Optimization of multi-origin travel search

- Artur Grigorev,
- Satyarth Mishra Sharma
- Vadim Liventsev
- Phan Van Duc
- Shreya Santra

Team: A, SVDs!
Skoltech

22 December, 2017

Problem description

Suppose n friends (who like travelling!) living in different cities around the world want to meet up at a common destination. When and where should they meet to get the cheapest flights?

Project Objective

Try different optimization methods to this problem and suggest a common destination at the cheapest price from multiple airports taking into account the arrival and departure dates.

Data Description

Data acquired (live!) from Sky Scanner API

- We can get the cheapest flights (anywhere) from an origin or between two cities
- Flight prices change on a whim, which is why crawling and caching the data is not effective.

Optimization Formulation

- We treat SkyScanner as a ‘black box’ and use optimization methods query it in the most promising way
- Adds a challenge - we are limited by the latency and specifications of the API. Need optimization!

Methods Applied & Performance Analysis

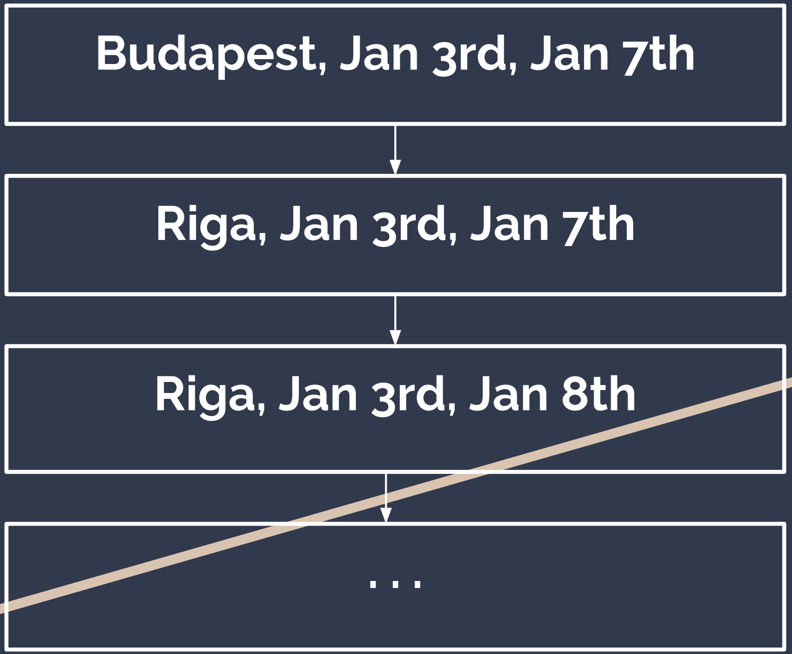
✓ Try different approaches

- Simulated Annealing
- Genetic Algorithm
- Regression Methods
- Branch & Bound

✓ Choose the best one (Spoiler alert: it’s B&B)

✓ Build a Telegram Bot around it

Simulated annealing



Each solution consists of:

- destination
- outbound date
- inbound date

Start with:

- Random triplet within the date range

Neighbourhood includes any solution reached by:

- changing destination city
- changing outbound date
- changing inbound date

Simulated annealing test

Sequence of best solutions for [London, Berlin, Brussels] on the date range from January 1st, 2018 to January 15th, 2018:

```
17.00197458267212 TO: COLO, COME: 2018-01-08,   LEAVE: 2018-01-11, PRICE: 25333.0
20.55383801460266 TO: COLO, COME: 2018-01-07,   LEAVE: 2018-01-11, PRICE: 24990.0
21.91395139694214 TO: COLO, COME: 2018-01-04,   LEAVE: 2018-01-11, PRICE: 21694.0
58.7781603336342 TO: MADR, COME: 2018-01-07,   LEAVE: 2018-01-14, PRICE: 21606.0
62.9499173164367 TO: MADR, COME: 2018-01-10,   LEAVE: 2018-01-13, PRICE: 21337.0
64.90201950073242 TO: MADR, COME: 2018-01-06,   LEAVE: 2018-01-13, PRICE: 16832.0
153.34158849716187 TO: BUDA,    COME: 2018-01-06,   LEAVE: 2018-01-13, PRICE: 16383.0
157.57617044448853 TO: BUDA,    COME: 2018-01-06,   LEAVE: 2018-01-12, PRICE: 8965.0
169.28521418571472 TO: BUDA,    COME: 2018-01-06,   LEAVE: 2018-01-10, PRICE: 8803.0
176.9501485824585 TO: BUDA, COME: 2018-01-05,   LEAVE: 2018-01-11, PRICE: 8239.0
```

6-days trip to Budapest for three people only for 8K roubles!

