Agent Type: Smartphone Advisor

Specifica dell’ambiente

* Performance
  + L’algoritmo considera i seguenti parametri:
    - Minimizzare lo scarto di:
      * Battery
      * Performance
      * Display
      * Camera
    - Massimizzare la data di uscita
      * Es: a parità di altri parametri, uno smartphone uscito nel 2019 sarà scelto al posto di uno uscito nel 2016
      * Preference Sorting
* Environment
  + Una collezione di “specs” (datasheet semplificati di smartphone)
    - Chiedere al prof! => caratteristiche dell’ambiente
* Actuators
  + Mostrare a video un set di “specs” conformi
* Sensors
  + Input di campi
  + Database
    - Noi leggiamo i dati da un form e il sistema manda il dataset dal db

Idea: usare un **algoritmo genetico**

x = lunghezza stringa finita

L’**individuo** è un’array di smartphone con x item all’interno.

F(s) = f(s[0])+ f(s[1])+ f(s[2])+ f(s[3])+ f(s[4])

Ci sono 4 funzioni obiettivo (battery, performance, display, camera).   
La funzione obiettivo deve minimizzare lo scarto. La funzione ritorna il valore dello scarto (Ad esempio con due input (3,3.5) la funzione ritornerà 0.5). L’obiettivo è minimizzare questo scarto.

Dobbiamo usare un’altra funzione di fitting che **massimizza la data d’uscita**?

Una funzione di ordinamento (Preferece Sorting in base alla data d’uscita) viene usata per fare filtraggio.  
Serve farlo nell’agente o alla fine?  
- Se lo facciamo nell’agente => vengono “scartati” quelli più vecchi

Dobbiamo scegliere:

1. Come fare il crossover => Single point
2. Mutazione => Random resetting => prendiamo uno smartphone a caso dal dataset
3. Terminazione => Tempo di esecuzione (max 4 secondi)
4. Preinizializzazione => Facciamo lo shuffle della lista prima di creare gli individui
5. Selezione degli individui => Roulette Wheel
6. Come scelgo chi fa crossing con chi? => Dovrebbero almeno invitarsi a cena una sera

Vincolo di esistenza => un individuo non può avere lo stesso smartphone + volte. Questo problema si può presentare in fase di mutazione e può andare avanti anche in fase di crossover.

Usiamo l’Elitism per mantenere in memoria gli individui più promettenti dopo ogni iterazione.