



Università degli Studi di Padova

DIPARTIMENTO DI MATEMATICA "TULLIO LEVI CIVITA"

Corso di laurea in INFORMATICA

**Implementazione di modelli di
programmazione matematica per
problemi di bin packing**

Relatore

LUIGI DE GIOVANNI
UNIVERSITÀ DI PADOVA

Laureando

DANIEL ROSSI

© DANIEL ROSSI
ALL RIGHTS RESERVED, 2018

Ringraziamenti

Ecco raggiunto un altro traguardo molto importante carico di felicità che vorrei condividere con coloro che mi sono stati vicino durante questi anni.

Prima di tutto vorrei porgere i miei ringraziamenti al Professor De Giovanni per il supporto fornitomi durante il periodo di stage e durante la stesura della tesi.

Ringrazio l'azienda Trans-Cel per l'opportunità di stage offertomi, grazie a Filippo, Nicola e Beatrice per le tante ore passate insieme.

Ringrazio la mia famiglia ed in particolare mia madre per lo sforzo fatto nello spronarmi a ricercare un futuro in cui mi sentissi realizzato.

Ai miei compagni di Università, per le giornate passate insieme in Torre Archimede, nostra seconda casa, per le risate, le giornate brutte e quelle belle, grazie di avermi fatto sentire apprezzato e coinvolto, grazie per il tempo dedicatomi, in particolare grazie a Victor e Mihai per gli importanti momenti passati insieme a studiare, sarà difficile trovare migliori compagni di corso di voi.

Infine voglio ringraziare Anna, non ultima per importanza, per avermi sempre sostenuto durante tutti questi anni, per avermi ascoltato nei momenti di difficoltà e aver condiviso con me la felicità dei momenti belli, non penso potrò mai ringraziarti abbastanza.

Padova, Dicembre 2018

Daniel Rossi

Abstract

Il documento qui di seguito illustra il lavoro svolto durante lo stage curriculare, della durata di trecentoventi ore, presso l'azienda di trasporti Trans-Cel.

Lo scopo di tale stage è stato quello di realizzare dei modelli di programmazione lineare per la risoluzione dello Strip Packing Problem, questi sono poi stati utilizzati per valutare oggettivamente la bontà delle soluzioni fornite dall'euristica aziendale, la quale si occupa della risoluzione di tale problema. Si è operato un confronto con le rispettive soluzioni ottenute con modelli matematici, messi in condizioni tali da operare con gli stessi vincoli imposti all'euristica.

Lo scopo era di fornire all'azienda uno strumento che permettesse di misurare oggettivamente la distanza delle soluzioni fornite dall'euristica dall'ottimo.

Gli obiettivi di maggiore importanza sono stati l'apprendimento del linguaggio di programmazione Python e delle librerie correlate utili per la rappresentazione delle soluzioni e la manipolazione dei dati. La libreria Google Or-Tools [1] per la prototipazione dei modelli.

L'obiettivo finale era la realizzazione di modelli di programmazione lineare che permettessero di produrre soluzioni corrette e ottime da confrontare con quelle fornite dall'euristica, per valutarne quantitativamente la bontà.

Indice

RINGRAZIAMENTI	ii
ABSTRACT	v
LISTA DELLE FIGURE	ix
LISTA DELLE TABELLE	xi
1 INTRODUZIONE	1
1.1 L'azienda	1
1.2 L'idea	2
1.3 Organizzazione del testo	2
1.4 Convenzioni tipografiche	3
1.5 Convenzioni modelli	3
2 PROCESSI E METODOLOGIE	5
2.1 Contesto	5
2.2 Introduzione al progetto	6
2.3 Vincoli temporali, tecnologici e metodologici	7
2.4 Requisiti e obiettivi	7
2.5 Pianificazione	10
2.6 Ambiente di lavoro	11
2.6.1 Metodi di sviluppo	11
2.6.2 Gestione di progetto	12
2.6.3 Linguaggio di programmazione e ambiente di sviluppo . .	12
2.7 Analisi dei rischi	13
3 INQUADRAMENTO DELLE ATTIVITÀ DI STAGE	15
3.1 Il progetto aziendale	15
3.2 Il progetto di stage	16
3.3 Astrazione del problema	16
3.4 Variante del Bin Packing	17
4 MODELLI PER IL BIN PACKING	19
4.1 Modelli di programmazione matematica	19
4.2 Modelli di programmazione lineari	20