~ 3min to get the solution

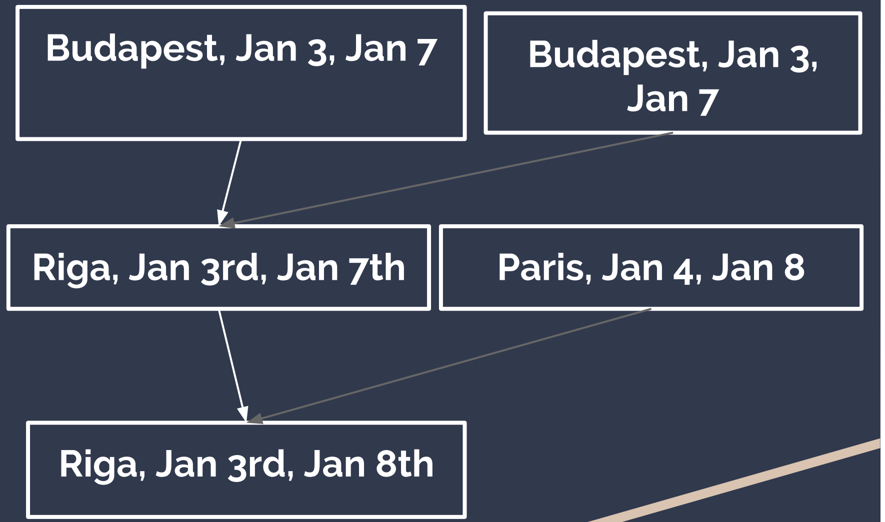
Genetic Algorithm

```
[TO: KULM,      COME: 2018-01-01,      LEAVE: 2018-01-10,
 TO: FLLA,      COME: 2018-01-07,      LEAVE: 2018-01-11,
 TO: CHIA,      COME: 2018-01-05,      LEAVE: 2018-01-13,
 TO: FLLA,      COME: 2018-01-02,      LEAVE: 2018-01-13,
 TO: MILA,      COME: 2018-01-11,      LEAVE: 2018-01-14,
 TO: SGNV,      COME: 2018-01-09,      LEAVE: 2018-01-14,
 TO: TPET,      COME: 2018-01-08,      LEAVE: 2018-01-11,
 TO: YMQA,      COME: 2018-01-01,      LEAVE: 2018-01-09,
 TO: PLSA,      COME: 2018-01-01,      LEAVE: 2018-01-05,
 TO: FDFA,      COME: 2018-01-03,      LEAVE: 2018-01-14,
 TO: BERI,      COME: 2018-01-07,      LEAVE: 2018-01-11,
 TO: MFMA,      COME: 2018-01-06,      LEAVE: 2018-01-12,
 TO: KULM,      COME: 2018-01-02,      LEAVE: 2018-01-05,
 TO: TPET,      COME: 2018-01-07,      LEAVE: 2018-01-14,
 TO: MILA,      COME: 2018-01-06,      LEAVE: 2018-01-10,
 TO: FLLA,      COME: 2018-01-10,      LEAVE: 2018-01-13,
 TO: THES,      COME: 2018-01-09,      LEAVE: 2018-01-14,
 TO: MILA,      COME: 2018-01-11,      LEAVE: 2018-01-14,
 TO: SGNV,      COME: 2018-01-09,      LEAVE: 2018-01-13,
 TO: FDFA,      COME: 2018-01-10,      LEAVE: 2018-01-14]
```

Example population

- Start by initializing our population with random solutions from our sample space
- In each iteration:
  - Evaluate the cost of each sample in our population
  - Discard the worst ones
  - ‘Crossover’ our best solutions: randomly pick and mix features from pairs of our best solutions
  - Some probability of ‘mutations’ - randomize parameters to increase genetic diversity

