



**Politecnico  
di Torino**

# Politecnico di Torino

Dipartimento di Ingegneria Gestionale e della Produzione

Corso di Laurea Triennale in Ingegneria Gestionale

Classe L8 – Ingegneria dell'informazione

A.A. 2023/2024

## **IoT & sorted waste: la transizione 4.0 di ASP e Comune di Asti**

**Relatore**

*Prof. Fulvio Corno*

**Candidato**

*Noemi Marchiaro  
Matricola 272162*

# Indice

<b>1</b>	<b>Proposta di progetto .....</b>	<b>3</b>
1.1	Titolo della proposta .....	3
1.2	Descrizione del problema proposto.....	3
1.3	Descrizione della rilevanza gestionale del problema.....	3
1.4	Descrizione dei data-set per la valutazione.....	4
1.5	Descrizione preliminare degli algoritmi coinvolti .....	5
1.6	Descrizione preliminare delle funzionalità previste per l'applicazione software .....	6
<b>2</b>	<b>Descrizione dettagliata del problema .....</b>	<b>7</b>
2.1	Raccolta differenziata .....	7
2.2	Procedimento di raccolta.....	8
2.3	Obiettivi .....	8
2.4	Criticità .....	9
2.4.1	Durata del viaggio .....	9
2.4.2	Contenuto presente nei cassonetti .....	10
<b>3</b>	<b>Descrizione del dataset .....</b>	<b>12</b>
3.1	Descrizione delle tabelle .....	13
3.1.1	Tabella cassonetti .....	13
3.1.2	Tabella luoghi.....	13
<b>4</b>	<b>Strutture dati e algoritmi utilizzati .....</b>	<b>14</b>
4.1	Struttura del progetto .....	14
4.2	Algoritmi .....	15
4.2.1	Ricerca del nearest neighbor .....	18
4.2.2	Parametri della ricorsione.....	19
<b>5</b>	<b>Diagramma delle classi principali .....</b>	<b>21</b>
<b>6</b>	<b>Interfaccia dell'applicazione .....</b>	<b>23</b>
6.1	Avvio del programma .....	23
6.2	Visualizzazione dello stato cassonetti.....	24
6.3	Calcolo del percorso di raccolta.....	26
6.4	Riepilogo del percorso di raccolta .....	29
<b>7</b>	<b>Esempio d'uso e risultati .....</b>	<b>30</b>
7.1	Visualizzazione dello stato.....	30
7.2	Calcolo del percorso.....	31
<b>8</b>	<b>Conclusioni e possibili miglioramenti .....</b>	<b>34</b>
	<b>Sitografia.....</b>	<b>35</b>

# Elenco delle figure

Figura 1: metodo riempিCassonetti() - Model .....	11
Figura 2: overview dei package.....	14
Figura 3: metodo calcolaPercorso() - Model.....	16
Figura 4: ciclo for che lancia la ricorsione – metodo init() - TrovaPercorso.....	17
Figura 5: ricerca del nearest neighbor – metodo cercaNearestNeighbor() - TrovaPercorso	18
Figura 6: durata del viaggio della soluzione parziale - metodo cercaNearestNeighbor() - TrovaPercorso.....	20
Figura 7: diagramma UML della classe Model .....	21
Figura 8: diagramma UML della classe TrovaPercorso .....	22
Figura 9: schermata di avvio .....	23
Figura 10: schermata per la visualizzazione dello stato dei cassonetti.....	24
Figura 11: schermata dello stato dei cassonetti del ‘vetro’ nella zona ‘Asti’ (tutta la città)	25
Figura 12: schermata dello stato dei cassonetti del ‘vetro’ nella zona ‘3’ .....	26
Figura 13: schermata per cercare del percorso di raccolta .....	27
Figura 14: schermata con il percorso di raccolta consigliato per la ‘plastica’ nella zona ‘1’ .....	28
Figura 15: schermata di riepilogo del percorso di raccolta accettato per la ‘plastica’ nella zona ‘1’ .....	29
Figura 16: schermata dello stato dei cassonetti della ‘carta nella zona ‘Asti’ (tutta la città)	31
Figura 17: schermata dello stato dei cassonetti del ‘vetro’ nella zona ‘4’ .....	31
Figura 18: schermata con il primo percorso di raccolta consigliato per la ‘plastica’ nella zona ‘1’ .....	31
Figura 19: schermata di riepilogo del primo percorso di raccolta accettato per la ‘plastica’ nella zona ‘1’ .....	31
Figura 20: schermata con il secondo percorso di raccolta consigliato per la ‘plastica’ nella zona ‘1’ .....	32
Figura 21: schermata di riepilogo del secondo percorso di raccolta accettato per la ‘plastica’ nella zona ‘1’ .....	32

# **1 Proposta di progetto**

## **1.1 Titolo della proposta**

IoT & sorted waste: la transizione 4.0 di ASP e Comune di Asti

## **1.2 Descrizione del problema proposto**

Asti Servizi Pubblici S.p.A e Comune di Asti stanno lavorando a un progetto innovativo di raccolta rifiuti.

“Il nuovo metodo, che sostituirà il servizio di raccolta porta a porta, verrà attuato con il posizionamento di nuovi cassonetti per la raccolta differenziata, raggruppati in ecoisole. Tali contenitori saranno dotati anche di un sensore in grado di rilevare il livello di riempimento, al fine di ottimizzare le procedure di raccolta, garantendo altresì sempre una volumetria disponibile all’Utenza.”

L’applicativo proposto dalla sottoscritta simulerà i volumi presenti nei contenitori, permettendone poi la visualizzazione mediante tabelle. Inoltre, sarà possibile trovare il percorso ottimizzato di raccolta (in base alla capacità dei mezzi disponibili e alla disponibilità degli operatori) abbassando così il consumo di carburante e le emissioni di CO2.

## **1.3 Descrizione della rilevanza gestionale del problema**

Si tratta di un progetto “reale”, per cui è stata anche appaltata la realizzazione di un sistema informatico in grado di monitorare il parco contenitori.

Le diverse funzionalità sopra elencate potranno essere sfruttate dal personale dell’azienda per controllare e pianificare la raccolta dei rifiuti.

L’ottimizzazione del percorso abbasserà il consumo di carburante (con un risparmio economico) e le emissioni di CO2 (riducendo l’impatto ambientale).

## **1.4 Descrizione dei data-set per la valutazione**

Il data-set utilizzato è stato fornito dall'azienda ASP di Asti, in forma confidenziale.

Si tratta di una raccolta di dati relativa al posizionamento dei cassonetti nel Comune di Asti realizzata dall'ufficio progettazione dell'igiene urbana.

Il data-set contiene i seguenti attributi relativi ai cassonetti:

- Id: codice identificativo aggiunto
- Latitudine
- Longitudine
- Indirizzo
- Tipo: carta, plastica, indifferenziata, vetro
- Dimensione: in Litri

Inoltre, sono stati forniti dati necessari al calcolo del percorso di raccolta:

- Dimensione del mezzo adibito alla raccolta, in base al tipo di rifiuto
- Durata indicativa di un turno di raccolta
- Percentuale di riduzione dei rifiuti nel mezzo di raccolta
- Durata svuotamento cassonetto
- Durata svuotamento camion in discarica
- Velocità media camion
- Località discarica, in base al tipo di rifiuto

## **1.5 Descrizione preliminare degli algoritmi coinvolti**

Il data-set fornito dall’azienda è un file Excel, è dunque necessaria la conversione a CSV e successivamente in linguaggio SQL.

L’applicazione viene realizzata in linguaggio Java e segue i pattern DAO (Data Access Object) e MVC (Model View Controller), ottenendo una separazione tra accesso ai dati, interfaccia utente e logica applicativa.

La prima funzionalità del tool è la simulazione dei volumi presenti nei contenitori (in base al tipo di rifiuto scelto) e la seguente visualizzazione mediante tabella.

Dopo una query al database su tutti i cassonetti del tipo selezionato i contenitori verranno “riempiti” in modo casuale e poi ordinati in base alla percentuale di riempimento.

In futuro i cassonetti saranno dotati di sensori in grado di rilevare tale percentuale e si avrà a disposizione un database aggiornato in tempo reale.

La seconda funzionalità è quella più complessa: trovare il percorso di raccolta, in base al tipo di rifiuto selezionato, che minimizzi il tempo e lo spazio percorso e massimizzi la quantità raccolta, rispettando i vincoli di tempo disponibile e dimensione del camion.

