	933	2950	83113

- all but `Neukoelln` from Transportation Network Problems online database
- tested on Intel Pentium 4 2.8GHz with 1 GB RAM, SuSE Linux
- optimality gap: 0.5%

Test Instances

Name	$\|V\|$	$\|A\|$	$\|K\|$
`Sioux Falls`	24	76	528
`Winnipeg`	1052	2836	4344
`Neukoelln`	1890	4040	3166
`Chicago Sketch`	933	2950	83113

- all but `Neukoelln` from Transportation Network Problems online database
- tested on Intel Pentium 4 2.8GHz with 1 GB RAM, SuSE Linux
- optimality gap: 0.5%

Test Instances

Name	$\|V\|$	$\|A\|$	$\|K\|$
`Sioux Falls`	24	76	528
`Winnipeg`	1052	2836	4344
`Neukoelln`	1890	4040	3166
`Chicago Sketch`	933	2950	83113

- all but `Neukoelln` from Transportation Network Problems online database
- tested on Intel Pentium 4 2.8GHz with 1 GB RAM, SuSE Linux
- optimality gap: 0.5%

General Findings

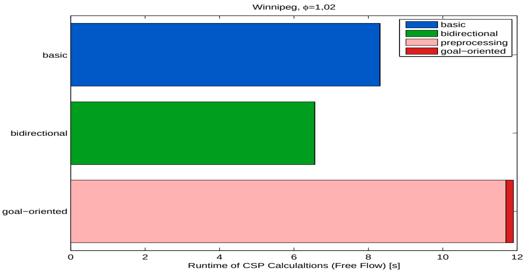
- runtime of query point generator negligible
  - almost all calculation time spent finding CSPs
- goal-directed approach slowest for free flow travel times
  - time for preprocessing longer than resulting speed-up

General Findings

- runtime of query point generator negligible
  - almost all calculation time spent finding CSPs
- goal-directed approach slowest for free flow travel times
  - time for preprocessing longer than resulting speed-up

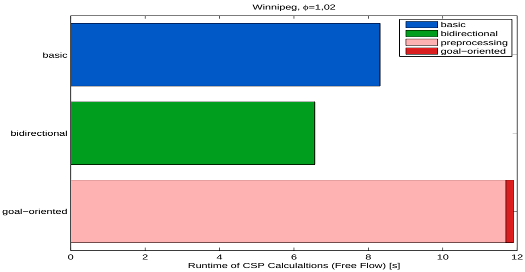
General Findings

- runtime of query point generator negligible
  - almost all calculation time spent finding CSPs
- goal-directed approach slowest for free flow travel times
  - time for preprocessing longer than resulting speed-up



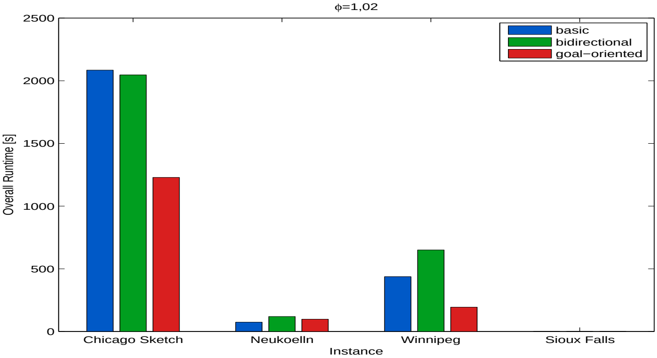
General Findings

- runtime of query point generator negligible
  - almost all calculation time spent finding CSPs
- goal-directed approach slowest for free flow travel times
  - time for preprocessing longer than resulting speed-up



Overall Runtime

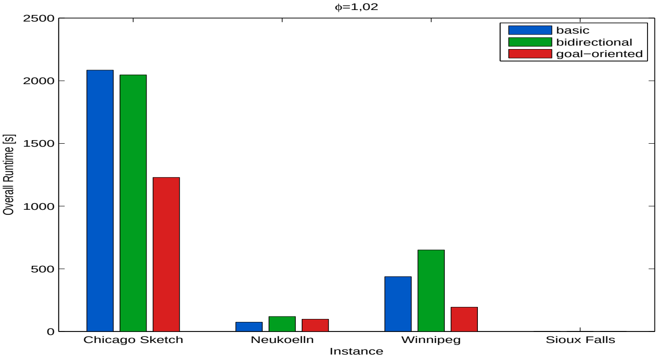
- total runtime: basic / bidirectional / goal-oriented



- why does goal-oriented algorithm perform best?

