Easy Pass

Documentazione

Easy Pass

Versione	1.0
Data	14/02/2022
Destinatario	Prof. F. Palomba
Presentato da	Montefusco Alberto Rinaldi Viviana Spina Gennaro

Sommario

Team members

Nome	Ruolo	Acronimo	Informazioni di contatto
Montefusco Alberto	Team member	MA	a.montefusco28@studenti.unisa.it
Viviana Rinaldi	Team member	VR	v.rinaldi26@studenti.unisa.it
Spina Gennaro	Team member	SG	g.spina5@studenti.unisa.it

Repository GitHub

Link
https://github.com/Alberto-00/EasyPass-AI

CAPITOLO 1

Introduzione

1.1 Sistema corrente

Per applicare le regole anti-Covid in ambito universitario, imposte dal Governo a seguito dell'emergenza sanitaria, è previsto un meccanismo di controllo del Green Pass degli studenti presenti in aula, effettuato da parte dei docenti che sostengono la lezione.
Al momento, il docente è tenuto a spostarsi fra i banchi per raggiungere lo studente che vuole controllare, comportando alcuni disagi, primo tra tutti il mancato rispetto delle distanze di sicurezza quando il docente deve effettuare la scansione.
Inoltre, nonostante nelle aule siano presenti posti contrassegnati che lo studente può occupare in modo tale da rispettare il distanziamento, queste misure non sono sempre rispettate.

1.2 Sistema proposto

Per eliminare i disagi esposti nel precedente paragrafo, Easy Pass viene sviluppato come una Web Application, accessibile da Internet, mirata all'informatizzazione della procedura di controllo.
Il Sistema è basato sull'utilizzo di sessioni di validazione identificate da un codice QR che, una volta condiviso dal docente (tramite il proiettore) e scansionato dagli studenti, permetterà a questi ultimi di inserire il proprio Green Pass per sottoporlo alla verifica.
Easy Pass prevede anche un modulo di Intelligenza Artificiale che permette di ricercare i posti che gli studenti potranno occupare nell'aula indicata dal docente, non solo rispettando il distanziamento tra questi, ma anche posizionando lo studente in un posto ritenuto ottimale per seguire la lezione.

Nota: attenendoci alle norme universitarie, il metro di distanza sarà rispettato se uno studente non ha colleghi seduti in maniera a lui adiacente. Questo significa anche che la capienza massima dell'aula è del 50%.

CAPITOLO 2

Definizione del problema

2.1 Obiettivi

Lo scopo del progetto è quello di realizzare un agente intelligente che sia in grado di trovare una soluzione ottimale tale da:

- disporre gli studenti in aula in modo da rispettare il metro di distanza tra due individui, imposto dalle norme sulla sicurezza del Ministero della Salute;
- scegliere un posto ritenuto migliore per la visualizzazione della lavagna o dello schermo proiettato.

Una volta realizzato l'agente intelligente, quest'ultimo dovrà essere integrato all'interno di Easy Pass.

2.2 Specifica PEAS

PEAS	
Performance	La misura di **performance** dell'agente si basa sull'accuratezza di assegnare un posto ad uno studente cercando di soddisfare gli obiettivi dati (*già descritti nel paragrafo 2.1*).
Enviroment	L'**ambiente** dell'agente è formato dall'aula, i singoli posti e gli studenti.
Actuators	Gli **attuatori** consistono nel visualizzare la *seating map* tramite una mappa grafica per mostrare a video la disposizione consigliata agli studenti.
Sensors	I **sensori** tramite il quale l'agente reperisce gli stimoli dall'ambiente sono rappresentati da un form per richiedere la visualizzazione della *seating map,* dopo che questa è stata configurata tramite l'inserimento dell'aula desiderata e del numero di studenti.

2.2.1 Caratteristiche dell'ambiente

L'ambiente è:

- **Completamente osservabile**: l'agente ha sempre accesso a tutte le informazioni relative alla disposizione dei posti;
- **Deterministico**: lo stato successivo dell'ambiente è completamente determinato dallo stato corrente e dall'azione eseguita dall'agente;
- **Sequenziale**: la posizione che lo studente può occupare è influenzata dalla sequenza di posti occupati precedentemente;
- **Statico**: durante la ricerca dei posti, il Docente attende l'esecuzione dell'algoritmo;
- **Discreto**: l'insieme delle possibili percezioni e azioni dell'agente sono distinte e definite;
- **Agente singolo**: l'ambiente consente la presenza di un unico agente.