Per fare ciò viene creato un grafo (completo, pesato e non orientato) in cui i vertici sono i cassonetti mentre gli archi rappresentano la distanza (in linea d’aria) tra ogni possibile coppia di contenitori. Verrà poi utilizzato un algoritmo ricorsivo in grado di trovare il percorso ottimo in base alle specifiche introdotte dall’utente. Al termine della ricorsione verrà mostrato il percorso e alcune statistiche.

## **1.6 Descrizione preliminare delle funzionalità previste per l'applicazione software**

L'applicazione potrà essere utilizzata dal personale addetto alla pianificazione giornaliera del servizio vuotatura dei contenitori.

In una prima schermata l'addetto potrà visualizzare, selezionando la tipologia di rifiuto, il livello di riempimento dei cassonetti in tempo reale e la quantità totale di rifiuto prodotto. Grazie alla sua esperienza potrà decidere quale raccolta ha la priorità.

Nella schermata successiva, dopo aver scelto la tipologia di rifiuto, otterrà il percorso ottimizzato di raccolta.

Inoltre, potrà visualizzare alcune statistiche come il numero di cassonetti svuotati e non, la durata del viaggio, la quantità raccolta e quella rimanente.

Tali informazioni saranno presenti sia numericamente che graficamente e aiuteranno l'operatore a decidere se “accettare” o meno il percorso consigliato.

In caso affermativo otterrà ulteriori dettagli sul viaggio programmato da fornire all'autista.

Potrà poi decidere se pianificare un ulteriore giro di raccolta per i cassonetti esclusi dal percorso precedente o programmare la raccolta per altre tipologie di rifiuto.

In caso negativo può effettuare una nuova pianificazione, eventualmente modificando i parametri (durata viaggio e capacità mezzo).

## 2 Descrizione dettagliata del problema

### 2.1 Raccolta differenziata

La raccolta differenziata è di fondamentale importanza nella salvaguardia dell'ambiente. Grazie ad essa si aiuta il pianeta a risparmiare le sue risorse e si riduce l'inquinamento. Con questo sistema di raccolta si raggruppano i rifiuti urbani in base alla loro tipologia (carta e cartone, plastica e alluminio, vetro, frazione organica e rifiuto indifferenziato) destinandoli al riciclaggio, con conseguente riutilizzo delle materie prime. Quasi tutti i materiali, infatti, possono avere nuova vita: il cartone, la plastica e il vetro possono essere riutilizzati per produrre nuovi oggetti o diventare materiale utile per altre produzioni.

Esistono due sistemi di raccolta:

- **Porta a porta:** i rifiuti vengono ritirati periodicamente presso l'abitazione. La raccolta avviene in giorni e con frequenze diverse a seconda della tipologia.
- **Verticale:** i rifiuti vengono depositati in casonetti nei pressi delle abitazioni.

La Città di Asti effettua la raccolta porta a porta, ad eccezione del vetro nel concentrico e nelle frazioni per cui sono presenti contenitori stradali.

Nel novembre del 2021, durante una conferenza stampa, Asp SpA e il Comune di Asti hanno annunciato il passaggio alla raccolta verticale. “Il nuovo metodo, che sostituirà gradualmente il servizio di raccolta porta a porta, verrà attuato con il posizionamento di nuovi casonetti per la raccolta differenziata, raggruppati in ecoisole, collocati su strada, adiacenti i condomini.” I contenitori sono dotati di un sensore in grado di rilevare il livello di riempimento, “permettendo un monitoraggio a distanza dei casonetti e dei materiali inseriti, in modo da efficientare i passaggi degli operatori per la raccolta che avrà così luogo solo quando, ma tutte le volte in cui, i contenitori saranno quasi pieni.”<sup>1</sup>

---

<sup>1</sup> [www.asp.asti.it/asp-e-il-comune-di-asti-presentano-la-tensione-del-servizio-di-raccolta-verticale/](http://www.asp.asti.it/asp-e-il-comune-di-asti-presentano-la-tensione-del-servizio-di-raccolta-verticale/)

## **2.2 Procedimento di raccolta**

Gli addetti alla raccolta partono con il camion dalla sede dell’azienda. “Gli agganci posizionati sulla cima dei contenitori permettono agli operatori di sollevarli grazie a un braccio meccanico di cui sono dotati i mezzi aziendali. Ciò riduce i rischi connessi alla sicurezza sul lavoro degli operatori e causati da attività quali il sollevamento di carichi pesanti e alla movimentazione dei contenitori su terreni sconnessi”.<sup>2</sup>

Si dirigono verso il primo cassonetto, lo svuotano e il contenuto presente nel camion viene compresso. Procedono verso il cassonetto successivo e ripetono tali azioni.

Terminata la raccolta si dirigono verso l’impianto di trattamento o di stoccaggio relativo alla frazione merceologica di rifiuto (nel programma per brevità viene usato il termine discarica). Dopo aver effettuato le operazioni di registrazione e una prima pesatura del mezzo scaricano il contenuto in appositi spazi, viene poi effettuata una seconda pesa per determinare il netto dei rifiuti conferiti, viene firmata la documentazione e fanno ritorno alla sede, terminando così il turno di lavoro.

## **2.3 Obiettivi**

L’obiettivo di questa applicazione è la gestione del servizio di raccolta dei rifiuti nella Città di Asti.

Sarà possibile visualizzare il livello di riempimento dei contenitori in base al tipo di rifiuto e alla zona della città, grazie a una lista di cassonetti ordinati per riempimento decrescente. L’operatore potrà dunque valutare la situazione e capire quale raccolta abbia la priorità.

Inoltre, si potrà trovare il percorso di raccolta che minimizzi lo spazio percorso e, conseguentemente, il tempo e massimizzi la quantità raccolta. Seguendo tale percorso i cassonetti maggiormente pieni verranno svuotati, garantendo alla comunità un buon

---

<sup>2</sup> [www.asp.asti.it/asp-e-il-comune-di-asti-presentano-la-tensione-del-servizio-di-raccolta-verticale/](http://www.asp.asti.it/asp-e-il-comune-di-asti-presentano-la-tensione-del-servizio-di-raccolta-verticale/)

servizio, ci saranno quindi meno mezzi circolanti che si muoveranno con maggiore efficienza, con conseguente riduzione di carburante consumato e di emissioni di CO<sub>2</sub>. L'operatore dovrà selezionare il tipo di rifiuto e la zona. L'applicazione suggerirà la capacità standard del mezzo adibito alla raccolta e la durata del turno di lavoro. Tali parametri potranno comunque essere modificati in base alla disponibilità di mezzi e personale. Terminata la ricerca, il programma mostrerà la lista di cassonetti da svuotare e quelli esclusi dal percorso, alcune statistiche come il numero di cassonetti svuotati e non, la durata del viaggio, la quantità raccolta e quella rimanente. Tali informazioni saranno presenti sia numericamente che graficamente e aiuteranno l'operatore a decidere se “accettare” o meno il percorso consigliato.

## 2.4 Criticità

Le principali criticità sono legate ad alcuni parametri reali di cui non è stato possibile tenere conto.

### 2.4.1 Durata del viaggio

La durata del viaggio è stata calcolata utilizzando la velocità media del camion, la durata di svuotamento del cassonetto e la durata di svuotamento del camion in impianto. Tali parametri spesso non rispecchiano la realtà.

La durata dello spostamento del mezzo tra un contenitore e l'altro o verso la destinazione finale è influenzata dal traffico, dalle condizioni atmosferiche e dalla presenza di eventuali lavori stradali.

La durata di svuotamento del cassonetto è influenzata dall'esperienza dell'addetto e da ostacoli presenti nelle vicinanze.

La durata di svuotamento del camion in impianto è influenzata dall'afflusso presente in esso e dal tempo impiegato per gestire la documentazione relativa al conferimento.

Inoltre, per calcolare la durata degli spostamenti viene usata la distanza in linea d'aria tra i diversi luoghi. Nella realtà, il camion si muove in strada percorrendo quindi una distanza maggiore.

Dunque, la durata del viaggio mostrata nella schermata è nettamente inferiore al tempo che il camion impiegherebbe a effettuare tale giro di raccolta.

Questa criticità può essere ignorata poiché l'obiettivo programma è quello di trovare il percorso che massimizzi la quantità raccolta impiegando il minor tempo possibile. Vincolo rispettato nonostante la bassa veridicità di tale dato.

#### **2.4.2 Contenuto presente nei cassonetti**

I cassonetti saranno dotati di un sensore in grado di rilevare il livello di riempimento e tramite il software tale informazione sarà presente in un database aggiornato in tempo reale. Tali sensori non sono ancora stati installati, motivo per cui ho dovuto simulare il contenuto presente nei vari contenitori.

