Indholdsfortegnelse

Algoritmer og databehandling leveret af Droids Agency

Simpel simulering

Den simple simulering tager udgangspunkt i den valgte lokation, hvorved den nuværende pulje af biler samt de tilknyttede ture vælges til simulering. Herudover har brugeren valgt en pulje af biler, der simuleres op imod den nuværende pulje. De valgte biler sorteres og prioriteres på baggrund af følgende kriterier; 1) cykler vil ligge først tilgængelig, 2) elcykler prioriteres på elforbrug, 3) elbiler prioriteres på elforbrug, Wh/km og 4) biler prioriteres på brændstofforbrug km/l.

Den valgte datoperiode splittes i tidsrum á et minut, som definerer køretøjernes mulige bookinger. Det betyder at et køretøj, der afslutter en tur klokken 12:15:16, ikke kan bookes til en tur der starter klokken 12:15:30. For hver tur tjekkes den tilgængelige køretøjspulje i den sorterede rækkefølge. Et køretøj kan acceptere på baggrund af en række kriterier der skal være opfyldt;

Gælder alle typer af køretøjer:

- Skal være ledige.
- Skal kunne køres på den tilladte distance for et køretøj.

Cykler:

- Skal være inde for det tilladte tidsrum
- Skal tildeles x procent af gangene den kvalificerer

Elbiler:

- Skal kunne hvile/lade det valgte antal timer hvert døgn

Trin 5, Tabu Search

Målet i trin 5 er at finde (næsten) optimale flådesammensætninger på baggrund af brugerinput. Fordi der arbejdes med to mål; CO2e ton/år og omkostning/år, skal der bruges multiobjektiv optimering, hvilket algoritmen tabu search, med lidt modifikationer, kan udnyttes til. Brugeren leverer målene, hvorved søgningen på løsninger starter. Tabu search arbejder i sammenspil med enten den simple simulering eller den intelligente simulering for at validere at kørselsbehovet kan tilfredsstilles og for at få nøjagtige tal på udledning og omkostninger.

Ud fra en given startløsning generer algoritmen naboløsninger – altså løsninger der ligger op ad den første løsning, men hvor den har skiftet enkelte køretøjer ud. Løsningerne tjekkes mod de tidligere træk, og om det ligger i den forbudte liste (tabu list, deraf navnet), men de tjekkes også på målene samt om de kan tilfredsstille kørselsbehovet. Den bedste naboløsning vælges og de træk der ligger til grund for den nye bedste løsning lægges i den blokerede liste for at undgå samme træk igen. Der arbejdes med forskellige kriterier for at undgå hele udfaldsrummet søges. Udfaldsrummet vokser hurtigt for hvert tilgængeligt køretøj og antallet af køretøjet der skal søges over. Ud fra den bedste naboløsning gentages processen indtil det definerede udfaldsrum er udtømt eller at stopkriteriet er opfyldt.

Tabu Search, fortsat – teknisk

Det er altafgørende for en effektiv søgning at begrænse antallet af mulige kombinationer af flåden. Følgende proces foretages før tabu search algoritmen starter:

- Lås biler i flådekombinationerne, således at biler med fortsat leasing ikke kan udskiftes
- Indlæs en repræsentation af puljens kørsel, en gennemsnitsdag samt en dag med høj belastning.
- For hver bil udregnes følgende værdier over 30000 km:
    - CO2e: udledning for køretøjet (POGI-metode)
    - Cost: omkostningerne for køretøjet inkl. driftsomkostning og samfundsøkonomiske omkostninger (POGI-metode)
    - Fitness værdi: sammenhæng mellem CO2e og omkostning (der foretages en vægtning ud fra brugerens input)
- Rekursiv søgning til definition af mindst antal biler nødvendig for tilfredsstillelse af kørselsbehovet:
    - Køretøj med længst rækkevidde vælges
    - Tjek antal ukørte ruter ved nuværende antal biler
    - Divide and conquer: hvis kørselsbehovet er tilfredsstillet, formindsk antallet af biler mellem det mindste tal, der kunne tilfredsstille kørselsbehovet, og det nuværende antal. Hvis kørselsbehovet ikke er tilfredsstillet, forøg antallet af biler mellem det nuværende tal, og det mindste tal der kunne tilfredsstille kørselsbehovet.
- Byg løsning således at de låste biler er til stede og det mindst nødvendige antal af biler nået. Løsningen repræsenteres med et array, hvor hver biltype har sit eget index.
- Generer mulige flådekombinationer
    - Sæt mål af antal biler til det mindst nødvendige plus 20%
    - Tilføj index, der virker som slack for at diversificere de genererede løsninger
    - Kombinationerne tvinges til at inkludere minimum de låste biler og antallet af mindst nødvendige biler
    - For hver type der tilføjes til kombinationerne tjekkes det om løsningerne overstiger de angivne mål, hvorved de fjernes og ikke færdiggøres.
- De genererede løsninger rangeres ved at tage produktet med fitnessværdien.

