Subject: How to model 3 topology levels in rail.ML
Posted by Thomas Langkamm on Fri, 26 Apr 2019 10:19:59 GMT
View Forum Message <> Reply to Message

Hello all,

I'm working at PSI Transcom. We learned about the NEAT infrastructure editor that will export rail.ML, and are very interested in using it as an external editor for our infrastructure using rail.ML 3.x. I'm checking out how to model all objects we need in rail.ML, and came across a little problem that I'd like to share: How could we model 3 or more topology levels?

We create track plans with all details (microscopic rail.ML model), but we usually import our timetables from planning programs and need to match the data from the planning programs to our track plans. Many planning programs use a 2-tier topology: One that is based on stations, and one that is more detailed (but not as detailed as the microscopic topology), usually on basis of station tracks and possibly parking tracks. This 2-tier topology is necessary, as timetable programs need to track connectivity and changes of direction in more detail than a station-based topology allows: Changes of direction can't be checked at all in a station based topology, and connectivity within a station is a thing because it might take passengers (or drivers switching to a different train journey) between a few seconds to get to a train on the other side of the same platform or several minutes to get to a different platform.

For example, a track plan like this

which has a basic A-B-C mesoscopic (station) topology could look like this in the "station track" based topology:

or

Connectivity is not induced by the technical ability to move between 2 station, but by what's in the planned timetable. For example, trains may be able to go directly from A2 to P3, but the edge may be missing because this is not something that will be planned in advance.

Now, if we get a timetable with a train journey ending at B, we get stations (and times) like C1 -> B2 -> P3, and need to match these locations to our track plan.

As far as I understand, rail.ML 3.1 currently defines 2 topology levels: A mesoscopic topology (station based) and a microscopic topology (full details). But how would we include an intermediate topology (that must be consistent with the other 2)?

Structurally I have some intuition how this could work, and work in a generic way: Each element is part of a (named) topology, and there can be inclusions. We can name the 3 or more standard topologies that we want to standardize (mesoscopic = nodes are stations, microscopic = nodes are netElements/track segments, and [insert cool name here] = nodes correspond to timetable topology), assign nodes (netElements) to topologies, and allow for inclusions. Inclusions can be

0..1 : 0..n (or 0..1 : 1..n in a complete model, where any element of a high-level topology has at least one element in a more detailed topology).

Thoughts?

Best regards, Thomas