All'avvio del programma, i cassonetti verranno riempiti grazie alla funzione del model *riempiCassonetti()*.

Grazie a un ciclo *for*, si scorre la lista di tutti i cassonetti presenti nella città.

Per ogni contenitore viene generato un numero casuale, scelto da una distribuzione normale, che viene utilizzato per stabilire la percentuale di riempimento. Questa non può rispecchiare la realtà ma è una buona approssimazione.

```

public void riempিCassonetti()
{
    // all'inizio tutti i cassonetti sono vuoti --> scorro la lista e li riempio
    for(Cassonetto c: this.cassonetti)
    {
        Integer dimensione = c.getDimensione();
        Integer percentuale;

        // genero un numero casuale con distribuzione normale
        // Returns the next pseudorandom, Gaussian ("normally") distributed double value
        // with mean 0.0 and standard deviation 1.0 from this random number generator's sequence.
        Random random = new Random();
        Double gauss = random.nextGaussian();

        if(gauss < -2)
        {
            // riempimento casuale tra 0% <= x < 2%
            Random rand = new Random();
            percentuale = rand.nextInt(2);
        }
        else if(gauss < -1)
        {
            // riempimento casuale tra 2% <= x < 16%
            Random rand = new Random();
            percentuale = rand.nextInt(14) + 2;
        }
        else if(gauss < 1)
        {
            // riempimento casuale tra 16% <= x < 84%
            Random rand = new Random();
            percentuale = rand.nextInt(68) + 16;
        }
        else if(gauss < 2)
        {
            // riempimento casuale tra 84% <= x < 98%
            Random rand = new Random();
            percentuale = rand.nextInt(14) + 84;
        }
        else // gauss >= 2
        {
            // riempimento casuale tra 98% <= x <= 100%
            Random rand = new Random();
            percentuale = rand.nextInt(3) + 98;
        }

        // ora in percentuale abbiamo un numero compreso tra 0 e 100 (distribuito normalmente)
        // i cassonetti però sono riempibili al massimo fino al 80%
        percentuale = (int) Math.round(percentuale * 0.8);

        Integer contenuto = (int) (dimensione*(percentuale/100.0));

        Double effettivo = 0.0;
        switch (c.getTipo())
        {
            case "carta": effettivo = contenuto * (1 - this.riduzioneCarta); break;
            case "indifferenziata": effettivo = contenuto * (1 - this.riduzioneIndifferenziata); break;
            case "plastica": effettivo = contenuto * (1 - this.riduzionePlastica); break;
            case "vetro": effettivo = contenuto * (1 - this.riduzioneVetro); break;
        }

        c.setContenuto(contenuto);
        c.setPercentuale(percentuale);
        c.setEffettivo(effettivo);
    }
}

```

Figura 1: metodo `riempিCassonetti()` - Model

### 3 Descrizione del dataset

Asti Servizi Pubblici ha fornito i dati relativi alla geolocalizzazione dei cassonetti in forma confidenziale è stato dunque necessario rielaborarli al fine di poterli pubblicare.

I dati utilizzati dall'applicazione sono di tre tipi principali:

- Dati relativi al posizionamento dei cassonetti (*cassonetti*)
- Dati relativi alla sede e alle discariche (*luoghi*)
- Dati necessari al calcolo del percorso di raccolta:
  - Dimensione mezzo adibito alla raccolta:
    - Vetro
    - Altre raccolte
  - Durata di un turno di lavoro
  - Percentuali di riduzione dei rifiuti nel mezzo di raccolta:
    - Carta
    - Plastica
    - Vetro
    - Indifferenziata
  - Durata di svuotamento di un cassonetto
  - Durata di svuotamento del camion in discarica:
    - Vetro
    - Altre raccolte

## 3.1 Descrizione delle tabelle

### 3.1.1 Tabella cassonetti

Ad Asti sono previste 278 ecoisole, ognuna con 4 cassonetti (carta, plastica, vetro, indifferenziata), suddivise in 7 zone in modo da renderne più efficiente la gestione.

La tabella *cassonetti* contiene le informazioni relative ai 1112 contenitori previsti in città:

- id: identificativo del cassonetto
- latitudine: latitudine della posizione del cassonetto
- longitudine: longitudine della posizione del cassonetto
- indirizzo: indirizzo del cassonetto
- zona: zona a cui appartiene il cassonetto
- tipo: tipo di rifiuto conferibile nel cassonetto
- dimensione: capienza del cassonetto

### 3.1.2 Tabella luoghi

I camion adibiti alla raccolta sono parcheggiati all'interno della sede dell'azienda e tutti i servizi iniziano e terminano presso tale luogo.

Inoltre, l'azienda usufruisce del servizio di due impianti di stoccaggio e trattamento, uno per la raccolta del vetro e uno per la raccolta di carta, plastica e indifferenziata.

Le informazioni relative a questi tre luoghi sono necessarie per calcolare il percorso di raccolta e sono contenute nella tabella *luoghi*:

- nome: nome del luogo (sede o discarica)
- indirizzo: indirizzo del luogo
- latitudine: latitudine del luogo
- longitudine: longitudine del luogo

## 4 Strutture dati e algoritmi utilizzati

### 4.1 Struttura del progetto

L'applicazione è stata sviluppata in linguaggio Java, seguendo i pattern MVC (*Model-View-Controller*) e DAO (Database Acces Object).

Il progetto dell'applicazione è disponibile al seguente link:

<https://github.com/TdP-prove-finali/MarchiaroNoemi>

Il progetto si suddivide in quattro package, ognuno dedicato allo svolgimento di determinate funzionalità:

- **SortedWaste**: contiene le classi per l'avvio dell'applicazione
- **controller**: dedicato al controllo dell'interfaccia grafica, contiene una classe *controller* per ogni schermata dell'interfaccia
- **db**: contiene le classi necessarie per connettersi al database e leggerne i dati memorizzati
- **model**: package principale dell'applicazione, contiene la parte algoritmica e le classi che definiscono gli oggetti coinvolti

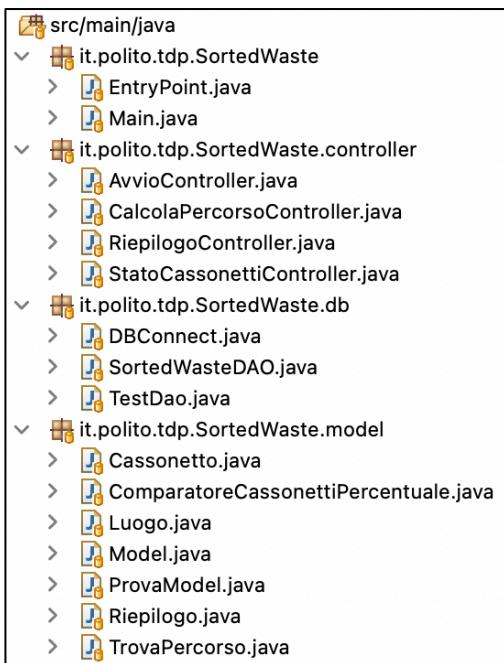


Figura 2: overview dei package

## 4.2 Algoritmi

Per la realizzazione del programma sono stati utilizzati molti algoritmi e tra questi il ruolo più importante è ricoperto dall'algoritmo ricorsivo che permette di trovare il percorso di raccolta.

Il risultato ottenuto non è ottimo. Per ottenere un percorso di raccolta ottimo su un grafo completo di  $N$  vertici è necessario provare tutti i possibili percorsi tramite un algoritmo ricorsivo. Questa operazione ha una complessità computazionale pari a  $O(N!)$  e dunque richiederebbe tempi di elaborazione inaccettabili.

È stato quindi utilizzato un algoritmo euristico che fornisce un risultato approssimato in tempi accettabili, una variante del “The nearest neighbour heuristic algorithm for the TSP problem”<sup>3</sup>. Tale algoritmo permette di risolvere il *travelling salesman problem*: data una lista di città e le distanze tra ogni coppia di città, qual è il percorso più breve possibile che visita ogni città esattamente una volta e torna alla città di origine? La chiave di questo metodo è visitare sempre la città più vicina, tra quelle non ancora visitate. Ciò semplifica molto il problema e su un grafo completo di  $N$  vertici questo algoritmo ha una complessità computazionale pari a  $O(N^2)$ .

