Documentazione Chemo

Caiazza Alfonso
0512109742

Giugliano Giuseppe
0512104762

Marcone Antonio
0512111101

https://github.com/alfoCaiazza/Chemo_FIA

Sommario

1. Introduzione

1.1 Definizione del problema

Chemo nasce dall'esigenza di digitalizzare il processo di schedulazione delle sedute chemioterapiche all'interno del reparto di Oncologia dell'ospedale San Giovanni di Dio e Ruggi D'Aragona di Salerno.
Infatti, ad oggi non esiste alcun tipo di sistema a cui il personale medico possa fare riferimento per la gestione delle sedute o dei medicinali necessari per le chemioterapie.

Dopo un'analisi preliminare del problema e dopo aver sviluppato in dettagli l'obiettivo da raggiungere, il nostro gruppo ha portato avanti lo sviluppo di un algoritmo genetico per la creazione di uno schedule "intelligente" che potesse garantire il minimo spreco di medicinali, considerando che un farmaco dopo essere stato aperto ha una scadenza di 24 ore.

Per rendere l'obiettivo raggiungibile, sono state fatte le seguenti considerazioni:

- Ogni paziente può partecipare ad una sola seduta settimanale;
- La terapia di ogni paziente prevede l'utilizzo di un solo medicinale;
- Le sedute hanno per approssimazione la stessa durata di 60 minuti.

1.2 Specifica PEAS e caratteristiche dell'ambiente

Riportiamo di seguito gli elementi della specifica PEAS per la descrizione dell'ambiente in cui opera l'algoritmo.

A. **Performance:** Le performance sono valutate secondo la quantità di pazienti vicini che consumano lo stesso farmaco e, globalmente, secondo la quantità consumata di ogni rispettivo farmaco.
B. **Environment:** L'ambiente risulta essere
   - **Stocastico:** data la presenza di componenti randomiche e probabilistiche, non è possibile determinare lo stato successivo a partire dallo stato corrente;
   - **Discreto:** Il numero delle percezioni dell'agente è limitato dalle azioni possibili;
   - **Statico:** L'agente agisce sull'insieme di pazienti da schedulare.
   - **Episodico;**
   - **Singolo:** Un unico agente opera nell'ambiente in esame**;**
C. **Actuators:** Consistone nello schedule prodotto a fine operazione;
D. **Sensors:** Consistono nella lista in input dei pazienti da schedulare e nella lista dei farmaci utilizzati dai pazienti.

1.3 Analisi del problema e scelta del dataset

Durante la formulazione e la discussione del problema, si è resa evidente la necessità di sviluppare un algoritmo di ottimizzazione. Poiché inoltre, durante il corso è cresciuto un ampio interesse nei riguardi degli Algoritmi Genetici, lo sviluppo della nostra soluzione si è diretto sin dall'inizio verso la costruzione di un algoritmo di questo tipo.

Considerato anche che il nostro progetto è stato sviluppato parallelamente alla creazione di Chemo, il dataset utilizzato per il nostro algoritmo genetico fa direttamente riferimento a quel database: stessi pazienti, stessi medicinali.

2. Funzionamento dell'algoritmo

L'algoritmo genetico sviluppato segue iterativamente le seguenti istruzioni:

A. Prende in input i pazienti selezionati per la schedulazione delle terapie;
B. Costruisce una popolazione inziale;
C. Calcola per tutti gli individui sviluppati il valore di fitness;
D. Seleziona un sottoinsieme di individui considerando il valore di fitness di ognuno di essi;
E. Gli individui selezionati vengono fatti accoppiare con una probabilità di crossover pari all'80%;
F. I nuovi individui ottenuti saranno soggetti ad una mutazione dei "geni" con una probabilità dello 0.1%.
G. Se viene soddisfatta almeno una shopping condition allora l'algoritmo termina e restituisce, considerando l'ultima generazione prodotta, l'individuo migliore; altrimenti l'algoritmo torna al passo (C).

3. Parametri

3.1. Rappresentazione ed inizializzazione degli individui

```
Patients number: 18
[0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0]
[0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0]
[0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0]
[1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1]
[0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0]
```

L'implementazione da noi proposta prevede che ogni individuo, ovvero ogni schedule, sia rappresentato tramite una matrice $n * n$, con n numero di pazienti da schedulare, dove le righe rappresentano i singoli appuntamenti e le colonne rappresentano i singoli pazienti: gli appuntamenti saranno delle liste di 0, per i pazienti non selezionati, e 1, per il solo paziente selezionato.

