University of Stuttgart
Institute for Road and Transport Science
Chair for Transport Planning and Traffic Engineering

Visum to LinTim Interface

Summary

This "Visum to LinTim" interface provides several python files and PTV Visum procedure sequences for reformatting Visum files into a format compatible with LinTim and vice versa. This makes it possible to combine infrastructure and demand modelling in PTV Visum with creating solutions for public transportation (PuT) supply in LinTim and comprehensive evaluation back in PTV Visum (see fig. 1).

Figure 1: workflow using PTV Visum and LinTim for public transport planning

LinTim "is a scientific software toolbox that has been under development since 2007, giving the possibility to solve the various planning steps in public transportation"[1]. For further information we refer to the LinTim documentation on https://www.lintim.net/. This document is meant to be a short guide for users who completed the modelling of their network in PTV Visum and want to create solutions for PuT supply with LinTim.

For using LinTim to solve plannings steps in PuT, first a LinTim dataset has to be created. The procedure sequence *VIS20_Gateway_Visum2Lintim.xml* generates files which can be read in LinTim to define a dataset. Further there's a second procedure sequence *VIS20_Gateway_Lintim2Visum.xml* to reformat LinTim's timetable output as file according to the PuT supply hierarchy of PTV Visum.

Folder Structure

Due to several dependencies and files being read or moved, we recommend the folder structure as shown in fig. 2. Otherwise you have to adjust the python script which sets the project directory in both procedure sequences and several other code snippets. Each scenario has a basic version file (without PuT supply) in the subfolder "Version". From this multiple instances can be derived. Each instance can have multiple solutions for PuT supply. Grey-shaded folders are generated as required.

[1] https://www.lintim.net/

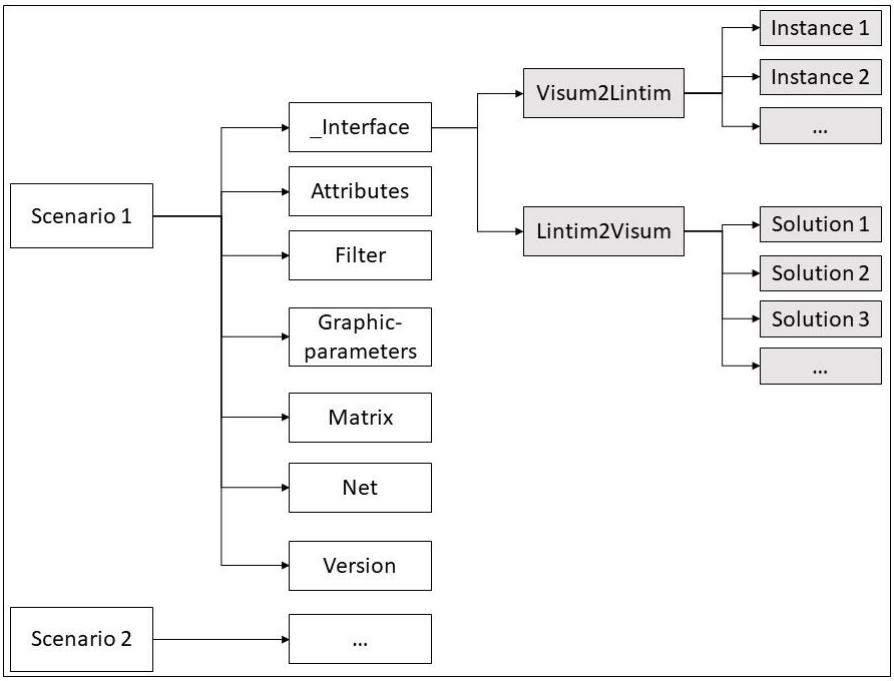

Figure 2: recommended (implemented) folder structure

Requirements

The procedure sequences are developed using PTV Visum 20 and Python 3.7. In addition the following non-standard python libraries are used:

- pandas (version 1.2.2) [2]
- networkx (version 2.5) [3]