2.3 Analisi del problema

Formulazione a stati completi del problema:

- **Stati**. Ogni disposizione nell'aula degli N studenti è uno stato.
- **Stato iniziale**. Aula con studenti già disposti.
- **Azioni**. Spostare gli studenti in modo che non ci siano conflitti e che il punteggio di fitness sia il più alto possibile.
- **Modello di transizione**. Restituisce l'aula con una nuova disposizione di studenti.
- **Test obiettivo**. Nell'aula sono disposti tutti gli studenti col minor numero di conflitti (laddove possibile, 0) e il valore di massimizzazione è il più alto.

Scelta dell'algoritmo. Formulato il problema, era chiara la necessità di utilizzare un algoritmo di ottimizzazione. In particolare, abbiamo implementato un algoritmo genetico multi-obiettivo in quanto gli scopi da raggiungere sono 2. È stato utilizzato NSGAII poiché tiene conto del trade-off tra i vari obiettivi contrastanti e, inoltre, è in grado di restituire un insieme di soluzioni non dominate in cui sono

favoriti gli individui più vicini al vero Fronte di Pareto. Il nostro primo pensiero era stato di utilizzare NSGA nella sua versione primitiva; tuttavia, abbiamo scelto di utilizzare questa variante in quanto integra meccanismi quali l'elitismo e la crowding distance nella selezione degli individui restituendo soluzioni migliori.

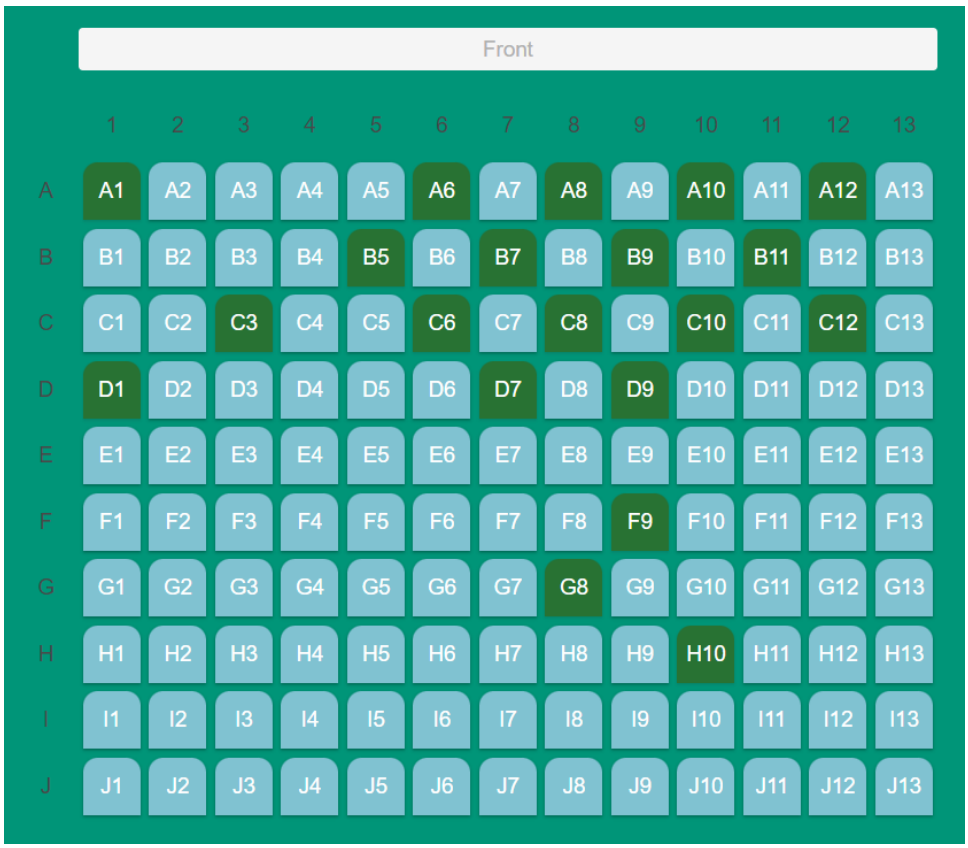
Soluzione.

La soluzione ottima è un individuo che ha 2 qualità:

- Tramite l'implementazione di una funzione di minimizzazione, si restituisce l'individuo col minor numero di conflitti;
- Tramite l'implementazione di una funzione di massimizzazione, avremo l'individuo con più alto punteggio di fitness, definito dalla posizione occupata in aula da ciascun studente.

Esempio:

- Aula 13 x 10;
- 20 studenti da disporre.



CAPITOLO 3

Algoritmo Genetico

3.1 Parametri

Durante la progettazione dell'algoritmo genetico si è cercato un modo per stabilire la bontà di una configurazione dei parametri. Di seguito, vengono elencati i parametri e gli operatori utilizzati per il setup dell'algoritmo genetico multi-obiettivo.

