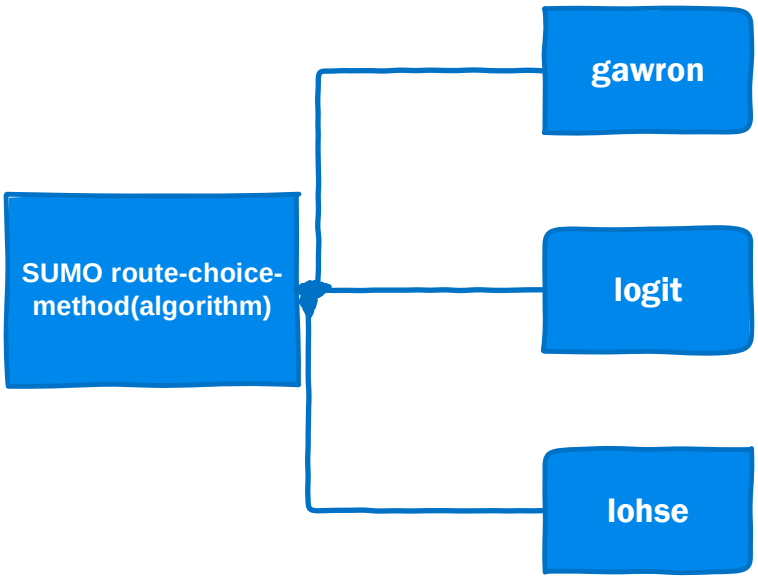
Traffic assignment methods(algorithms)

