Operation Research and Transport
Braess's Paradox

V. Leclère (ENPC)

April 8th, 2020

What will this course be about ?

- Understanding how people choose their way through a transportation network.
- having an idea on how to compute efficiently :
    - the shortest path on a network
    - the equilibrium on a network
- A practical work to compute this equilibrium on a computer
- Snapshots of other problems

Contents

Transportation Planning Process

1. Organization and definition
2. Base year inventory
3. Model analysis
    1. trip generation
    2. trip distribution
    3. modal split
    4. traffic assignement
4. Travel forecast
5. Network evaluation

Urban Transportation Network Analysis

Input of the analysis:

- transportation infrastructure and services (street, intersections...)
- transportation system and control policies
- demand for travel.

Two stage analysis:

- First stage: determining the congestion, i.e. calculating the flow through each component of the network.
- Second stage : computing measure of interests according to the flow.
  - travel time and costs,
  - revenue and profit of ancilliary services,
  - welfare measures (accessibility, equity),
  - flow by-products (pollution, change in land-value)...

Why do we need a system approach ?

- Some decision could be taken according to local measure. For example traffic light can be timed according to data on current usual traffic at the intersection.
- However most decision will impact the travel time / confort. Hence, some people will adapt their usual transit route.
- Consequently, the congestion on the network will change, changing time / confort of other part of the system and inducing other people to adapt their path...
- After some time these ripple effect will lessen, and the system will reach a new equilibrium.

Equilibrium in Markets

- For a given product, in a perfectly competitive market we have:
  - a production function giving the number of product companies are ready to make for a given price;
  - a demand function giving the number of product consumer are ready to buy for a given price.
- In some cases, especially in transportation, the price is not the only determinant factor. Regularity, fiability, ease of use, comfort are other determinant factor.
- In the remaining of the course we will be speaking of cost of each path, the cost factoring in all of this factors.

Nash Equilibrium : Prisonner's Dilemna

Two guys got caught while dealing chocolate. As he is missing hard evidence the judge offer them a deal.

- If both deny their implication they will get 2 month each.
- If one speak, and the other deny, the first will get 1 month while the other will get 5 months.
- If both speak they get 4 month each.

Question : what is the equilibrium ?

Nash Equilibrium

- In game theory we consider multiple agents $a \in \mathcal{A}$, each having a set of possible action $u_a \in \mathcal{U}_a$.
- Each agent earn a reward $r_a(u)$ depending on his action, as well as the other actions.
- A (pure) Nash equilibrium is a set of actions $\{u_a\}_{a\in\mathcal{A}}$, such that no player can increase his reward by changing is action if the other keep these actions :

$$\forall a \in \mathcal{A}, \quad \forall u'_a \in \mathcal{U}_a, \qquad r_a(u'_a, u_{-a}) \leq r_a(u_a, u_{-a}).$$

- A recommandation can be followed only if it is a Nash Equilibrium.

Game Theory : a few classes

- Number of player
  - 2 (most results)
  - $n > 2$ (hard, even with 3)
  - an infinity.
- Objective
  - zero-sum game (e.g. chess)
  - cooperative : everybody share the same objective (e.g. pandemia)
  - generic (e.g. Prisonner dilemna)

Game theory : a few definitions

Definition

A Nash equilibrium is a set of action such that no player can unilaterally improve its pay-off by changing his action.

Definition

A Pareto efficient solution is a set of action such that no other set of actions can strictly improve at least one player pay-off without decreasing at least another.

Definition

A social optimum is a set of action minimizing the pay-off average.

Exercises :

- what about Prisonner's Dilemma ?
- what about Zero Sum games ?

Exercise: A beautiful mind

A beautiful mind : https://youtu.be/a9k4UJrCdKg

- Is the solution proposed by Nash a Nash equilibrium ?
- Is the solution proposed by Nash a Pareto Optimum ?
- Is the solution proposed by "Smith" a Nash equilibrium ?
- Is the solution proposed by "Smith" a Pareto Optimum ?
- Any other suggestion ?

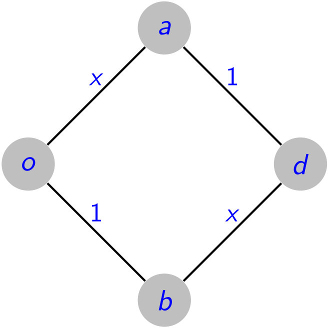
Contents

Game theory in road network

- People choose their means of transport (e.g. car versus public transport), their time of departure, their itinerary.
- Each user choose in its own interest (mainly the shortest time / lowest cost).
- The time depends on the congestion, which means on the choice of other users.
- Hence, we are in a game framework : users interact with conflicting interest.

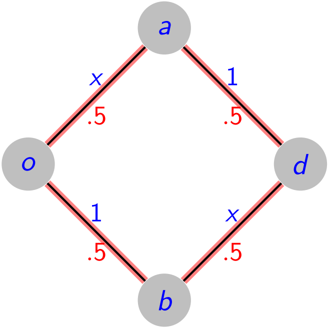
A very simple framework

- Consider a large group of who person want to go from the same origin o to the destination d, at the same time, with the same car.
- We look at a very simple graph with two roads, each composed of two edges.
- The time on each edges of the road is given as a function of the number of person taking the given edge.



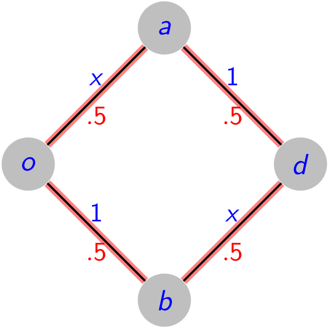
A very simple framework

- Consider a large group of who person want to go from the same origin o to the destination d, at the same time, with the same car.
- We look at a very simple graph with two roads, each composed of two edges.
- The time on each edges of the road is given as a function of the number of person taking the given edge.



A very simple framework

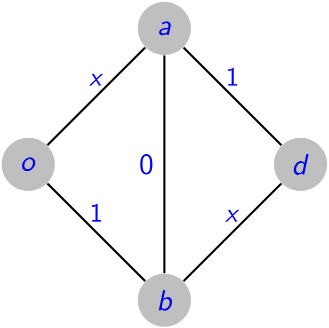
- Consider a large group of who person want to go from the same origin o to the destination d, at the same time, with the same car.
- We look at a very simple graph with two roads, each composed of two edges.
- The time on each edges of the road is given as a function of the number of person taking the given edge.



Total time : 1.5

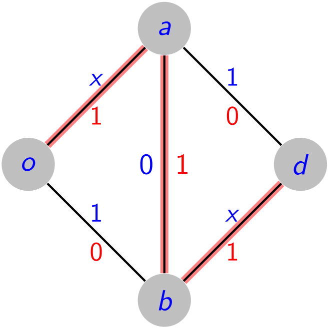
Adding a road

- Now someone decide to construct a new, very efficient road with cost 0.
- What is the new equilibrium ?



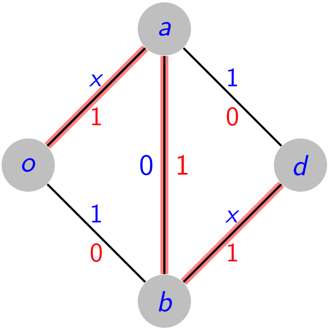
Adding a road

- Now someone decide to construct a new, very efficient road with cost 0.
- What is the new equilibrium ?



Adding a road

- Now someone decide to construct a new, very efficient road with cost 0.
- What is the new equilibrium ?
- Notice that the time for every user as increased ! This is the price of anarchy.



Total time : 2

Another explanation

https://www.youtube.com/watch?v=ZiauQXIKs3U (7')

And a physical demonstration :

https://www.youtube.com/watch?v=nMrYlspifuo

Definitions snapshot

On this example we can compare :

- User Equilibrium (UE), with global cost 2
- System Optimum (SO), with global cost 1.5
- price of anarchy : $4/3$.

Definition

A Wardrop (User) Equilibrium, is a repartition of flow such that no single user can improve its cost (travel time) by unilaterally changing routes.

Real case examples

- 42d Street of New York. (New York Times, 25/12/1990).
- Stuttgart 1969 (a newly built road was closed again), Seoul 2003 (6 lanes highway was turned into a park).
- New York 2009 (closed some places with success)
- In 2008, researcher found road in Boston and NYC that should be closed to diminish traffic.
- Steinberg and Zangwill showed that Braess paradox is more or less as likely to occur as not.
- Rapoport's experiment (2009):
  - A group of 18 students is presented with the problem of repetively (40 times) choosing its road on the graph, earning money for the experiment : fastest meaning more money.
  - Then the graph is modified (either by adding the 0 cost road, or retiring it).
  - Conclusion : after a few iteration the observed repartition is close to the theoretical one with some oscillations.
  - Then tested on a bigger network.

Exercise

- Two nodes : a and b
- Two edges : (from a to b): 1 and 2
- Total number of trips : 1000
- Costs : $c_1(x_1) = 5 + 2x_1$, $c_2(x_2) = 10 + x_2$.
- Question : what is the repartition of the trips along the two edges ?
- Same question with $c_1(x_1) = 15(1 + 0.15(\frac{x_1}{1000})^4)$, $c_2(x_2) = 20(1 + 0.15(\frac{x_2}{3000})^4)$?

Another Nash Equilibrium: Split or Steal

The prisonner's Dilemna has been used as the final part of TV game show called "split or steal".

The rules :

- The two remaining contestants have a certain amount of money M.
- They each have to choose "split" or "steal"
- If both "split" they each get half: $M/2$.
- If one "steal" while the other "split", the stealing one get M and the other 0.
- If they both "steal" they get nothing.

Here is an example :
https://www.youtube.com/watch?v=yM38mRHY150&list=PLq4_sHebc4IWI2VQnqaKXf0YXEj88jcK0&index=5
Here is a very nice example of why reality is more complex than math : https://www.youtube.com/watch?v=S0qjK3TWZE8

Operation Research and Transport
Shortest Path Algorithm

V. Leclère (ENPC)

April 22th, 2020

In the previous episode

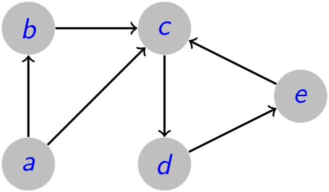
We have seen :

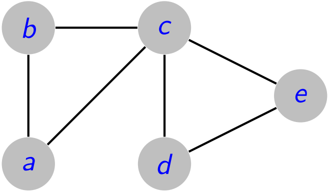
- A few definitions about game theory (Nash equilibrium, Pareto efficient point, Social optimum)
- Examples of Braess paradox
- Applications of the course in the industry

Contents

What is a Graph ?

- A graph is one of the elementary modelisation tools of Operation Research.
- A directed graph (V, E) is defined by
  - A finite set of n vertices V
  - A finite set of m edges each linked to an origin and a destination.
- A graph is said to be undirected if we do not distinguish between the origin and the destination.





A few definitions

Consider a directed graph (V, E).

- If $(u, v) \in E$, u is a predecessor of v, and v is a successor of u.
- A path is a sequence of edges $\{e_k\}_{k \in [\![1,n]\!]}$, such that the destination of one edge is the origin of the next. The origin of the first edge is the origin of the path, and the destination of the last edge is the destination of the path.
- A (directed) graph is connected if for all $u, v \in V$, there is a u-v-path.
- A cycle is a path where the destination vertex is the origin.

A weighted graph

- A weighted (directed) graph is a (directed) graph (V, E) with a weight function $\ell : E \to \mathbb{R}$.
- The weight of a $s - t-$path p is sum of the weights of the edges contained in the path :

$$\ell(p) := \sum_{e \in p} \ell(e).$$

- The shortest path from o to d is the path of minimal weight with origin o and destination d.
- An absorbing cycle is a cycle of strictly negative weight.

Contents

An optimality condition

The methods we are going to present are based on a label function over the vertices. This function should be understood as an estimate cost of the shortest path cost between the origin and the current vertex.

Theorem

Suppose that there exists a function $\lambda : V \mapsto \mathbb{R} \cup \{+\infty\}$, *such that*

$$\forall (i,j) \in E, \qquad \lambda_j \leq \lambda_i + \ell(i,j).$$

Then, if p *is an* s-t*-path, we have* $\ell(p) \leq \lambda(t) - \lambda(s)$[a]
In particular, if p *is such that*

$$\forall (i,j) \in p, \qquad \lambda_j = \lambda_i + \ell(i,j),$$

then p *is a shortest path.*

[a] with the convention $\infty - \infty = \infty$.

A generic algorithm

We keep a list of candidates vertices $U \subset V$, and a label function $\lambda : V \mapsto \mathbb{R} \cup \{+\infty\}$.