3.1.1 Encoding degli individui

Un individuo è una disposizione di N studenti in un'aula di $m \; x \; n$ posti (dove m rappresenta il numero di righe e n il numero di colonne).
La codifica utilizzata è una codifica reale in quanto più compatta e più vicina al problema in questione (poiché prende degli input numerici).

3.1.2 Selezione

Per la selezione degli individui migliori si è scelto di utilizzare la strategia "Binary Tournament" che, nel contesto del NSGAII, confronta due individui sulla base di due fattori: la crowding distance e il rango. Ciò accade per favorire da una parte soluzioni non troppo simili (che porterebbero altrimenti ad una convergenza prematura e a una popolazione finale poco migliorata) e dall'altra per premiare le soluzioni che hanno un maggiore punteggio di fitness. Quest'ultimo concetto è rafforzato dall'utilizzo dell'elitismo.

3.1.3 Crossover

La strategia adottata è stata "K-Point Crossover" che consiste nel dividere due individui di una stessa generazione in due parti (questo è valido nel nostro caso in quanto $K = 1$) in un punto randomico e scambiarle tra di loro per aumentare la diversità della popolazione. La probabilità che avvenga il crossover è 0.8%.

3.1.4 Mutazione

La "Polynomial Mutation" riguarda i singoli geni di ciascun individuo, il cui valore muterà in un valore randomico, generato mediante opportuni calcoli, secondo una probabilità di mutazione dello 0.01%.

3.1.5 Vincoli

Oltre ad indicare il limite superiore e inferiore di ogni variabile, è stato imposto un altro vincolo per prevenire che l'algoritmo posizioni uno studente in un posto già assegnato ad altri (in altri termini, non possono essere generate all'interno di uno stesso individuo due geni uguali).

3.1.6 Calcolo dei conflitti

Nel contesto del nostro problema, i conflitti vengono rilevati ogni qualvolta due studenti sono seduti a meno di un metro di distanza. Ossia, ciò accade se i due studenti sono seduti direttamente uno dietro l'altro o di fianco. Non rileviamo conflitti se sono seduti sulla stessa diagonale.

```
private int calculateConflicts(List<Double> encoding) {
    int conflicts = 0;

    for (int i = 0; i < encoding.size(); i += 2){
        int x = (int) Math.floor(encoding.get(i));
        int y = (int) Math.floor(encoding.get(i + 1));

        for (int j = i + 2; j < encoding.size(); j += 2){
            int xa = (int) Math.floor(encoding.get(j));
            int ya = (int) Math.floor(encoding.get(j + 1));

            if ((xa - 1) >= 0) {
                if (xa - 1 == x && y == ya)
                    conflicts++;
            } if ((xa + 1) < this.ROW){
                if (xa + 1 == x && y == ya)
                    conflicts++;
            } if ((ya + 1) < this.COL){
                if (xa == x && ya + 1 == y)
                    conflicts++;
            } if ((ya - 1) >= 0){
                if (xa == x && ya - 1 == y)
                    conflicts++;
            }

        }
    }
    return conflicts;
}
```

3.1.7 Funzione di fitness

La funzione di fitness è implementata tramite una funzione di massimizzazione in cui a ogni studente viene associato un punteggio a seconda dell'ottimalità del settore in cui si trova. Sommando i punteggi così ottenuti, si ha il valore della funzione di fitness di quella disposizione (individuo).
Nel nostro problema, abbiamo deciso di suddividere l'aula in 9 settori ognuno dei quali ha un punteggio in un range da 1 a 4, definito dalla sua posizione rispetto alla lavagna. In particolare, ogni studente è rappresentato da un gene, il quale indica le coordinate del posto da lui occupato.

Divisione in settori.
Ogni aula analizzata è divisa in 9 settori. Un settore è una matrice contenente due coppie di coordinate (che indicano rispettivamente lo spigolo in alto a sinistra, in cui inizia il settore, e lo spigolo in basso a destra, in cui termina il settore) e il punteggio del settore.

Per calcolare le dimensioni di ogni settore verifichiamo:

- se il modulo prodotto dal rapporto tra il numero di righe della matrice per 3 è diverso da 0, allora i primi due settori avranno **numero di righe = (righe della matrice / 3) + 1** mentre il terzo settore avrà il restante numero di righe;
- altrimenti, il **numero di righe del settore = (righe della matrice / 3)**.