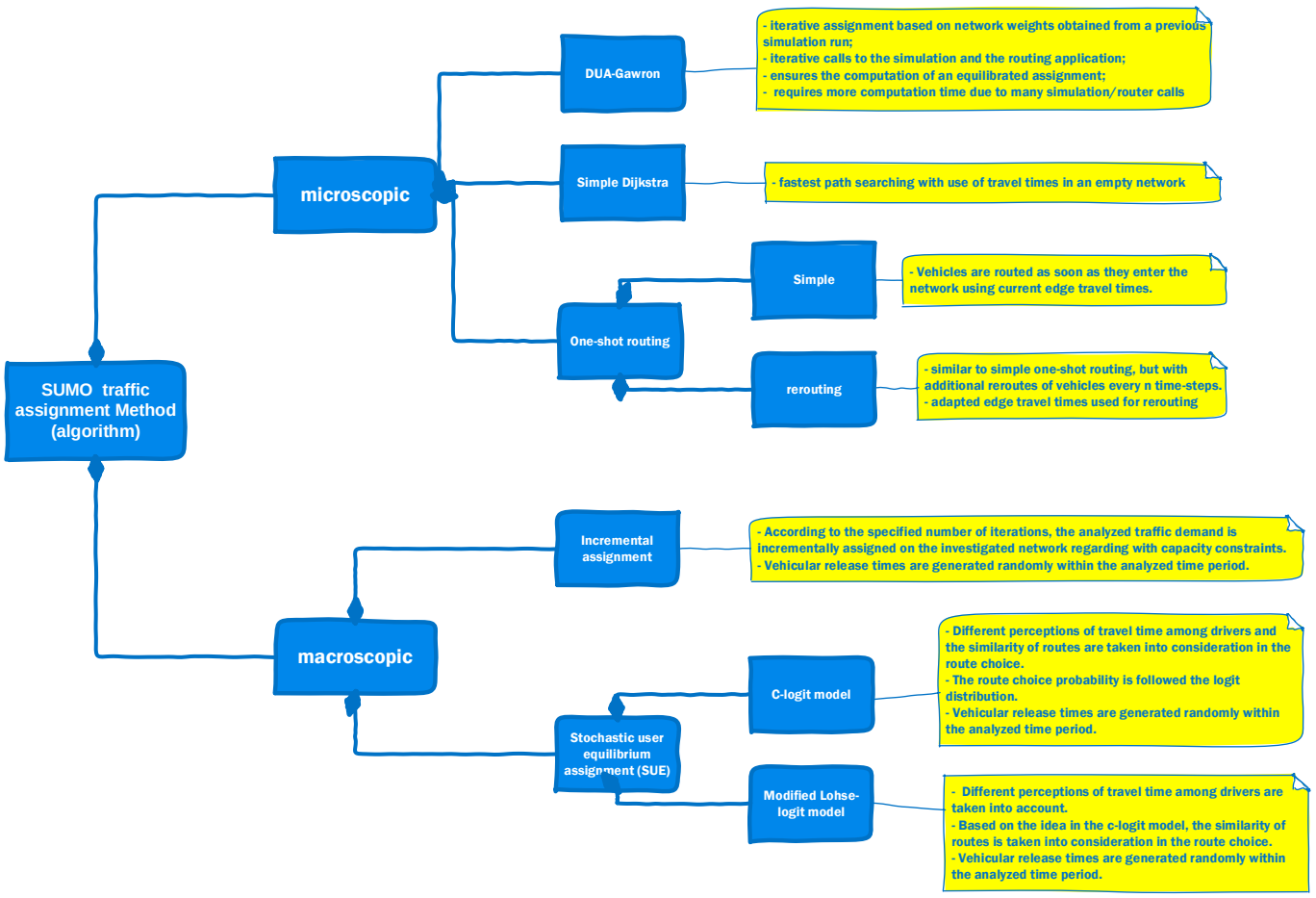
MAROUTER

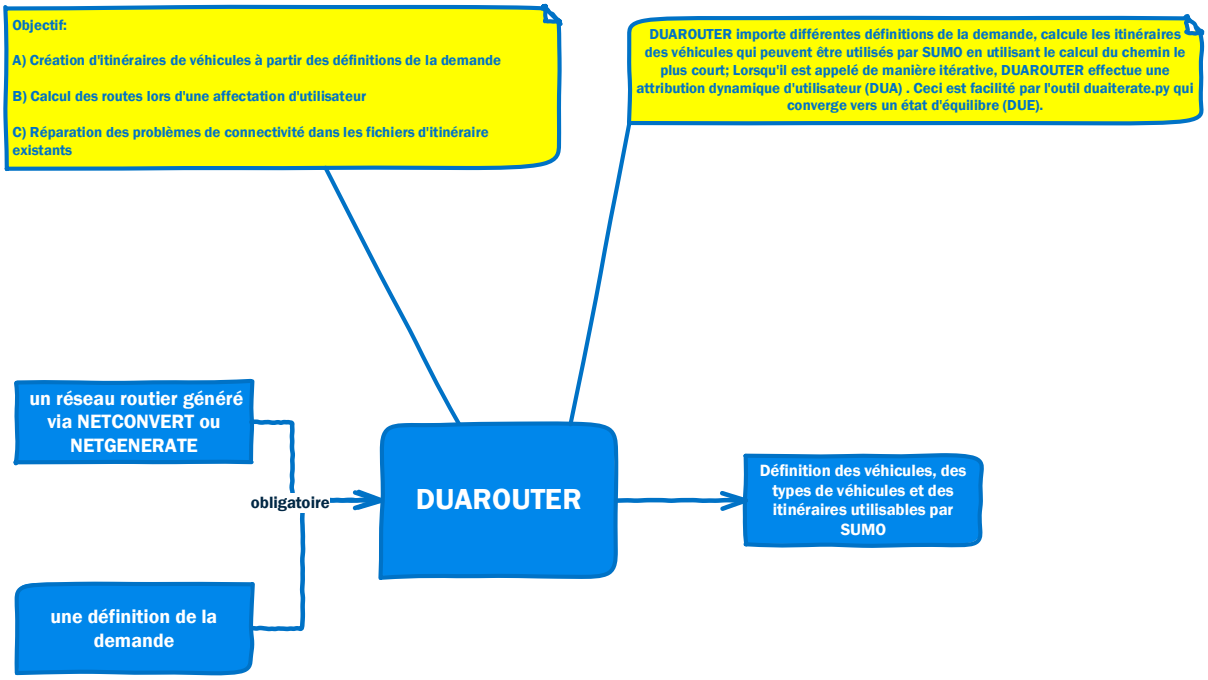
<BOOL>	
--additive-traffic *<BOOL>*	Keep traffic flows of all time slots in the net; *default:* **false**
--assignment-method *<STRING>*	Choose a assignment method: incremental, UE or SUE; *default:* **incremental**
--tolerance *<FLOAT>*	Use FLOAT as tolerance when checking for SUE stability; *default:* **0.001**
--left-turn-penalty *<FLOAT>*	Use left-turn penalty FLOAT to calculate link travel time when searching routes; *default:* **0**
--paths *<INT>*	Use INTEGER as the number of paths needed to be searched for each OD pair at each iteration; *default:* **1**
--paths.penalty *<FLOAT>*	Penalize existing routes with FLOAT to find secondary routes; *default:* **1**
--upperbound *<FLOAT>*	Use FLOAT as the upper bound to determine auxiliary link cost; *default:* **0.5**
--lowerbound *<FLOAT>*	Use FLOAT as the lower bound to determine auxiliary link cost; *default:* **0.15**
-i *<INT>* **--max-iterations** *<INT>*	maximal number of iterations for new route searching in incremental and stochastic user assignment; *default:* **20**
--max-inner-iterations *<INT>*	maximal number of inner iterations for user equilibrium calcuation in the stochastic user assignment; *default:* **1000**
--route-choice-method *<STRING>*	Choose a route choice method: gawron, logit, or lohse; *default:* **logit**
--gawron.beta *<FLOAT>*	Use FLOAT as Gawron's beta; *default:* **0.3**