Il limite più grande di questo algoritmo è la forte dipendenza del percorso dal primo vertice visitato. Motivo per cui è stato utilizzato un *repetitive nearest neighbor algorithm*: si prova ogni vertice come punto di partenza e poi si sceglie il percorso migliore. Si scorre la lista di tutti i vertici e si fa partire l'argoritmo *nearest neighbor* da ognuno di essi. Su un grafo completo di  $N$  vertici questo comporta  $N$  ripetizioni di tale algoritmo, portando la complessità computazionale a  $O(N^3)$ , nettamente inferiore a quella dell'algoritmo *brute force* citato a inizio paragrafo.

---

<sup>3</sup> <https://jgrapht.org/javadoc/org.jgrapht.core/org/jgrapht/alg/tour/NearestNeighborHeuristicTSP.html>

```

public List<Cassonetto> calcolaPercorso(String tipo, String zona, Integer capacitaMezzo, Integer durataTurno)
{
    this.creaGrafo(tipo, zona);
    this.trovaPercorso = new TrovaPercorso(this.grafo);
    this.sede = dao.getSede();
    this.discarica = null;
    if(tipo.compareTo("vetro") == 0)
    {
        this.discarica = dao.getDiscaricaVetro();
    }
    else
    {
        this.discarica = dao.getDiscaricaAltro();
    }
    this.trovaPercorso.init(tipo, zona, capacitaMezzo, durataTurno, this.sede, this.discarica);
    this.percorso = this.trovaPercorso.getPercorso();
    return this.percorso;
}

```

Figura 3: metodo *calcolaPercorso()* - Model

Per trovare il percorso di raccolta sono necessari i seguenti parametri:

- Tipo di rifiuto
- Zona in cui si vuole effettuare la raccolta
- Capacità del mezzo a disposizione
- Durata del turno di lavoro dell'addetto alla raccolta

Tali dati vengono inseriti dall'operatore e vengono passati dal controller al model, il quale, dopo aver ricevuto dal database le informazioni relative alla sede e alla discarica, crea e avvia un'istanza di una classe di appoggio unicamente destinata alla ricerca del percorso (*TrovaPercorso*).

La classe *TrovaPercorso* contiene le costanti necessarie al calcolo del percorso e gestisce l'algoritmo ricorsivo che permette di trovarlo.

Grazie a un metodo di inizializzazione si acquisiscono i dati passati dal model e si lancia la ricorsione aggiungendo ogni cassonetto, se non vuoto, come primo del percorso.

```

// inserisco ogni cassonetto come primo del percorso e lancio la ricorsione
for(Cassonetto primo: this.grafo.vertexSet())
{
    // se il cassonetto NON è vuoto, inizio il percorso da lui
    if(primo.getContenuto() != 0)
    {
        parziale.add(primo);

        // calcolo quantità raccolta --> contenuto del cassonetto "compresso" nel camion
        Double effettivo = primo.getEffettivo();
        // calcolo durata viaggio
        LatLng coord1 = primo.getPosizione();
        LatLng coord2 = this.sede.getPosizione();
        Double distanza = LatLngTool.distance(coord1, coord2, LengthUnit.KILOMETER);
        Integer durataViaggio = (int) (distanza / this.velocitaCamion * 60); // velocità in km/h --> durata in minuti

        // viaggio verso la discarica e svuotamento
        Double distanzaDiscarica = LatLngTool.distance(
            primo.getPosizione(), this.discarica.getPosizione(), LengthUnit.KILOMETER);
        // viaggio in sede
        Double distanzaSede = LatLngTool.distance(
            this.discarica.getPosizione(), this.sede.getPosizione(), LengthUnit.KILOMETER);
        Integer viaggioDiscarica = (int) (distanzaDiscarica / this.velocitaCamion * 60);
        Integer viaggioSede = (int) (distanzaSede / this.velocitaCamion * 60);

        // lancio la ricorsione
        cercaNearestNeighbor(parziale,
            durata + durataViaggio + this.durataSvuotamentoCassonetto + viaggioDiscarica +
            this.durataSvuotamentoDiscarica + viaggioSede,
            quantità + effettivo);

        // backtracking --> rimuovo l'ultimo cassonetto
        parziale.remove(parziale.size()-1);
    }
}

```

Figura 4: ciclo for che lancia la ricorsione – metodo init() - TrovaPercorso

#### 4.2.1 Ricerca del nearest neighbor

Nel metodo ricorsivo viene controllata la validità del percorso, cioè che i limiti di tempo e quantità non vengano superati. Se la soluzione parziale è valida, si stabilisce se sia la soluzione migliore controllandone la quantità raccolta.

Dopo tali controlli inizia la ricerca del nearest neighbor, il vertice con distanza minima dall'ultimo inserito nella soluzione parziale. Bisogna però controllare che il cassetto non faccia già parte del percorso e che non sia vuoto.

Si richiama poi il metodo ricorsivo dopo aver calcolato i parametri necessari.

```
// proseguo il percorso nel nearest neighbor non ancora svuotato
// trovo il vicino più vicino che:
// NON deve far già parte del percorso
// NON deve essere vuoto

Cassonetto ultimo = parziale.get(parziale.size()-1);
Double distanzaMinima = Double.MAX_VALUE;
Cassonetto piuVicino = null;

for(Cassonetto vicino: Graphs.neighborListOf(grafo, ultimo))
{
    if(!parziale.contains(vicino) && vicino.getContenuto() != 0)
    {
        Double distanza = this.grafo.getEdgeWeight(this.grafo.getEdge(vicino, ultimo));

        if(distanza < distanzaMinima)
        {
            distanzaMinima = distanza;
            piuVicino = vicino;
        }
    }
}

if(piuVicino == null)
{
    // i cassonetti sono o tutti vuoti o tutti già presenti nel percorso
    // fermo la ricorsione
    return;
}

// aggiungo il cassetto alla soluzione parziale
parziale.add(piuVicino);
```

Figura 5: ricerca del nearest neighbor – metodo cercaNearestNeighbor() - TrovaPercorso

## 4.2.2 Parametri della ricorsione

Il metodo ricorsivo necessita di tre parametri:

- List<Cassonetto> *parziale*: soluzione parziale
- Double *durata*: durata del viaggio della soluzione parziale
- Double *quantità*: quantità raccolta della soluzione parziale

I parametri *parziale* e *quantità* non creano grandi problemi, in quanto basta aggiungere un cassonetto alla lista e sommare la quantità raccolta. Per quanto riguarda invece la *durata* ci sono molti fattori da tenere in considerazione.

Durante l'inizializzazione, per il primo cassonetto, la durata si calcola sommando i tempi necessari alle seguenti azioni:

- Viaggio dalla sede al cassonetto
- Svuotamento del cassonetto
- Viaggio verso la discarica
- Svuotamento del camion in discarica
- Viaggio verso la sede

Nel metodo ricorsivo vengono aggiunti nuovi cassonetti alla lista, i quali avranno tempi di viaggio verso la discarica differenti. Di conseguenza lanciando la ricorsione bisogna:

- Aggiungere la durata del viaggio tra l'ultimo cassonetto presente nel percorso e il nuovo cassonetto
- Aggiungere la durata di svuotamento del cassonetto
- Rimuovere dalla durata il tempo di viaggio dall'ultimo cassonetto alla discarica
- Aggiungere alla durata il tempo di viaggio dal nuovo cassonetto alla discarica
- NON aggiungere il tempo di svuotamento del camion in discarica
- NON aggiungere il tempo di viaggio dalla discarica alla sede

Questi ultimi due tempi sono già stati inseriti nella durata nel metodo di inizializzazione in quanto restano invariabili, a prescindere dall'ultimo cassonetto inserito nel percorso.

```
// calcolo quantità raccolta --> contenuto del cassonetto "compresso" nel camion
Double effettivo = piuVicino.getEffettivo();

// calcolo durata viaggio tra ultimo e piuVicino
Double distanza = this.grafo.getEdgeWeight(this.grafo.getEdge(piuVicino, ultimo));
Integer durataViaggio = (int) (distanza / this.velocitaCamion * 60); // velocità in km/h --> durata in minuti

// proseguendo su questo ramo, aggiungo un altro cassonetto
// questo avrà una diversa distanza dalla discarica rispetto all'ultimo inserito

// viaggio verso la discarica di piuVICINO
Double distanzaDiscaricaPlus = LatLngTool.distance(
    piuVicino.getPosizione(), this.discarica.getPosizione(), LengthUnit.KILOMETER);
Integer viaggioDiscaricaPlus = (int) (distanzaDiscaricaPlus / this.velocitaCamion * 60);

// viaggio verso la discarica di ULTIMO
Double distanzaDiscaricaMinus = LatLngTool.distance(
    ultimo.getPosizione(), this.discarica.getPosizione(), LengthUnit.KILOMETER);
Integer viaggioDiscaricaMinus = (int) (distanzaDiscaricaMinus / this.velocitaCamion * 60);

// lancio la ricorsione
cercaNearestNeighbor(parziale,
    durata + durataViaggio + this.durataSvuotamentoCassonetto
    + viaggioDiscaricaPlus
    - viaggioDiscaricaMinus,
    quantità + effettivo);

// backtracking
// rimuovo l'ultimo cassonetto
parziale.remove(parziale.size()-1);
```

Figura 6: durata del viaggio della soluzione parziale - metodo `cercaNearestNeighbor()` - `TrovaPercorso`

## 5 Diagramma delle classi principali

Gli algoritmi principali di questa applicazione sono contenuti nelle classi *Model* e *TrovaPercorso* delle quali vengono riportati i diagrammi UML.