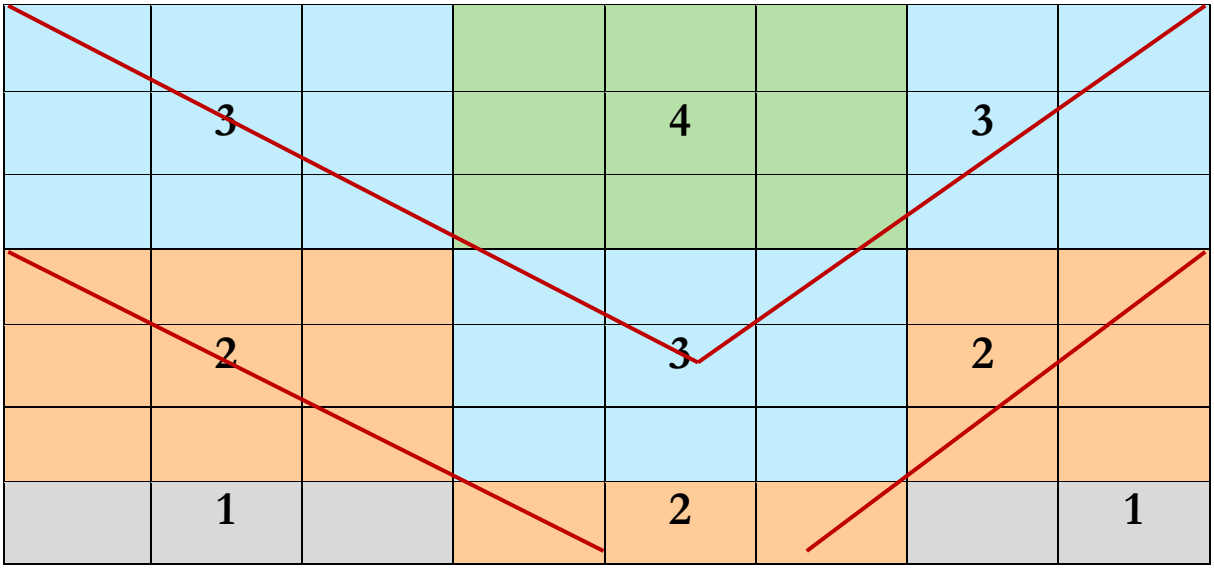
In modo analogo calcoliamo il numero di colonne per ogni settore.

Assegnazione punteggio al settore.
La distribuzione dei punteggi avviene secondo uno schema a "V" utilizzato anche in altri problemi di assegnazione dei posti (quali teatri, cinema, etc.):

Settore 1:	punteggio **3**;	**Settore 4**:	punteggio **2**;	**Settore 7**:	punteggio **1**;
Settore 2:	punteggio **4**;	**Settore 5**:	punteggio **3**;	**Settore 8**:	punteggio **2**;
Settore 3:	punteggio **3**;	**Settore 6**:	punteggio **2**;	**Settore 9**:	punteggio **1**;

Esempio divisione in settori.

Consideriamo un'aula formata da 7 righe e 8 colonne. La divisione in settori, con il relativo punteggio, sarà strutturata in questo modo:



3.1.8 Criteri di arresto

L'algoritmo si arresta nel momento in cui ha eseguito in totale la funzione di valutazione 100.000 volte.

3.2 Passi dell'algoritmo

Inizializzazione: innanzitutto vengono impostati gli obiettivi, i vincoli del problema, la taglia della popolazione, il numero di valutazioni e le probabilità di crossover e mutazione; inoltre, sono indicati anche i bound di ogni variabile. Viene quindi generata una disposizione randomica di studenti in un'aula con m righe e n colonne;

Valutazione popolazione: generata la popolazione, viene effettuato il calcolo dei conflitti e della funzione di fitness. Successivamente, è controllato il numero di valutazioni corrente e, se questo risulta essere inferiore al numero di valutazioni indicato nel problema, si accederà al ciclo per migliorare ancora la popolazione;

Selezione: tramite la "Binary Tournament Selection" otteniamo un mating pool di 100 individui, in cui una parte si è direttamente *salvata* grazie all'elitismo mentre un'altra ha dovuto superare la selezione basata su rango e crowding distance;

Crossover: dal mating pool, un numero casuale di individui viene combinato a coppie per aumentare la diversità genetica tramite "Single-Point Crossover";

Mutazione: quindi, per mezzo della "Polynomial Mutation" il valore di alcuni geni potrebbe essere cambiato per evitare la convergenza prematura;

Aggiornamento della popolazione: a questo punto, si è arrivati a una nuova generazione che viene nuovamente valutata e, se i criteri di arresto sono soddisfatti, viene restituita come risultato dell'algoritmo;

Risultato: nel nostro caso, ai fini del nostro problema, estrarremo dalla popolazione finale l'individuo migliore, valutando come tale quello che ha in assoluto il minor numero di conflitti e, come secondo criterio, quello che ha il punteggio di fitness più alto. Qualora vi siano due soluzioni uguali, sarà restituita la prima riscontrata.