$U := \{o\}$;
$\lambda(o) := 0$;
$\forall v \neq o, \quad \lambda(v) = +\infty$;

while $U \neq \emptyset$ **do**
  choose $u \in U$;
  for v *successor of* u **do**
    if $\lambda(v) > \lambda(u) + \ell(u, v)$ **then**
      $\lambda(v) := \lambda(u) + \ell(u, v)$;
      $U := U \cup \{v\}$;
  $U := U \setminus \{u\}$;

Algorithm properties

- If $\lambda(u) < \infty$ then $\lambda(u)$ is the cost of a o-u-path.
- If $u \notin U$ then
  - either $\lambda(i) = \infty$ (never visited)
  - or

$$\text{for all successor } v \text{ of } u, \qquad \lambda(v) \leq \lambda(u) + \ell(u, v).$$

- If the algorithm end $\lambda(u)$ is the smallest cost to go from o to u.
- Algorithm end iff there is no path starting at o and containing an absorbing circuit.

Dijkstra's algorithm

Assume that all cost are non-negative.

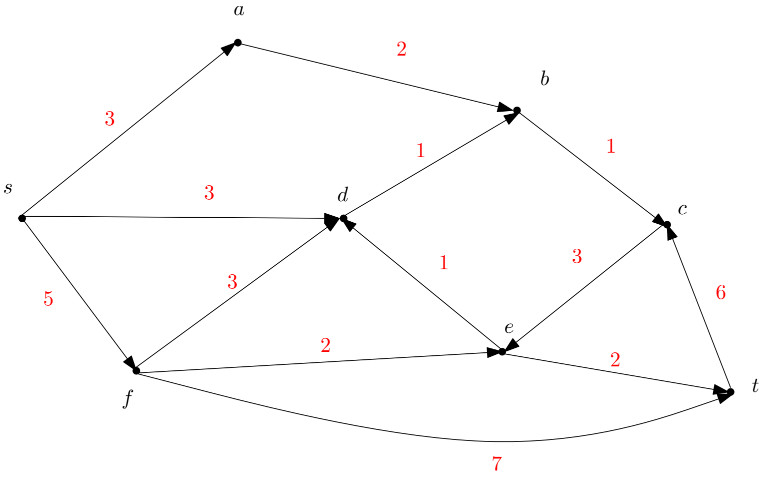
$U := \{o\}$;
$\lambda(o) := 0$;
$\forall v \neq o, \quad \lambda(v) = +\infty$;

while $U \neq \emptyset$ **do**
  choose $u \in \arg\min_{u' \in U} \lambda(u')$;
  for v *successor of* u **do**
    if $\lambda(v) > \lambda(u) + \ell(u, v)$ **then**
      $\lambda(v) := \lambda(u) + \ell(u, v)$;
      $U := U \cup \{v\}$;
  $U := U \setminus \{u\}$;

Algorithm 1: Dijkstra algorithm

A video explanation

https://www.youtube.com/watch?v=zXfDYaahsNA


a
2
b
3
1
1
s
3
d
c
1
3
6
5
3
e
2
2
t
f
7

Application example

s	a	b	c	d	e	f	t
(0)	∞	∞	∞	∞	∞	∞	∞

Application example

s	a	b	c	d	e	f	t
(0)	∞	∞	∞	∞	∞	∞	∞
0	(3)	∞	∞	(3)	∞	(5)	∞

Application example

s	a	b	c	d	e	f	t
(0)	∞	∞	∞	∞	∞	∞	∞
0	(3)	∞	∞	(3)	∞	(5)	∞
0	3	(5)	∞	(3)	∞	(5)	∞

Application example

s	a	b	c	d	e	f	t
(0)	∞	∞	∞	∞	∞	∞	∞
0	(3)	∞	∞	(3)	∞	(5)	∞
0	3	(5)	∞	(3)	∞	(5)	∞
0	3	(4)	∞	3	∞	(5)	∞

Application example

s	a	b	c	d	e	f	t
(0)	∞	∞	∞	∞	∞	∞	∞
0	(3)	∞	∞	(3)	∞	(5)	∞
0	3	(5)	∞	(3)	∞	(5)	∞
0	3	(4)	∞	3	∞	(5)	∞
0	3	4	(5)	3	∞	(5)	∞

Application example

s	a	b	c	d	e	f	t
(0)	∞	∞	∞	∞	∞	∞	∞
0	(3)	∞	∞	(3)	∞	(5)	∞
0	3	(5)	∞	(3)	∞	(5)	∞
0	3	(4)	∞	3	∞	(5)	∞
0	3	4	(5)	3	∞	(5)	∞
0	3	4	5	3	(8)	(5)	∞

Application example

s	a	b	c	d	e	f	t
(0)	∞	∞	∞	∞	∞	∞	∞
0	(3)	∞	∞	(3)	∞	(5)	∞
0	3	(5)	∞	(3)	∞	(5)	∞
0	3	(4)	∞	3	∞	(5)	∞
0	3	4	(5)	3	∞	(5)	∞
0	3	4	5	3	(8)	(5)	∞
0	3	4	5	3	(7)	5	(12)

Application example

s	a	b	c	d	e	f	t
(0)	∞	∞	∞	∞	∞	∞	∞
0	(3)	∞	∞	(3)	∞	(5)	∞
0	3	(5)	∞	(3)	∞	(5)	∞
0	3	(4)	∞	3	∞	(5)	∞
0	3	4	(5)	3	∞	(5)	∞
0	3	4	5	3	(8)	(5)	∞
0	3	4	5	3	(7)	5	(12)
0	3	4	5	3	7	5	(9)

Application example

s	a	b	c	d	e	f	t
(0)	∞	∞	∞	∞	∞	∞	∞
0	(3)	∞	∞	(3)	∞	(5)	∞
0	3	(5)	∞	(3)	∞	(5)	∞
0	3	(4)	∞	3	∞	(5)	∞
0	3	4	(5)	3	∞	(5)	∞
0	3	4	5	3	(8)	(5)	∞
0	3	4	5	3	(7)	5	(12)
0	3	4	5	3	7	5	(9)
0	3	4	5	3	7	5	9

Shortest path complexity with positive cost

Theorem

Let $G = (V, E)$ be a directed graph, $o \in V$ and a cost function $\ell : E \to \mathbb{R}_+$.
When applying Dijkstra's algorithm, each node is visited at most once. Once a node v has been visited it's label is constant accross iterations and equal to the cost of shortest o-v-path.
In particular, a shortest path from o to any vertex v can be found in $O(n^2)$, where $n = |V|$.

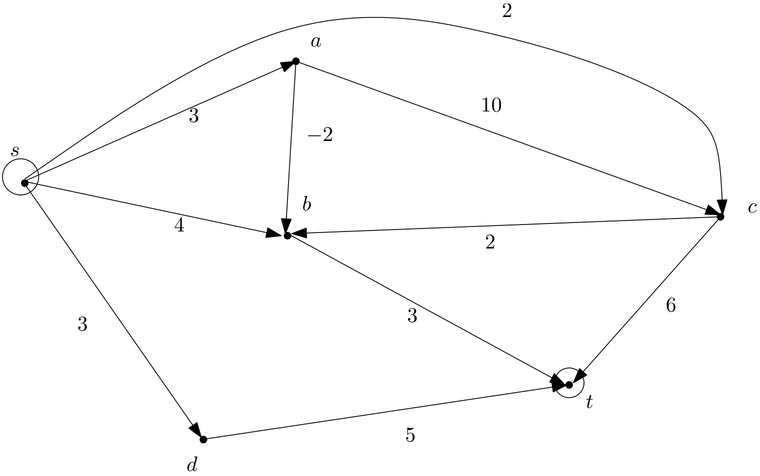
Note that with specific implementation (e.g. in binary tree of nodes) we can obtain a complexity in $O(n + m\log(\log(m)))$.

Contents

Recall on DFS

Deep First Search is an algorithm to visit every nodes on a graph. It consists in going as deep as possible (taking any children of a given node), and backtracking when you reach a leaf.
https://www.youtube.com/watch?v=fI6X6IBkzcw

Acircuitic graph



Topological Ordering

Definition

A topological ordering of a graph is an ordering (injective function from V to $\mathbb{N}$) of the vertices such that the starting endpoint of every edge occurs earlier in the ordering than the ending endpoint of the edge.

Applications :

- courses prerequisite
- compilation order
- manufacturing
- ...

Topological order is equivalent to acircuitic.

Theorem

A directed graph is acyclic if and only if there exist a topological ordering. A topological ordering can be found in $O(|V| + |E|)$.

Proof :

- If G has a topological ordering then it is acyclic. (by contradiction).
- If G is a DAG, then it has a root node (with no incoming edges). (by contradiction).
- If G is a DAG then G has a topological ordering (by induction).
- Done in $O(|V| + |E|)$ (maintain $count(v)$: number of incoming edges, S: set of remaining nodes with no incoming edges).

video explanation

https://www.youtube.com/watch?v=gyddxytyAiE (They use DFS to count the in-degree, it is simply a fancy way of looping on arcs)

Contents

Bellman's idea

A part of an optimal path is still optimal.

$\lambda(v) :=$ minimum cost of o-v-path, with $\lambda(v) := \infty$ if such a path doesn't exist.

Bellman's equation

$$\lambda(v) = \min_{(u,v)\in E} \big(\lambda(u) + \ell(u,v)\big)$$

There exist a predecessor u of v such that the shortest path between o and v is given by the shortest path between o and u adding the edge (u,v).

Bellman's idea

A part of an optimal path is still optimal.

$\lambda(v) :=$ minimum cost of o-v-path, with $\lambda(v) := \infty$ if such a path doesn't exist.

Bellman's equation

$$\lambda(v) = \min_{(u,v)\in E} \big(\lambda(u) + \ell(u,v)\big)$$

There exist a predecessor u of v such that the shortest path between o and v is given by the shortest path between o and u adding the edge (u,v).

Bellman's idea

A part of an optimal path is still optimal.

$\lambda(v) :=$ minimum cost of o-v-path, with $\lambda(v) := \infty$ if such a path doesn't exist.

Bellman's equation

$$\lambda(v) = \min_{(u,v)\in E} \big(\lambda(u) + \ell(u,v)\big)$$

There exist a predecessor u of v such that the shortest path between o and v is given by the shortest path between o and u adding the edge (u, v).

Dynamic Programming algorithm

Assume that the graph is connected and without cycle.

Data: Graph, cost function

$\lambda(s) := 0$;

$\forall v \neq s, \quad \lambda(v) = +\infty$;

while $\exists v \in V, \quad \lambda(v) = \infty$ **do**

  choose a vertex v such that all predecessors u have a finite label ;

  $\lambda(v) := \min\{\lambda(u) + \ell(u, v) \quad | \quad (u, v) \in E\}$;

Algorithm 2: Bellman Forward algorithm

The while loop can be replaced by a for loop over the nodes in a topological order.

Algorithm

Theorem

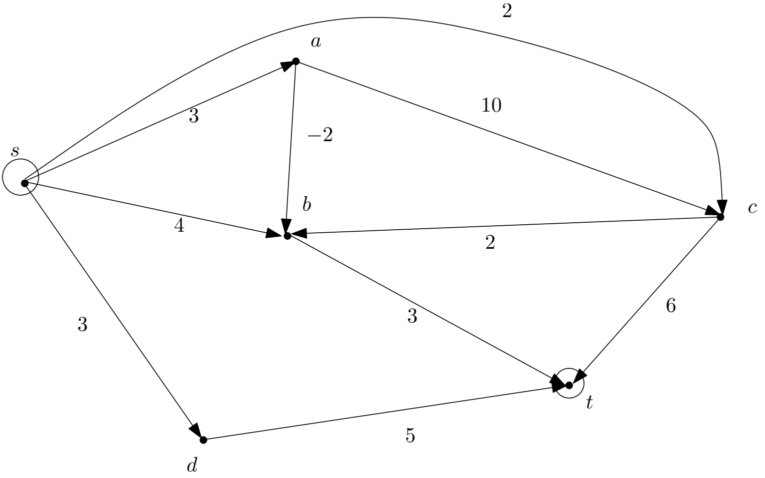
Let $D = (V, E)$ be a directed graph without cycle, and $w : E \to \mathbb{R}$ a cost function. The shortest path from o to any vertex $v \in V$ can be computed in $O(n + m)$.

Note that we do not require the costs to be positive for the Bellman algorithm. In particular we can also compute the longest path.