4.3	Modello 2D	21
4.4	Modello 2D con rotazione	23
4.5	Modello 2D con rotazione e sequenza scarico	26
4.6	Modello 3D con rotazione e sovrapposizione	31
5	TEST E VALIDAZIONE	37
5.1	Istanze	37
5.2	Numero di oggetti	38
5.3	Validazione soluzioni fornite	39
6	TECNOLOGIE E STRUMENTI	41
6.1	Tecnologie	41
6.1.1	Python	41
6.1.2	Google Or-Tools	42
6.1.3	Cbc	42
6.1.4	Boost	43
6.1.5	Pandas	43
6.1.6	Matplotlib	44
6.1.7	Multiprocessing	44
6.2	Strumenti	45
6.2.1	Dropbox	45
6.2.2	GitHub	45
6.2.3	Visual Studio Code	45
6.2.4	Jupyter Notebook	46
6.2.5	PIP	46
6.2.6	Taiga	47
7	RISULTATI	49
7.1	Errori	50
7.2	Risultati modello 2DR	51
7.3	Risultati modello 2DRS	52
7.4	Risultati modello 3D	53
8	CONCLUSIONE	55
8.1	Consuntivo finale	55
8.2	Raggiungimento degli obiettivi	57
8.3	Conoscenze acquisite	57
8.4	Valutazione personale	57
	GLOSSARIO	59
	ACRONIMI	60

Lista delle figure

2.1	Diagramma d Gantt	10
3.1	Contenitori Bin Packing	17
4.1	Veduta area - piano cartesiano	21
4.2	Pacchi ruotati	23
4.3	Vie scaricamento pacco	26
4.4	Corretta sequenza di scarico	26
4.5	Grafico con merci 3D	31
5.1	Grafico tempi esecuzione	39
6.1	Logo Python	41
6.2	Logo Or-Tools	42
6.3	Logo Cbc	42
6.4	Logo Boost	43
6.5	Logo Pandas	43
6.6	Logo Matplotlib	44
6.7	Logo Multiprocessing	44
6.8	Logo Dropbox	45
6.9	Logo GitHub	45
6.10	Logo Visual Studio Code	45
6.11	Logo Jupyter Notebook	46
6.12	Logo Pip	46
6.13	Logo Taiga	47

Lista delle tabelle

2.1	Pianificazione concordata nel piano di lavoro	11
5.1	Le dimensioni dei gruppi di istanze eseguite	38
7.1	Risultati 2DR ottimi	51
7.2	Risultati 2DR best bound	51
7.3	Risultati 2DRS ottimi	52
7.4	Risultati 2DRS best bound	52
7.5	Risultati 3D ottimi	53
7.6	Risultati 3D best bound	53
8.1	Ripartizione reale delle ore di stage	56

1

Introduzione

1.1 L'azienda

Trans-Cel è un'azienda di trasporti che opera nel settore da oltre trent'anni, ha una numerosa flotta composta da bilici e motrici con cui trasporta merci nel nord e centro Italia. Tra le qualità che contraddistinguono questa azienda c'è la tecnica del groupage ed il trasporto di merci pericolose.

L'azienda ha sede ad Albignasego in provincia di Padova nel cui ufficio operativo vengono organizzati in tempo reale i viaggi dei mezzi per trasportare le merci dei clienti, fornendo anche un servizio di deposito se richiesto.

Da questa realtà si evince come si debba essere sempre pronti alla comunicazione con il cliente, fornendo loro un servizio di trasporto adeguato che li soddisfi ma che permetta all'azienda di realizzare un profitto, è da qui che nasce la necessità di un software decisionale di supporto.