CAPITOLO 4

Testing dei parametri scelti

4.1 Testing

I parametri a cui si fa riferimento, oltre quelli già indicati sono:

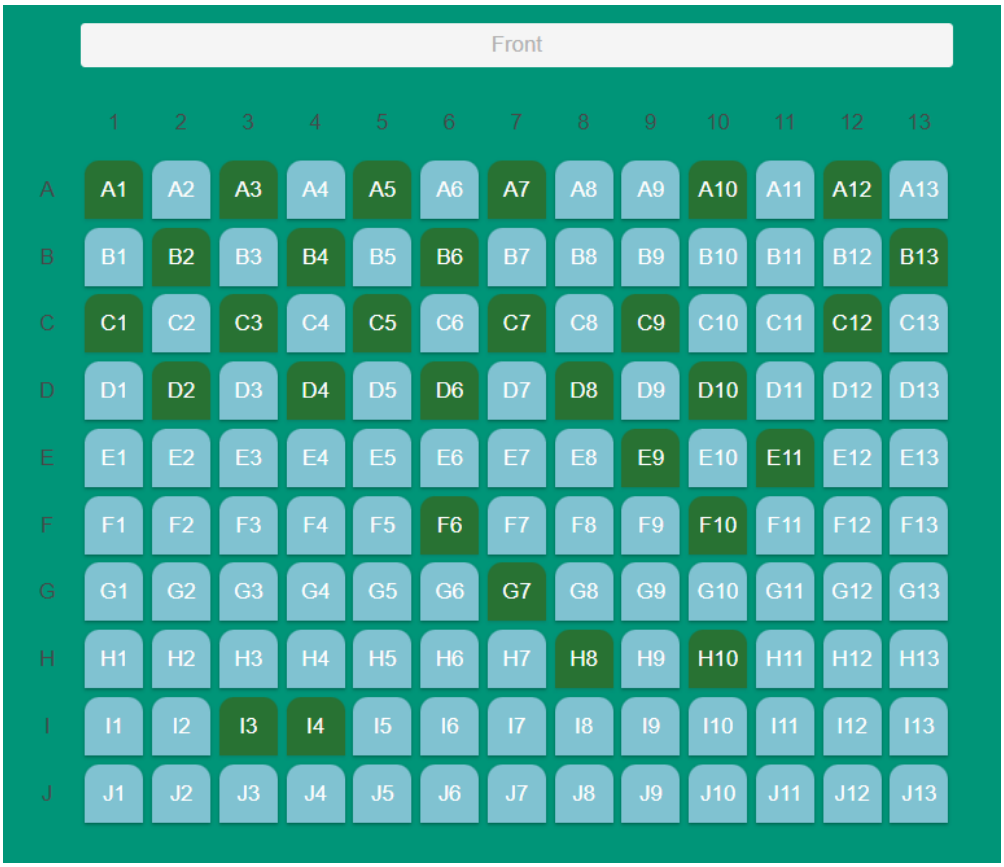
- dimensioni dell'aula: 10 x 13 (capienza dell'aula al 50% quindi 65 posti max occupabili);
- numero di valutazioni: 100.000.

1) Caso

Numero studenti: 30 Dimensioni popolazione: 20

```
'Vision Range Problem' INFO Best Solution:

Population Size: 20
Students Size: 30
Solution: 2 8 1 5 7 9 0 11 5 9 4 8 3 1 0 6 0 2 2 2 3 7 3 3 3 5 0 4 5 5 2 6 0 9 6 6 1 12 2 11 3 9 8 3 7 7 2 0 4 10 2 4 1 1 1 3 8 2 0 0
Objectives: 1.0 conflicts   -93.0 score
Constraints: 0.0
Total execution time: 3221 ms
```

