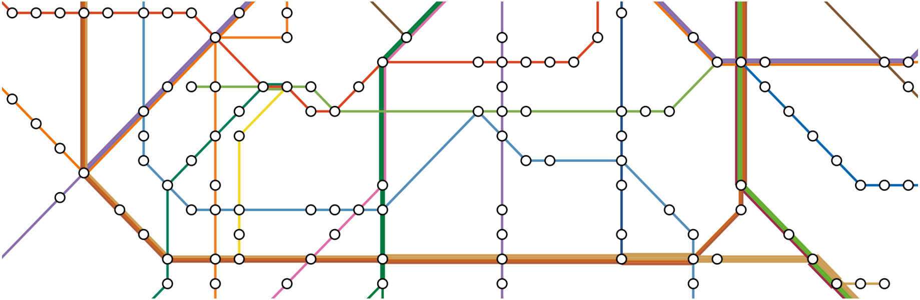
Mathematical Aspects of Public Transportation Networks

Niels Lindner



May 14, 2018

Chapter 2

Shortest Routes in Public Transportation Networks

§2.4 Multi-Modal Routing

Multi-Modal Routing

Multi-modal routing is a holistic routing approach including

- road networks
- public transportation networks
- flight networks
- ...

The routing on road networks could include private cars, taxis, bicycles, footpaths, ...

Idea

Merge road networks with graphs coming from the realistic time-dependent model for public transportation/flight networks.

Road and Flight Networks

Road networks

Road networks are modeled as a directed graph in the naive way:

- vertices are intersections of roads,
- edges are road segments.

We can also include footpaths into this model. Additionally put a label on each edge specifying whether the corresponding segment is a highway, a local street, a cycle lane, a footpath, ... This is important for estimating the travel time.

Flight networks

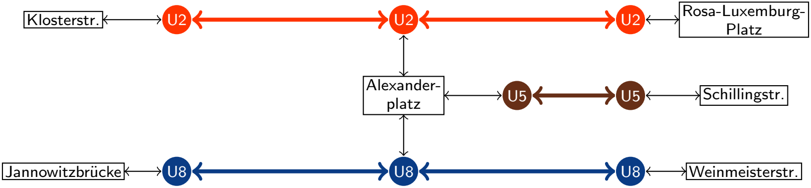
Flight networks can be modeled in the same way as public transportation networks, e.g., as in the realistic time-dependent model. It makes sense to distinguish transfers within an alliance from other transfers.

Linking

Question

How to link road networks with public transportation networks?

Recall that the vertex set of the realistic time-dependent model for a public transportation network comprises *station vertices* and *route vertices*:



Route vertices are neither startpoints nor endpoints of a journey.

Idea

Introduce a directed edge from every station vertex to the nearest vertex (i.e., intersection) of the road network. Also add an edge in the backward direction.

Linking

More on linking

- The travel time on such a link may be estimated by geographical distance divided by minimum walking speed.
- Station vertices may also be linked to several nearest vertices.
- There is no point in linking every vertex of a road network to the nearest station, as this results in long footpaths.
- The linking process between flight and road network is similar.
- Flight and public transport networks should also be linked directly.

Result

The result is a directed graph, where earliest arrival queries can be solved by applying (time-dependent) Dijkstra.

Label-Constrained Shortest Walks

Problems

- This produces useless journeys, e.g., private car - train - private car.
- Even a journey private car - train is useless for people without cars.

Solution

Restrict the possible sequences of transport modes in a journey. This is called the *label-constrained shortest walk problem*. Here, we label each edge by its mode of transportation.

Formal Languages

Definition

Let Σ be a non-empty finite set (**alphabet**).

- A **word** on Σ is a finite sequence $a_1 \cdots a_n$, where $a_1, \dots, a_n \in \Sigma$.
- Σ^* denotes the set of all words on Σ, including the *empty word* ε. (**Kleene star**)
- If x and y are two words in Σ^*, their **concatenation** is $xy \in \Sigma^*$.
- A **language** L on the alphabet Σ is simply a subset of Σ^*.

Definition

Let $G = (V, E)$ be a weighted directed graph, and $s, t \in V$. Further let $\sigma : E \to \Sigma$ be a labeling of the edges in E with letters from an alphabet Σ, and let $L \subseteq \Sigma^*$ be a language.

The **label-constrained shortest** s-t-**walk problem (LCSWP)** is to find an s-t-walk $(e_1, \dots, e_k)$ of minimum length such that $\sigma(e_1) \cdots \sigma(e_k) \in L$.

Regular Languages

Theorem (Barret/Jacob/Marathe, 2000)

If L is a regular language, then the LCSWP on L can be solved in polynomial time.