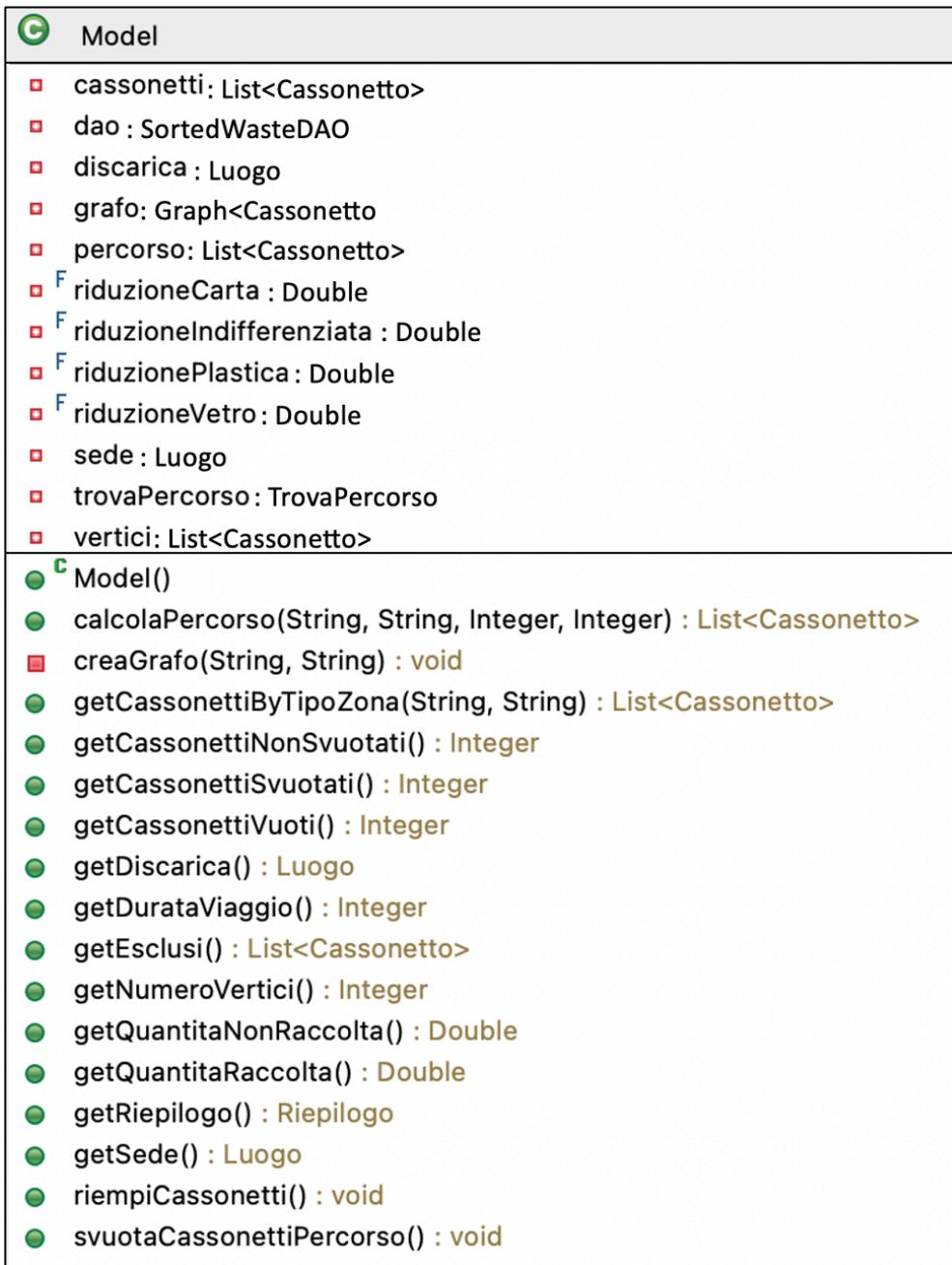


Figura 7: diagramma UML della classe Model

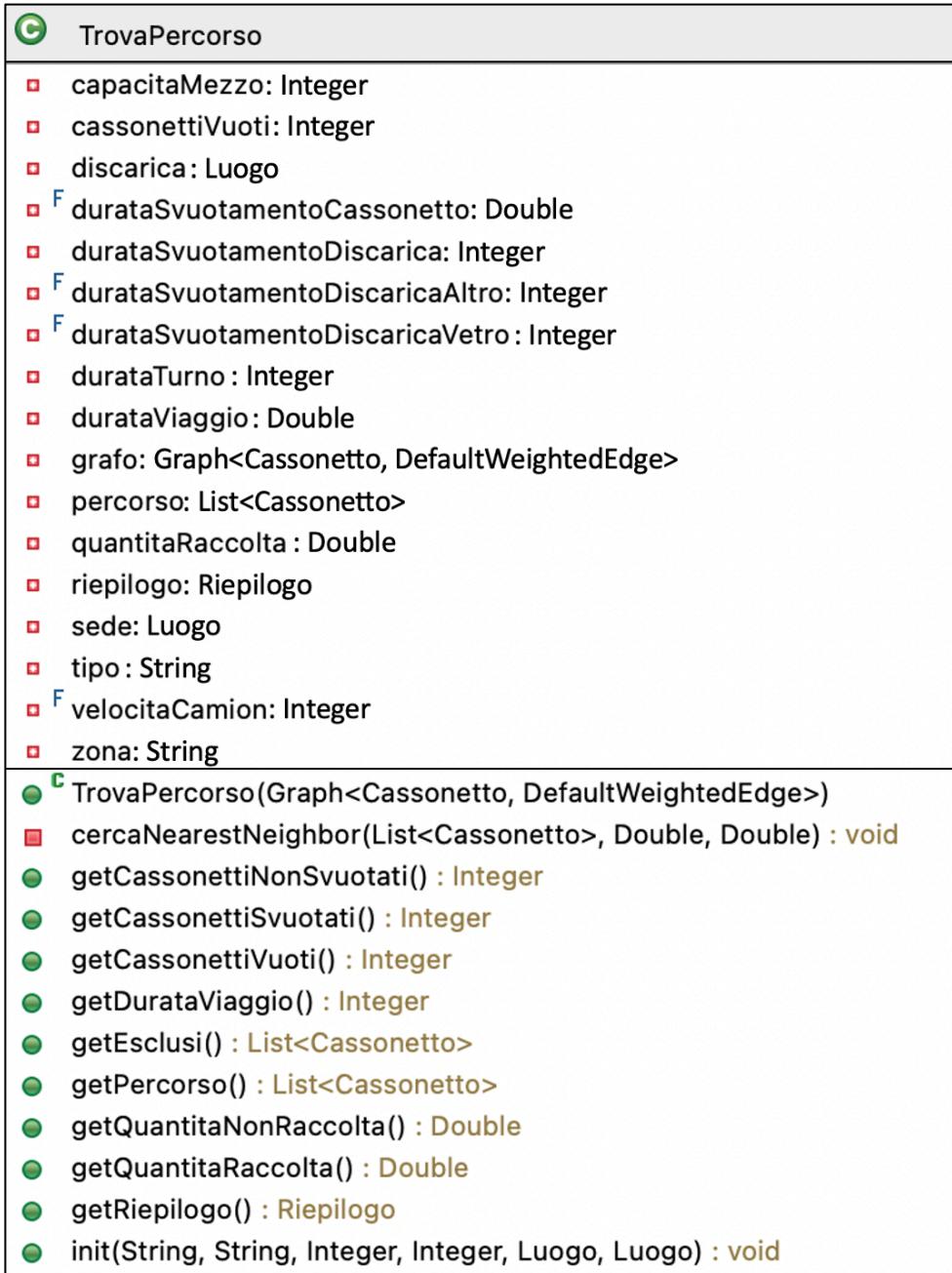


Figura 8: diagramma UML della classe TrovaPercorso

## 6 Interfaccia dell'applicazione

L'applicazione si compone di quattro schermate, ognuna dedicata allo svolgimento delle diverse funzionalità del software.