Video explanation

https://www.youtube.com/watch?v=TXkDpqjDMHA (up to 6:30)

Acircuitic graph



Application example

s	a	c	b	d	t
0	∞	∞	∞	∞	∞

Application example

s	a	c	b	d	t
0	∞	∞	∞	∞	∞
0	$0+3$	∞	∞	∞	∞

Application example

s	a	c	b	d	t
0	∞	∞	∞	∞	∞
0	$0+3$	∞	∞	∞	∞
0	3	$\min\{0+2, 10+3\}$	∞	∞	∞

Application example

s	a	c	b	d	t
0	∞	∞	∞	∞	∞
0	$0+3$	∞	∞	∞	∞
0	3	$\min\{0+2, 10+3\}$	∞	∞	∞
0	3	2	$\min\{0+4, 3-2, 2+2\}$	∞	∞

Application example

s	a	c	b	d	t
0	∞	∞	∞	∞	∞
0	$0+3$	∞	∞	∞	∞
0	3	$\min\{0+2, 10+3\}$	∞	∞	∞
0	3	2	$\min\{0+4, 3-2, 2+2\}$	∞	∞
0	3	2	1	$0+3$	∞

Application example

s	a	c	b	d	t
0	∞	∞	∞	∞	∞
0	$0+3$	∞	∞	∞	∞
0	3	$\min\{0+2, 10+3\}$	∞	∞	∞
0	3	2	$\min\{0+4, 3-2, 2+2\}$	∞	∞
0	3	2	1	$0+3$	∞
0	3	2	1	3	4

Contents

Algorithm Principle

- To reach destination d from origin o in a weighted directed graph we keep a label function $\lambda(n)$.
- The label function is defined as a sum $\lambda = g + h$, where
  - $g(n)$ is the best cost of a o-n-path
  - $h(n)$ is an (user-given) heuristic of the cost of a n-d-path

```
U := {s} ; λ(s) := h(s) ; ∀v ≠ s,   λ(v) = g(v) = +∞ ;
while U ≠ ∅ do
    choose u ∈ arg min_{u'∈U} λ(u') ;
    for v successor of u do
        if g(v) > g(u) + ℓ(u, v) then
            g(v) := g(u) + ℓ(u, v);
            λ(v) := g(v) + h(v);
            U := U ∪ {v};
    U := U \ {u} ;
```

Algorithm 3: $A^\star$ algorithm

Heuristic definitions

Definition (admissible heuristic)

A heuristic is admissible if it underestimate the actual cost to get to the destination, i.e. if for all vertex $v \in V$, $h(v)$ is lower or equal to the cost of a shortest path from v to d.

Example : in the case of a graph in $\mathbb{R}^2$ with a cost proportional to the euclidean distance, an admissible heuristic is the euclidean distance between v and t (the "direct flight" distance).

Definition (consistent heuristic)

The heuristic h is consistent if it is admissible and for every $(u, v) \in E$, $h(u) \leq \ell(u, v) + h(v)$.

A consistent heuristic satisfies a "triangle inequality".

Consistent heuristic

- $h \equiv 0$ is consistent. In this case $A^\star$ reduced to Dijkstra.
- If h is consistent, $A^\star$ can be implemented more efficiently.
- Roughly speaking, no node needs to be processed more than once, and $A^\star$ is equivalent to running Dijkstra's algorithm with the reduced cost $\tilde{\ell}(u, v) = \ell(u, v) + h(v) - h(u)$.

Choice of heuristic

- If $h \equiv 0$, we have Dijkstra algorithm.
- If h is admissible, $A^\star$ yields the shortest path.
- If h is consistent we have Dijkstra's algorithm with the reduced cost $\tilde{\ell}(u, v) = \ell(u, v) + h(v) - h(u)$.
- If h is exact we explore only the best path.
- If h is not admissible the algorithm might not yield the shortest path, but can be fast to find a good path.

Video explanation

Detailed explanation of A* :
https://www.youtube.com/watch?v=eSOJ3ARN5FM
Some comparison of the algorithm :
https://www.youtube.com/watch?v=GC-nBgi9r0U
A quick run of A* :
https://www.youtube.com/watch?v=19h1g22hby8

Wardrop Equilibrium

V. Leclère (ENPC)

May 5th, 2021

Contents

Convex set

- A set $C \subset \mathbb{R}^n$ is *convex* iff

$$\forall x, y \in C, \quad \forall t \in [0,1], \qquad tx + (1-t)y \in C.$$

- Intersection of convex sets is convex.
- A closed convex set C is equal to the intersection of all half-spaces containing it.

Convex function

- The *epigraph* of a function $f : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ is

$$\text{epi}(f) := \Big\{ (x, t) \in \mathbb{R}^n \times \mathbb{R} \quad \Big| \quad t \geq f(x) \Big\}.$$

- The *domain* of a function f is

$$\mathrm{dom}(f) := \Big\{ x \in \mathbb{R}^n \quad \Big| \quad f(x) < +\infty \Big\}$$

- The function f is said to be *convex* iff its epigraph is convex, in other words iff

$$\forall t \in [0,1], \qquad f(tx + (1-t)y) \leq tf(x) + (1-t)f(y).$$

- The function f is said to be *strictly convex* iff

$$\forall t \in (0,1), \qquad f(tx + (1-t)y) < tf(x) + (1-t)f(y).$$

Convexity and differentiable

We assume sufficient regularity for the written object to exist.

- If $f : \mathbb{R} \to \mathbb{R}$.
    - f is convex iff f' non-decreasing.
    - If $f' > 0$ then f is strictly convex.
    - f is convex iff $f'' \geq 0$.
    - If $f'' > 0$ then f is strictly convex.
- If $f : \mathbb{R}^n \to \mathbb{R}$
    - f is convex iff ∇f non-decreasing (i.e. $(\nabla f(y) - \nabla f(x)) \cdot (y - x) \geq 0$).
    - f is convex iff $\nabla^2 f(x) \succeq 0$ for all x.
    - If $\nabla^2 f(x) \succ 0$ for all x then f is strictly convex.

Video explanation

https://www.youtube.com/watch?v=qF0aDJfEa4Y

Convex differentiable optimization problem

Consider the following optimization problem.

$$\begin{aligned} \min_{x\in\mathbb{R}^n} \quad & f(x) & & (P) \\ \text{s.t.} \quad & g_i(x) = 0 & & \forall i \in [n_E] \\ & h_j(x) \leq 0 & & \forall j \in [n_I] \end{aligned}$$

with

$$X := \left\{ x \in \mathbb{R}^n \mid \forall i \in [n_E], \quad g_i(x) = 0, \quad \forall j \in [n_I], \quad h_j(x) \leq 0 \right\}.$$

- (P) is a *convex optimization problem* if f and X are convex.
- (P) is a *convex differentiable optimization problem* if f, and h_j (for $j \in [n_I]$) are convex differentiable and g_i (for $i \in [n_E]$) are affine

Convex differentiable optimization problem

Consider the following optimization problem.

$$\begin{aligned} \min_{x\in\mathbb{R}^n} \quad & f(x) & & (P) \\ \text{s.t.} \quad & g_i(x) = 0 & & \forall i \in [n_E] \\ & h_j(x) \leq 0 & & \forall j \in [n_I] \end{aligned}$$

with

$$X := \{x \in \mathbb{R}^n \mid \forall i \in [n_E], \quad g_i(x) = 0, \quad \forall j \in [n_I], \quad h_j(x) \leq 0\}.$$

- (P) is a *convex optimization problem* if f and X are convex.
- (P) is a *convex differentiable optimization problem* if f, and h_j (for $j \in [n_I]$) are convex differentiable and g_i (for $i \in [n_E]$) are affine

KKT conditions

Theorem (KKT)

Let $x^\sharp$ be an optimal solution to a differentiable optimization problem (P). If the constraints are qualified at $x^\sharp$ then there exists optimal multipliers $\lambda^\sharp \in \mathbb{R}^{n_E}$ and $\mu^\sharp \in \mathbb{R}^{n_I}$ satisfying

$$\begin{cases} \nabla f(x^\sharp) + \sum_{i=1}^{n} \lambda_i^\sharp \nabla g_i(x^\sharp) + \sum_{j=1}^{n_I} \mu_i^\sharp \nabla h_j(x^\sharp) = 0 & \textit{first order condition} \\ g(x^\sharp) = 0 & \textit{primal admissibility} \\ h(x^\sharp) \leq 0 & \\ \mu \geq 0 & \textit{dual admissibility} \\ \mu_i g_i(x^\sharp) = 0, \quad \forall i \in [\![1, n_I]\!] & \textit{complementarity} \end{cases}$$

The three last conditions are sometimes compactly written

$$0 \leq g(x^\sharp) \perp \mu \geq 0.$$

Video explanation

Intro to constrained optimization
https://www.youtube.com/watch?v=vwUV2IDLP8Q
Explaining tangeancy of multipliers
https://www.youtube.com/watch?v=yuqB-d5MjZA
Marginal interpretation of multipliers
https://www.youtube.com/watch?v=m-G3K2GPmEQ

Slater condition

A convex optimization problem (P) satisfies the *Slater* condition if there exists a strictly admissible $x_0 \in \mathbb{R}^n$ that is

$$\forall i \in [n_E], \quad g_i(x_0) = 0, \quad \forall j \in [n_I], \quad h_j(x_0) < 0.$$

If the Slater condition is satisfied, then the constraints are qualified at any $x \in X$.

Another optimality condition (convex case)

Theorem

If (P) *is a convex differentiable optimization problem, then* $x^\sharp \in X$ *is an optimal solution iff*

$$\forall y \in X, \quad \nabla f(x) \cdot (y - x) \geq 0.$$

Contents

The set-up

- $G = (V, E)$ is a directed graph
- x_e for $e \in E$ represent the flux (number of people per hour) taking edge e
- $\ell_e : \mathbb{R} \to \mathbb{R}^+$ the cost incurred by a given user to take edge e
- We consider K origin-destination vertex pair $\{o^k, d^k\}_{k \in [\![1,K]\!]}$, such that there exists at least one path from o^k to d^k.
- r_k is the rate of people going from o^k to d^k
- $\mathcal{P}_k$ the set of all simple (i.e. without cycle) path form o^k to d^k
- We denote f_p the flux of people taking path $p \in \mathcal{P}_k$

The set-up

- $G = (V, E)$ is a directed graph
- x_e for $e \in E$ represent the flux (number of people per hour) taking edge e
- $\ell_e : \mathbb{R} \to \mathbb{R}^+$ the cost incurred by a given user to take edge e
- We consider K origin-destination vertex pair $\{o^k, d^k\}_{k \in [\![1,K]\!]}$, such that there exists at least one path from o^k to d^k.
- r_k is the rate of people going from o^k to d^k
- $\mathcal{P}_k$ the set of all simple (i.e. without cycle) path form o^k to d^k
- We denote f_p the flux of people taking path $p \in \mathcal{P}_k$

Some physical relations

People going from o^k to d^k have to choose a path

$$r^k = \sum_{p \in \mathcal{P}^k} f_p.$$

People going through an edge are on a simple path taking this edge

$$x_e = \sum_{p \ni e} f_p.$$

The flux are non-negative

$$\forall p \in \mathcal{P}, \quad f_p \geq 0, \qquad \text{and} \qquad , \forall e \in E, \quad x_e \geq 0$$

Some physical relations

People going from o^k to d^k have to choose a path

$$r^k = \sum_{p \in \mathcal{P}^k} f_p.$$

People going through an edge are on a simple path taking this edge

$$x_e = \sum_{p \ni e} f_p.$$

The flux are non-negative

$$\forall p \in \mathcal{P}, \quad f_p \geq 0, \qquad \text{and} \qquad , \forall e \in E, \quad x_e \geq 0$$

Some physical relations

People going from o^k to d^k have to choose a path

$$r^k = \sum_{p \in \mathcal{P}^k} f_p.$$

People going through an edge are on a simple path taking this edge

$$x_e = \sum_{p \ni e} f_p.$$

The flux are non-negative

$$\forall p \in \mathcal{P}, \quad f_p \geq 0, \qquad \text{and} \qquad , \forall e \in E, \quad x_e \geq 0$$

System optimum problem

The system optimum consists in minimizing the sum of all costs over the admissible flux $x = (x_e)_{e \in E}$

- Given x, the cost of taking edge e for one person is $\ell_e(x_e)$.
- The cost for the system for edge e is thus $x_e \ell_e(x_e)$.
- Thus minimizing the system costs consists in solving

$$\begin{aligned}
\min_{x,f} \quad & \sum_{e \in E} x_e \ell_e(x_e) & & (SO) \\
s.t. \quad & r_k = \sum_{p \in \mathcal{P}_k} f_p & & k \in [\![1, K]\!] \\
& x_e = \sum_{p \ni e} f_p & & e \in E \\
& f_p \geq 0 & & p \in \mathcal{P}
\end{aligned}$$

System optimum problem

The system optimum consists in minimizing the sum of all costs over the admissible flux $x = (x_e)_{e \in E}$

- Given x, the cost of taking edge e for one person is $\ell_e(x_e)$.
- The cost for the system for edge e is thus $x_e \ell_e(x_e)$.
- Thus minimizing the system costs consists in solving

$$
\begin{aligned}
\min_{x,f} \quad & \sum_{e \in E} x_e \ell_e(x_e) && (SO) \\
s.t. \quad & r_k = \sum_{p \in \mathcal{P}_k} f_p && k \in [\![1, K]\!] \\
& x_e = \sum_{p \ni e} f_p && e \in E \\
& f_p \geq 0 && p \in \mathcal{P}
\end{aligned}
$$

System optimum problem

The system optimum consists in minimizing the sum of all costs over the admissible flux $x = (x_e)_{e \in E}$