Procedure sequence *VIS20_Gateway_Visum2Lintim.xml*

The procedure sequence *VIS20_Gateway_Visum2Lintim.xml* creates all necessary files to import the network model in LinTim. These files are:

- *infrastructure.net* contains information about links, nodes, stops and zones
- *config.net* contains general settings for LinTim
- *walk_times.att* contains walk times between stops and zones as well as between zones and zones
- *od.att* contains demand for all origin-destination pairs on zone level
- *veh_units.att* contains settings of transport systems

[2] https://pandas.pydata.org/
[3] https://networkx.org/

If the model contains existing PuT supply of transport systems which are not considered for optimization, following files are created additionally:

- *visum-fixed-lines.net*
- *visum-fixed-times.net*

The procedure sequence contains two main parts: general settings and file generation (see figure 3). More details are provided in the next subsections.

Number: 60	Procedure		Reference object(s)
1	Group VIS20_Gateway_Visum2Lintim ...	...	
2	Group set project directory	...	3
3	Run script	...	
4	Group create necessary UDAs ...	...	5
6	Group ### General Settings ### ...	...	7 - 21
22	Group ### Create Files for LinTim ##### ...	...	
23	Group infrastructure.net, fixed-lines.net, walk_times.att ...	...	24 - 27
28	Group demand: od.att ...	...	29 - 30
31	Group Calculate walk time ...	...	32 - 60

Figure 3: Overview of procedure sequence *VIS20_Gateway_Visum2Lintim.xml*

General Settings

In this group all attributes used in LinTim are set. Table 1 shows attributes which define general settings. As already mentioned multiple instances can be distinguished. Instances can vary in minor changes only for public transport planning, e.g. net reduction or variations of connectors or terminals. Attributes regarding parameters for the utility function are shown in table 2.

Further the characteristics of transport systems need to be exported. They are set in PTV Visum → *Lists* → *PuT operation* → *vehicle units*.

On stop point level there are three attributes necessary, see table 3.

File Generation

All necessary files are generated while running the procedure sequence, in particular:

→ Save attributes of transport systems in *veh_units.att* (run internal script, step 25)

→ Save values nodes, zones, links and stop points in *infrastructure.net*, further save active PuT supply as fixed lines in *visum-fixed-lines.net* (Links, Lines, Line routes, Line route items, Vehicle journeys) and *visum-fixed-times.net* (Links, Lines, Line routes, Line route items, Time profiles, Time profile items, Vehicle journeys) (run script *ExportVisumInfrastructureLintim.py*, step 26)

Table 1: general settings

attribute	**example value**	**explanation**
scenario_name	"01_example"	Scenario prefix / folder name
instance_name	"I1"	instance prefix
lintim_period_length	3600	Length of period in timetable [s]
lintim_time_units_per_minute	60	
lintim_base_unit_for_headway	1	Factor for headway
lintim_tsys_for_adapting	"B"	Transport system for which PuT supply will be designed. Other transport systems are only considered as fixed PuT supply
lintim_defdwelltime	20	Default dwell time for PuT supply to be designed [s]
lintim_prepreptime	90	Default layover time for PuT supply to be designed [s]
lintim_postpreptime	90	Default layover time for PuT supply to be designed [s]

Table 2: parameters for utility function

attribute	**example value**	**explanation**
lintim_walktime_utility	1.5	
lintim_transfer_utility	5	
lintim_min_transfertime	180	Lower bound for transfer time [s] Take care, this counts also as walking (e.g. is multiplied with 1.5)

Table 3: necessary attributes stop points

attribute	example value	explanation
is_Terminal	True	defines if stop is possible terminal for lines
is_Transfer	True	defines if stop is possible transfer station for lines
DefDwellTime	20	Lower bound for dwell time [s]

→ Save general attributes and parameters for the utility function in *config.net* (run internal script, step 27)

Export Demand *od.att*:

→ Save relevant demand matrix (needs to be defined) as *od.mtx*, using $O-Format (step 29)

→ Change header of *od.mtx* and save matrix as *od.att* (run internal script, step 30)

Export of walk times *walk_times.att*:

→ Run group *Calculate walk time*. To get the shortest path between stops and zones as well as between zones and zones, there are zones added during the procedure sequence. All changes to the network are removed in the end.