1.2 L'idea

All'interno del prodotto software che sta realizzando l'azienda, è in via di sviluppo una euristica che permette di organizzare al meglio le merci all'interno del container del camion in modo automatico. L'idea è quella di valutare la bontà delle *soluzioni*^[9] fornite dalla stessa, confrontandole con quelle fornite dal modello e individuare se e con quali tipi di pacchi queste riportino differenze maggiori.

1.3 Organizzazione del testo

Di seguito viene riportata per ogni capitolo una piccola descrizione delle tematiche trattate:

- **Capitolo 2:** in questo capitolo vengono riportati gli obiettivi generali e la pianificazione concordata con l'azienda, inoltre vengono riportate le metodologie e strumenti utilizzati in generale, infine una analisi dei rischi;
- **Capitolo 3:** viene descritto il problema generale e come è stato risolto dall'azienda, viene illustrato lo scopo dello stage, viene data una definizione astratta del problema in esame;
- **Capitolo 4:** vengono riportati i modelli utilizzati e una descrizione dell'idea di base, per ciascuno di essi verrà riportato integralmente la lista delle variabili utilizzate, descrivendo inoltre anche cosa modellino, viene riportato per ciascun modello una descrizione dei vincoli, a corredo di tutto questo ci sarà del materiale grafico utilizzato durante lo stage;
- **Capitolo 5:** vengono riportate le modalità con cui si sono eseguiti i test, illustrando come siano stati strutturati e come sia stato possibile verificare le soluzioni fornite dai modelli;
- **Capitolo 6:** vengono riportati gli strumenti adottati per lo svolgimento delle attività, corredati da una breve descrizione che riporti come sono stati utilizzati;
- **Capitolo 7:** vengono riportati i risultati ottenuti per ciascun modello, fornendo una serie di osservazioni sulle peculiarità dei gruppi di istanze usate nei test;
- **Capitolo 8:** vengono riportate le conclusioni relative al numero di obiettivi soddisfatti e all'effettiva suddivisione delle ore rispetto tali obiettivi.

1.4 Convenzioni tipografiche

Il testo adotta le seguenti convenzioni tipografiche:

- ogni acronimo, abbreviazione, parola ambigua o tecnica viene spiegata e chiarificata alla fine del testo;
- ogni parola di glossario alla prima apparizione verrà etichetta come segue: *parola*^[g];
- ogni riga di un elenco puntato terminerà con un *a* parte l'ultima riga che si concluderà con un punto.

1.5 Convenzioni modelli

Ogni qual volta si dovrà fare riferimento ad un modello vi si farà riferimento attraverso i seguenti acronimi:

- **2D**: modello in 2 dimensioni, considera solo profondità e larghezza;
- **2DR**: modello in 2 dimensioni con la rotazione;
- **2DRS**: modello in 2 dimensioni con rotazione e sequenza di scarico;
- **3D**: modello in 3 dimensioni con rotazione e sovrapposizione, la rotazione è la stessa disponibile nei modelli 2D, vengono considerate quindi larghezza, profondità e altezza.

2

Processi e metodologie

In questo capitolo verranno riportati in modo approfondito lo scopo e gli obiettivi dello stage, contestualizzazione delle attività alla realtà aziendale e scadenziario delle stesse.

2.1 Contesto

Il progetto generale nasce dalla visione di Filippo Sottovia, titolare dell'azienda, di realizzare un software decisionale di supporto che permettesse di soddisfare molteplici obiettivi quali:

- agevolazione degli operatori nello svolgimento delle loro mansioni;
- facilitazione del processo decisionale richiedendo così lavoratori meno esperti;
- nuove attività frutto del tempo risparmiato grazie all'aumento della produttività;
- condivisione in tempo reale delle informazioni sullo stato dei trasporti;
- stima di costi e profitti disponibile in ogni momento.

Il software, fornisce un'interfaccia grafica web intuitiva che fa dell'usabilità la propria punta di diamante. Essa infatti persegue l'idea di rendere il più semplice possibile operare sul sistema senza però mancare di professionalità. Il sistema inoltre è equipaggiato con algoritmi che permettono di ottimizzare gli ordini organizzando al meglio le merci da trasportare ripartendole nei diversi camion della flotta, tenendo conto degli orari di carico e scarico, delle sedi dei clienti/fornitori, delle condizioni di traffico sulle strade, delle pause dovute per legge agli autisti e della particolarità delle merci.