- Given x, the cost of taking edge e for one person is $\ell_e(x_e)$.
- The cost for the system for edge e is thus $x_e \ell_e(x_e)$.
- Thus minimizing the system costs consists in solving

$$
\begin{aligned}
\min_{x,f} \quad & \sum_{e \in E} x_e \ell_e(x_e) & & (SO) \\
s.t. \quad & r_k = \sum_{p \in \mathcal{P}_k} f_p & & k \in [\![1, K]\!] \\
& x_e = \sum_{p \ni e} f_p & & e \in E \\
& f_p \geq 0 & & p \in \mathcal{P}
\end{aligned}
$$

Path intensity formulation

- We can reformulate the (SO) problem only using path-intensity $f = (f_p)_{p\in\mathcal{P}}$.
- Define $x_e(f) := \sum_{p\ni e} f_p$, and $x = (x_e)_{e\in E}$.
- Define the loss along a path $\ell_p(f) = \sum_{e\in p} \ell_e\big(\underbrace{\sum_{p'\ni e} f_{p'}}_{x_e(f)}\big)$
- The total cost is thus

$$C(f) = \sum_{p\in\mathcal{P}} f_p \ell_p(f) = \sum_{e\in E} x_e \ell_e(x_e(f)) = C(x(f)).$$

Path intensity formulation

- We can reformulate the (SO) problem only using path-intensity $f = (f_p)_{p \in \mathcal{P}}$.
- Define $x_e(f) := \sum_{p \ni e} f_p$, and $x = (x_e)_{e \in E}$.
- Define the loss along a path $\ell_p(f) = \sum_{e \in p} \ell_e\big(\underbrace{\sum_{p' \ni e} f_{p'}}_{x_e(f)}\big)$
- The total cost is thus

$$C(f) = \sum_{p \in \mathcal{P}} f_p \ell_p(f) = \sum_{e \in E} x_e \ell_e(x_e(f)) = C(x(f)).$$

Path intensity formulation

- We can reformulate the (SO) problem only using path-intensity $f = (f_p)_{p \in \mathcal{P}}$.
- Define $x_e(f) := \sum_{p \ni e} f_p$, and $x = (x_e)_{e \in E}$.
- Define the loss along a path $\ell_p(f) = \sum_{e \in p} \ell_e\big(\underbrace{\sum_{p' \ni e} f_{p'}}_{x_e(f)}\big)$
- The total cost is thus

$$C(f) = \sum_{p \in \mathcal{P}} f_p \ell_p(f) = \sum_{e \in E} x_e \ell_e(x_e(f)) = C(x(f)).$$

Path intensity formulation

- We can reformulate the (SO) problem only using path-intensity $f = (f_p)_{p\in\mathcal{P}}$.
- Define $x_e(f) := \sum_{p\ni e} f_p$, and $x = (x_e)_{e\in E}$.
- Define the loss along a path $\ell_p(f) = \sum_{e\in p} \ell_e\Big(\underbrace{\sum_{p'\ni e} f_{p'}}_{x_e(f)}\Big)$
- The total cost is thus

$$C(f) = \sum_{p\in\mathcal{P}} f_p \ell_p(f) = \sum_{e\in E} x_e \ell_e(x_e(f)) = C(x(f)).$$

Path intensity problem

$$
\begin{aligned}
\min_{f} \quad & \sum_{p \in \mathcal{P}} f_p \ell_p(f) && (SO) \\
s.t. \quad & r_k = \sum_{p \in \mathcal{P}_k} f_p && k \in [\![1, K]\!] \\
& f_p \geq 0 && p \in \mathcal{P}
\end{aligned}
$$

Equilibrium definition

John Wardrop defined a traffic equilibrium as follows. "Under equilibrium conditions traffic arranges itself in congested networks such that all used routes between an O-D pair have equal and minimum costs, while all unused routes have greater or equal costs."

A mathematical definition reads as follows.

Definition

A user flow f is a User Equilibrium if

$$\forall k \in [\![1, K]\!], \quad \forall (p, p') \in \mathcal{P}_k^2, \qquad f_p > 0 \quad \Longrightarrow \quad \ell_p(f) \leq \ell_{p'}(f).$$

Equilibrium definition

John Wardrop defined a traffic equilibrium as follows. "Under equilibrium conditions traffic arranges itself in congested networks such that all used routes between an O-D pair have equal and minimum costs, while all unused routes have greater or equal costs."

A mathematical definition reads as follows.

Definition

A user flow f is a User Equilibrium if

$$\forall k \in [\![1, K]\!], \quad \forall (p, p') \in \mathcal{P}_k^2, \qquad f_p > 0 \quad \Longrightarrow \quad \ell_p(f) \leq \ell_{p'}(f).$$

A new cost function

We are going to show that a user-equilibrium f is defined as a vector satisfying the KKT conditions of a certain optimization problem.

Let define a new edge-loss function by

$$L_e(x_e) := \int_0^{x_e} \ell_e(u)du.$$

The Wardrop potential is defined (for edge intensity) as

$$W(f) = W(x(f)) = \sum_{e \in E} L_e(x_e(f)).$$

A new cost function

We are going to show that a user-equilibrium f is defined as a vector satisfying the KKT conditions of a certain optimization problem.
Let define a new edge-loss function by

$$L_e(x_e) := \int_0^{x_e} \ell_e(u)du.$$

The Wardrop potential is defined (for edge intensity) as

$$W(f) = W(x(f)) = \sum_{e \in E} L_e(x_e(f)).$$

User optimum problem

Theorem

A flow f is a user equilibrium if and only if it satisfies the first order KKT conditions of the following optimization problem

$$\begin{aligned}
\min_{x,f} \quad & W(x) & \\
s.t. \quad & r_k = \sum_{p \in \mathcal{P}_k} f_p & k \in [\![1, K]\!] \\
& x_e = \sum_{p \ni e} f_p & e \in E \\
& f_p \geq 0 & p \in \mathcal{P}
\end{aligned}$$

Proof

In path intensity formulation

$$\begin{aligned} \min_f \quad & \sum_{e \in E} L_e\Big(\sum_{p \ni e} f_p\Big) & \\ s.t. \quad & r_k = \sum_{p \in \mathcal{P}_k} f_p & k \in [\![1, K]\!] \\ & f_p \geq 0 & p \in \mathcal{P} \end{aligned}$$

with Lagrangian

$$L(f, \lambda, \mu) := W(f) + \sum_{k=1}^{K} \lambda_k \Big(r_k - \sum_{p \in \mathcal{P}_k} f_p\Big) + \sum_{p \in \mathcal{P}} \mu_p f_p.$$

Proof

In path intensity formulation

$$\begin{aligned} \min_f \quad & \sum_{e\in E} L_e\Big(\sum_{p\ni e} f_p\Big) & \\ s.t. \quad & r_k = \sum_{p\in\mathcal{P}_k} f_p & k\in[\![1,K]\!] \\ & f_p \geq 0 & p\in\mathcal{P} \end{aligned}$$

with Lagrangian

$$L(f,\lambda,\mu) := W(f) + \sum_{k=1}^K \lambda_k\Big(r_k - \sum_{p\in\mathcal{P}_k} f_p\Big) + \sum_{p\in\mathcal{P}} \mu_p f_p.$$

Proof II

Now note that we have

$$\begin{aligned}\frac{\partial W}{\partial f_p}(f) &= \frac{\partial}{\partial f_p}\Big(\sum_{e\in E} L_e(\sum_{p'\ni e} f_{p'})\Big)\\ &= \sum_{e\in p}\frac{\partial}{\partial x_e}L_e(x_e(f))\\ &= \sum_{e\in p}\ell_e(x_e(f)) = \ell_p(f),\end{aligned}$$

Recall that $L_e(x_e) := \int_0^{x_e}\ell_e(u)du$.

Proof III

The constraints of (UE) are qualified. Consequently its first-order KKT conditions reads

$$\begin{cases} \dfrac{\partial L(f,\lambda,\mu)}{\partial f_p} = \ell_p(f) - \lambda_k + \mu_p = 0 & \forall p \in \mathcal{P}_k, \forall k \in [\![1,K]\!] \\ \dfrac{\partial L(f,\lambda,\mu)}{\partial \lambda_k} = r_k - \sum_{p\in\mathcal{P}_k} f_p = 0 & \forall k \in [\![1,K]\!] \\ \mu_p = 0 \text{ or } f_p = 0 & \forall p \in \mathcal{P} \\ \mu_p \leq 0, f_p \geq 0 & \forall p \in \mathcal{P} \end{cases}$$

f satisfies the KKT conditions iff for all origin-destination pair $k \in [\![1,K]\!]$, and all path $p \in \mathcal{P}_k$ we have

$$\begin{cases} \ell_p(f) = \lambda_k & \text{if } f_p > 0 \\ \ell_p(f) \geq \lambda_k & \text{if } f_p = 0 \end{cases}$$

In other words, if the path $p \in \mathcal{P}_k$ is used, then its cost is λ_k, and all other path $p' \in \mathcal{P}_i$ have a greater or equal cost, which is the definition of

Proof III

The constraints of (UE) are qualified. Consequently its first-order KKT conditions reads

$$\begin{cases} \dfrac{\partial L(f,\lambda,\mu)}{\partial f_p} = \ell_p(f) - \lambda_k + \mu_p = 0 & \forall p \in \mathcal{P}_k, \forall k \in [\![1, K]\!] \\ \dfrac{\partial L(f,\lambda,\mu)}{\partial \lambda_k} = r_k - \sum_{p \in \mathcal{P}_k} f_p = 0 & \forall k \in [\![1, K]\!] \\ \mu_p = 0 \text{ or } f_p = 0 & \forall p \in \mathcal{P} \\ \mu_p \leq 0, f_p \geq 0 & \forall p \in \mathcal{P} \end{cases}$$

f satisfies the KKT conditions iff for all origin-destination pair $k \in [\![1, K]\!]$, and all path $p \in \mathcal{P}_k$ we have

$$\begin{cases} \ell_p(f) = \lambda_k & \text{if } f_p > 0 \\ \ell_p(f) \geq \lambda_k & \text{if } f_p = 0 \end{cases}$$

In other words, if the path $p \in \mathcal{P}_k$ is used, then its cost is λ_k, and all other path $p' \in \mathcal{P}_i$ have a greater or equal cost, which is the definition of

Proof III

The constraints of (UE) are qualified. Consequently its first-order KKT conditions reads

$$\begin{cases} \dfrac{\partial L(f,\lambda,\mu)}{\partial f_p} = \ell_p(f) - \lambda_k + \mu_p = 0 & \forall p \in \mathcal{P}_k, \forall k \in [\![1,K]\!] \\ \dfrac{\partial L(f,\lambda,\mu)}{\partial \lambda_k} = r_k - \sum_{p \in \mathcal{P}_k} f_p = 0 & \forall k \in [\![1,K]\!] \\ \mu_p = 0 \text{ or } f_p = 0 & \forall p \in \mathcal{P} \\ \mu_p \leq 0, f_p \geq 0 & \forall p \in \mathcal{P} \end{cases}$$

f satisfies the KKT conditions iff for all origin-destination pair $k \in [\![1,K]\!]$, and all path $p \in \mathcal{P}_k$ we have

$$\begin{cases} \ell_p(f) = \lambda_k & \text{if } f_p > 0 \\ \ell_p(f) \geq \lambda_k & \text{if } f_p = 0 \end{cases}$$

In other words, if the path $p \in \mathcal{P}_k$ is used, then its cost is λ_k, and all other path $p' \in \mathcal{P}_i$ have a greater or equal cost, which is the definition of a Wardrop Equilibrium.

Convex case : equivalence

If the loss functions (in edge-intensity) are non-decreasing then the Wardrop potential W is convex.

Theorem

Assume that the loss function ℓ_e are non-decreasing for all $e \in E$. Then there exists at least one user equilibrium, and a flow f is a user equilibrium if and only if it solves (UE)

Proof : the cost is convex as composition of convex and affine functions, thus KKT is a necessary and sufficient condition for optimality.

Convex case : equivalence

If the loss functions (in edge-intensity) are non-decreasing then the Wardrop potential W is convex.

Theorem

Assume that the loss function ℓ_e are non-decreasing for all $e \in E$. Then there exists at least one user equilibrium, and a flow f is a user equilibrium if and only if it solves (UE)

Proof : the cost is convex as composition of convex and affine functions, thus KKT is a necessary and sufficient condition for optimality.

Convex case : characterization

define the system cost of a flow f for a given flow f', as

$$C^{f}(f) := \sum_{e \in E} x_e(f) \ell_e\big(x_e(f')\big).$$

Theorem

Assume that the cost functions ℓ_e are continuous and non-decreasing. Then, f^{UE} is a user equilibrium iff

$$\forall f \in F^{ad}, \qquad C^{f^{UE}}(f^{UE}) \leq C^{f^{UE}}(f),$$

where F^{ad} is the set of admissible flows.

Convex case : characterization

define the system cost of a flow f for a given flow f', as

$$C^{f}(f) := \sum_{e\in E} x_e(f)\ell_e\big(x_e(f')\big).$$

Theorem

Assume that the cost functions ℓ_e are continuous and non-decreasing. Then, f^{UE} is a user equilibrium iff

$$\forall f \in F^{ad}, \qquad C^{f^{UE}}(f^{UE}) \leq C^{f^{UE}}(f),$$

where F^{ad} is the set of admissible flows.

