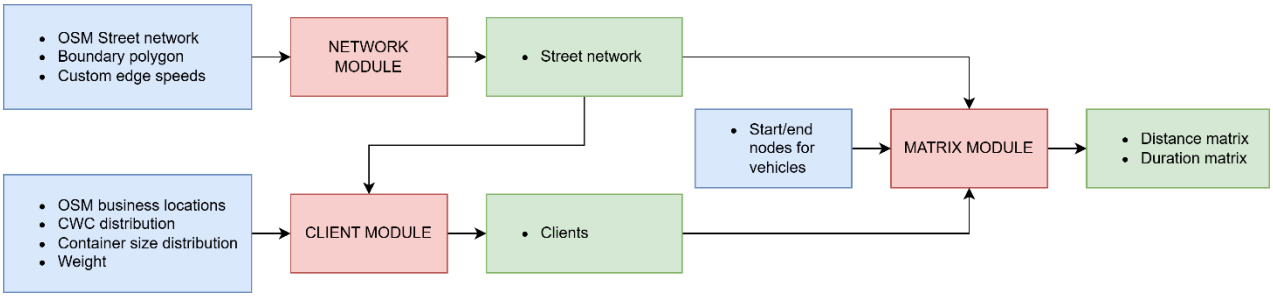
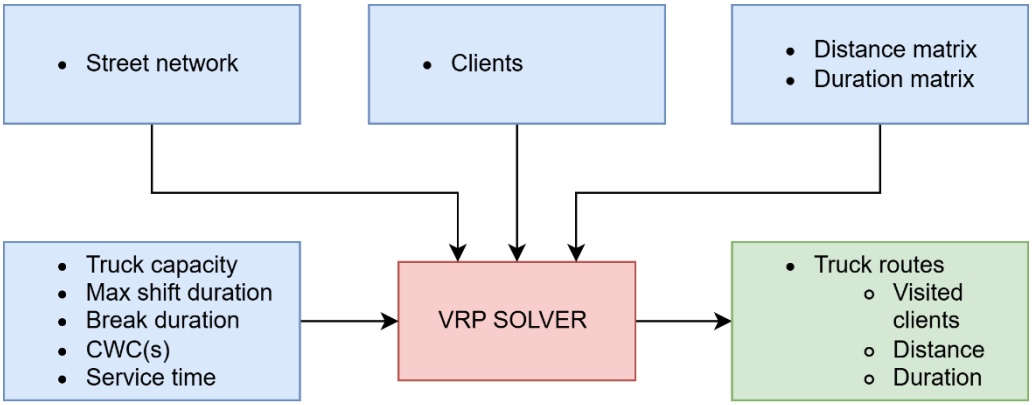
Overzicht
De code genereert 3 belangrijke input bestanden: het stratennetwerk, het klantenbestand en de afstand/tijd matrices. Zie hieronder.



Deze input wordt gebruikt door de VRP solver om routes uit te rekenen.



Bestanden
Onder een overzicht van de Python bestanden.

1. **generate_input.py**
   Genereert het stratennetwerk en vervangt de max speeds door realistische gemiddelde snelheden. Verder laadt het alle plekken in het gebied met de volgende OpenStreetMap (OSM) tags:

   {'office':True,'shop':True, 'amenity':['restaurant','bar','cafe','fast_food','food_court','pub','school']}

   Deze plekken worden gebruikt als klantenlocaties.Er wordt een polygon gebruikt om het gebied wat geladen is uit OSM precies af te snijden tot de gewenste study area.

2. **clients.py**
   Beperkt het aantal klanten tot het gewenste aantal (in dit geval 1550) en verrijkt het klantenbestand met de volgende data:

   - Containergrootte per klant (en daarmee service time en gewicht)
   - Inzamelaar per klant
   - Containerlocatie per klant (dichtstbijzijnde plek op het stratennetwork omdat klanten niet op het stratennetwerk liggen)

3. matrix_calculator.py en run_matrix_calculator.py
De matrix_calculator.py creëert de afstand/tijd matrix en wordt opgeroepen door run_matrix_calculator.py. **Deze moet je runnen in external system terminal.**

De input is een numpy array met nodes op het stratennetwerk (in dit geval allCWC_entryv3_nodes.npy) en de toegepaste weight (*length* voor de afstand of *travel_time* voor de reistijd).

Van de array met alle nodes (depots en containers van klanten) wordt een lijst gemaakt met alle mogelijke *pairs*. Ofwel, als er 3 klanten A,B,C zijn worden de volgende paren gegenereerd:

AA, AB, AC, BA, BB, BC, CA, CB, CC

Van ieder paar wordt de afstand/tijd uitgerekend. Het script gebruikt parallel processing om het uitrekenen te versnellen.

Let op, het uitrekenen van 1 matrix kost veel tijd, zo'n 8 uur voor 1550. De tijd om een matrix te genereren schaalt kwadratisch met het aantal klanten.

4. route_calculator.py
Dit is de VRP solver die de routes uitrekent en plot.

Definities:

waste_collector: dit geeft aan welke inzamelaars je uit kunt kiezen (puur voor de gebruiker). De letters L,M,S geeft de grootte van een inzamelaar aan.
collector: de (klanten van de) inzamelaar die gesimuleerd wordt. Mag 1 of meer zijn.
vehicle_list: start en eindpunt van de voertuigen. 0 = een voertuig begint op molenvliet, 1 = een voertuig begint in Rdam Zuid, 2 = een voertuig begint in Rdam Noord. Het aantal getallen geeft het aantal beschikbare voertuigen aan. Bijvoorbeeld [1,1,2] betekent dat er 3 voertuigen beschikbaar zijn waarvan 2 uit Rdam Zuid en 1 vanuit Rdam Noord.
capacity: de capaciteit van een vrachtwagen in kg
max_shift_duration: max duur van een shift in uur
break_duration: duur van de pauze in minuten
time_limit_meta: tijdslimiet voor de VRP solver (langer laten runnen is betere oplossing)

Optioneel: vanaf line 330 kunnen er constraints geïmplementeerd worden. Deze constraints kunnen alleen over individuele voertuigen geïmplementeerd worden en niet over groepen voertuigen. De constraint houdt in dat ieder voertuig een x aantal klanten bediend of x hoeveelheid geld verdient.

Met search_parameters.log_search = True wordt de log van de VRP solver weergegeven.

De route calculator genereert vier bestanden:

- Een lijst met alle nodes van alle klanten
- Een lijst met de gereden routes
- Een jpg bestand met de routes gevisualiseerd
- Een txt bestand met route data per voertuig.

Requirements – important package versions

matplotlib-base	3.8.4
matplotlib-inline	0.1.7
networkx	3.3
numpy	1.26.4
ortools	9.11.4210
osmnx	1.9.4
pandas	2.2.3
geopandas	0.14.4