È stato realizzato un video dimostrativo sull'uso dell'applicazione, disponibile al seguente link: [https://youtu.be/Acg3TNwZ\\_5I](https://youtu.be/Acg3TNwZ_5I)

### 6.1 Avvio del programma

All'avvio del programma viene mostrata un'immagine contenente la mappa con il posizionamento dei cassonetti presenti nella città di Asti. Per procedere con l'utilizzo è necessario premere il pulsante *start*, il quale aggiorna il database contenente le informazioni sul livello di riempimento dei contenitori e abilita i due pulsanti che permettono di passare alle omonime schermate successive.

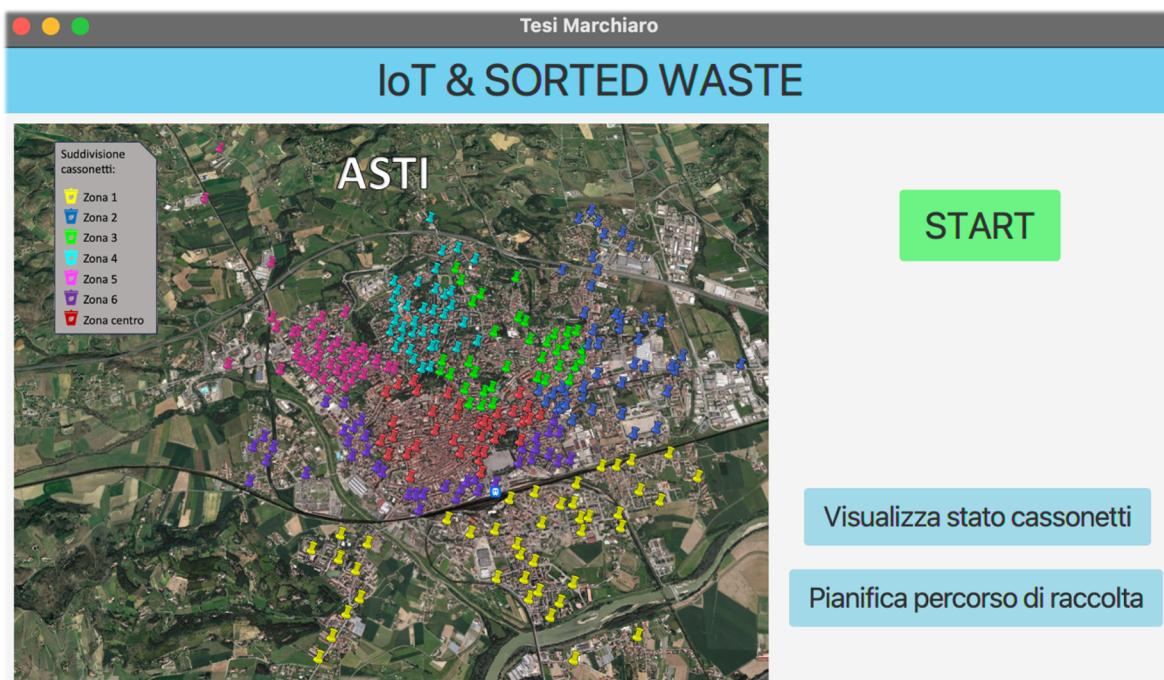


Figura 9: schermata di avvio

## 6.2 Visualizzazione dello stato cassonetti

Questa schermata permette di visualizzare il livello di riempimento dei cassonetti presenti nella città. È necessario scegliere la tipologia di rifiuto e la zona di cui si vuole effettuare la valutazione tramite i due menù a tendina e, successivamente, premere il pulsante *visualizza stato*. Viene così popolata la tabella e la zona dedicata alle statistiche riassuntive della situazione.

Per passare invece alla schermata successiva basta premere il pulsante *calcola percorso*.

The screenshot shows a software application window titled "Tesi Marchiaro" with a blue header bar. The main title "STATO CASSONETTI" is centered above a grid-based interface. On the left side, there is a vertical panel containing dropdown menus for "Selezione tipo:" and "Selezione zona:", a "Visualizza stato" button, and a "Calcola percorso" button. Below this panel are three input fields labeled "N cassonetti:", "Litri totali:", and "Litri camion:". To the right of the input fields is a large grid table with columns labeled "Indirizzo", "Percentuale", "Dimensione (L)", "Contenuto (L)", and "Zona". A message "Nessun contenuto nella tabella" is displayed in the center of the grid area. At the bottom of the window, a note in a blue-bordered box reads: "Selezione il tipo di rifiuto e la zona per visualizzare lo stato dei cassonetti. Per procedere al calcolo del percorso di raccolta premere il 'pulsante calcola percorso'."

Figura 10: schermata per la visualizzazione dello stato dei cassonetti

Tesi Marchiaro

## STATO CASSONETTI

vetro

Asti

Visualizza stato

Calcola percorso

Indirizzo	Percentuale	Dimensione (L)	Contenuto (L)	Zona
corso Savona 150	80	2250	1800	1
viale Vittoria 101	80	2250	1800	centro
via Pietro Micca 7	80	2250	1800	3
via Pietro Micca 37	80	2250	1800	3
via Tosi 34	80	2250	1800	3
via Vigna 27	79	2250	1777	6
corso Alba 236	79	2250	1777	1
via Don Gallo 54	79	2250	1777	6
corso Alessandria 390	78	2250	1755	2
via Liguria	78	2250	1755	2
via Manzoni 3	78	2250	1755	4
via Cagna 21	78	2250	1755	3
via Pallio 26	78	2250	1755	3
via Puccini 12	78	2250	1755	5
viale Vittoria 17	78	2250	1755	centro

N cassonetti: 278

Litri totali: 254713

Litri camion: 25000

In tabella sono presenti i cassonetti del tipo e zona selezionati, ordinati per riempimento decrescente.  
Inoltre è possibile calcolare un percorso di raccolta.

Figura 11: schermata dello stato dei cassonetti del ‘vetro’ nella zona ‘Asti’ (tutta la città)

Tesi Marchiaro

## STATO CASSONETTI

Indirizzo	Percentuale	Dimensione (L)	Contenuto (L)	Zona
via Pietro Micca 7	80	2250	1800	3
via Pietro Micca 37	80	2250	1800	3
via Tosi 34	80	2250	1800	3
via Cagna 21	78	2250	1755	3
via Pallio 26	78	2250	1755	3
via Verdi 22	77	2250	1732	3
strada Acacie 2	74	2250	1665	3
via Badoni 2	72	2250	1620	3
via Calcaterra 24	67	2250	1507	3
via Platone	66	2250	1485	3
via A. Certosa 22	65	2250	1462	3
via Fontana 2	64	2250	1440	3
via Borelli 26	59	2250	1327	3
strada Fortino 12	58	2250	1305	3
via Monterainero 22	54	2250	1215	3

In tabella sono presenti i cassonetti del tipo e zona selezionati, ordinati per riempimento decrescente.  
Inoltre è possibile calcolare un percorso di raccolta.

Figura 12: schermata dello stato dei cassonetti del 'vetro' nella zona '3'

### 6.3 Calcolo del percorso di raccolta

La schermata principale dell'applicazione è quella che permette di trovare il percorso di raccolta. L'utente deve selezionare dai menù a tendina la zona e la tipologia di rifiuto di cui si vuole calcolare il percorso. La scelta di quest'ultimo riempirà automaticamente le aree di testo sottostanti con i valori standard della capacità del mezzo e della durata del turno, i quali saranno comunque modificabili.

Alla pressione del pulsante *calcola* viene popolata l'intera schermata. Le tue tabelle contengono, rispettivamente, il percorso di raccolta consigliato e la lista dei cassonetti esclusi, ordinata per riempimento decrescente. Sono presenti le statistiche di riepilogo sia sotto forma numerica che grafica.