Proof

By convexity (f^{UE}) is an optimal solution to (UE) iff

$$\nabla W(f^{UE}) \cdot (f - f^{UE}) \geq 0, \qquad \forall f \in F^{ad}$$

which is equivalent to

$$\sum_{p \in \mathcal{P}} \underbrace{\frac{\partial W}{\partial f_p}(f^{UE})}_{\ell_p(f^{UE})} f_p \quad \geq \quad \sum_{p \in \mathcal{P}} \underbrace{\frac{\partial W}{\partial f_p}(f^{UE})}_{\ell_p(f^{UE})} f_p^{UE}, \qquad \forall f \in F^{ad}$$

which can be written

$$C^{f^{UE}}(f^{UE}) \leq C^{f^{UE}}(f), \qquad \forall f \in F^{ad}.$$

Proof

By convexity (f^{UE}) is an optimal solution to (UE) iff

$$\nabla W(f^{UE}) \cdot (f - f^{UE}) \geq 0, \qquad \forall f \in F^{ad}$$

which is equivalent to

$$\sum_{p \in \mathcal{P}} \underbrace{\frac{\partial W}{\partial f_p}(f^{UE})}_{\ell_p(f^{UE})} f_p \quad \geq \quad \sum_{p \in \mathcal{P}} \underbrace{\frac{\partial W}{\partial f_p}(f^{UE})}_{\ell_p(f^{UE})} f_p^{UE}, \qquad \forall f \in F^{ad}$$

which can be written

$$C^{f^{UE}}(f^{UE}) \leq C^{f^{UE}}(f), \qquad \forall f \in F^{ad}.$$

Proof

By convexity (f^{UE}) is an optimal solution to (UE) iff

$$\nabla W(f^{UE}) \cdot (f - f^{UE}) \geq 0, \qquad \forall f \in F^{ad}$$

which is equivalent to

$$\sum_{p \in \mathcal{P}} \underbrace{\frac{\partial W}{\partial f_p}(f^{UE})}_{\ell_p(f^{UE})} f_p \quad \geq \quad \sum_{p \in \mathcal{P}} \underbrace{\frac{\partial W}{\partial f_p}(f^{UE})}_{\ell_p(f^{UE})} f_p^{UE}, \qquad \forall f \in F^{ad}$$

which can be written

$$C^{f^{UE}}(f^{UE}) \leq C^{f^{UE}}(f), \qquad \forall f \in F^{ad}.$$

Contents

Definition

Definition

Consider increasing loss functions ℓ_e. Let f^{UE} be a user equilibrium, and f^{SO} be a system optimum. Then the price of anarchy of our network is given by

$$PoA := \frac{C(f^{UE})}{C(f^{SO})} \geq 1.$$

Theorem

Let ℓ_e be the affine function $x_e \mapsto b_e x_e + c_e$, with $b_e, c_e \geq 0$. Then the price of anarchy is lower than $4/3$, and the bound is tight.

Proof

Let f be a feasible flow, and f^{UE} be the user equilibrium. For ease of notation we fix $x^{UE} = x(f^{UE})$, and $x = x(f)$.

By Theorem we have

$$
\begin{aligned}
C(f^{UE}) &\leq C^{f^{UE}}(f) \\
&= \sum_{e\in E} (b_e x_e^{UE} + c_e) x_e \\
&\leq \sum_{e\in E} \Big[(b_e x_e + c_e) x_e + \frac{1}{4} b_e (x_e^{UE})^2 \Big] \qquad \text{as } (x_e - x_e^{UE}/2)^2 \geq 0 \\
&\leq C(f) + \frac{1}{4} \sum_{e\in E} (b_e x_e^{UE} + c_e) x_e^{UE} \qquad \text{as } c_e x_e^{UE} \geq 0 \\
&= C(f) + \frac{1}{4} C^{f^{UE}}(f^{UE})
\end{aligned}
$$

Hence we have $3/4 C(f^{UE}) \leq C(f)$.

Minimizing over admissible flow f ends the proof.

Proof

Let f be a feasible flow, and f^{UE} be the user equilibrium. For ease of notation we fix $x^{UE} = x(f^{UE})$, and $x = x(f)$.
By Theorem we have

$$\begin{aligned}
C(f^{UE}) &\leq C^{f^{UE}}(f) \\
&= \sum_{e\in E} (b_e x_e^{UE} + c_e) x_e \\
&\leq \sum_{e\in E} \Big[(b_e x_e + c_e)x_e + \frac{1}{4} b_e (x_e^{UE})^2\Big] && \text{as } (x_e - x_e^{UE}/2)^2 \geq 0 \\
&\leq C(f) + \frac{1}{4}\sum_{e\in E} (b_e x_e^{UE} + c_e) x_e^{UE} && \text{as } c_e x_e^{UE} \geq 0 \\
&= C(f) + \frac{1}{4} C^{f^{UE}}(f^{UE})
\end{aligned}$$

Hence we have $3/4 C(f^{UE}) \leq C(f)$.
Minimizing over admissible flow f ends the proof.

Proof

Let f be a feasible flow, and f^{UE} be the user equilibrium. For ease of notation we fix $x^{UE} = x(f^{UE})$, and $x = x(f)$.
By Theorem we have

$$\begin{aligned}
C(f^{UE}) &\leq C^{f^{UE}}(f) \\
&= \sum_{e \in E} \big(b_e x_e^{UE} + c_e\big) x_e \\
&\leq \sum_{e \in E} \Big[\big(b_e x_e + c_e\big) x_e + \frac{1}{4} b_e \big(x_e^{UE}\big)^2 \Big] && \text{as } (x_e - x_e^{UE}/2)^2 \geq 0 \\
&\leq C(f) + \frac{1}{4} \sum_{e \in E} \big(b_e x_e^{UE} + c_e\big) x_e^{UE} && \text{as } c_e x_e^{UE} \geq 0 \\
&= C(f) + \frac{1}{4} C^{f^{UE}}(f^{UE})
\end{aligned}$$

Hence we have $3/4C(f^{UE}) \leq C(f)$.
Minimizing over admissible flow f ends the proof.

Proof

Let f be a feasible flow, and f^{UE} be the user equilibrium. For ease of notation we fix $x^{UE} = x(f^{UE})$, and $x = x(f)$.
By Theorem we have

$$
\begin{aligned}
C(f^{UE}) &\leq C^{f^{UE}}(f) \\
&= \sum_{e\in E} \big(b_e x_e^{UE} + c_e\big) x_e \\
&\leq \sum_{e\in E} \Big[\big(b_e x_e + c_e\big)x_e + \frac{1}{4} b_e \big(x_e^{UE}\big)^2\Big] && \text{as } (x_e - x_e^{UE}/2)^2 \geq 0 \\
&\leq C(f) + \frac{1}{4}\sum_{e\in E} \big(b_e x_e^{UE} + c_e\big) x_e^{UE} && \text{as } c_e x_e^{UE} \geq 0 \\
&= C(f) + \frac{1}{4} C^{f^{UE}}(f^{UE})
\end{aligned}
$$

Hence we have $3/4C(f^{UE}) \leq C(f)$.
Minimizing over admissible flow f ends the proof.

Proof

Let f be a feasible flow, and f^{UE} be the user equilibrium. For ease of notation we fix $x^{UE} = x(f^{UE})$, and $x = x(f)$.
By Theorem we have

$$\begin{aligned}
C(f^{UE}) &\leq C^{f^{UE}}(f) \\
&= \sum_{e \in E} \big(b_e x_e^{UE} + c_e\big) x_e \\
&\leq \sum_{e \in E} \Big[\big(b_e x_e + c_e\big) x_e + \frac{1}{4} b_e \big(x_e^{UE}\big)^2\Big] \qquad \text{as } (x_e - x_e^{UE}/2)^2 \geq 0 \\
&\leq C(f) + \frac{1}{4} \sum_{e \in E} \big(b_e x_e^{UE} + c_e\big) x_e^{UE} \qquad \text{as } c_e x_e^{UE} \geq 0 \\
&= C(f) + \frac{1}{4} C^{f^{UE}}(f^{UE})
\end{aligned}$$

Hence we have $3/4 C(f^{UE}) \leq C(f)$.
Minimizing over admissible flow f ends the proof.

Proof

Let f be a feasible flow, and f^{UE} be the user equilibrium. For ease of notation we fix $x^{UE} = x(f^{UE})$, and $x = x(f)$.
By Theorem we have

$$
\begin{aligned}
C(f^{UE}) &\leq C^{f^{UE}}(f) \\
&= \sum_{e\in E} \big(b_e x_e^{UE} + c_e\big) x_e \\
&\leq \sum_{e\in E} \Big[\big(b_e x_e + c_e\big) x_e + \frac{1}{4} b_e \big(x_e^{UE}\big)^2\Big] && \text{as } (x_e - x_e^{UE}/2)^2 \geq 0 \\
&\leq C(f) + \frac{1}{4} \sum_{e\in E} \big(b_e x_e^{UE} + c_e\big) x_e^{UE} && \text{as } c_e x_e^{UE} \geq 0 \\
&= C(f) + \frac{1}{4} C^{f^{UE}}(f^{UE})
\end{aligned}
$$

Hence we have $3/4C(f^{UE}) \leq C(f)$.
Minimizing over admissible flow f ends the proof.

Proof

Let f be a feasible flow, and f^{UE} be the user equilibrium. For ease of notation we fix $x^{UE} = x(f^{UE})$, and $x = x(f)$.
By Theorem we have

$$\begin{aligned} C(f^{UE}) &\leq C^{f^{UE}}(f) \\ &= \sum_{e\in E} \big(b_e x_e^{UE} + c_e\big)x_e \\ &\leq \sum_{e\in E} \Big[\big(b_e x_e + c_e\big)x_e + \frac{1}{4}b_e\big(x_e^{UE}\big)^2\Big] && \text{as } (x_e - x_e^{UE}/2)^2 \geq 0 \\ &\leq C(f) + \frac{1}{4}\sum_{e\in E}\big(b_e x_e^{UE} + c_e\big)x_e^{UE} && \text{as } c_e x_e^{UE} \geq 0 \\ &= C(f) + \frac{1}{4}C^{f^{UE}}(f^{UE}) \end{aligned}$$

Hence we have $3/4C(f^{UE}) \leq C(f)$.
Minimizing over admissible flow f ends the proof.

Pigou's Example

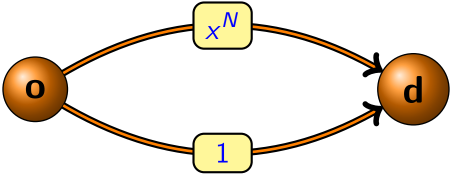


Figure: Pigou example

On a graph with two nodes: one origin, one destination, a total flow of 1, a fixed cost of 1 on one edge, and a cost of x^N on the other, where $N \in \mathbb{N}$ and x is the intensity of the flow using this edge (see Figure 1).

1. Compute the system optimum for a given N.
2. Compute the user equilibrium for a given N.
3. Compute the price of anarchy on this network when $N \to \infty$.

Exercise for next week (3.2)

Consider a (finite) directed, strongly connected, graph $G = (V, E)$. We consider K origin-destination vertex pair $\{o^k, d^k\}_{k \in [\![1,K]\!]}$, such that there exists at least one path from o^k to d^k.

We want to find bounds on the price of anarchy, assuming that, for each arc e, $\ell_e : \mathbb{R}^+ \to \mathbb{R}^+$ is non-decreasing, and that we have

$$x\ell_e(x) \leq \gamma L_e(x), \qquad \forall x \in \mathbb{R}^+$$

1. Recall which optimization problems solves the social optimum x^{SO} and the user equilibrium x^{UE}.
2. Let x be a feasable vector of arc-intensity. Show that $W(x) \leq C(x) \leq \gamma W(x)$.
3. Show that the price of anarchy $C(x^{UE})/C(x^{SO})$ is lower than γ.
4. If the cost per arc ℓ_e are polynomial of order at most p with non-negative coefficient, find a bound on the price of anarchy. Is this bound sharp ?