Definition (Regular languages/Regular expressions)

Let Σ be an alphabet. A language $L \subseteq \Sigma^*$ is **regular** if it can be constructed using the following rules:

- $\emptyset$ is regular.
- $\{\varepsilon\}$ is regular.
- $\{a\}$ is regular for all $a \in \Sigma$.
- If L_1 is regular, then so is $L_1^* := \{x_1 \cdots x_n \mid x_1, \ldots, x_n \in L_1, n \in \mathbb{N}_0\}$.
- If L_1 and L_2 are regular, then so is $L_1 L_2 := \{xy \mid x \in L_1, y \in L_2\}$.
- if L_1 and L_2 are regular, then so is $L_1 \cup L_2$.

Regular Languages: Example

Example

Let $\Sigma = \{c, t, w\}$ (car ride, train ride, walk). Then $L = \{cw^*tw^*\}$ is regular, where w^* denotes an arbitrary finite sequence of w's. That is,

$$L = \{ct, cwt, ctw, cwwt, cwtw, ctww, cwwwt, cwwtw, cwtww, ctwww, \dots\}$$

Construction of L:

1. $L_c = \{c\}$, $L_w = \{w\}$ and $L_t = \{t\}$ are regular languages.
2. L_w^* is regular.
3. The concatenation $L_c L_w^* L_t L_w^*$ is regular.

Remark

Expressions of the form cw^*tw^* are called *regular expressions*. Regular languages are precisely the languages generated by regular expressions.

Deterministic Finite Automata

Definition

A **deterministic finite automaton (DFA)** is a 5-tuple $M = (Q, \Sigma, \delta, q_0, F)$, where

- Q is a finite set of *states*,
- Σ is an *input alphabet*,
- $\delta : Q \times \Sigma \to Q$ is a *transition function*,
- $q_0 \in Q$ is a *start state*,
- $F \subseteq Q$ is a set of *final states*.

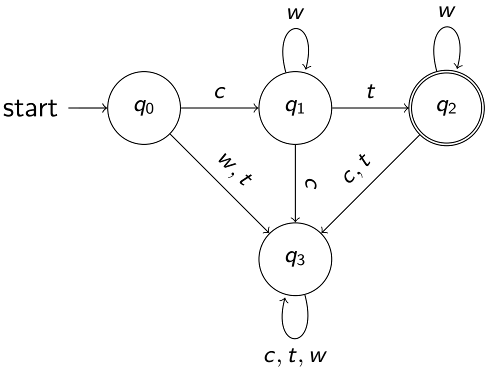
The **language accepted by** M is

$$\left\{ a_1 \cdots a_n \in \Sigma^* \,\middle|\, \begin{array}{c} \exists\, q_1, \ldots, q_{n-1} \in Q, q_n \in F : \\ \delta(q_{i-1}, a_i) = q_i \text{ for } i = 1, \ldots, n \end{array} \right\}.$$

DFA: Example

Example

Consider the following DFA:



- $Q = \{q_0, q_1, q_2, q_3\}$
- $\Sigma = \{c, t, w\}$
- $F = \{q_2\}$
-

δ	c	t	w
q_0	q_1	q_3	q_3
q_1	q_3	q_2	q_1
q_2	q_3	q_3	q_2
q_3	q_3	q_3	q_3

This DFA accepts all words on a directed walk from q_0 to q_2, i. e., all regular expressions of the form cw^*tw^*.

Regular Languages, DFA and NFA

Theorem

Let L be a language. Then the following are equivalent:

- *L is regular.*
- *L is accepted by some DFA.*
- *L is accepted by some NFA.*

Definition

A **non-deterministic finite automaton (NFA)** is a 5-tuple $N = (Q, \Sigma, \delta, S, F)$, where

- Q, Σ, F are as in the DFA case,
- $\delta : Q \times \Sigma \to \mathcal{P}(Q)$ takes values in the power set of Q,
- S is a *set* of start states.

The **language accepted by** N is

$$\left\{ a_1 \cdots a_n \in \Sigma^* \;\middle|\; \begin{array}{c} \exists\, q_0 \in S, q_1, \ldots, q_{n-1} \in Q, q_n \in F : \\ q_i \in \delta(q_{i-1}, a_i) \text{ for } i = 1, \ldots, n \end{array} \right\}.$$

DFA and NFA

Remarks

- NFA can be drawn as directed graphs in a similar way.
- NFAs accept words for which there is a valid directed walk. Unlike in DFAs, there might be more than one walk corresponding to a word.
- Any DFA is trivially an NFA.
- Every NFA can be turned into an equivalent, but potentially much bigger DFA.
- NFAs are a good choice when alternatives should be modeled, i.e., the union of two regular languages.
- NFAs can *die* in the sense that $\delta(q, a) = \emptyset$ for the current state q and input letter a.
- In our example, we could therefore construct a smaller NFA by omitting the state q_3.

LCSWP: Algorithm

Let G be a weighted digraph, $s, t \in V(G)$, $L \subseteq \Sigma^*$ a regular language and $\sigma : E(G) \to \Sigma$.