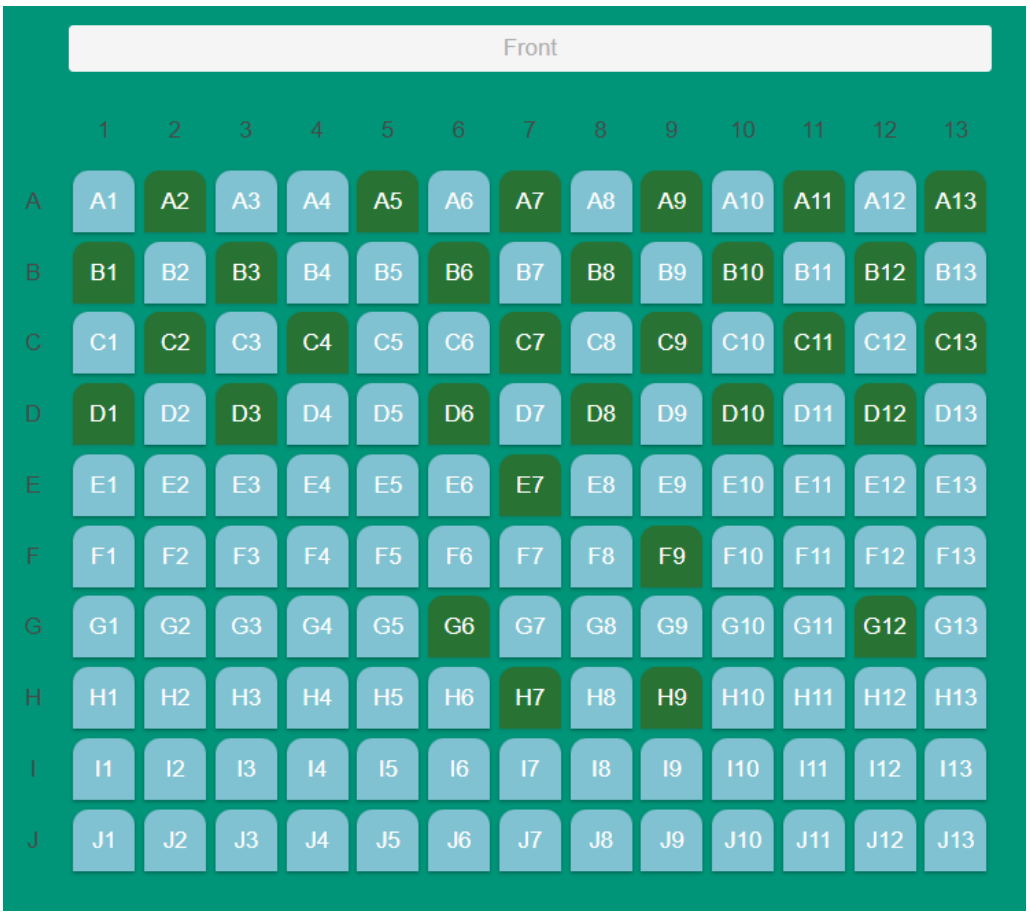


2) Caso

Numero studenti: 30 Dimensioni popolazione: 50

```
'Vision Range Problem' INFO Best Solution:

Population Size: 50
Students Size: 30
Solution: 0 12 3 5 2 1 5 8 7 6 0 6 1 7 0 8 2 6 1 5 0 1 7 8 2 10 6 11 1 11 2 3 3 0 1 9 3 2 1 2 3 7 2 12 2 8 0 4 3 9 1 0 6 5 0 10 4 6 3 11
Objectives: 0.0 conflicts   -99.0 score
Constraints: 0.0
Total execution time: 8901 ms
```

