About me

Tinkered with OTP since ~2011

• City of Helsinki 2014
• Helsinki Region Transport & Digitransit 2015–2018
• Kyyti 2020
• Entur 2021–
Overordnet komponentskisse
Team Ruter og Rensplanlægger
Gjort i begynnelsen 2022-06-10

[Diagram image]

Dokumentasjonen

åtkomst: ruter, sprekter, Plans, håndbok, W-1 og bruksanvisningene
https://www.ensport.no/ensport/ensport/ensport
https://www.ensport.no/ensport
https://github.com/lunar

Komponenten

Innhold:

- Innhold
- Prosess
- Vernet
- Rekommendasjoner
- Bruksanvisning
- Andre aktører

Kontakt:

Entur

Forfatter:

[Signature]
Agenda

Past — OpenTripPlanner 1

Present — OpenTripPlanner 2
  • How it works
  • New features
  • Sandbox extensions
  • Simplified setup

Future — Roadmap
Past
Pain points with OTP 1

Time-dependent A* search with trip banning

- Insufficient performance for nationwide deployments
- Focus on research capabilities
  - Solved by split into OTP and R5
- Lack of architectural vision and focus
- Fragmented development
  - Each organization had its own fork
Present
OTP Routing Process

- **Street Search**: Searches for the nearest stops, where the user can board a transit vehicle.
- **Transit Search**: Searches the scheduled routes between the stops close to the origin and destination.
- **Itinerary Filtering/Decorating**: Filters down the itineraries presented to the user and enriches them eg. with price.
- **Flex Search**: Extends the stops the user can reach by utilizing flexible transit, which has no fixed timetables/route.
- **Transfer Optimizer**: Optimizes where to transfer between each leg pair in the itineraries.
- **API Routing Request**
- **API Routing Response**
Street Search
Fixed Route
Hail and Ride Sections

- Fixed route and schedule
- Between stops 3 and 6, you can board or alight anywhere along the route
Flexible Areas

- Door to door anywhere within a service area
Flexible Areas

- One area for boarding and another for alighting
Flexible Areas

- Any number of areas, some with only boarding, some with only alighting
Fixed Stops in an Area

- Stop to stop within an area
Feeder services

- Area to stop and vice versa
Complex services
Raptor

- **Raptor** works in rounds
  - Implicit graph model using memory layout
  - One round for each transit trip & transfer
  - Exploring the transit network following transit routes.
  - Find all pareto optimal paths by
    - \([\text{Arrival time} | \text{Number of transfers}]\) – Given \text{departure time}

- **Range Raptor**
  - Iterates backwards over \text{departure time} within a \text{search window}
  - Only explores new trips not reached by previous rounds
  - Pareto optimal by
    - \([\text{Departure time} | \text{Arrival time} | \text{Number of transfers}]\)

- **Multi-criteria Range Raptor**
  - One or more additional criteria – with \text{performance penalty}
  - Pareto optimal by
    - \([\text{Departure time} | \text{Arrival time} | \text{Number of transfers} | \text{Generalized cost}]\)
<table>
<thead>
<tr>
<th>State</th>
<th>Transit Strategy</th>
<th>Search direction</th>
<th>Optimization</th>
<th>Result</th>
<th>Response time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standard</td>
<td>Standard</td>
<td>Forward</td>
<td>-</td>
<td>Paths [ time, transfers ]</td>
<td>66 ms</td>
</tr>
<tr>
<td>Standard</td>
<td>Standard</td>
<td>Reverse</td>
<td>-</td>
<td>Paths [ time, transfers ]</td>
<td>68 ms</td>
</tr>
<tr>
<td>BestTime</td>
<td>Standard</td>
<td>Forward</td>
<td>-</td>
<td>Best time &amp; hops, No paths</td>
<td>63 ms</td>
</tr>
<tr>
<td>BestTime</td>
<td>Standard</td>
<td>Reverse</td>
<td>-</td>
<td>Best time &amp; hops, No paths</td>
<td>60 ms</td>
</tr>
<tr>
<td>Standard</td>
<td>NoWait</td>
<td>Forward</td>
<td>1 iteration</td>
<td>Paths [ time, transfers ]</td>
<td>49 ms</td>
</tr>
<tr>
<td>Standard</td>
<td>NoWait</td>
<td>Reverse</td>
<td>1 iteration</td>
<td>Paths [ time, transfers ]</td>
<td>48 ms</td>
</tr>
<tr>
<td>BestTime</td>
<td>NoWait</td>
<td>Forward</td>
<td>1 iteration</td>
<td>Best time &amp; hops, No paths</td>
<td>41 ms</td>
</tr>
<tr>
<td>BestTime</td>
<td>NoWait</td>
<td>Reverse</td>
<td>1 iteration</td>
<td>Best time &amp; hops, No paths</td>
<td>37 ms</td>
</tr>
<tr>
<td>MC</td>
<td>MC</td>
<td>Forward</td>
<td>-</td>
<td>Paths [ time, transfers, cost ]</td>
<td>508 ms</td>
</tr>
<tr>
<td>MC</td>
<td>MC</td>
<td>Forward</td>
<td>Heuristic Destination Check</td>
<td>Paths [ time, transfers, cost ]</td>
<td>320 ms</td>
</tr>
</tbody>
</table>

28 Samples
Search Window 2 - 20 h
Dataset Norway
Transfer optimization

Where to transfer between a pair of trips?