2.2 Introduzione al progetto

L'azienda per permettere di stimare lo spazio occupato dalle merci ha sviluppato un'*euristica*^[9], questa riceve in input le merci da trasportare divise nei diversi ordini, le approssima a parallelepipedi e le dispone al meglio sul pianale del camion, sovrapponendole dove possibile, con l'obiettivo di ridurre al minimo lo spazio lineare occupato dalle stesse, così facendo si risparmia spazio per eventuali altre merci da caricare qualora arrivassero nuovi ordini. A rendere ancora più complicato il lavoro dell'euristica riportiamo tre principali problemi da tenere in considerazione:

- Stabilità degli oggetti: la faccia inferiore di ciascun oggetto deve poggiare per intero sulle facce superiori di altri oggetti sotto di sé o sul pianale del camion;
- Sequenza di scarico: ogni oggetto deve poter essere scaricato lateralmente o frontalmente, questo implica che non può avere intorno a sé merci che ne blocchino lo scarico, in quanto appartenenti ad ordini successivi;
- Baricentro degli oggetti: peso delle merci deve essere equamente distribuito sul pianale del camion;
- Sovrapponibilità delle merci: alcune merci non sono sovrapponibili.

2.3 Vincoli temporali, tecnologici e metodologici

Nel periodo di stage svolto presso l'azienda mi è stato chiesto di tenere un *diario* condiviso utilizzando DropBox [2], nel suddetto diario mi si richiedeva di annotare giornalmente l'avanzare del lavoro riportando idee, osservazioni riguardanti criticità rilevate e positività riscontrate negli strumenti utilizzati. Nella cartella condivisa mi è stato chiesto di includere anche gli articoli accademici letti e le presentazioni fatte in azienda man mano che si proseguiva con lo stage.

La presentazione di metà stage è stato un evento importante alla quale ha presenziato tutto il team di sviluppo, parte del personale d'amministrazione e il Professor De Giovanni che mi ha fornito preziosi consigli su cosa orientare gli sforzi per la seconda metà dello stage alla luce di quanto ottenuto nella prima metà.

Prima di iniziare lo stage è stato concordato con l'azienda un piano di lavoro su un totale di 320 ore, lavorando 5 giorni a settimana, 8 ore per ciascun giorno.

2.4 Requisiti e obiettivi

Nella tabella riportata di seguito vengono elencati gli obiettivi dello stage, a corredo degli stessi vi sarà un codice univoco ed una breve descrizione.

Ogni obiettivo è provvisto di un codice identificativo formato da una delle seguenti stringhe ob,de,op rappresentante il livello di importanza e da un numero incrementale positivo, che rispetta la seguente nomenclatura:

[importanza][identificativo].

Il livello di importanza di ciascun obiettivo può essere uno tra i seguenti:

- Obbligatorio: individuato dalla stringa *ob*, sono obiettivi fondamentali per la riuscita del progetto, il loro soddisfacimento dovrà verificarsi obbligatoriamente entro la fine dello stage, pena il fallimento dello stesso;
- Desiderabile: individuato dalla stringa *de*, sono obiettivi secondari su cui però si nutre dell'interesse, il loro soddisfacimento è auspicabile entro la fine dello stage;
- Opzionale: individuato dalla stringa *op*, sono obiettivi di contorno su cui si nutre poco interesse, la loro realizzazione si verificherà nel momento in cui

si dovesse soddisfare tutti gli obiettivi obbligatori e desiderabili prima della fine dello stage.