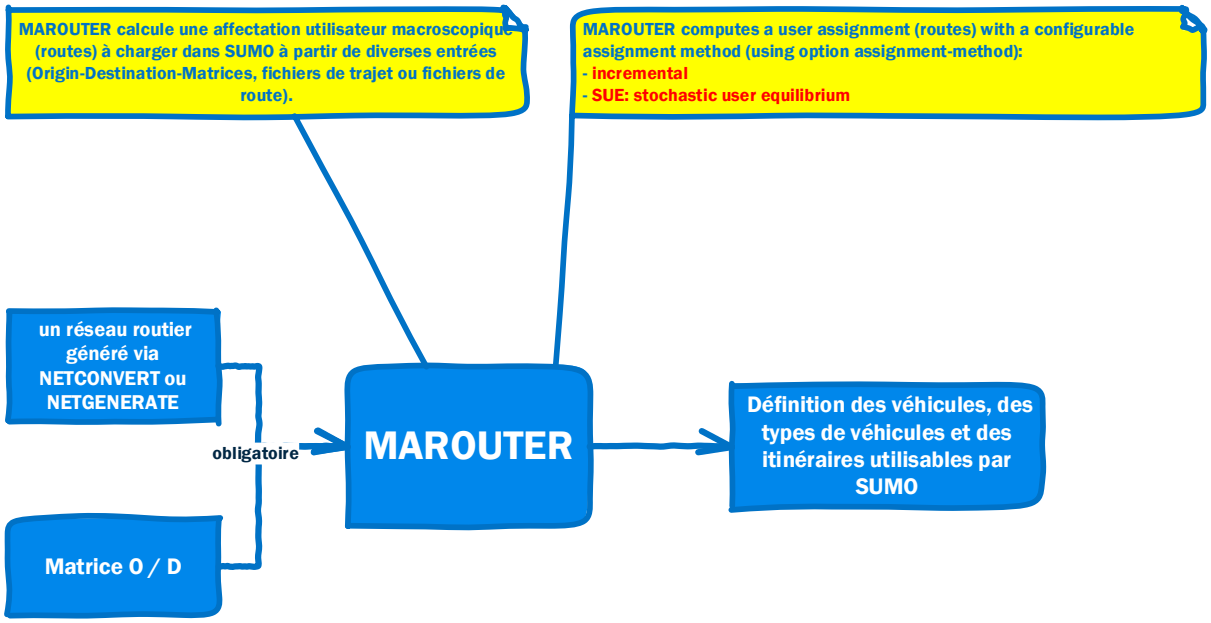
DUAROUTER

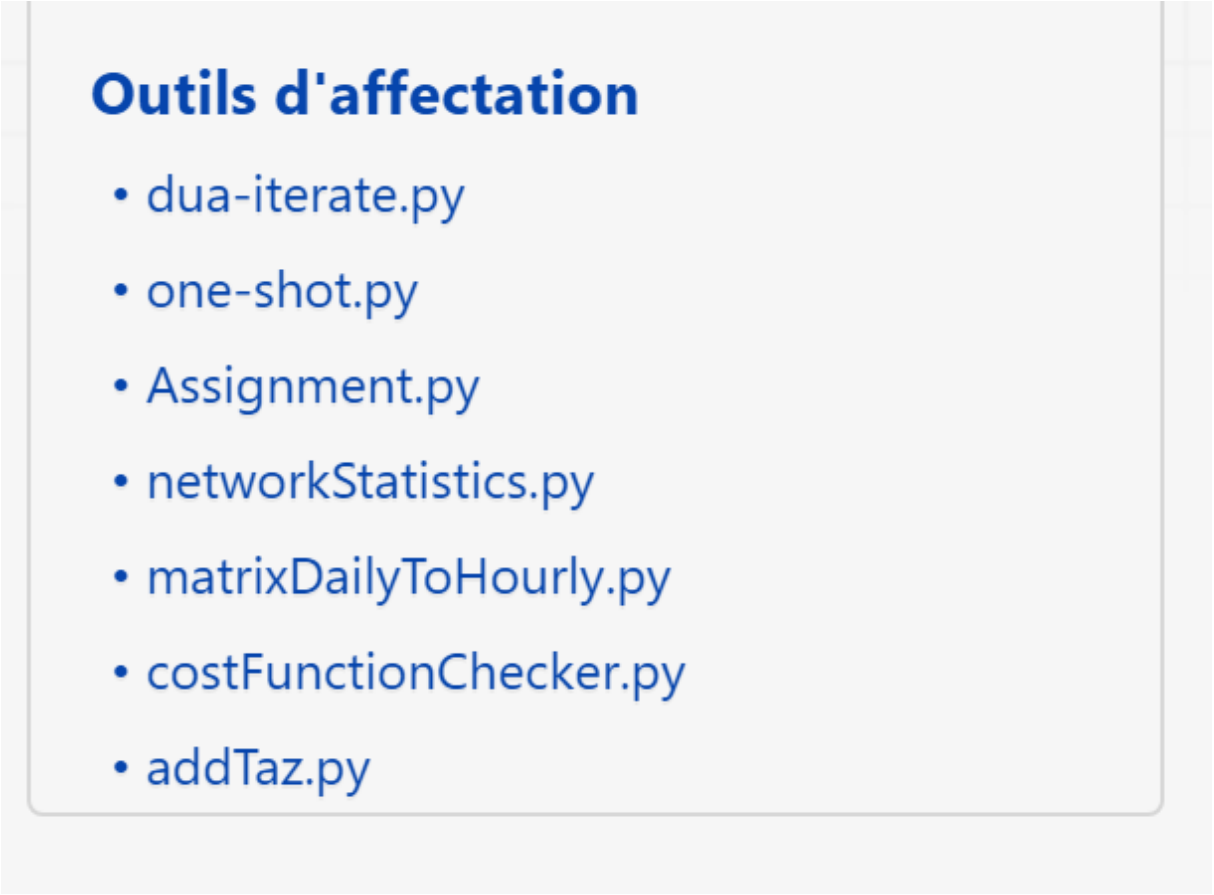
--astar.all-distances *<FILE>*	Initialize lookup table for astar from the given file (generated by marouter --all-pairs-output)
--astar.landmark-distances *<FILE>*	Initialize lookup table for astar ALT-variant from the given file
--astar.save-landmark-distances *<FILE>*	Save lookup table for astar ALT-variant to the given file
--gawron.beta *<FLOAT>*	Use FLOAT as Gawron's beta; *default:* **0.3**
--gawron.a *<FLOAT>*	Use FLOAT as Gawron's a; *default:* **0.05**
--keep-all-routes *<BOOL>*	Save routes with near zero probability; *default:* **false**
--skip-new-routes *<BOOL>*	Only reuse routes from input, do not calculate new ones; *default:* **false**
--ptline-routing *<BOOL>*	Route all public transport input; *default:* **false**
--logit *<BOOL>*	Use c-logit model (deprecated in favor of --route-choice-method logit); *default:* **false**
--route-choice-method *<STRING>*	Choose a route choice method: gawron, logit, or lohse; *default:* **gawron**
--logit.beta *<FLOAT>*	Use FLOAT as logit's beta; *default:* **-1**