La matrice quadrata è una conseguenza del fatto che il numero di appuntamenti prodotti è uguale al numero di pazienti da schedulare: l'unico controllo stabilito consiste nel verificare che il numero di pazienti non superi il numero di sedute effettuabili in una sola settimana (numero di posti * numero di ore * numero di giorni).

Poiché la generazione di tale matrice è fatta in maniera totalmente casuale, il numero di conflitti, cioè di pazienti selezionati più volte, nella prima generazione è molto alto.

3.2 Calcolo della funzione di fitness

La funzione di fitness utilizzata dall'algoritmo, valuta nel seguente modo la bontà di un individuo:

- Se nello schedule sono presenti tre pazienti consecutivi che utilizzano lo stesso farmaco, allora la funzione di fitness viene incrementata di un punteggio pari a +0.3;
- Se nello schedule sono presenti solo due pazienti consecutivi che utilizzano lo stesso farmaco, allora la funzione di fitness viene incrementata di un punteggio pari a +0.2;
- Se la quantità di un farmaco utilizzato dai pazienti nello schedule viene consumata totalmente allora la funzione di fitness viene incrementata di un punteggio pari a +0.5;
- Se la quantità di un farmaco utilizzato dai pazienti nello schedule viene consumata per più della sua metà allora la funzione di fitness viene incrementata di un punteggio pari a +0.2;

3.3 Selezione

La selezione degli individui per la creazione della successiva generazione avviene mediante l'implementazione della strategia Roulette Wheel: viene assegnata una probabilità di selezione per ogni individuo in base al valore della loro fitness relativa nella popolazione. L'operatore seleziona casualmente (tenendo conto delle probabilità) un individuo, che passerà alla generazione successiva.

Questa operazione è ripetuta tante volte quanti sono gli individui presenti nella popolazione, con ogni individuo che può vincere più di una volta.

3.4 Crossover

Per come abbiamo deciso di struttura gli individui, l'implementazione dell'operatore di crossover è stata secondo il Single Point Crossover: la prima meta del patrimonio genetico del primo individuo viene inserita nel nuovo individuo, mentre l'altra metà viene prelevata dal secondo individui, evitando le ripetizioni.
Abbiamo deciso di utilizzare l'operatore con probabilità pari all'80%.

Il problema principale, in questo caso, consisteva nell'evitare di inserire nel nuovo individuo creato delle ripetizioni. La soluzione da noi adottata è stata quella di verificare di volta in volta la presenza o meno di conflitti e, al termine del crossover, inserire casualmente gli individui mancanti, cioè non presenti in alcun appuntamento.

3.5 Mutazione

La scelta della particolare rappresentazione degli individui come matrice ci ha portato sin da subito a pensare, per la scelta dell'operatore di mutazione, alla costruzione di un operatore di tipo Swap: vengono infatti scelte a caso due righe della matrice e scambiate tra di loro.

La probabilità con cui un individuo vada incontro a mutazione è pari allo 0.1%.

3.6 Stopping Condition

L'algoritmo termina se viene soddisfatta almeno una delle seguenti stopping condition:

- Il valore massimo delle fitness calcolate per tutti gli individui della nuova generazione è più piccolo rispetto al valore massimo delle fitness della generazione precedente. In merito a questa situazione, abbiamo inoltre considerato la presenza di una "chance" di proseguimento, notando che tale condizione si verificava solo per le prime generazioni prodotte.
- Il numero di individui del mating pool è minore di 2, perché in questa condizione non sarebbe possibile effettuare il crossover.

4. Conclusioni

Per lo sviluppo di questo progetto abbiamo scelto di non utilizzare nessuna particolare libreria per la costruzione dell'algoritmo genetico, preferendo infatti sviluppare da zero tutte le funzionalità e gli operatori necessari, al fine di comprendere e apprezzare di più l'argomento trattato durante il corso.

Nonostante i risultati dell'algoritmo possono risultare non brillanti, l'algoritmo sviluppato permette comunque di raggiungere l'obiettivo prefissato, rendendoci quindi molto soddisfatti.