1. Introduzione e contesto del problema

Shall we go è una piattaforma che fa del suo basarsi sulla community degli utenti dei suoi servizi la sua base. Data la sua natura, questa è stata una scelta quasi obbligata da compiere, descritta in maniera sommaria anche nel documento contenente il problem statement. Se è vero che esistono servizi come Moovit o lo stesso Google Maps che forniscono un’ottima mole di dati riguardante il *topic* che Shall we Go approccia, ovvero quello dei trasporti e tutto ciò che li riguarda, è altrettanto vero che questi ultimi presentano dei limiti che si fanno evidenti soprattutto quando si vanno a trattare aziende di trasporto medio-piccole che non comunicano (o non possono comunicare) i dati sulla loro organizzazione delle fermate e delle corse che espletano.

Nel caso specifico dell’Agro Nocerino-Sarnese, per presentare un esempio pratico, operano diverse aziende di trasporto private (come la ex *Buonotourist* o la *Leonetti & Gallucci*). Nelle due applicazioni sopracitate, tuttavia, i loro punti di fermata (che molto spesso differiscono da quelli delle aziende statali, come Busitalia Campania e la SITA) e lo scheduling delle loro corse non sono minimamente menzionati in quanto queste ultime verosimilmente non li comunicano ai gestori delle piattaforme. Ed è in questo caso che entra in gioco il concetto di crowdsourcing su cui Shall we Go si basa.

Gli utenti, di conseguenza, giocano un ruolo fondamentale per il funzionamento ed il successo della piattaforma, perché possono applicare le loro conoscenze al servizio degli altri segnalando la presenza di fermate in un determinato punto e le linee che le sfruttano.

Tuttavia, anche questo approccio non è esente da problemi di natura pratica. Quando si mette nelle mani del pubblico la possibilità di contribuire a dei dati che poi sono destinati ad essere consultati anche da altre persone, non è sempre possibile assumere la buona fede di chi effettua delle segnalazioni, in quanto potrebbero verificarsi episodi di *trolling* o di vero e proprio vandalismo, come accade di tanto in tanto sulle voci dell’enciclopedia online Wikipedia. È stato quindi da subito necessario pensare ad un qualche tipo di contromisura a questo fenomeno. La prima feature che è stata presa in considerazione è quelal che consiste nel prevedere una sorta di “carriera” all’interno della piattaforma che funziona in modo simile al social *Reddit* in cui gli utenti sono caratterizzati da un certo livello di “Karma”, che certifica la loro attività derivata dalla permanenza sulla piattaforma ed il contributo che esso apporta (nel caso di Reddit, in termini di post, upvote o commenti ai post nelle varie boards). Vien da sé che quanto più un utente contribuisce alla piattaforma, tanto più il suo *karma* aumenta. Un approccio del genere potrebbe essere adottato anche in Shall we Go. Ad esempio, quando un utente effettua la segnalazione di una fermata in un determinato punto, essa verrà analizzata da uno (o più) utenti “fidati” in base a diversi fattori come karma, tempo di permanenza sulla piattaforma e posizione geografica (uno dei temi principali che saranno affrontati più avanti in questo documento). Nel caso questa segnalazione si riveli corretta, essa verrà integrata nella piattaforma, chi segnala verrà notificato e “premiato” con aumento del proprio karma.

Il karma di un utente, quindi, può fungere da discriminante sulla sua possibilità di avere “potere decisionale” sulle segnalazioni che arrivano, garantendo quindi il mantenimento di un certo livello di qualità sui dati presenti sulla piattaforma.

1. Come approcciarsi al problema della scelta degli utenti verificatori? Una proposta.

Fino ad ora, tuttavia, si è discusso molto sul problema di come garantire la qualità dei dati ma poco dell’infrastruttura “under the hood”. In particolare, sorge immediatamente la questione prettamente tecnica del come andare a scegliere chi devono essere gli incaricati ad esaminare le segnalazioni di ciascun utente. Banalmente, un utente di Milano ha una probabilità decisamente minore di saper analizzare la segnalazione di una fermata che si trova nel comune di Fisciano rispetto ad un utente che abita in quel comune o comunque poco lontano da lì.

Oltre ad un criterio prettamente tecnico (che è quello della posizione geografica del verificatore), c’è poi da affrontare tutto il discorso sul karma e sul periodo di appartenenza alla piattaforma. Una situazione del genere, specialmente se ci si proietta in un contesto con molti utenti e quindi su larga scala, non sempre è approcciabile con una semplice ricerca lineare tra tutti gli utenti della piattaforma, al netto di un possibile lavoro di preprocessing (che sarà discusso in un’appendice di questo documento e che comunque non sempre può andare a buon fine). Nel contesto del corso di *Fondamenti di Intelligenza Artificiale*, tenuto per la prima volta nel corrente anno accademico (2020/2021) dai prof. Fabio Palomba e Fabio Narducci, sono stati discussi vari tipi di algoritmi di ottimizzazione del processo di ricerca, compresa una classe di algoritmi che si presta molto bene a questo tipo di problema: gli Algoritmi Genetici (da qui in avanti, *GA*).

I GA, più precisamente, sono una *metaeuristica* generica, ovvero forniscono una serie di “regole” che permette di definire degli algoritmi facilmente adattabili ad un numero molto alto di tipologie di problemi e che si basano sul concetto di Evoluzione come inteso da Darwin nel XIX secolo.

L’idea alla base dei GA è quella di andare a costruire inizialmente un insieme di soluzioni ad un problema, i cosiddetti “individui” di una popolazione, che man mano verranno fatti “evolvere” usando sostanzialmente tre operatori genetici: **selezione**, **crossover** e **mutazione**, ottenendo man mano soluzioni sempre diverse migliori in termini di una metrica chiamata **fitness**, che esprime in termini quantitativi la bontà di una soluzione.

Il vantaggio del paradigma GA, essendo una metaeuristica, risiede proprio nella sua adattabilità a molti problemi, tra cui vi è proprio quello affrontato da Shall we Go in questo contesto. Di seguito verrà proposto un possibile setup, fase che rappresenta una delle difficoltà tipiche della progettazione dell’algoritmo, altrimenti molto intuitivo da applicare

1. Algoritmo genetico: un possibile setup.

Si è detto che i principali *step* da affrontare durante la progettazione di un GA sono:

* Come codificare gli individui che compongono una popolazione
* Come stabilire una *funzione di fitness*
* Come applicare i tre *operatori genetici* descritti in precedenza (selezione, crossover e mutazione di un individuo)
* Come stabilire una *condizione di terminazione,* ovvero quando far terminare il processo di “evoluzione”.
  1. Codifica degli individui

Nel contesto degli algoritmi genetici, possiamo interpretare un individuo come un insieme di geni, che lo caratterizzano e permettono la creazione di nuovi individui tramite l’applicazione dei tre operatori genetici. Calandoci nel contesto del problema che si vuole affrontare, si nota subito che un individuo, contrariamente a quanto ci si potrebbe aspettare, non può coincidere con un singolo utente “candidato” alla verifica delle segnalazioni, in quanto composto da caratteristiche che non possono essere mutate o incrociate tra di loro (banalmente, si creerebbe un utente che nella realtà non esiste, di fatto vanificando lo sforzo dell’algoritmo). È necessario quindi “salire di livello”. Si potrebbe provare a passare dalla relazione *singolo utente = individuo* a quella, più generale, *gruppo di utenti = individuo*