Aalto-yliopisto
Perustieteiden
korkeakoulu

Robust passenger assignment for line planning in public transport

Eero Ketola

9.12.2023

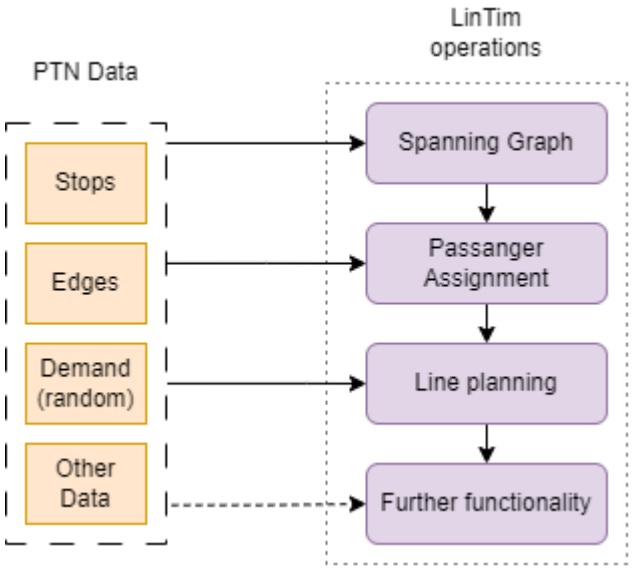
Advisor: Moritz Stinzendörfer

Supervisor: *Philine Schiewe*

Systeemianalyysin
laboratorio

LINTIM

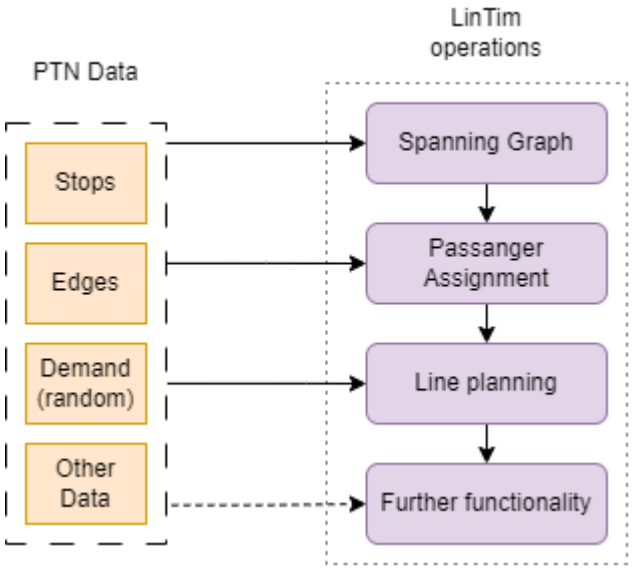
- LinTim is a tool for planning and analysing public transport supply
- Various functionality
  - Passenger assignment
  - Line planning
  - Timetabling
- Doesn't currently have functionality for random demand



Crude outline of LinTim operation

Random demand

- Demand has effects on nearly all stages of planning
- Random demand might cause current algorithms to find non-optimal line plans
- Simplest solution: use average demand
  - This is how random demand is currently dealt with



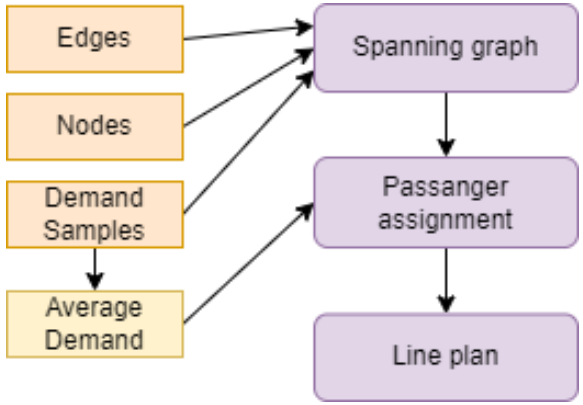
Crude outline of LinTim operation

Statement of objectives

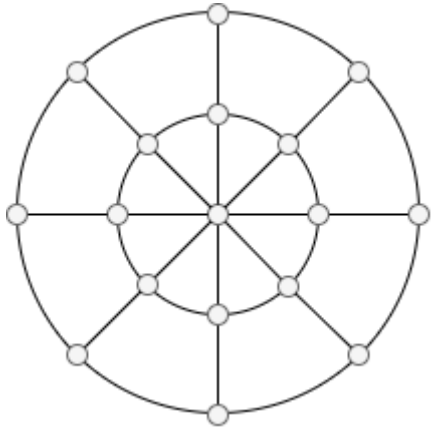
- We are given:
  - A graph representing all possible stops and edges
  - A method for sampling demand
  - Some evaluation metric
- We want:
  - The best (or at least a good) line concept according to the evaluation metric
    - This will naturally depend on the nature of demand. The best line concept for one demand sample will often not be the best for another.