Den ovenstående søgning sker på en begrænset periode af data, som er en repræsentation af det kørte på lokationen. Den bedst rangerede løsning på udsnittet af data, er ikke nødvendigvis den bedste eller en løsning der tilfredsstiller kørselsbehovet på hele dataperioden – naturligvis fordi kørslen varierer en smule fra dag til dag.

Den bedste rangeret løsning vælges, og der generes en ”tabu struktur”, som er de mulige træk på index-niveau. Trækkene evalueres på det fulde data for den valgte periode, hvor den bedste af disse vælges. Såfremt den bedste findes i tabulisten, vælges trækket kun hvis den er bedre end den nuværende bedste løsning. For at minimere mængden flådesammensætninger, der skal tjekkes på den fulde mængde data, tjekkes det om algoritmen i længere tid bevæger sig ned ad en sti, der ikke forbedrer udgangspunktet. Hvis der efter 10 træk ikke er fundet en bedre fitness værdi forlades de nuværende naboløsninger.

Data

Startsteder

Simuleringsværktøjet og ikke mindst data er afhængig af præ-definerede startsteder i kommunen. For hver lokation hentes en adresse og GPS-koordinater til lokationen. For at forbedre præcisionen af logningen, er det anbefalet for lokationer med flere parkeringsmuligheder at levere flere GPS-koordinater for samme lokation. Disse startsteder bliver defineret som puljer, og bliver de mulige lokationer, der kan vælges i simuleringsværktøjet.

Startsteder indeholder:

- id
- Adresse
- GPS Koordinater

Bildata

For hver bil, der er i de valgte puljer, bliver der hentet metadata fra flådestyringssystemerne. Da det varierer hvilken mængde der bliver leveret herfra, skal disse data beriges af brugerne. Som minimum skal der for et køretøj være forbrug, enten WLTP-el (Wh/km) eller WLTP-fossil (Wh/km), og omkostning pr. år. Omkostning pr. år er den værdi man forventer at skulle betale fast hvert år over leasingperioden, men eksklusivt brændstofforbrug da simuleringsværktøjet udregner forbruget ud fra de allokerede ruter i simuleringen. For køretøjer kan der noteres en rækkevidde, som især er vigtig for elbiler. Her kan der også tilføjes en kapacitetsnedskrivning for at realisere elbilens egentlige rækkevidde i eksempelvis koldt vejr.

Bildata indeholder:

- Id
- Registreringsnummer
- Mærke
- Model
- Køretøjstype (elbil, fossilbil, elcykel, cykel)
- Drivmiddel (el, benzin, diesel, hybrid etc.)
- WLTP-fossil, km/l
- WLTP-el, Wh/km
- Kapacitets nedskrivning
- CO2 gr. udledning pr. km
- Rækkevidde
- Omkostning pr. år
- Start sted/lokation id
- Start leasing
- Slut leasing
- Leasing type
- Tilladt km/år
- Opladningstid

Tur data

Turene der simuleres på, kommer fra flådestyringssystemer og baserer sig på individuelle GPS-logninger fra køretøjet er blevet tændt til det er slukket. Det kan ikke umiddelbart bruges til simulering, da man så vil tildele køretøjer ture, der ikke nødvendigvis ender og slutter det samme sted; på hjemmelokationen. Derfor foretages en aggregering af logninger til egentlige ture, som baserer sig på en række regler.

- En tur kan starte, når køretøjet befinder sig indenfor 100 meter af et startsted
- En tur skal altid stoppe det samme sted, som den er startet +-100 meter.

- En tur allokeres til den lokation, hvor køretøjet der har kørt den hører til. (Dvs. en bil der bliver udlånt til at køre andre steder for en periode, vil tage turene "med hjem")
- Der findes fejl i GPS-koordinaterne, hvilket influerer aggregeringen. Det mitigeres ved at; 1) fjerne logninger der har samme sluttidspunkt, 2) fjerne logninger der viser at en bil har været tændt i mere end 24 timer, 3) erstatte distance på logninger med fugleflugt distance såfremt fugleflugt er længere end den noterede distance.
- En tur er defineret ved minimum distance på 0,5 km.
- En tur, der varer længere end 17 timer (eksempelvis fordi bilen har kørt fra steder, der ikke er et registreret startsted), splittes op til flere ture. For hver gang bilen har holdt stille i en time eller mere afsluttes en tur.

Det er vigtigt at notere, at der simuleres på ture og ikke vagtplaner. Dvs. køreplanen, der ligger til grund for simuleringen, ikke tager højde for at en vagt skal fuldføres i det samme køretøj.