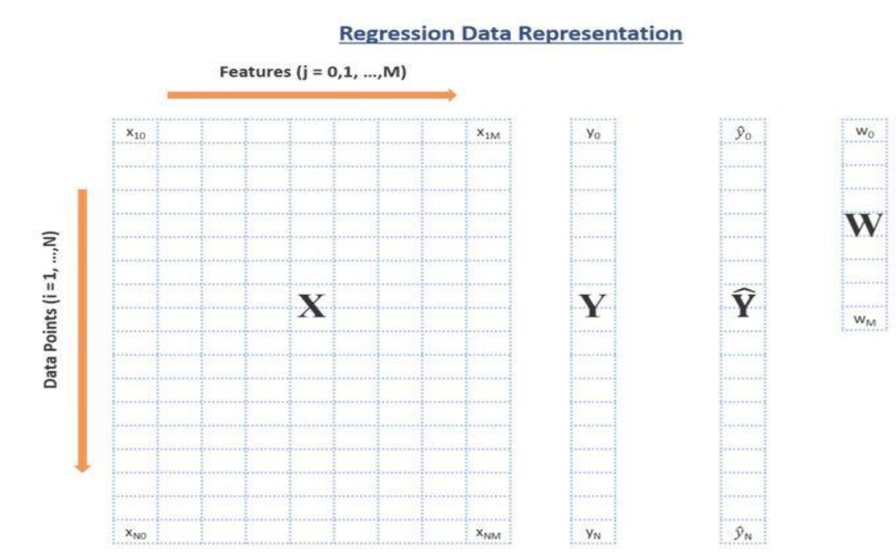
Genetic algorithm



Works reasonably well, but in practice not great compared to other methods:

- Technical reason: Getting costs for our population at each iteration is expensive in terms of time and API calls
- Parameters need to be tuned to include right amount of genetic diversity at each step - otherwise we get ‘inbreeding’
- Still promising, with a better technical implementation (smarter parallelization of network calls)