Further video content

This is a research seminar by one of the expert in the domain. The first half is very interesting to get a better intuition of the concepts. The second half is more dedicated to the proof of the result presented in the talk.

https://www.youtube.com/watch?v=e3O_tMsN2t8

Numerical Methods

V. Leclère (ENPC)

May 6th, 2020

Contents

The set-up

- $G = (V, E)$ is a directed graph
- x_e for $e \in E$ represent the flux (number of people per hour) taking edge e
- $\ell_e : \mathbb{R} \to \mathbb{R}^+$ the cost incurred by a given user to take edge e
- We consider K origin-destination vertex pair $\{o^k, d^k\}_{k \in [\![1,K]\!]}$, such that there exists at least one path from o^k to d^k.
- r_k is the rate of people going from o^k to d^k
- $\mathcal{P}_k$ the set of all simple (i.e. without cycle) path form o^k to d^k
- We denote f_p the flux of people taking path $p \in \mathcal{P}_k$

Some physical relations

People going from o^k to d^k have to choose a path

$$r^k = \sum_{p \in \mathcal{P}^k} f_p.$$

People going through an edge are on a simple path taking this edge

$$x_e = \sum_{p \ni e} f_p.$$

The flux are non-negative

$$\forall p \in \mathcal{P}, \quad f_p \geq 0, \qquad \text{and} \qquad , \forall e \in E, \quad x_e \geq 0$$

Some physical relations

People going from o^k to d^k have to choose a path

$$r^k = \sum_{p \in \mathcal{P}^k} f_p.$$

People going through an edge are on a simple path taking this edge

$$x_e = \sum_{p \ni e} f_p.$$

The flux are non-negative

$$\forall p \in \mathcal{P}, \quad f_p \geq 0, \qquad \text{and} \qquad , \forall e \in E, \quad x_e \geq 0$$

Some physical relations

People going from o^k to d^k have to choose a path

$$r^k = \sum_{p \in \mathcal{P}^k} f_p.$$

People going through an edge are on a simple path taking this edge

$$x_e = \sum_{p \ni e} f_p.$$

The flux are non-negative

$$\forall p \in \mathcal{P}, \quad f_p \geq 0, \qquad \text{and} \qquad , \forall e \in E, \quad x_e \geq 0$$

System optimum problem

The system optimum consists in minimizing the sum of all costs over the admissible flux $x = (x_e)_{e \in E}$

- Given x, the cost of taking edge e for one person is $\ell_e(x_e)$.
- The cost for the system for edge e is thus $x_e \ell_e(x_e)$.
- Thus minimizing the system costs consists in solving

$$\begin{aligned}
\min_{x,f} \quad & \sum_{e \in E} x_e \ell_e(x_e) && (SO) \\
s.t. \quad & r_k = \sum_{p \in \mathcal{P}_k} f_p && k \in [\![1, K]\!] \\
& x_e = \sum_{p \ni e} f_p && e \in E \\
& f_p \geq 0 && p \in \mathcal{P}
\end{aligned}$$

System optimum problem

The system optimum consists in minimizing the sum of all costs over the admissible flux $x = (x_e)_{e \in E}$

- Given x, the cost of taking edge e for one person is $\ell_e(x_e)$.
- The cost for the system for edge e is thus $x_e \ell_e(x_e)$.
- Thus minimizing the system costs consists in solving

$$\begin{aligned}
\min_{x,f} \quad & \sum_{e \in E} x_e \ell_e(x_e) & & (SO) \\
s.t. \quad & r_k = \sum_{p \in \mathcal{P}_k} f_p & & k \in [\![1, K]\!] \\
& x_e = \sum_{p \ni e} f_p & & e \in E \\
& f_p \geq 0 & & p \in \mathcal{P}
\end{aligned}$$

System optimum problem

The system optimum consists in minimizing the sum of all costs over the admissible flux $x = (x_e)_{e \in E}$

- Given x, the cost of taking edge e for one person is $\ell_e(x_e)$.
- The cost for the system for edge e is thus $x_e \ell_e(x_e)$.
- Thus minimizing the system costs consists in solving

$$
\begin{aligned}
\min_{x,f} \quad & \sum_{e \in E} x_e \ell_e(x_e) & & (SO) \\
s.t. \quad & r_k = \sum_{p \in \mathcal{P}_k} f_p & & k \in [\![1, K]\!] \\
& x_e = \sum_{p \ni e} f_p & & e \in E \\
& f_p \geq 0 & & p \in \mathcal{P}
\end{aligned}
$$

Path intensity formulation

- We can reformulate the (SO) problem only using path-intensity $f = (f_p)_{p\in\mathcal{P}}$.
- Define $x_e(f) := \sum_{p\ni e} f_p$, and $x = (x_e)_{e\in E}$.
- Define the loss along a path $\ell_p(f) = \sum_{e\in p} \ell_e\big(\underbrace{\sum_{p'\ni e} f_{p'}}_{x_e(f)}\big)$
- The total cost is thus

$$C(f) = \sum_{p\in\mathcal{P}} f_p \ell_p(f) = \sum_{e\in E} x_e \ell_e(x_e(f)) = C(x(f)).$$

Path intensity formulation

- We can reformulate the (SO) problem only using path-intensity $f = (f_p)_{p \in \mathcal{P}}$.
- Define $x_e(f) := \sum_{p \ni e} f_p$, and $x = (x_e)_{e \in E}$.
- Define the loss along a path $\ell_p(f) = \sum_{e \in p} \ell_e\big(\underbrace{\sum_{p' \ni e} f_{p'}}_{x_e(f)}\big)$
- The total cost is thus

$$C(f) = \sum_{p \in \mathcal{P}} f_p \ell_p(f) = \sum_{e \in E} x_e \ell_e(x_e(f)) = C(x(f)).$$

Path intensity formulation

- We can reformulate the (SO) problem only using path-intensity $f = (f_p)_{p \in \mathcal{P}}$.
- Define $x_e(f) := \sum_{p \ni e} f_p$, and $x = (x_e)_{e \in E}$.
- Define the loss along a path $\ell_p(f) = \sum_{e \in p} \ell_e\big(\underbrace{\sum_{p' \ni e} f_{p'}}_{x_e(f)}\big)$
- The total cost is thus

$$C(f) = \sum_{p \in \mathcal{P}} f_p \ell_p(f) = \sum_{e \in E} x_e \ell_e(x_e(f)) = C(x(f)).$$

Path intensity formulation

- We can reformulate the (SO) problem only using path-intensity $f = (f_p)_{p\in\mathcal{P}}$.
- Define $x_e(f) := \sum_{p\ni e} f_p$, and $x = (x_e)_{e\in E}$.
- Define the loss along a path $\ell_p(f) = \sum_{e\in p} \ell_e\big(\underbrace{\sum_{p'\ni e} f_{p'}}_{x_e(f)}\big)$
- The total cost is thus

$$C(f) = \sum_{p\in\mathcal{P}} f_p \ell_p(f) = \sum_{e\in E} x_e \ell_e(x_e(f)) = C(x(f)).$$

Path intensity problem

$$
\begin{aligned}
\min_{f} \quad & \sum_{p \in \mathcal{P}} f_p \ell_p(f) && (SO) \\
s.t. \quad & r_k = \sum_{p \in \mathcal{P}_k} f_p && k \in [\![1, K]\!] \\
& f_p \geq 0 && p \in \mathcal{P}
\end{aligned}
$$

Contents

Equilibrium definition

John Wardrop defined a traffic equilibrium as follows. "Under equilibrium conditions traffic arranges itself in congested networks such that all used routes between an O-D pair have equal and minimum costs, while all unused routes have greater or equal costs."

A mathematical definition reads as follows.

Definition

A user flow f is a User Equilibrium if

$$\forall k \in [\![1, K]\!], \quad \forall (p, p') \in \mathcal{P}_k^2, \qquad f_p > 0 \quad \Longrightarrow \quad \ell_p(f) \leq \ell_{p'}(f).$$

Equilibrium definition

John Wardrop defined a traffic equilibrium as follows. "Under equilibrium conditions traffic arranges itself in congested networks such that all used routes between an O-D pair have equal and minimum costs, while all unused routes have greater or equal costs."
A mathematical definition reads as follows.

Definition

A user flow f is a User Equilibrium if

$$\forall k \in [\![1, K]\!], \quad \forall (p, p') \in \mathcal{P}_k^2, \qquad f_p > 0 \quad \Longrightarrow \quad \ell_p(f) \leq \ell_{p'}(f).$$

A new cost function

We are going to show that a user-equilibrium f is defined as a vector satisfying the KKT conditions of a certain optimization problem.

Let define a new edge-loss function by

$$L_e(x_e) := \int_0^{x_e} \ell_e(u)du.$$

The Wardrop potential is defined (for edge intensity) as

$$W(f) = W(x(f)) = \sum_{e \in E} L_e(x_e(f)).$$

A new cost function

We are going to show that a user-equilibrium f is defined as a vector satisfying the KKT conditions of a certain optimization problem.

Let define a new edge-loss function by

$$L_e(x_e) := \int_0^{x_e} \ell_e(u)du.$$

The Wardrop potential is defined (for edge intensity) as

$$W(f) = W(x(f)) = \sum_{e \in E} L_e(x_e(f)).$$

User optimum problem

Theorem

A flow f *is a user equilibrium if and only if it satisfies the first order KKT conditions of the following optimization problem*

$$
\begin{aligned}
\min_{x,f} \quad & W(x) & \\
s.t. \quad & r_k = \sum_{p \in \mathcal{P}_k} f_p & k \in [\![1, K]\!] \\
& x_e = \sum_{p \ni e} f_p & e \in E \\
& f_p \geq 0 & p \in \mathcal{P}
\end{aligned}
$$

Convex case : equivalence

If the loss functions (in edge-intensity) are non-decreasing then the Wardrop potential W is convex.

Theorem

Assume that the loss function ℓ_e are non-decreasing for all $e \in E$. Then there exists at least one user equilibrium, and a flow f is a user equilibrium if and only if it solves (UE)

Contents

Descent methods

Consider the unconstrained optimization problem

$$\min_{x \in \mathbb{R}^n} \quad f(x). \tag{2}$$

A *descent direction algorithm* is an algorithm that constructs a sequence of points $(x^{(k)})_{k \in \mathbb{N}}$, that are recursively defined with:

$$x^{(k+1)} = x^{(k)} + t^{(k)} d^{(k)} \tag{3}$$

where

- $x^{(0)}$ is the initial point,
- $d^{(k)} \in \mathbb{R}^n$ is the descent direction,
- $t^{(k)}$ is the step length.

Descent methods

Consider the unconstrained optimization problem

$$\min_{x\in\mathbb{R}^n} \quad f(x). \tag{2}$$

A *descent direction algorithm* is an algorithm that constructs a sequence of points $(x^{(k)})_{k\in\mathbb{N}}$, that are recursively defined with:

$$x^{(k+1)} = x^{(k)} + t^{(k)}d^{(k)} \tag{3}$$

where

- $x^{(0)}$ is the initial point,
- $d^{(k)} \in \mathbb{R}^n$ is the descent direction,
- $t^{(k)}$ is the step length.

Descent methods

Consider the unconstrained optimization problem

$$\min_{x \in \mathbb{R}^n} \quad f(x). \tag{2}$$

A *descent direction algorithm* is an algorithm that constructs a sequence of points $(x^{(k)})_{k\in\mathbb{N}}$, that are recursively defined with:

$$x^{(k+1)} = x^{(k)} + t^{(k)} d^{(k)} \tag{3}$$

where

- $x^{(0)}$ is the initial point,
- $d^{(k)} \in \mathbb{R}^n$ is the descent direction,
- $t^{(k)}$ is the step length.

Video explanation

https://www.youtube.com/watch?v=n-YOSDSOfUI

Descent direction

For a differentiable objective function f, $d^{(k)}$ will be a descent direction iff $\nabla f(x^{(k)}) \cdot d^{(k)} \leq 0$, which can be seen from a first order development:

$$f(x^{(k)} + t^{(k)} d^{(k)}) = f(x^{(k)}) + t\langle \nabla f(x^{(k)}) \, , d^{(k)} \rangle + o(t).$$

The most classical descent direction is $d^{(k)} = -\nabla f(x^{(k)})$, which correspond to the gradient algorithm.

Descent direction

For a differentiable objective function f, $d^{(k)}$ will be a descent direction iff $\nabla f(x^{(k)}) \cdot d^{(k)} \leq 0$, which can be seen from a first order development:

$$f(x^{(k)} + t^{(k)} d^{(k)}) = f(x^{(k)}) + t \langle \nabla f(x^{(k)}), d^{(k)} \rangle + o(t).$$

The most classical descent direction is $d^{(k)} = -\nabla f(x^{(k)})$, which correspond to the gradient algorithm.

Step-size choice

The step-size $t^{(k)}$ can be:

- fixed $t^{(k)} = t^{(0)}$, for all iteration,
- optimal $t^{(k)} \in \arg\min_{t \geq 0} f(x^{(k)} + td^{(k)})$,
- a "good" step, following some rules (e.g Armijo's rules).

Finding the optimal step size is a special case of unidimensional optimization (or linear search).

Step-size choice

The step-size $t^{(k)}$ can be:

- fixed $t^{(k)} = t^{(0)}$, for all iteration,
- optimal $t^{(k)} \in \arg\min_{t \geq 0} f(x^{(k)} + td^{(k)})$,
- a "good" step, following some rules (e.g Armijo's rules).

Finding the optimal step size is a special case of unidimensional optimization (or linear search).

Contents

Unidimensional optimization

We assume that the objective function $J : \mathbb{R} \to \mathbb{R}$ is strictly convex.
We are going to consider two types of methods:

- interval reduction algorithms: constructing $[a^{(l)}, b^{(l)}]$ containing the optimal point;
- successive approximation algorithms: approximating J and taking the minimum of the approximation.