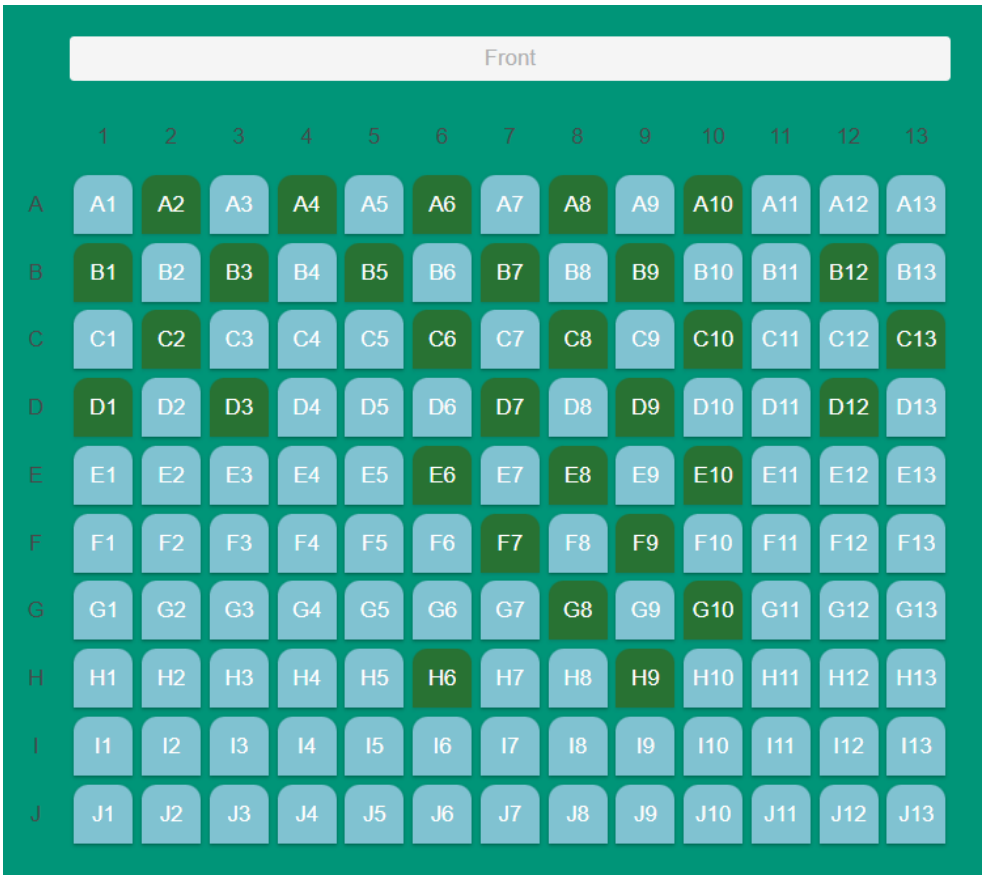


3) Caso

Numero studenti: 30 Dimensioni popolazione: 100

```
'Vision Range Problem' INFO Best Solution:

Population Size: 100
Students Size: 30
Solution: 4 9 5 6 3 8 1 6 3 0 5 8 1 8 4 7 2 9 2 5 2 1 3 6 6 9 0 1 3 11 2 12 4 5 1 0 3 2 6 7 1 2 1 4 0 3 0 9 2 7 7 5 0 7 7 8 0 5 1 11
Objectives: 0.0 conflicts   -100.0 score
Constraints: 0.0
Total execution time: 11777 ms
```



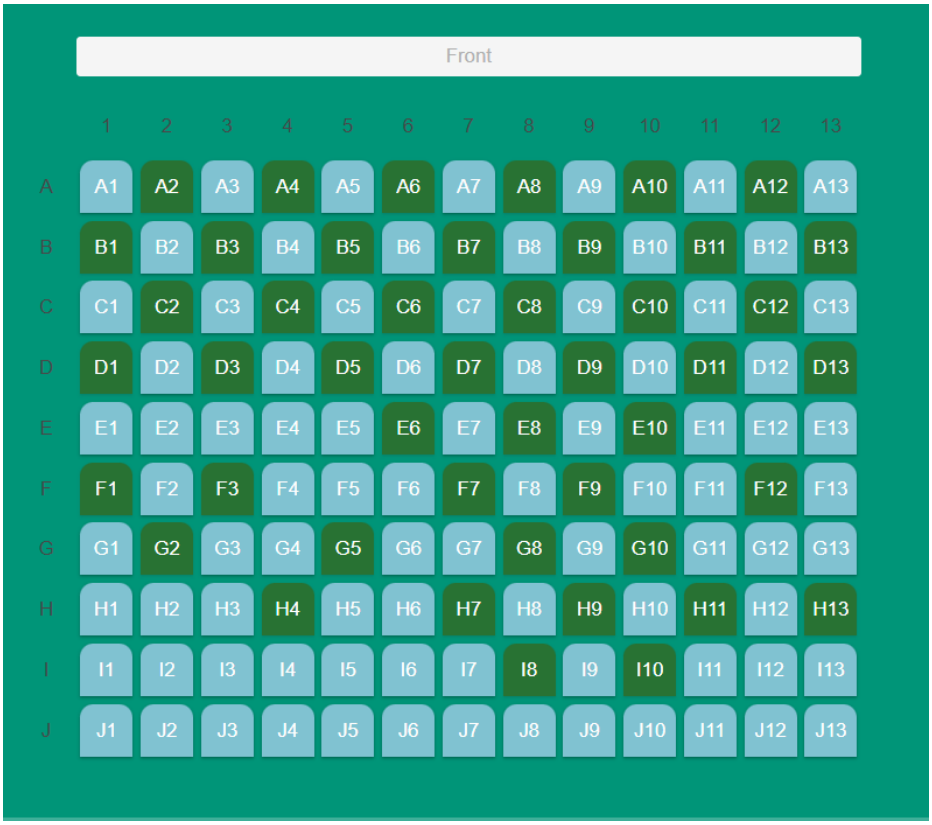
4) Caso

Numero studenti: 45 Dimensioni popolazione: 100

```
'Vision Range Problem' INFO Best Solution:

Population Size: 100
Students Size: 45
Solution: 1 0 0 3 5 11 5 8 6 1 4 7 0 11 5 6 2 7 1 4 1 8 5 0 4 5 3 8 0 9 2 11 6 4 3 10 7 3 2 1 8 7 4 9 7 6 0 7 7 10 7 12 2 5 6 9 8 9 1 6 2 9 1 12 1 2 7 8 0 1 3 0 2 3 6 7
Objectives: 0.0 conflicts   -135.0 score
Constraints: 0.0
Total execution time: 13438 ms
```

```
3 4 3 2 0 5 5 2 3 6 3 12 1 10
```