LCSWP Algorithm

1. Construct an NFA $N = (Q, \Sigma, \delta, S, F)$ accepting precisely L.
2. Construct the *product network* $G^\times$ as follows:
   - $V(G^\times) := V(G) \times Q$
   - $E(G^\times) := \{((v_1, q_1), (v_2, q_2)) \mid (v_1, v_2) \in E(G), q_2 \in \delta(q_1, \sigma(v_1, v_2))\}$, keep the weights.
3. Compute all shortest paths from (s, q_s) to (t, q_f) for all $q_s \in S$ and $q_f \in F$.
4. Determine the path of minimum length and return its projection to G (or return that no walk exists).

LCSWP: Remarks

Correctness

- Any path in $G^\times$ projects to a walk in G and to a walk in the state graph of the NFA N. In particular, any (s, q_s)-(t, q_f)-path in $G^\times$ gives an s-t-walk in G labeled with a word accepted by N, i.e., a word in L.
- Concept: Minimize over all s-t-walks and all possible words on them.
- Note that the solution to the LCSWP might include repeated vertices.

Complexity

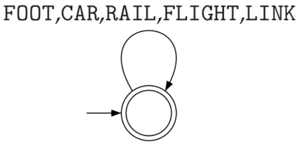
The product network has $|V(G)| \cdot |Q|$ vertices and $\mathcal{O}(|E(G)| \cdot |Q|)$ edges. The Dijkstra algorithm needs hence

$$\mathcal{O}(|S||F|(|V(G)||Q| \log(|V(G)||Q|) + |E(G)||Q|))$$

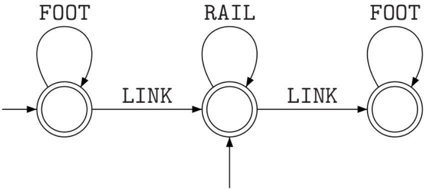
elementary operations to solve the many-to-many shortest-path problem. Since $|S|, |F| \leq |Q|$, this is polynomial if we can bound $|Q|$. Given a regular expression with ℓ characters, *Thompson's construction* yields an NFA with $O(\ell)$ states, so $|Q|$ is linear in the input size of L.

Multi-Modal Routing: DFA

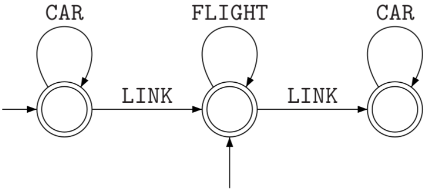
What are good regular languages for multi-modal routing?



(a) Everything mixed.



(b) Foot & Railways.

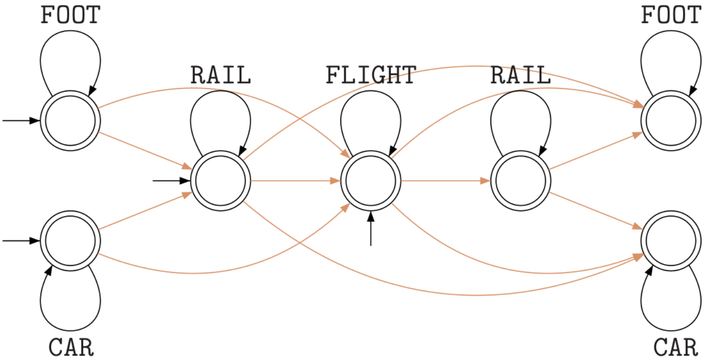


(c) Car & Flights.

Thomas Pajor: Multi-Modal Route Planning, Diplomarbeit, 2009.

Multi-Modal Routing: NFA

What are good regular languages for multi-modal routing?



Thomas Pajor: Multi-Modal Route Planning, Diplomarbeit, 2009.

More on Multi-Modal Routing

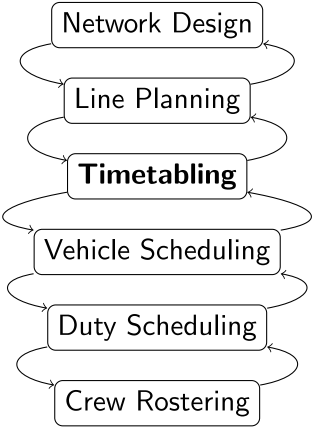
- Of course, running Dijkstra's algorithm on the product network is not the end of the story.
- Several speed-ups (mostly from road network techniques) are available.
- However, some preprocessing strategies do not allow a user to specify his preferences (e.g., is there a private car available?).
- Multi-criteria optimization is important as well (number of changes of transport modes, price) $\rightsquigarrow$ *Multi-Modal Multi-Criteria RAPTOR*.

Chapter 3

Periodic Timetabling

§3.1 Overview

Public Transport Planning Cycle



strategic planning

operational planning