Bisection method

We assume that J is differentiable over $[a, b]$. Note that, for $c \in [a, b]$, $t* < c$ iff $J'(c) > 0$. From this simple remark we construct the bisection method.

```
while b^(l) − a^(l) > ε do
    c^(l) = (b^(l) + a^(l)) / 2 ;
    if J'(c^(l)) > 0 then
        a^(l+1) = a^(l) ; b^(l+1) = c^(l) ;
    else if J'(c^(l)) < 0 then
        a^(l+1) = c^(l) ; b^(l+1) = b^(l) ;
    else
        return interval [a^(l), b^(l)]
    l = l + 1
```

Note that $L_l = b^{(l)} - a^{(l)} = \frac{L_0}{2^l}$.

Golden section

Consider $a < t_1 < t_2 < b$, we are looking for $t^* = \arg\min_{t \in [a,b]} J(t)$

Note that

- if $J(t_1) < J(t_2)$, then $t^* \in [a, t_2]$;
- if $J(t_1) > J(t_2)$, then $t^* \in [t_1, b]$;
- if $J(t_1) = J(t_2)$, then $t^* \in [t_1, t_2]$.

Hence, at each iteration the interval $[a^{(l)}, b^{(l)}]$ is updated into $[a^{(l)}, t_2^{(l)}]$ or $[t_1^{(l)}, b^{(l)}]$.

Golden section I

Consider $a < t_1 < t_2 < b$, we are looking for $t^* = \arg\min_{t \in [a,b]} J(t)$

Note that

- if $J(t_1) < J(t_2)$, then $t^* \in [a, t_2]$;
- if $J(t_1) > J(t_2)$, then $t^* \in [t_1, b]$;
- if $J(t_1) = J(t_2)$, then $t^* \in [t_1, t_2]$.

Hence, at each iteration the interval $[a^{(l)}, b^{(l)}]$ is updated into $[a^{(l)}, t_2^{(l)}]$ or $[t_1^{(l)}, b^{(l)}]$.

Golden section II

We now want to know how to choose $t_1^{(l)}$ and $t_2^{(l)}$. To minimize the worst case complexity we want equity between both possibility, hence $b^{(l)} - t_1^{(l)} = t_2^{(l)} - a^{(l)}$. Now assume that $J(t_1^{(l)}) < J(t_2^{(l)})$. Hence $a^{(l+1)} = a^{(l)}$, and $b^{(l+1)} = t_2$. We would like to reuse the computation of $J(t_1^{(l)})$ by defining $t_1^{(k+1)} = t_2^{(l)}$.

In order to satisfy this constraint we need to have

$$\begin{cases} L_2 + L_1 = L \\ \dfrac{L_2}{L} = \dfrac{L_1}{L_2} =: R \end{cases} \tag{4}$$

where $L = b^{(l)} - a^{(l)}$, $L_1 = t_1^{(l)} - a^{(l)}$ and $L_2 = t_2^{(l)} - a^{(l)}$.

This implies

$$1 + R = \frac{1}{R} \tag{5}$$

Golden section II

We now want to know how to choose $t_1^{(l)}$ and $t_2^{(l)}$. To minimize the worst case complexity we want equity between both possibility, hence $b^{(l)} - t_1^{(l)} = t_2^{(l)} - a^{(l)}$. Now assume that $J(t_1^{(l)}) < J(t_2^{(l)})$. Hence $a^{(l+1)} = a^{(l)}$, and $b^{(l+1)} = t_2$. We would like to reuse the computation of $J(t_1^{(l)})$ by defining $t_1^{(k+1)} = t_2^{(l)}$.
In order to satisfy this constraint we need to have

$$\begin{cases} L_2 + L_1 = L \\ \dfrac{L_2}{L} = \dfrac{L_1}{L_2} =: R \end{cases} \tag{4}$$

where $L = b^{(l)} - a^{(l)}$, $L_1 = t_1^{(l)} - a^{(l)}$ and $L_2 = t_2^{(l)} - a^{(l)}$.
This implies

$$1 + R = \frac{1}{R} \tag{5}$$

Golden section II

We now want to know how to choose $t_1^{(l)}$ and $t_2^{(l)}$. To minimize the worst case complexity we want equity between both possibility, hence $b^{(l)} - t_1^{(l)} = t_2^{(l)} - a^{(l)}$. Now assume that $J(t_1^{(l)}) < J(t_2^{(l)})$. Hence $a^{(l+1)} = a^{(l)}$, and $b^{(l+1)} = t_2$. We would like to reuse the computation of $J(t_1^{(l)})$ by defining $t_1^{(k+1)} = t_2^{(l)}$.
In order to satisfy this constraint we need to have

$$\begin{cases} L_2 + L_1 = L \\ \frac{L_2}{L} = \frac{L_1}{L_2} =: R \end{cases} \tag{4}$$

where $L = b^{(l)} - a^{(l)}$, $L_1 = t_1^{(l)} - a^{(l)}$ and $L_2 = t_2^{(l)} - a^{(l)}$.
This implies

$$1 + R = \frac{1}{R} \tag{5}$$

Golden section III

$$R = \frac{\sqrt{5}-1}{2}. \tag{6}$$

Finally, in order to satisfy equity and reusability it is enough to set

$$t_1^{(l)} = a^{(l)} + (1-R)(b^{(l)} - a^{(l)})$$
$$t_1^{(l)} = a^{(l)} + R(b^{(l)} - a^{(l)})$$

The same happens for the $J(t_1^{(l)}) > J(t_2^{(l)})$ case.

Golden section algorithm

$a^{(0)} = a, \quad b^{(0)} = b;$

$t_1^{(0)} = a + (1-R)b, \quad t_2^{(0)} = a + Rb;$

$J_1 = J(t_1^{(0)}), \quad J_2 = J(t_2^{(0)});$

while $b^{(l)} - a^{(l)} > \varepsilon$ **do**

  if $J_1 < J_2$ **then**

    $a^{(l+1)} = a^{(l)}$; $b^{(l+1)} = t_2^{(l)}$;

    $t_1^{(l+1)} = a^{(l+1)} + (1-R)b^{(l+1)}$; $t_2^{(l+1)} = t_1^{(}l)$;

    $J_2 = J_1;$

    $J_1 = J(t_1^{(l+1)});$

  else

    $a^{(l+1)} = t_1^{(l)}$; $b^{(l+1)} = b^{(l)}$;

    $t_1^{(l+1)} = t_2^{(l)}$; $t_2^{(l+1)} = a^{(l+1)} + Rb^{(l+1)}$;

    $J_1 = J_2;$

    $J_2 = J(t_2^{(l+1)});$

  $l = l + 1$

Note that $L_l = R^l L_0$.

Video explantion

Golden section

https://www.youtube.com/watch?v=6NYp3td3cjU

Curve fitting : Newton method

If J is twice-differentiable (with non-null second order derivative) is to determine $t^{(k+1)}$ as the minimum of the second order Taylor's of J at $t^{(k)}$:

$$t^{(l+1)} - t^{(l)} = \arg\min_t J(t^{(l)}) + J'(t^{(l)})t + \frac{t^2}{2} J''(t^{(l)})$$
$$= \big(J''(t^{(l)})\big)^{-1} J'(t^{(l)})$$

This is the well known, and very efficient, Newton method.

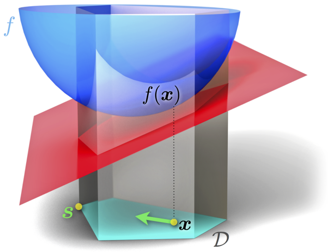
Conditional gradient algorithm

We address an optimization problem with convex objective function f and compact polyhedral constraint set X, i.e.

$$\min_{x \in X \subset \mathbb{R}^n} \quad f(x)$$

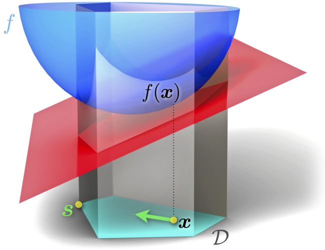
where

$$X = \{x \in \mathbb{R}^n \quad | \quad Ax \leq b, \quad \tilde{A}x = \tilde{b}\}$$



Conditional gradient algorithm

It is a descent algorithm, where we first look for an admissible descent direction $d^{(k)}$, and then look for the optimal step.

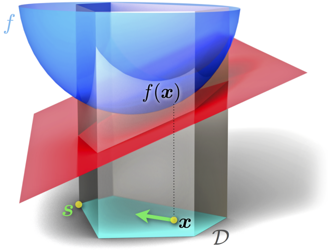


Conditional gradient algorithm

It is a descent algorithm, where we first look for an admissible descent direction $d^{(k)}$, and then look for the optimal step.

As f is convex, we know that for any point $x^{(k)}$,

$$f(y) \geq f(x^{(k)}) + \nabla f(x^{(k)}) \cdot (y - x^{(k)})$$

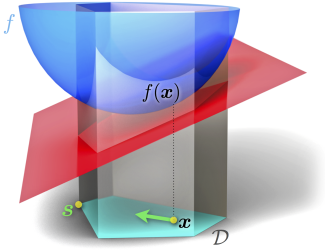


Conditional gradient algorithm

It is a descent algorithm, where we first look for an admissible descent direction $d^{(k)}$, and then look for the optimal step.
As f is convex, we know that for any point $x^{(k)}$,

$$f(y) \geq f(x^{(k)}) + \nabla f(x^{(k)}) \cdot (y - x^{(k)})$$

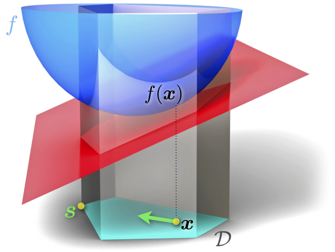
The conditional gradient method consists in choosing the descent direction that minimize the linearization of f over X.



Conditional gradient algorithm

The conditional gradient method consists in choosing the descent direction that minimize the linearization of f over X. More precisely, at step k we solve

$$y^{(k)} \in \arg\min_{y \in X} \quad f(x^{(k)}) + \nabla f(x^{(k)}) \cdot (y -$$



Remarks on conditional gradient

$$y^{(k)} \in \arg\min_{y \in X} \quad f(x^{(k)}) + \nabla f(x^{(k)}) \cdot (y - x^{(k)}).$$

- This problem is linear, hence easy to solve.
- By the convexity inequality, the value of the linearized Problem is a lower bound to the true problem.
- As $y^{(k)} \in X$, $d^{(k)} = y^{(k)} - x^{(k)}$ is a *feasable direction*, in the sense that for all $t \in [0,1]$, $x^{(k)} + td^{(k)} \in X$.
- If $y^{(k)}$ is obtained through the simplex method it is an extreme point of X, which means that, for $t > 1$, $x^{(k)} + td^{(k)} \notin X$.
- If $y^{(k)} = x^{(k)}$ then we have found an optimal solution.
- We also have $y^{(k)} \in \arg\min_{x \in X} \nabla f(x^{(k)}) \cdot y$, the lower-bound being obtained easily.

Remarks on conditional gradient

$$y^{(k)} \in \arg\min_{y \in X} \quad f(x^{(k)}) + \nabla f(x^{(k)}) \cdot (y - x^{(k)}).$$

- This problem is linear, hence easy to solve.
- By the convexity inequality, the value of the linearized Problem is a lower bound to the true problem.
- As $y^{(k)} \in X$, $d^{(k)} = y^{(k)} - x^{(k)}$ is a *feasable direction*, in the sense that for all $t \in [0, 1]$, $x^{(k)} + td^{(k)} \in X$.
- If $y^{(k)}$ is obtained through the simplex method it is an extreme point of X, which means that, for $t > 1$, $x^{(k)} + td^{(k)} \notin X$.
- If $y^{(k)} = x^{(k)}$ then we have found an optimal solution.
- We also have $y^{(k)} \in \arg\min_{x \in X} \nabla f(x^{(k)}) \cdot y$, the lower-bound being obtained easily.

Remarks on conditional gradient

$$y^{(k)} \in \arg\min_{y \in X} \quad f(x^{(k)}) + \nabla f(x^{(k)}) \cdot (y - x^{(k)}).$$

- This problem is linear, hence easy to solve.
- By the convexity inequality, the value of the linearized Problem is a lower bound to the true problem.
- As $y^{(k)} \in X$, $d^{(k)} = y^{(k)} - x^{(k)}$ is a *feasable direction*, in the sense that for all $t \in [0, 1]$, $x^{(k)} + td^{(k)} \in X$.
- If $y^{(k)}$ is obtained through the simplex method it is an extreme point of X, which means that, for $t > 1$, $x^{(k)} + td^{(k)} \notin X$.
- If $y^{(k)} = x^{(k)}$ then we have found an optimal solution.
- We also have $y^{(k)} \in \arg\min_{x \in X} \nabla f(x^{(k)}) \cdot y$, the lower-bound being obtained easily.