Si prevede lo svolgimento dei seguenti obiettivi:

- Obbligatori

- ob01: individuazione e analisi *constraints* modello 2D;
- ob02: modello matematico 2D con *framework* di modellazione algebrica;
- ob03: traduzione modello in Python con l'ausilio di Google OR-Tools;
- ob04: test sul modello 2D e confronto con l'euristica;
- ob05: individuazione e analisi *constraints* modello 2DR;
- ob06: modello matematico 2DR con *framework* di modellazione algebrica;
- ob07: traduzione modello in Python con l'ausilio di Google OR-Tools;
- ob08: test sul modello 2DR e confronto con l'euristica;

- Desiderabili

- de01: individuazione e analisi *constraints* modello 3D;
- de02: modello matematico 3D con *framework* di modellazione algebrica;
- de03: traduzione modello in Python con l'ausilio di Google OR-Tools;
- de04: test sul modello 3D e confronto con l'euristica;

- Opzionali

- op01: evoluzione euristica, fornita dall'azienda, con nuove funzionalità;

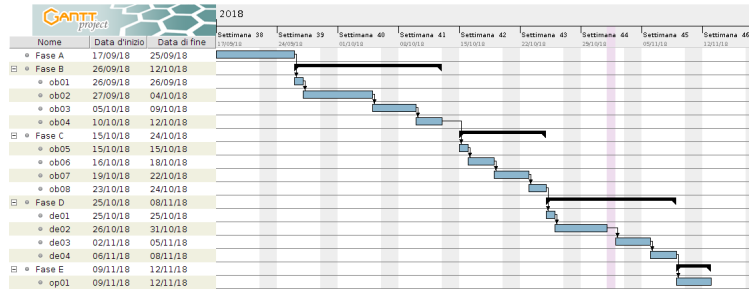


Figure 2.1: Diagramma di Gantt dello stage.

2.5 Pianificazione

Con le ore a disposizione per questo stage si è proceduto a organizzare come segue le attività:

- **Formazione:** si è visto necessario approfondire la programmazione lineare e la letteratura correlata al problema del bin packing, imparare il linguaggio di programmazione Python [3], l'ambiente fornito dallo strumento Jupyter notebook [4] e imparare ad usare il framework Google Or-Tools [1].
- **Modello 2D:** questo periodo di stage ha richiesto la prototipazione, realizzazione e test del modello 2D e il confronto con l'euristica.
- **Modello 2DR:** questo periodo di stage ha richiesto la prototipazione, realizzazione e test del modello 2DR e il confronto con l'euristica.
- **Modello 3D:** questo periodo di stage ha richiesto la prototipazione, realizzazione e test del modello 3D e il confronto con l'euristica.
- **Lavoro sviluppo euristica:** periodo richiesto per realizzare delle funzionalità da integrare nell'euristica.

Ogni periodo di sviluppo di un modello sarà poi diviso a sua volta nelle seguenti attività:

- analisi dei constraints
- prototipazione
- test del modello

- confronto con l'euristica.

La pianificazione delle attività è stata distribuita come mostrato nella tabella (2.1).

Durata in ore		Descrizione dell'attività
56		A: Formazione
	8	<ul style="list-style-type: none"> • Ricerca <i>framework</i> di modellazione algebrica • Studio di tale <i>framework</i> • Studio Google OR - Tools
	24	
	24	
104		B: Versione modello 2D
	8	<ul style="list-style-type: none"> • Individuazione ed analisi <i>constraints</i> • Prototipazione modello • Traduzione in Python - Google OR - Tools • Test e confronto con euristica
	48	
	24	
	24	
72		C: Versione modello 2D con rotazione
	8	<ul style="list-style-type: none"> • Individuazione ed analisi <i>constraints</i> • Prototipazione modello • Traduzione in Python - Google OR - Tools • Test e confronto con euristica
	24	
	16	
	16	
72		D: Versione modello 3D con sovrapposizione
	8	<ul style="list-style-type: none"> • Individuazione ed analisi <i>constraints</i> • Prototipazione modello • Traduzione in Python - Google OR - Tools • Test e confronto con euristica
	32	
	16	
	24	
16		E: Lavoro sviluppo euristica
	16	<ul style="list-style-type: none"> • Realizzazione nuove funzioni euristica
Totale: 320		