Overall Runtime

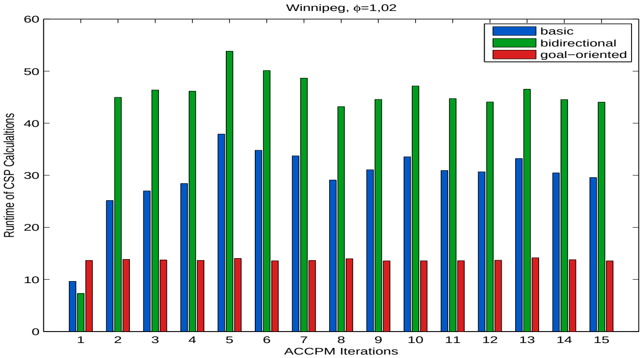
- total runtime: basic / bidirectional / goal-oriented



- why does goal-oriented algorithm perform best?

Effect of CSP Acceleration

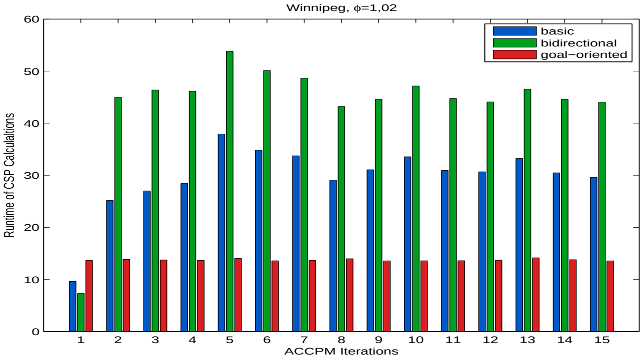
- CSP-runtime over ACCPM iterations for Winnipeg



- runtimes of basic and bidirectional algorithms increase!

Effect of CSP Acceleration

- CSP-runtime over ACCPM iterations for Winnipeg



- runtimes of basic and bidirectional algorithms increase!

Explanation

- edge length for CSP calculations: dual variables *u*
- as we approach optimum, *u* approaches CSO travel times

⇒ in congested networks, direct paths become unattractive

⇒ basic labeling algorithm is deflected from target

  ⇝ infeasible paths are explored first

- goal orientation dominates this effect

Explanation

- edge length for CSP calculations: dual variables u
- as we approach optimum, u approaches CSO travel times

$\Rightarrow$ in congested networks, direct paths become unattractive

$\Rightarrow$ basic labeling algorithm is deflected from target

  $\rightsquigarrow$ infeasible paths are explored first

- goal orientation dominates this effect

Explanation

- edge length for CSP calculations: dual variables u
- as we approach optimum, u approaches CSO travel times

$\Rightarrow$ in congested networks, direct paths become unattractive

$\Rightarrow$ basic labeling algorithm is deflected from target

  $\rightsquigarrow$ infeasible paths are explored first

- goal orientation dominates this effect

Explanation

- edge length for CSP calculations: dual variables *u*
- as we approach optimum, *u* approaches CSO travel times

⇒ in congested networks, direct paths become unattractive

⇒ basic labeling algorithm is deflected from target

  ⇝ infeasible paths are explored first

- goal orientation dominates this effect

Explanation

- edge length for CSP calculations: dual variables u
- as we approach optimum, u approaches CSO travel times

$\Rightarrow$ in congested networks, direct paths become unattractive

$\Rightarrow$ basic labeling algorithm is deflected from target

  $\rightsquigarrow$ infeasible paths are explored first

- goal orientation dominates this effect

Explanation

- edge length for CSP calculations: dual variables u
- as we approach optimum, u approaches CSO travel times

$\Rightarrow$ in congested networks, direct paths become unattractive

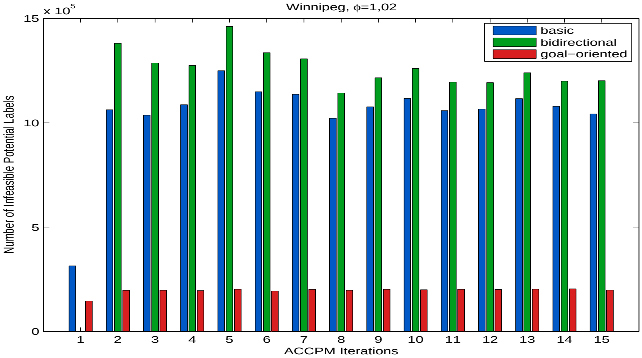
$\Rightarrow$ basic labeling algorithm is deflected from target

  - $\rightsquigarrow$ infeasible paths are explored first

- goal orientation dominates this effect

Exploration of Nodes on Infeasible Paths

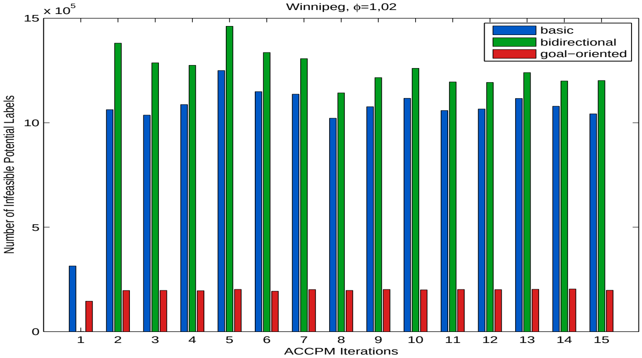
- potential labels violating length bounds for Winnipeg