Remarks on conditional gradient

$$y^{(k)} \in \arg\min_{y \in X} \quad f(x^{(k)}) + \nabla f(x^{(k)}) \cdot (y - x^{(k)}).$$

- This problem is linear, hence easy to solve.
- By the convexity inequality, the value of the linearized Problem is a lower bound to the true problem.
- As $y^{(k)} \in X$, $d^{(k)} = y^{(k)} - x^{(k)}$ is a *feasable direction*, in the sense that for all $t \in [0,1]$, $x^{(k)} + td^{(k)} \in X$.
- If $y^{(k)}$ is obtained through the simplex method it is an extreme point of X, which means that, for $t > 1$, $x^{(k)} + td^{(k)} \notin X$.
- If $y^{(k)} = x^{(k)}$ then we have found an optimal solution.
- We also have $y^{(k)} \in \arg\min_{x \in X} \nabla f(x^{(k)}) \cdot y$, the lower-bound being obtained easily.

Remarks on conditional gradient

$$y^{(k)} \in \arg\min_{y \in X} \quad f(x^{(k)}) + \nabla f(x^{(k)}) \cdot (y - x^{(k)}).$$

- This problem is linear, hence easy to solve.
- By the convexity inequality, the value of the linearized Problem is a lower bound to the true problem.
- As $y^{(k)} \in X$, $d^{(k)} = y^{(k)} - x^{(k)}$ is a *feasable direction*, in the sense that for all $t \in [0, 1]$, $x^{(k)} + td^{(k)} \in X$.
- If $y^{(k)}$ is obtained through the simplex method it is an extreme point of X, which means that, for $t > 1$, $x^{(k)} + td^{(k)} \notin X$.
- If $y^{(k)} = x^{(k)}$ then we have found an optimal solution.
- We also have $y^{(k)} \in \arg\min_{x \in X} \nabla f(x^{(k)}) \cdot y$, the lower-bound being obtained easily.

Remarks on conditional gradient

$$y^{(k)} \in \arg\min_{y \in X} \quad f(x^{(k)}) + \nabla f(x^{(k)}) \cdot (y - x^{(k)}).$$

- This problem is linear, hence easy to solve.
- By the convexity inequality, the value of the linearized Problem is a lower bound to the true problem.
- As $y^{(k)} \in X$, $d^{(k)} = y^{(k)} - x^{(k)}$ is a *feasable direction*, in the sense that for all $t \in [0,1]$, $x^{(k)} + td^{(k)} \in X$.
- If $y^{(k)}$ is obtained through the simplex method it is an extreme point of X, which means that, for $t > 1$, $x^{(k)} + td^{(k)} \notin X$.
- If $y^{(k)} = x^{(k)}$ then we have found an optimal solution.
- We also have $y^{(k)} \in \arg\min_{x \in X} \nabla f(x^{(k)}) \cdot y$, the lower-bound being obtained easily.

Frank Wolfe algorithm

Data: objective function f, constraints, initial point $x^{(0)}$, precision ε
Result: ε-optimal solution $x^{(k)}$, upperbound $f(x^{(k)})$, lowerbound $\underline{f}$
$\underline{f} = -\infty$;
$k = 0$;

while $f(x^{(k)}) - \underline{f} > \varepsilon$ **do**

- solve the LP $\min_{y \in X} f(x^{(k)}) + \nabla f(x^{(k)}) \cdot (y - x^{(k)})$;
- let $y^{(k)}$ be an optimal solution, and $\underline{f}$ the optimal value ;
- set $d^{(k)} = y^{(k)} - x^{(k)}$;
- solve $t^{(k)} \in \arg\min_{t \in [0,1]} f\left(x^{(k)} + td^{(k)}\right)$;
- update $x^{(k+1)} = x^{(k)} + t^{(k)} d^{(k)}$;
- $k = k + 1$;

Contents

All-or nothing

A very simple heuristic consists in:

1. Set $k = 0$.
2. Assume initial cost per edge $\ell_e^{(k)} = \ell_e(x_e^{ref})$.
3. For each origin-destination pair (o_i, d_i) find the shortest path associated with $\ell^{(k)}$.
4. Associate the full flow r_i to this path, which form a flow of user $f^{(k)}$.
5. Deducing the travel cost per edge is $\ell_e^{(k+1)} = \ell_e(f^{(k)})$.
6. Go to step 3.

This method is simple and requires only to compute the shortest path in a fixed cost graph.

However it is not converging as it can cycle.

All-or nothing

A very simple heuristic consists in:

1. Set $k = 0$.
2. Assume initial cost per edge $\ell_e^{(k)} = \ell_e(x_e^{ref})$.
3. For each origin-destination pair (o_i, d_i) find the shortest path associated with $\ell^{(k)}$.
4. Associate the full flow r_i to this path, which form a flow of user $f^{(k)}$.
5. Deducing the travel cost per edge is $\ell_e^{(k+1)} = \ell_e(f^{(k)})$.
6. Go to step 3.

This method is simple and requires only to compute the shortest path in a fixed cost graph.

However it is not converging as it can cycle.

All-or nothing

A very simple heuristic consists in:

1. Set $k = 0$.
2. Assume initial cost per edge $\ell_e^{(k)} = \ell_e(x_e^{ref})$.
3. For each origin-destination pair (o_i, d_i) find the shortest path associated with $\ell^{(k)}$.
4. Associate the full flow r_i to this path, which form a flow of user $f^{(k)}$.
5. Deducing the travel cost per edge is $\ell_e^{(k+1)} = \ell_e(f^{(k)})$.
6. Go to step 3.

This method is simple and requires only to compute the shortest path in a fixed cost graph.
However it is not converging as it can cycle.

Smoothed all-or-nothing

The all-or-nothing method can be understood as follow: each day every user choose the shortest path according to the traffice on the previous day. We can smooth the approach by saying that only a fraction ρ of user is going to update its path from one day to the next.

Hence the smoothed all-or-nothing approach reads

1. Set $k = 0$.
2. Assume initial cost per arc $\ell_e^{(k)} = \ell_e(x_e^{ref})$.
3. For each pair origin destination (o_i, d_i) find the shortest path associated with $\ell^{(k)}$.
4. Associate the full flow r_i to this path, which form a flow of user $\tilde{f}^{(k)}$.
5. Compute the new flow $f^{(k)} = (1-\rho)f^{(k-1)} + \rho\tilde{f}^{(k)}$.
6. Deducing the travel cost per arc as $\ell_e^{(k+1)} = \ell_e(f^{(k)})$.
7. Go to step 3.

Smoothed all-or-nothing

The all-or-nothing method can be understood as follow: each day every user choose the shortest path according to the traffice on the previous day. We can smooth the approach by saying that only a fraction ρ of user is going to update its path from one day to the next.

Hence the smoothed all-or-nothing approach reads

1. Set $k = 0$.
2. Assume initial cost per arc $\ell_e^{(k)} = \ell_e(x_e^{ref})$.
3. For each pair origin destination (o_i, d_i) find the shortest path associated with $\ell^{(k)}$.
4. Associate the full flow r_i to this path, which form a flow of user $\tilde{f}^{(k)}$.
5. Compute the new flow $f^{(k)} = (1 - \rho)f^{(k-1)} + \rho\tilde{f}^{(k)}$.
6. Deducing the travel cost per arc as $\ell_e^{(k+1)} = \ell_e(f^{(k)})$.
7. Go to step 3.

Contents

UE problem

Recall that, if the arc-cost functions are non-decreasing finding a user-equilibrium is equivalent to solving

$$\begin{aligned} \min_{f \geq 0} \quad & W(x(f)) \\ s.t. \quad & r_k = \sum_{p \in \mathcal{P}_k} f_p & k \in [\![1, K]\!] \end{aligned}$$

where

$$W(f) = W(x(f)) = \sum_{e \in E} L_e(x_e(f)),$$

with

$$L_e(x_e) := \int_0^{x_e} \ell_e(u) du,$$

and

$$x_e(f) = \sum_{p \ni e} f_p.$$

Frank-Wolfe for UE I

Let's compute the linearization of the objective function. Consider an admissible flow $f^{(\kappa)}$ and a path $p \in \mathcal{P}_i$. We have

$$\frac{\partial W \circ x}{\partial f_p}(f^{(\kappa)}) = \frac{\partial}{\partial f_p}\Big(\sum_{e \in E} L_e(\sum_{p' \ni e} f^{(\kappa)}_{p'})\Big)$$
$$= \sum_{e \in p} \frac{\partial}{\partial x_e} L_e(x_e(f^{(\kappa)}))$$
$$= \sum_{e \in p} \ell_e(x_e(f^{(\kappa)}) = \ell_p(f^{(\kappa)}).$$

Hence, the linearized problem around $f^{(k)}$ reads

$$\min_{\{y_p\}_{p \in \mathcal{P}}} \quad \sum_{p \in \mathcal{P}} y_p \ell_p(f^{(\kappa)})$$
$$s.t \quad r_k = \sum_{p \in \mathcal{P}_k} y_p \qquad k \in [\![1, K]\!]$$

Frank-Wolfe for UE

I

Let's compute the linearization of the objective function. Consider an admissible flow $f^{(\kappa)}$ and a path $p \in \mathcal{P}_i$. We have

$$
\begin{aligned}
\frac{\partial W \circ x}{\partial f_p}(f^{(\kappa)}) &= \frac{\partial}{\partial f_p}\Big(\sum_{e \in E} L_e\big(\sum_{p' \ni e} f_{p'}^{(\kappa)}\big)\Big) \\
&= \sum_{e \in p} \frac{\partial}{\partial x_e} L_e(x_e(f^{(\kappa)})) \\
&= \sum_{e \in p} \ell_e(x_e(f^{(\kappa)}) = \ell_p(f^{(\kappa)}).
\end{aligned}
$$

Hence, the linearized problem around $f^{(k)}$ reads

$$
\begin{aligned}
\min_{\{y_p\}_{p \in \mathcal{P}}} \quad & \sum_{p \in \mathcal{P}} y_p \ell_p(f^{(\kappa)}) \\
s.t \quad & r_k = \sum_{p \in \mathcal{P}_k} y_p \qquad\qquad k \in [\![1, K]\!]
\end{aligned}
$$

Frank-Wolfe for UE II

$$\begin{aligned}
\min_{\{y_p\}_{p\in\mathcal{P}}} \quad & \sum_{p\in\mathcal{P}} y_p \ell_p(f^{(\kappa)}) & \\
s.t \quad & r_k = \sum_{p\in\mathcal{P}_k} y_p & k \in [\![1, K]\!] \\
& y_p \geq 0 & p \in \mathcal{P}
\end{aligned}$$

Note that this problem is an all-or-nothing iteration and can be solved (o,d)-pair by (o,d)-pair by solving a shortest path problem.
As the cost $t_a^k := \ell_e(f^{(\kappa)})$ is non-negative we can use Djikstra's algorithm to solve this problem.

Frank-Wolfe for UE II

$$
\begin{aligned}
\min_{\{y_p\}_{p\in\mathcal{P}}} \quad & \sum_{p\in\mathcal{P}} y_p \ell_p(f^{(\kappa)}) & \\
s.t \quad & r_k = \sum_{p\in\mathcal{P}_k} y_p & k \in [\![1, K]\!] \\
& y_p \geq 0 & p \in \mathcal{P}
\end{aligned}
$$

Note that this problem is an all-or-nothing iteration and can be solved (o, d)-pair by (o, d)-pair by solving a shortest path problem. As the cost $t_a^k := \ell_e(f^{(\kappa)})$ is non-negative we can use Djikstra's algorithm to solve this problem.

Frank-Wolfe for UE III

aving found $y^{(\kappa)}$, we now have to solve

$$\min_{t\in[0,1]} J(t) := W\Big((1-t)f^{(\kappa)} + ty^{(\kappa)}\Big).$$

As J is convex, the bisection method seems adapted. We have

$$\begin{aligned} J'(t) &= \nabla W\Big((1-t)f^{(\kappa)} + ty^{(\kappa)}\Big) \cdot (y^{(\kappa)} - f^{(\kappa)}) \\ &= \sum_{p\in\mathcal{P}} (y_p^{(\kappa)} - f_p^{(\kappa)})\ell_p\big((1-t)f^{(\kappa)} + ty^{(\kappa)}\big) \end{aligned}$$

hence the bisection method is readily implementable.

Frank Wolfe is a smoothed all-or-nothing

Data: cost function ℓ, constraints, initial flow $f^{(0)}$
Result: equilibrium flow $f^{(\kappa)}$
$\underline{W} = -\infty$;
$k = 0$;
compute starting travel time $c_e^{(0)} = \ell_e(x(f^{(\kappa)}))$;

while $W(x^{(\kappa)}) - \underline{W} > \varepsilon$ **do**
  foreach *pair origin-destination* (o_i, d_i) **do**
    find a shortest path p_i from o_i to d_i for the loss $c^{(\kappa)}$;
  deduce an auxiliary flow $y^{(\kappa)}$ by setting r_i to p_i ;
  set descent direction $d^{(\kappa)} = y^{(\kappa)} - f^{(\kappa)}$;
  find optimal step $t^{(\kappa)} \in \arg\min_{t\in[0,1]} W\left(x^{(\kappa)} + td^{(\kappa)}\right)$;
  update $f^{(k+1)} = f^{(\kappa)} + t^{(\kappa)} d^{(\kappa)}$;
  $\kappa = \kappa + 1$;