SUMO route-choice-method(algorithm)
gawron
logit
lohse


SUMO traffic assignment Method (algorithm)
microscopic
DUA-Gawron
- iterative assignment based on network weights obtained from a previous simulation run;
- iterative calls to the simulation and the routing application;
- ensures the computation of an equilibrated assignment;
- requires more computation time due to many simulation/router calls
Simple Dijkstra
- fastest path searching with use of travel times in an empty network
One-shot routing
Simple
- Vehicles are routed as soon as they enter the network using current edge travel times.
rerouting
- similar to simple one-shot routing, but with additional reroutes of vehicles every n time-steps.
- adapted edge travel times used for rerouting
macroscopic
Incremental assignment
- According to the specified number of iterations, the analyzed traffic demand is incrementally assigned on the investigated network regarding with capacity constraints.
- Vehicular release times are generated randomly within the analyzed time period.
Stochastic user equilibrium assignment (SUE)
C-logit model
- Different perceptions of travel time among drivers and the similarity of routes are taken into consideration in the route choice.
- The route choice probability is followed the logit distribution.
- Vehicular release times are generated randomly within the analyzed time period.
Modified Lohse-logit model
- Different perceptions of travel time among drivers are taken into account.
- Based on the idea in the c-logit model, the similarity of routes is taken into consideration in the route choice.
- Vehicular release times are generated randomly within the analyzed time period.


Objectif:
A) Création d'itinéraires de véhicules à partir des définitions de la demande
B) Calcul des routes lors d'une affectation d'utilisateur
C) Réparation des problèmes de connectivité dans les fichiers d'itinéraire existants
DUAROUTER importe différentes définitions de la demande, calcule les itinéraires des véhicules qui peuvent être utilisés par SUMO en utilisant le calcul du chemin le plus court; Lorsqu'il est appelé de manière itérative, DUAROUTER effectue une attribution dynamique d'utilisateur (DUA) . Ceci est facilité par l'outil duaiterate.py qui converge vers un état d'équilibre (DUE).
un réseau routier généré via NETCONVERT ou NETGENERATE
obligatoire
une définition de la demande
DUAROUTER
Définition des véhicules, des types de véhicules et des itinéraires utilisables par SUMO


MAROUTER calcule une affectation utilisateur macroscopique (routes) à charger dans SUMO à partir de diverses entrées (Origin-Destination-Matrices, fichiers de trajet ou fichiers de route).
MAROUTER computes a user assignment (routes) with a configurable assignment method (using option assignment-method):
- incremental
- SUE: stochastic user equilibrium
un réseau routier généré via NETCONVERT ou NETGENERATE
obligatoire
Matrice O / D
MAROUTER
Définition des véhicules, des types de véhicules et des itinéraires utilisables par SUMO



https://sumo.dlr.de/docs/Tools/Assign.html#assignmentpy