L'utente può calcolare ulteriori percorsi di raccolta, cambiando il tipo, la zona, la capacità del mezzo e/o la durata del turno. Può inoltre tornare alla schermata

precedente o accettare il percorso consigliato tramite l'apposito pulsante. Quest'ultima azione azzerà il livello di riempimento dei cassonetti presenti nel percorso, come se il camion fosse già passato a svuotarli (operazione necessaria vista la mancanza di un database aggiornato in tempo reale), e permette il passaggio alla schermata successiva.

**Tesi Marchiaro**

## CALCOLA PERCORSO

<p>Selezione tipo: <input type="button" value="▼"/></p> <p>Selezione zona: <input type="button" value="▼"/></p> <p>Capacità mezzo (L): <input type="text"/></p> <p>Durata viaggio massima (m): <input type="text"/></p> <p><input type="button" value="Calcola"/> <input type="button" value="Accetta percorso"/> <input type="button" value="Indietro"/></p>	<p>Percorso consigliato:</p> <table border="1" style="width: 100%; height: 150px;"> <thead> <tr> <th>Indirizzo</th> <th>Percentuale</th> </tr> </thead> <tbody> <tr><td colspan="2">Nessun contenuto nella tabella</td></tr> </tbody> </table> <p>Cassonetti esclusi:</p> <table border="1" style="width: 100%; height: 150px;"> <thead> <tr> <th>Indirizzo</th> <th>Percentuale</th> </tr> </thead> <tbody> <tr><td colspan="2">Nessun contenuto nella tabella</td></tr> </tbody> </table>	Indirizzo	Percentuale	Nessun contenuto nella tabella		Indirizzo	Percentuale	Nessun contenuto nella tabella	
Indirizzo	Percentuale								
Nessun contenuto nella tabella									
Indirizzo	Percentuale								
Nessun contenuto nella tabella									
<p>Cassonetti svuotati: <input type="text"/></p> <p>Cassonetti non svuotati: <input type="text"/></p> <p>Cassonetti vuoti: <input type="text"/></p> <p>Quantità raccolta (L): <input type="text"/></p> <p>Quantità non raccolta (L): <input type="text"/></p> <p>Durata effettiva viaggio (m): <input type="text"/></p>	<p><b>Cassonetti</b></p> <p><b>Rifiuti</b></p>								
<p>Selezione il tipo di rifiuto e la zona prima di procedere al calcolo del percorso.</p>									

Figura 13: schermata per cercare del percorso di raccolta

Tesi Marchiaro

## CALCOLA PERCORSO

plastica

1

Capacità mezzo (L):

Durata viaggio massima (m):

Calcola
Accetta percorso
Indietro

Percorso consigliato:

Indirizzo	Percentuale
via Gianotti 14	17
via del Barcailo 21	34
lungo Tanaro Pescatori	41
via Bosio 4	18
corso Savona 220	40
corso Savona 287	18
via Gancia 22	27
corso Savona 150	25
corso Savona 161	30

Cassonetti esclusi:

Indirizzo	Percentuale
corso Alba 46	79
via Tagliamento 4	67
via 101 B.Garibaldi	62
via Terraccini 9	61
via Pacotto 22	54
strada Trincere 48	44
via Cuneo 5	37
piazzale Nenni 7	34
corso Alba 97	26

Cassonetti



- Cassonetti svuotati
- Cassonetti vuoti
- Cassonetti non svuotati

Rifiuti



- Quantità raccolta
- Quantità non raccolta

Viene visualizzato il percorso ottimo con le relative statistiche.  
Si può accettare il percorso consigliato, calcolare un nuovo percorso con parametri diversi o tornare alla schermata iniziale.

Figura 14: schermata con il percorso di raccolta consigliato per la 'plastica' nella zona '1'

## 6.4 Riepilogo del percorso di raccolta

L'ultima schermata dell'applicazione mostra un riepilogo del percorso di raccolta accettato in precedenza. Rappresenta una sorta di foglio di viaggio che l'addetto alla raccolta deve seguire durante il turno di lavoro. Sono presenti, infatti, tutte le fermate da effettuare, dalla partenza in sede, ai cassonetti da svuotare fino all'indicazione della discarica assegnata al tipo di rifiuto raccolto. Sono inoltre indicate la tipologia del rifiuto da raccogliere, la zona in cui effettuare la raccolta e la capacità del camion assegnato.

L'utente dell'applicazione può decidere poi se pianificare un nuovo percorso o visualizzare lo stato dei cassonetti dopo la vuotatura.

Tesi Marchiaro

### RIEPILOGO PERCORSO

Tipologia rifiuto:	plastica											
Zona:	1											
Capacità camion (L):	20000											
Cassonetti da svuotare:	27											
Durata prevista (m):	75.5											
<b>Pianifica un nuovo percorso</b>												
<b>Visualizza stato cassonetti</b>												
Partenza da:	via delle Corse 2 (ASP Cantiere Igiene Urbana)											
Cassonetti:	<table border="1"><thead><tr><th>Indirizzo</th></tr></thead><tbody><tr><td>via Gianotti 14</td></tr><tr><td>via del Barcaiole 21</td></tr><tr><td>lungo Tanaro Pescatori</td></tr><tr><td>via Bosio 4</td></tr><tr><td>corso Savona 220</td></tr><tr><td>corso Savona 287</td></tr><tr><td>via Gancia 22</td></tr><tr><td>corso Savona 150</td></tr><tr><td>corso Savona 161</td></tr><tr><td>corso Venezia 39</td></tr></tbody></table>	Indirizzo	via Gianotti 14	via del Barcaiole 21	lungo Tanaro Pescatori	via Bosio 4	corso Savona 220	corso Savona 287	via Gancia 22	corso Savona 150	corso Savona 161	corso Venezia 39
Indirizzo												
via Gianotti 14												
via del Barcaiole 21												
lungo Tanaro Pescatori												
via Bosio 4												
corso Savona 220												
corso Savona 287												
via Gancia 22												
corso Savona 150												
corso Savona 161												
corso Venezia 39												
Discarica:	frazione Quarto Inferiore 273 D (G.A.I.A.)											
Ritorno a:	via delle Corse 2 (ASP Cantiere Igiene Urbana)											

Viene visualizzato il riepilogo del percorso ottimo calcolato.  
Inoltre è possibile tornare alle schermate precedenti.

Figura 15: schermata di riepilogo del percorso di raccolta accettato per la 'plastica' nella zona '1'

## 7 Esempio d'uso e risultati

I risultati ottenuti dall'applicazione sono estremamente variabili. Ciò è causato dalla generazione casuale del livello di riempimento dei cassonetti, che cambia ogni volta che si avvia il programma. Questa grande variabilità non è comunque considerabile come una limitazione dell'applicativo, in quanto la produzione di rifiuti da parte della cittadinanza è altrettanto imprevedibile.

### 7.1 Visualizzazione dello stato

I risultati ottenuti variano in base ai parametri in input: tipo di rifiuto e zona di raccolta selezionati.

Vengono mostrate le schermate relative ai risultati ottenuti visualizzando la situazione dei cassonetti del vetro nella zona 4 e della carta in tutta la città.

È doveroso notare la diversa dimensione dei cassonetti e dei camion per i tipi selezionati:

- Vetro: cassonetto 2.250 Litri, camion 25.000 Litri
- Carta: cassonetto 3.000 litri, camion 20.000 Litri (dati validi anche per plastica e indifferenziata)

Come spiegato nel paragrafo 3.1.1, nella città di Asti sono previsti 278 cassonetti per ogni tipologia di rifiuto e questi sono suddivisi in 7 zone, ognuna con in media 40 cassonetti. Ciò viene evidenziato nell'area dedicata al numero dei cassonetti, diversa ovviamente nelle due schermate mostrate.

STATO CASSONETTI					
	Indirizzo	Percentuale	Dimensione (L)	Contenuto (L)	Zona
vetro	via Petrarca 75	79	2250	1777	4
4	via Falcone 46	78	2250	1755	4
	via Boccaccio 11	74	2250	1665	4
	via Manzoni 3	71	2250	1597	4
	via Rabiglio 2	65	2250	1462	4
	via G. Musso 1	61	2250	1372	4
	via Manzoni 13	60	2250	1350	4
	via Conte Verde 111	60	2250	1350	4
	via Valence 2	60	2250	1350	4
	corso Dante 257	58	2250	1305	4
	via Conte Verde 148	57	2250	1282	4
	via Foscolo 8	54	2250	1215	4
	via G. Roretto 1	54	2250	1215	4
	corso Dante 134	54	2250	1215	4
	via Arduino 34	51	2250	1147	4

In tabella sono presenti i cassonetti del tipo e zona selezionati, ordinati per riempimento decrescente.  
Inoltre è possibile calcolare un percorso di raccolta.