Further the procedure sequence makes sure all filters are set correctly.

Some Notes on "Visum to LinTim"

- Stop points have to be node-based stop points in PTV Visum. In LinTim the set of stop points is a subset of given nodes.
- Make sure, your infrastructure is non-directed, e.g. a link has the same length and run time as it's opposite direction.
- Each link has an attribute "internal_volume". If it is above zero for a link, this link will necessarily become part of the resulting line net.
- At the moment, a line cannot skip stops. You have to consider this while modelling the infrastructure.
- Don't use "-" in names of fixed lines. LinTim uses "-" as a separator.
- Make sure your fixed timetable is periodic between 7 - 8 o'clock. LinTim deals with periodic timetables and this time interval is used for transformation.

Generation of PuT supply solution in LinTim based on exported Visum files

This section contains a short step-by-step instruction how to create a timetable for your dataset. Be aware that LinTim contains lots of algorithms, objective functions and parameters which you have to consider. Also it's possible that the default settings won't lead to a feasible solution. For further details, such as the meaning of variables and the possibility to change settings, use the LinTim documentation https://www.lintim.net/.

- copy all files from the folder $Visum2Lintim \rightarrow$ *instance name* to a lintim dataset folder.
- change default settings by creating a file $basis \rightarrow Private\text{-}Config.cnf$ in your dataset, which contains the following statements
  - `sl_model; all`
  - `sl_forbidden_edges; true`
  - `lpool_restrict_forbidden_edges; true`
  - `lc_respect_fixed_lines; true`
  - `lc_respect_forbidden_edges; true`
  - `tim_respect_fixed_times; true`

To create a new dataset, use the dataset template folder in LinTim to ensure the existence of required makefiles. You can create a solution typing the following commands in your *Ubuntu terminal*.

- make sure your dataset folder is set as directory in the ubuntu terminal
- `make config-fill-config-from-visum`
- `make ptn-read-infrastructure-from-visum`
- `make od-read-node-od-from-visum`
- `make sl-stop-location`
- `make od-distribute-from-nodes`
- `make ptn-regenerate-load`
- `make lpool-line-pool`
- `make lc-read-fixed-lines-from-visum`
- `make lc-line-concept`

Table 4: attributes to set for import of PuT supply

attribute	**example value**	**explanation**
scenario_name	"01_example"	Scenario prefix / folder name
instance_name	"I1"	instance prefix
lintim_version_name	"A_2b_1_2_2"	name of solution / folder name in $\_Interface \rightarrow Lintim2Visum$

- `make ean`
- `make tim-read-fixed-times-from-visum`
- `make tim-timetable`
- `make tim-transform-to-visum`

To reimport the resulting PuT supply in PTV Visum, copy the dataset folder into your Visum folder structure $\_Interface \rightarrow Lintim2Visum$ and rename it.

Procedure sequence *VIS20_Gateway_Lintim2Visum.xml*

The procedure sequence *VIS20_Gateway_LinTim2Visum* imports the PuT supply solution generated in Lintim. This is achieved by generation of a *.net file* containing lines, line routes, line route items, time profiles, time profile items and vehicle journeys. Furthermore a *.gpax file* is created to set the graphic parameters of lines. These files are named based on the solution name (**lintim_version_name** + file extension].

→ Adjust parameters mentioned in table 4 (Group "set attributes")

→ Run procedure sequence, in particular

- Generate & load ***lintim_version_name****.net file* file containing elements of PuT supply (run script *LinTimToVisumNetPublicTransport.py* in step 13).
- Generate & load ***lintim_version_name****.gpax* file containing graphic parameters (run script *LinTimToVisumNetPublicTransport.py* in step 16).

→ Save solution as **scenario_name** + **instance_name** + **lintim_version_name**.ver