- number of these labels proportional to runtime

Exploration of Nodes on Infeasible Paths

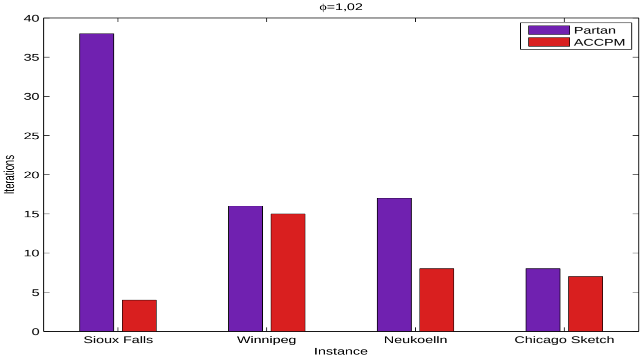
- potential labels violating length bounds for Winnipeg



- number of these labels proportional to runtime

Comparison with Partan

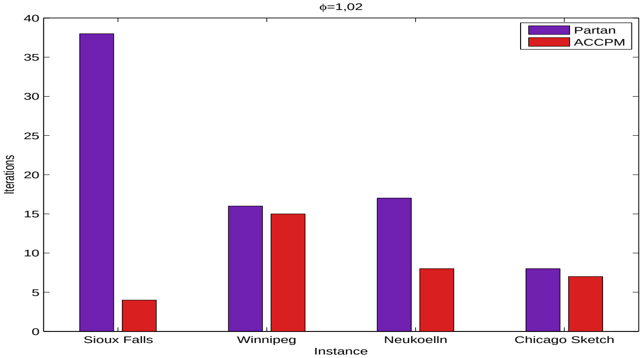
- number of iterations compared with Partan



- ACCPM needs less iterations

Comparison with Partan

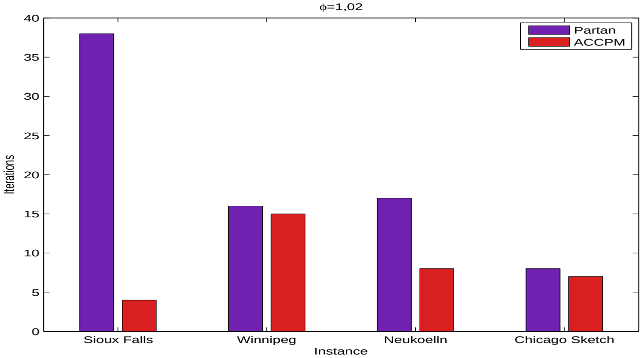
- number of iterations compared with Partan



- ACCPM needs less iterations

Comparison with Partan

- number of iterations compared with Partan



⇒ ACCPM needs less runtime

Outline

Summary

- constrained system optimum delivers equally good and fair solutions for traffic optimization
- proximal-ACCPM can be used to solve a Lagrangian relaxation of CSO
- algorithm outperforms previous approaches:
  $\approx \frac{1}{2}$ the runtime of Partan algorithm
- interesting relationship

  dual variables
  $\leftrightarrow$ level of congestion
  $\leftrightarrow$ runtime of different CSP algorithms

Summary

- constrained system optimum delivers equally good and fair solutions for traffic optimization
- proximal-ACCPM can be used to solve a Lagrangian relaxation of CSO
- algorithm outperforms previous approaches:
  $\approx \frac{1}{2}$ the runtime of Partan algorithm
- interesting relationship

  dual variables
  $\leftrightarrow$ level of congestion
  $\leftrightarrow$ runtime of different CSP algorithms

Summary

- constrained system optimum delivers equally good and fair solutions for traffic optimization
- proximal-ACCPM can be used to solve a Lagrangian relaxation of CSO
- algorithm outperforms previous approaches:
  $\approx \frac{1}{2}$ the runtime of Partan algorithm
- interesting relationship

  dual variables
  $\leftrightarrow$ level of congestion
  $\leftrightarrow$ runtime of different CSP algorithms

Summary

- constrained system optimum delivers equally good and fair solutions for traffic optimization
- proximal-ACCPM can be used to solve a Lagrangian relaxation of CSO
- algorithm outperforms previous approaches:
  $\approx \frac{1}{2}$ the runtime of Partan algorithm
- interesting relationship

dual variables
$\leftrightarrow$ level of congestion
$\leftrightarrow$ runtime of different CSP algorithms