Turdata indeholder:

- Start tidspunkt
- Slut tidspunkt
- Start GPS-koordinater
- Slut GPS-koordinater
- Distance
- Bil id
- Startsted/lokation id

POGI

Miljøværktøjet udviklet i POGI-regi bruges til at beregne CO2e, driftsomkostninger og samfundsøkonomiske omkostninger, som bruges bredt i simuleringsværktøjet. CO2e er valgt for at kunne sammenligne el – og fossilbiler, da elbilerne ved blot CO2 gr. udledning pr. km. ikke vil have nogen udledning.

Følgende værdier er sat i udregningen af CO2e:

- El: 0,09 kg CO2e/kWh
- Benzin: 2,52 kg CO2e/liter[1]
- Diesel: 2,98 kg CO2e/liter[2]

Følgende værdier er sat i udregning af driftsomkostninger:

- El: 2,13 kr./kWh
- Benzin: 12,33 kr./liter
- Diesel: 10,83 kr./liter

Følgende værdier er sat i udregning af samfundsøkonomiske omkostninger:

- 1500 kr. pr ton CO2e

[1] IPCC's Fifth Assessment Report, fra 2014 – fra miljoestyrelsen-tco-vaerktoej-motorkoeretoejer-operationel-leasing-2021.xlsm
[2] IPCC's Fifth Assessment Report, fra 2014 – fra miljoestyrelsen-tco-vaerktoej-motorkoeretoejer-operationel-leasing-2021.xlsm

Ukørte ture

Ture der ikke bliver allokeret, bliver straffet i simuleringen ved at antage, at disse ture bliver kørt af en bil, der ikke er en del af flåden. Denne bil er en standard fossil/benzin bil, der kører 20 km/l. De ukørte km. bliver ligeledes opsummeret fra den valgte dato-periode til et år. Udover tillægget på udledning, kommer der også "kørepenge" for de ukørte ture. Her gælder det, at de første 840 kilometer betales en høj takst på 3,44 kr./km. og en lav takst på 1.9 kr./km. for de efterfølgende kilometer.

Algoritmer leveret af Qampo

I det følgende beskrives de algoritmer, som er understøttet af Qampos algoritmer i forbindelse med leverancen til Intelligent Flådestyring. Dette dækker ikke nødvendigvis over alle de funktioner, som rent faktisk er implementeret i front-enden, og derfor er tilgængelige for brugerne.

Formål og forudsætninger

"Trin 2" algoritmen sørger for at ens bilflåde bliver udnyttet på den bedst mulige måde med hensyn til løbende omkostninger (brændstof, slitage, osv.) samtidig med at CO2 udledningen bliver så lav som mulig.

En nødvendighed for at køre algoritmen er kendskab til følgende input:

- Ens flåde med antal og type af hver bil, herunder viden om de løbende omkostninger per kilometer og CO2 udledning per kilometer. Der kan også angives en maksimumrækkevidde bilen kan køre om dagen (af hensyn til enten batteristørrelse ved elbiler eller f.eks. kilometer antal på leasingkontrakt).
- Ture der skal planlægges på en dag for et givent sted hvor flåden er placeret. For hver tur skal vi vide hvornår på dagen den starter og slutter, samt hvor langt der er kørt. Det antages at turen altid starter og slutter på stedet hvor flåden er placeret.

Algoritmen afvejer de løbende omkostninger med tons CO2 udledning via en parameter som også kan varieres, f.eks. til at simulere forskellige priser på CO2 kvoter.

Outputtet fra algoritmen er en plan for hver bil med ture den skal køre.

Algoritmen kan bruges til at simulere effekten af en anden flådesammensætning på historisk data eller sammenligne virkeligheden med den optimale måde at udnytte ens eksisterende flåde på.

Algoritmens tekniske detaljer

I udviklingen af "Algoritmen" er der udviklet og eksperimenteret med tre forskellige algoritmer som hver især kan ligge hele puslespillet.

De to af dem opnår det samme, nemlig en *eksakt optimal løsning*, men via forskellige metoder, nemlig henholdsvis via *integer programming* og *contraint programming.* Hver især har de deres fordele i forhold til hvad de kan modellere hvilket er vigtigt i forhold til videreudvikling af nye *features*. Disse algoritmer er bygget på Google's *OR-Tools* kode og bruger *state-of-the-art* teknologi indenfor *operations research*.

Den tredje algoritme er en *greedy* algoritme som ikke opnår det bedste resultat, men i stedet simulerer hvordan tildelingen af biler foregår mange steder i dag. Den kan altså bruges som en *benchmark* til de to *eksakte algoritmer.* Denne algoritmer rangerer alle ture efter starttidspunkt og tildeler til den billigste ledige bil.