Figura 17: schermata dello stato dei cassonetti del ‘vetro’ nella zona ‘4’

STATO CASSONETTI					
	Indirizzo	Percentuale	Dimensione (L)	Contenuto (L)	Zona
carta	via Cagna 21	80	3000	2400	3
Asti	via Borriida 5	79	3000	2370	1
	corso Alessandria 239	78	3000	2340	2
	strada Sesia 17	78	3000	2340	1
	via Vigna 37	78	3000	2340	6
	via Terracini 8	78	3000	2340	1
	corso G. Ferraris 4	78	3000	2340	6
	via Giovanni XXIII 13	78	3000	2340	4
	corso Don Minzoni 113	78	3000	2340	6
	strada Fortino 12	77	3000	2310	3
	Viale Partigiani 2	77	3000	2310	4
	via Catalani 138	76	3000	2280	5
	via Torchio 45	75	3000	2250	1
	corso Dante 164	75	3000	2250	4
	piazza I Maggio 4	75	3000	2250	2

In tabella sono presenti i cassonetti del tipo e zona selezionati, ordinati per riempimento decrescente.  
Inoltre è possibile calcolare un percorso di raccolta.

Figura 16: schermata dello stato dei cassonetti della ‘carta’ nella zona ‘Asti’ (tutta la città)

## 7.2 Calcolo del percorso

I risultati ottenuti variano in base ai parametri in input: tipo di rifiuto e zona di raccolta selezionati, capacità del mezzo a disposizione e durata del turno di lavoro.

Vengono mostrate le schermate relative ai risultati ottenuti cercando il percorso di raccolta per la plastica nella zona 1 della città, utilizzano i parametri standard per la capacità del mezzo e la durata del turno di lavoro (20.000 Litri e 6:20 ore).

CALCOLA PERCORSO	
plastica	Indirizzo
1	corso Savona 220
	via Bosio 4
	corso Savona 287
	via Gancia 22
	corso Savona 150
	corso Savona 161
	corso Venezia 39
	corso Venezia 89
	corso Venezia 115
Cassonetti svuotati:	24
Cassonetti non svuotati:	19
Cassonetti vuoti:	0
Quantità raccolta (L):	19992.0
Quantità non raccolta (L):	24330.0
Durata effettiva viaggio (m):	72
Cassonetti	Cassonetti svuotati
Rifiuti	Quantità raccolta

Venne visualizzato il percorso ottimo con le relative statistiche.  
Si può accettare il percorso consigliato, calcolare un nuovo percorso con parametri diversi o tornare alla schermata iniziale.

Figura 18: schermata con il primo percorso di raccolta consigliato per la ‘plastica’ nella zona ‘1’

RIEPILOGO PERCORSO	
Tipologia rifiuto:	Indirizzo
plastica	corso Savona 220
Zona:	1
Capacità camion (L):	20000
Cassonetti da svuotare:	24
Durata prevista (m):	72.0
Partenza da:	via delle Corse 2 (ASP Cantiere Igiene Urbana)
Cassonetti:	Indirizzo
	corso Savona 220
	via Bosio 4
	corso Savona 287
	via Gancia 22
	corso Savona 150
	corso Savona 161
	corso Venezia 39
	corso Venezia 89
	corso Venezia 115
	corso Savona 40
Discarica:	frazione Quarto Inferiore 273 D (G.A.I.A.)
Ritorno a:	via delle Corse 2 (ASP Cantiere Igiene Urbana)

Venne visualizzato il riepilogo del percorso ottimo calcolato.  
Inoltre è possibile tornare alle schermate precedenti.

Figura 19: schermata di riepilogo del primo percorso di raccolta accettato per la ‘plastica’ nella zona ‘1’

Si può notare come venga massimizzata la quantità di rifiuti raccolta (con una rimanenza nel camion di solamente 8 Litri) rimanendo abbondantemente nei vincoli di tempo. Quest'ultima voce, come indicato nel paragrafo 2.4.1, è una stima della durata del viaggio, ipotizzando non ci sia traffico o altri imprevisti, che in ogni caso non si discosta molto dalla realtà.

Accettando tale percorso, i 24 cassonetti vengono svuotati e pianificando un nuovo percorso di raccolta con gli stessi parametri inseriti in precedenza si ottiene il seguente output.



Figura 21: schermata con il secondo percorso di raccolta consigliato per la 'plastica' nella zona '1'



Figura 20: schermata di riepilogo del secondo percorso di raccolta accettato per la 'plastica' nella zona '1'

In questo caso, i 19 cassonetti esclusi dal percorso precedente vengono tutti svuotati durante questo nuovo giro di raccolta. La durata prevista non si discosta molto da quella precedente, nonostante il numero inferiore di cassonetti da svuotare. Ciò è dovuto alla maggiore distanza tra i singoli cassonetti e/o dalla distanza verso la sede e la discarica e anche questo non si discosta molto dalla realtà.

È doveroso precisare il motivo della discordanza tra la quantità non raccolta nel primo percorso (24.330 Litri) e quella raccolta nel secondo (17.031 litri). Nonostante i due percorsi permettano di svuotare tutti i cassonetti questi due valori, come si potrebbe immaginare, non coincidono. Questo è dovuto al compattamento dei rifiuti nel mezzo di raccolta, con percentuali di riduzione diversa in base al tipo di rifiuto, che per la plastica è pari al 30%. Quindi, la quantità non raccolta nel primo percorso è pari alla quantità di rifiuti presenti nei cassonetti esclusi, i quali una volta svuotati durante il secondo giro vengono compattati nel camion, passando da 24.330 Litri a 17.031 Litri ( $=24.330 * 0.7$ ).

## 8 Conclusioni e possibili miglioramenti

L'applicazione sviluppata raggiunge tutti gli obiettivi posti in fase di progettazione, indicati nel capitolo 1, nonostante le limitazioni indicate al paragrafo 2.4.

Essa permette di visualizzare lo stato dei cassonetti della città e di calcolare percorsi di raccolta in base alle necessità e disponibilità.

Inoltre, come mostrato negli esempi, l'interfaccia semplice e intuitiva, con l'aiuto dei suggerimenti situati in fondo a ogni schermata, permette l'utilizzo dell'applicativo anche a un utente meno esperto rispetto a un operatore specializzato del settore.

L'applicazione può essere implementata in svariati modi.

Con la messa in funzione dei sensori nei cassonetti, un database aggiornato in tempo reale permetterebbe una maggior veridicità nei risultati ottenuti.

L'aggiunta di un database contenente le informazioni relative ai mezzi di raccolta disponibili permetterebbe la scelta del camion tramite un apposito menù, evitando all'utente l'inserimento manuale di tale dato.

Un'ulteriore ottimizzazione del programma potrebbe essere quella di gestire in modo diverso le situazioni in cui si vogliono effettuare più giri di raccolta durante lo stesso turno di lavoro. In questo modo, dopo aver conferito i rifiuti in impianto, si potrebbe evitare il ritorno in sede e iniziare subito il nuovo giro di raccolta.

Un esempio di software reale, di cui questo progetto si può considerare la versione base, è un applicativo di Smart Waste Management implementato per il comune di Delft in Olanda volto ad ottimizzare e migliorare le attività di raccolta dei rifiuti della città attraverso l'installazione di sensori per la lettura del livello di riempimento dei cestini e analisi delle informazioni raccolte. Grazie ad algoritmi predittivi e funzioni di ranking dei vari cassonetti, viene trovato il percorso di raccolta ottimo.<sup>4</sup>

---

<sup>4</sup> [www.retealtatecnologia.it/technology-report/smart-waste-management-basato-su-machine-learning-il-comune-di-delft](http://www.retealtatecnologia.it/technology-report/smart-waste-management-basato-su-machine-learning-il-comune-di-delft)

## Sitografia

- <https://www.asp.asti.it/asp-e-il-comune-di-asti-presentano-l'estensione-del-servizio-di-raccolta-verticale/>
- <https://jgrapht.org/javadoc/org.jgrapht.core/org/jgrapht/alg/tour/NearestNeighborHeuristicTSP.html>
- <https://www.retealtatecnologia.it/technology-report/smart-waste-management-basato-su-machine-learning-il-comune-di-delft>



[IoT & sorted waste: la transizione 4.0 di ASP e Comune di Asti](#) © 2023 by [Noemi Marchiaro](#) is licensed under [CC BY-NC-SA 4.0](#)