Regression Methods

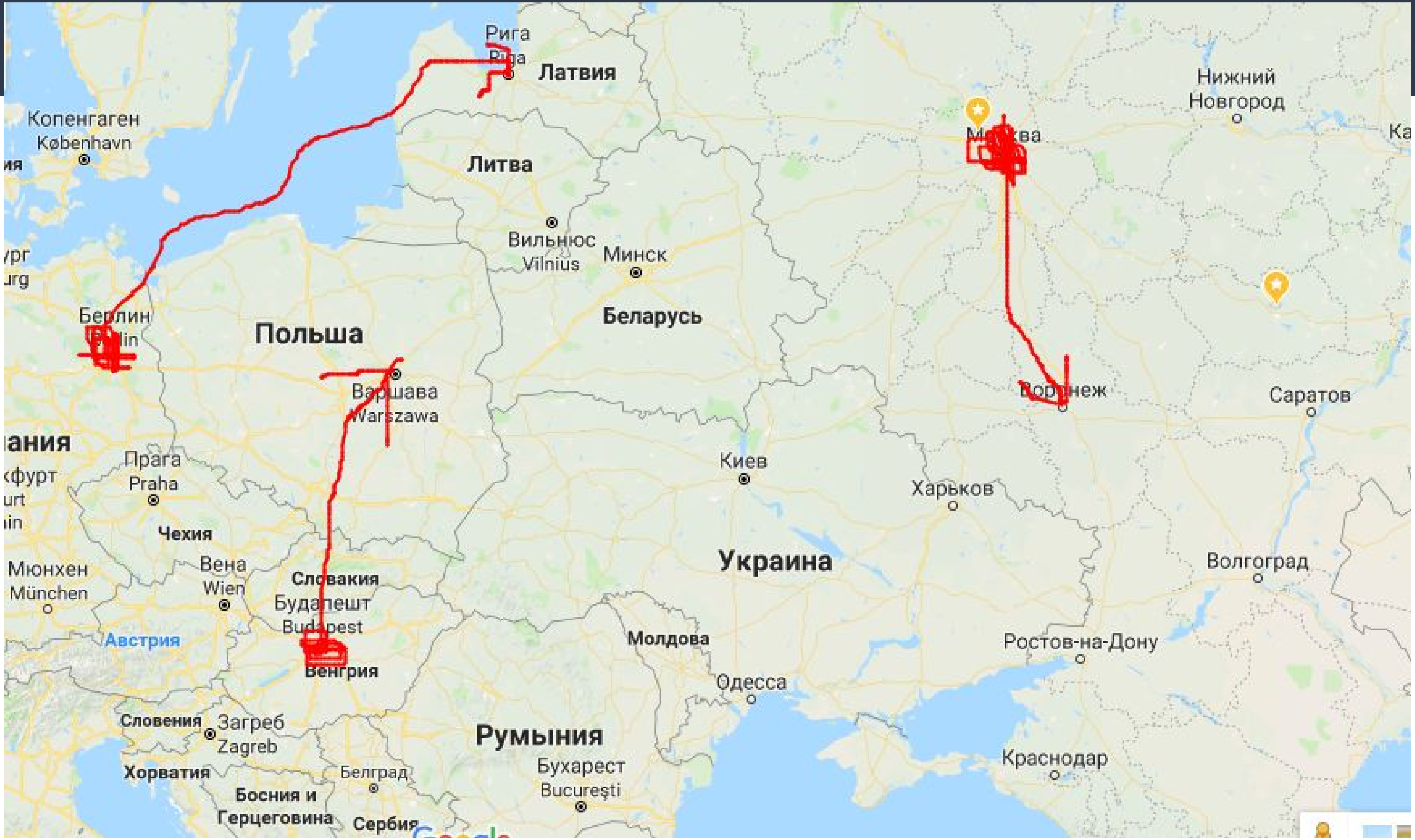


- Several Regression methods were used to predict the expected minimum price for a given origin date, departure date and their places.
- Ridge Regression has 66.48% and Lasso Regression with 66.89% provided an indication of the goodness of fit of a set of predictions to the value.

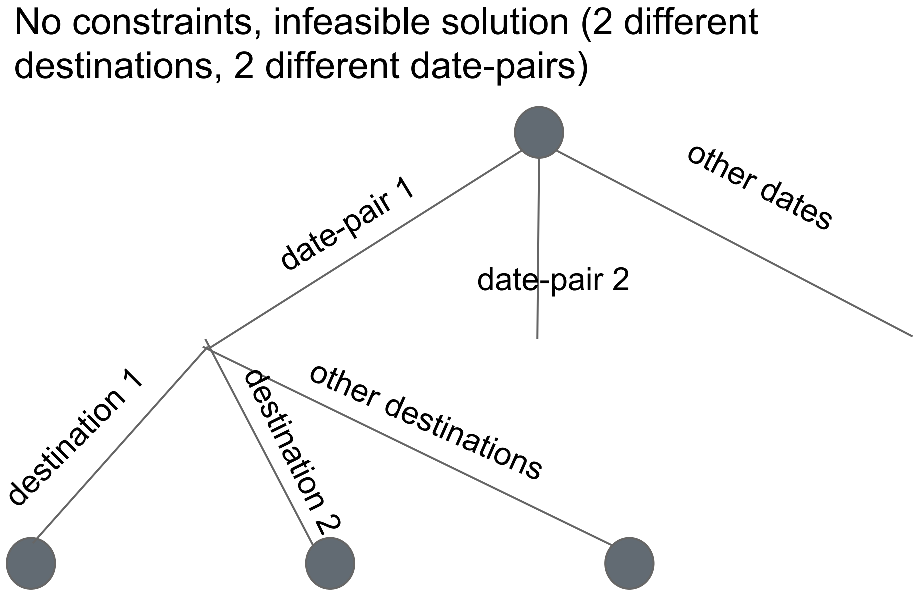
$$Cost(W) = RSS(W) + \lambda * (sum\ of\ squares\ of\ weights) = \sum_{n=1}^{N} (y_i - \hat{y}_i)^2 + \lambda \sum_{j=0}^{M} ||w_j||_2^2$$

$$Cost(W) = RSS(W) + \lambda * (sum\ of\ squares\ of\ weights) = \sum_{n=1}^{N} (y_i - \sum_{j=1}^{M} w_j x_{ij})^2 + \lambda \sum_{j=0}^{M} ||w_j||_1$$

Lower-bounding the cost



Branch and Bound (for 2 origins)



4 types of constraints:

- Required destination
- Banned (taboo) destination
- Required travel dates
- Banned (taboo) travel dates

SVD Approach

Using a truncated database (sourced from Skyscanner), a matrix was created with the Minimum Prices from various origins to various destinations.

	Origin	MinPrice	Destination
0	MUNI	9150.0	MOSC
1	VIEN	8679.0	MOSC
2	BRUS	9197.0	MOSC

However, the SVD failed to converge.

Also, we could not come up with a proper application of low-rank approximation in this project

Demonstration Time!

To our knowledge, this is currently the best implemented solution for this problem.

DAVAI

Learning Outcomes

Vadim:

- I can use optimization in my hobby (travel)!

Shreya:

- My first serious coding challenge!

Sat:

- If you want to make everyone happy, don’t be a leader. Sell ice cream!

Learning Outcomes

Artur:

- Learning outcomes are unnecessary: I pursue knowledge for the sake of knowledge.

Duc:

- Apply machine learning methods to unconventional data.

THANK YOU!