MODELO DE ??

Identificação do problema:

Nesta vertente de utilização da app, pretende-se disponibilizar ao utilizador o ponto de recolha mais próximo em que pode depositar o tipo e a quantidade de lixo que pretende, bem como o itinerário para lá chegar.

Dados de entrada:

* type – tipo de lixo que se pretende depositar (plástico, vidro, papel, etc)
* quantity – quantidade de lixo que se pretende depositar
* G = (V, E) – grafo dirigido pesado, composto por:
  + V – vértices, que representam interseções, compostos por:
    - Dist – distância a L
    - Adj ⊆ E – conjunto de arestas que partem do vértice
  + E – arestas, que representam vias de comunicação, compostas por:
    - W – peso da aresta (distância entre os dois vértices)
    - ID – identificador único da aresta
    - Vi ∈ V – vértice de partida
    - Vf ∈ V – vértice de chegada
* P – conjunto de pontos de recolha, cada um caracterizado por:
  + type – tipo de lixo que recolhem
  + Q – quantidade do lixo que lá está
  + Qmáx – quantidade máxima do lixo que pode lá estar
  + R ∈ E – aresta onde está
  + D – distância ao vértice de chegada da aresta
* L – localização do utilizador, caracterizada por:
  + R ∈ E – aresta onde está
  + D – distância ao vértice de chegada da aresta

Dados de saída:

* Pf ∈ P – ponto de recolha do tipo correspondente mais próximo do utilizador com capacidade para a quantidade de lixo que se pretende depositar
* C = {e ∈ E : 1 ≤ i ≤ |E| } – sequência de arestas, correspondente ao caminho a percorrer (ei = i-ésima aresta)

Restrições:

* Dados de entrada:
  + quantity > 0, visto que representa peso ou volume
  + ∀ p ∈ P, Q ≥ 0 ∧ Qmáx > 0, pelo mesmo motivo
  + ∀ e ∈ E, W > 0, visto que o peso das arestas representa uma distância
  + ∀ L ∧ ∀ p ∈ P, D ≥ 0, pelo mesmo motivo
  + ∀ v ∈ V, Dist ≥ 0, pelo mesmo motivo
* Dados de saída:
  + R(Pf) = efinal

Funções objetivo:

A solução ótima do problema passa por minimizar a distância que o utilizador tem de percorrer para depositar o lixo, de modo que pretende-se minimizar a seguinte função:

f = ∑c∈C W(c)

MODELO DE NEGÓCIO

Identificação do problema:

Nesta vertente da app, pretende-se disponibilizar ao utilizador um itinerário desde a sua localização até à localização da central de reciclagem, passando por domicílios que tenham o tipo de resíduos que pretendem recolher, sem nunca ultrapassar a quantidade possibilitada pelo utilizador.

Dados de entrada:

* type – tipo de lixo que se pretende recolher (plástico, vidro, papel, etc)
* quantity – quantidade de lixo que se pretende recolher
* G = (V, E) – grafo dirigido pesado, composto por:
  + V – vértices, que representam interseções, compostos por:
    - Dist – distância a Li
    - Adj ⊆ E – conjunto de arestas que partem do vértice
  + E – arestas, que representam vias de comunicação, compostas por:
    - W – peso da aresta (distância entre os dois vértices)
    - ID – identificador único da aresta
    - Vi ∈ V – vértice de partida
    - Vf ∈ V – vértice de chegada
* Pi – conjunto de domicílios que pretendem recolha, cada um caracterizado por:
  + type – tipo de lixo que recolhem
  + Q – quantidade do lixo que lá está
  + R ∈ E – aresta onde está
  + D – distância ao vértice de chegada da aresta
* Li – localização do utilizador, caracterizada por:
  + R ∈ E – aresta onde está
  + D – distância ao vértice de chegada da aresta
* Lf – localização da central de reciclagem:
  + R ∈ E – aresta onde está
  + D – distância ao vértice de chegada da aresta

Dados de saída:

* C = {e ∈ E : 1 ≤ i ≤ |E| } – sequência de arestas, correspondente ao itinerário a percorrer (ei = i-ésima aresta)
* Pf = {p ∈ P : 1 ≤ i ≤ |P| } – sequência de domicílios de onde o utilizador deve recolher resíduos

Restrições:

* Dados de entrada:
  + quantity > 0, visto que representa peso ou volume
  + ∀ p ∈ P, Q ≥ 0, pelo mesmo motivo
  + ∀ e ∈ E, W > 0, visto que o peso das arestas representa uma distância
  + ∀ L ∧ ∀ p ∈ P, D ≥ 0, pelo mesmo motivo
  + ∀ v ∈ V, Dist ≥ 0, pelo mesmo motivo
* Dados de saída:
  + ∀ p ∈ Pf, R(p) ∈ C
  + ∑p∈Pf Q(p) < quantity

Funções objetivo:

A solução do problema passa por maximizar a quantidade recolhida, sem ultrapassar a quantidade possível de recolher, e por minimizar a distância percorrida, dando prevalência ao primeiro critério. Assim, queremos minimizar as seguintes funções:

f = quantity - ∑p∈Pf Q(p) (sendo que nunca pode ser < 0)

g = ∑c∈C W(c)