Constraint: Spanning graphs, orb webs

- Spanning graphs determine, which edges must be used in a graph
- A spanning graph can be used to generate a passenger assignment, which can then be used to generate a line plan
- We'll create an algorithm, which finds a spanning graph
  - This spanning graph should on average generate the best line plan
- Now we need a line plan:
  - We could generate this with average demand
  - We could also create a similar algorithm for generating the line plan



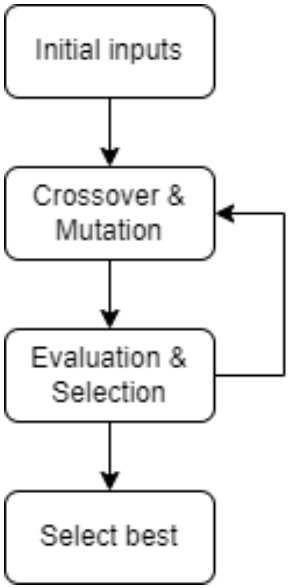
Idea of the algorithm for finding the end line plan



Orb webs, which will be used for testing the graph

Genetic algorithms

- Genetic algorithms mimic natural selection
- Key components:
  - Fitness function
  - Mutation and crossover compatible genotype
- Advantages:
  - Mutation can prevent convergence in a local minimum
  - Very flexible
  - Fitness function has few requirements
    - Can be discontinuous for example
- Drawbacks:
  - Genetic drift
  - Random mutation



Abstract implementation of genetic algorithms

Implementation

```
function genetic_spanning_graph():
    input: (ptn: Graph[Node, Edge], evol_steps: int, new_os: int,
            os_chosen: int, od_samples: Array[Map[Edge, float]])

    // Generate initial spanning graphs. Random graphs can be used.
    // The generator should evaluate the spanning graphs as well.
    offspring: Array[Set[Edge], float] = gen_epsloads(ptn, od_samples

    for _ in range(evol_steps):
        // Generate and evaluate new offspring
        for _ in range(new_os):
            pair = (random(offspring), random(offspring))
            new_os = mutate(crossover(pair))
            offspring += (new_os, evaluate(ptn, new_os, od_samples))
        // Choose best offspring and repeat
        offspring.sort(x: x[1])
        offspring = offspring.head(os_chosen)

    // Best output from evolution
    best: Set[Edge] = offspring[0][0]

    return best
```

Pseudocode for finding a spanning graph

```
function evaluate():
    input: (ptn: Graph[Node, Edge],
            sp_graph: Set[Edge],
            od_samples: Array[Map[Edge, float]])

    // Square sum of evaluations
    eval_square_sum: float = 0

    for od in od_samples:
        // Generate passanger assignment. This is
        // done with a shortest path method
        // using Floyd-Warshall algorithm
        pa = gen_pa(od, ptn, sp_graph)
        // Generate line plan. This is a core
        // functionality of LinTim
        line_plan = gen_lp(pa, od, ptn)
        // Evaluate generated line plan. This is
        // also a core functionality
        eval = gen_eval(lp, od, ptn)
        eval_square_sum += eval^2

    return eval_square_sum
```

Pseudocode for evaluating a spanning graph

Dates

- Presentation of the topic: 9.12.2023
- Results ready: end of 2023
- Second seminar: January 2024
- First draft for thesis: January 2024
- Thesis ready: February 2024

References and literature

- Heinrich, I., Herrala, O., Schiewe, P., Terho T., (2023). Using Light Spanning Graphs for Passenger Assignment in Public Transport. In 23rd Symposium on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems (ATMOS 2023). Open Access Series in Informatics (OASIcs), Volume 115, pp. 2:1-2:16, Schloss Dagstuhl - Leibniz-Zentrum für Informatik
- LinTim: https://lintim.net/
- Ventura, S., Luna, J., Moyano, J. (2022), eds. ‘Genetic Algorithms’. London: IntechOpen, Print.
- Schöbel, A. (2011) ‘Line planning in public transportation: Models and methods’, OR Spectrum, 34(3), pp. 491–510.