- Transfer priority cost
  - Station transfer priority*
  - Guaranteed transfers
  - In-seat transfers

- Optimal wait time
  - Avoid very short transfer times*
  - Avoid back-travel*

* Not in raptor
Itinerary Filtering & Decorating

- Limit the number of results
  - Worse but optimal results
  - Grouping too similar results
  - Park & ride, where the car is parked almost immediately

- Decorate results
  - Real-time alerts
  - Price calculation

- Sorting of results
NeTEx–GTFS

- New internal data model independent of the import format
  - OTP 1 used GTFS POJOs internally
  - New entities from NeTEX not existing in GTFS

Different formats have different benefits
- GTFS
  - Easy to produce and consume
  - YAGNI — Requires producer and consumer before appending spec
- Netex
  - Much more complex and nuanced
  - Caters for almost all use cases
Sandbox extensions

• New feature in OTP 2 for code not suited for core
  • Extremely successful, currently 22 extensions
    • New APIs
      • GraphQL
      • Travel time
      • Vector tiles
    • New data formats
      • Data overlay
      • SIRI
    • New functionality under development
      • GTFS-Fares v2
      • GTFS-Flex v2
    • Deployment-specific code
      • Non-GBFS vehicle rental updaters
      • Cloud integrations
GraphQL APIs

Two APIs with different vocabularies

- GTFS
- Transmodel
Vector tiles

- Mapbox vector tile format
- Multiple layer types available
  - Stops & stations
  - Rental stations & vehicles
  - Car and bike parking
- Configurable mapping from internal model to tile layer
  - Real-time info
  - Multilingual
Travel time analysis

• Requested feature from OTP 1
• Two output formats
  • GeoJSON — isochrones
  • GeoTIFF — travel time rasters
• Configurable street & transit modes
Simplified operations

- Abstracted data sources
  - Local file system
  - HTTPS
  - Cloud storage services
    - GCP storage
    - Azure blob storage
    - AWS S3 (open PR pending somebody using it)
- All input and output paths can be configured
  - Data can be read from or written to an data source
- Improved monitoring support
  - Prometheus endpoint
# Graph Build Configuration

This table lists all the JSON properties that can be defined in a `build-config.json` file. These will be stored in the graph itself, and affect any server that subsequently loads that graph. Sections follow that describe particular settings in more depth.

<table>
<thead>
<tr>
<th>Config Parameter</th>
<th>Type</th>
<th>Summary</th>
<th>Req./Opt.</th>
<th>Default Value</th>
<th>Since</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>areaVisibility</code></td>
<td>boolean</td>
<td>Perform visibility calculations.</td>
<td>Optional</td>
<td>false</td>
<td>1.5</td>
</tr>
<tr>
<td><code>banDiscouragedBiking</code></td>
<td>boolean</td>
<td>Should biking be allowed on OSM ways tagged with <code>bicycle=discouraged</code></td>
<td>Optional</td>
<td>false</td>
<td>2.0</td>
</tr>
<tr>
<td><code>banDiscouragedWalking</code></td>
<td>boolean</td>
<td>Should walking be allowed on OSM ways tagged with <code>foot=discouraged</code></td>
<td>Optional</td>
<td>false</td>
<td>2.0</td>
</tr>
<tr>
<td><code>blockBasedInterlining</code></td>
<td>boolean</td>
<td>Whether to create stay-seated transfers in between two trips with the same block id.</td>
<td>Optional</td>
<td>true</td>
<td>2.2</td>
</tr>
<tr>
<td><code>buildReportDir</code></td>
<td>uri</td>
<td>URI to the directory where the graph build report should be written to.</td>
<td>Optional</td>
<td></td>
<td>2.0</td>
</tr>
<tr>
<td><code>configVersion</code></td>
<td>string</td>
<td>Deployment version of the <code>build-config.json</code>.</td>
<td>Optional</td>
<td></td>
<td>2.1</td>
</tr>
<tr>
<td><code>dataImportReport</code></td>
<td>boolean</td>
<td>Generate nice HTML report of Graph errors/warnings</td>
<td>Optional</td>
<td>false</td>
<td>2.0</td>
</tr>
<tr>
<td><code>discardMinTransferTimes</code></td>
<td>boolean</td>
<td>Should minimum transfer times in GTFS file be discarded</td>
<td>Optional</td>
<td>false</td>
<td>2.2</td>
</tr>
</tbody>
</table>
//build-config.json
{
  "transitModelTimeZone": "Europe/Brussels",
  "osmCacheDataInMem": "true",
  "osm": [
    {
      "source": "https://download.geofabrik.de/europe/belgium-latest.osm.pbf",
      "osmTagMapping": "germany"
    }
  ],
  "transitFeeds": [
    {
      "type": "gtfs",
      "feedId": "NMBS",
      "source": "http://gtfs.irail.be/nmbs/gtfs/latest.zip"
    },
    {
      "type": "gtfs",
      "feedId": "LIJN",
      "source": "http://gtfs.irail.be/de-lijn/de_lijn-gtfs.zip"
    },
    {
      "type": "gtfs",
      "feedId": "TEC",
      "source": "https://gtfs.irail.be/tec/tec-gtfs.zip"
    },
    {
      "type": "gtfs",
      "feedId": "MIVB",
      "source": "https://gtfs.irail.be/mivb/mivb-gtfs.zip"
    }
  ]
}
Future
Competition neutrality

• New Raptor criteria
• Fixed size bitset for used authority/operator group

• Operator 1, departure from A at 18:00, arrival at B at 21:00
• Operator 2, departure from A at 18:01, arrival at B at 20:59
• => Only operator 2 is showing up
Unified GraphQL API

- Currently two GraphQL APIs and one REST API
  - Deprecate REST API
- New unified GraphQL API
  - One structure
  - Two dialects, GTFS and Transmodel
  - Use translation file go from internal model to API
Useful links

• https://www.opentripplanner.org/
• https://docs.opentripplanner.org/en/dev-2.x/
• https://github.com/opentripplanner/OpenTripPlanner
• https://gitter.im/opentripplanner/OpenTripPlanner
• https://otp-performance.leonard.io/