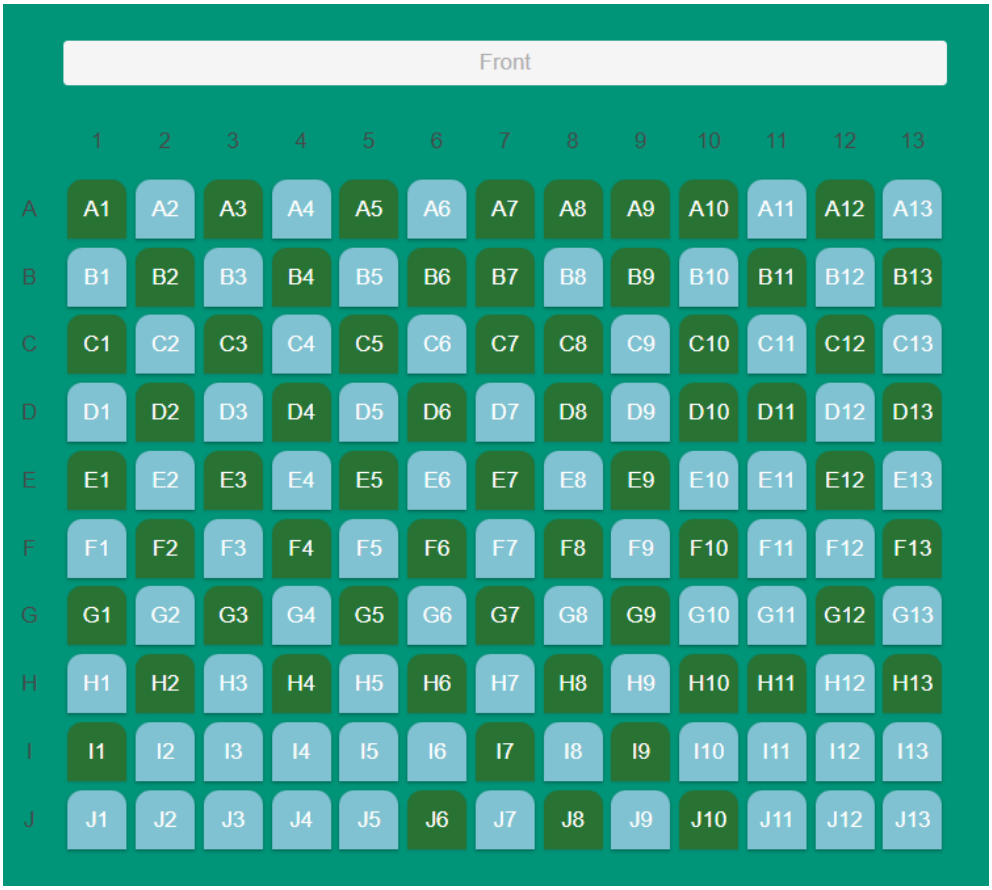
5) Caso

Numero studenti: 60 Dimensioni popolazione: 100

```
'Vision Range Problem' INFO Best Solution:

Population Size: 100
Students Size: 60
Solution: 2 7 6 0 5 1 3 1 2 6 0 2 7 12 0 0 5 9 4 11 1 8 3 7 4 4 8 8 6 2 6 6 7 3 0 9 9 7 1 10 2 11 2 2 9 9 3 10 0 4 4 6 6 8 7 1 1 1 5 12 1 5 7 10 4 2 0 8 8 0 6 4 5 5 3 3
Objectives: 12.0 conflicts  -171.0 score
Constraints: 0.0
Total execution time: 12770 ms
```

```
4 0 3 12 2 0 1 12 8 6 0 6 6 11 2 9 5 3 7 7 5 7 1 6 0 11 7 9 9 5 7 5 3 5 2 4 0 7 3 9 1 3 4 8
```



CAPITOLO 5

Conclusione

5.1 Considerazioni finali

Come si può notare dai test effettuati, risulta evidente che una popolazione più numerosa porti a soluzioni con un numero di conflitti inferiori seppur il tempo di elaborazione aumenti. D'altra parte, il numero di conflitti aumenta inevitabilmente con l'avvicinarsi del numero degli studenti alla capienza massima dell'aula.

Inoltre, non sono stati effettuati test considerando un numero più elevato di valutazioni a causa dell'aumento del tempo di esecuzione dell'algoritmo genetico.

In conclusione, siamo soddisfatti del risultato in quanto è evidente come l'algoritmo raggruppi gli studenti nei settori con migliore visibilità, mantenendo comunque, per quanto possibile, le misure di distanziamento espresse.