Table 2.1: Pianificazione concordata nel piano di lavoro

2.6 Ambiente di lavoro

2.6.1 Metodi di sviluppo

Il *ciclo di vita*^[9] di un prodotto in Trans-Cel segue il *modello incrementale*^[9], formato da un periodo di concezione dell'idea, analisi della stessa, progettazione ed

infine partendo dagli obiettivi più importanti si realizza il prodotto rilasciando periodicamente una versione dello stesso, questo dovrà mostrare le nuove funzionalità sviluppate e dimostri così l'*incremento*^[g] fatto. In quest'ottica lo sviluppo di un modello può essere associato ad un incremento, le cui *milestone*^[g] è la ricezione dei risultati, a sua volta lo sviluppo di ciascun modello segue il modello incrementale, che può essere visto come l'insieme di tre principali attività:

- Analisi letteratura: lettura articoli accademici riportanti modelli simili o idee per lo sviluppo degli stessi.
- Scrittura: scrittura del modello e integrazione con il sistema grazie al framework Or-Tools.
- Verifica: testing massivo e verifica delle soluzioni fornite.

Inoltre è possibile cogliere delle analogie con il modello di ciclo di vita *agile*^[g] per quanto riguarda i *brainstorming*^[g], momenti di confronto creativo che stimolano il ragionamento in modo critico, utili per fare il punto della situazione oltre che esporre le problematiche che si stanno incontrando.

2.6.2 Gestione di progetto

Per quanto riguarda la gestione di progetto sono stati utilizzati alcuni strumenti descritti con maggiore dettaglio nel **Capitolo 6**. In generale per la gestione dei task da eseguire si è fatto uso di Taiga [5], uno strumento di *project management*^[g], per la gestione della comunicazione e condivisione informazioni si è fatto uso dell'applicazione Telegram [6], per la condivisione di documentazione e articoli si è fatto uso del servizio DropBox [2], per il versionamento si è fatto uso del servizio GitHub [7] per la familiarità dello strumento, per quanto riguarda l'interfaccia con cui versionare ho utilizzato git [8] da terminale.

2.6.3 Linguaggio di programmazione e ambiente di sviluppo

Per la totalità dello stage si è lavorato utilizzando Jupyter notebook [4]. Con questo strumento è stato possibile scrivere programmi in linguaggio Python [3] in modo molto agevole. Questo linguaggio di programmazione è orientato agli

oggetti ed interpretato dinamicamente al momento dell'esecuzione da un interprete. Python [3] risulta veramente versatile in quanto fornisce incredibili funzionalità utilizzabili in modo semplice e intuitivo, dispone di moltissimi moduli che permettono le più svariate cose, inoltre è stato possibile grazie alla libreria Boost [9] esporre costrutti C++ [10] a Python [3] permettendo di utilizzarli nei propri programmi, questo ha permesso di mantenere efficienza e modularità.

2.7 Analisi dei rischi

In questa sezione vengono riportati i principali rischi che si prefiguravano all'inizio dello stage. Ciascuno di essi oltre ad avere una breve descrizione riporta il livello di rischio e come si possa fare per evitarlo:

- **Difficoltà nelle tecnologie adottate**

Ad inizio stage è stato chiaro che Python [3] avrebbe avuto un ruolo dominante nel progetto ma la mole di librerie e estensioni rendeva il linguaggio troppo vasto da poter approfondire nella sua interezza, ed oltre a questo vi era il modulo Or-Tools e Pandas [11] da approfondire e utilizzare.

- **Livello di rischio:** Basso;
- **Contromisure:** Studiare approfonditamente il linguaggio Python [3] e i moduli sopra citati.

- **Difficoltà di integrazione nel team**

Di fondamentale importanza per la riuscita di un progetto è la cooperazione con i colleghi e la creazione di un ambiente di lavoro sano e che stimoli la produttività, essendo un nuovo arrivato inserito in un ambiente soggetto a forte stress per le stringenti scadenze vi era la possibilità di entrare in conflitto con qualche collega.

- **Livello di rischio:** Basso;
- **Contromisure:** Perseguire un atteggiamento positivo, critico e oggettivo.

- **Contrattempi dovuti a malattie e impegni**

Un rischio da tenere in considerazione è quello dovuto a impegni o malattie

che precludano la possibilità di recarsi nel luogo di lavoro, data la durata dello stage è sicuramente possibile possa verificarsi.

- **Livello di rischio:** Basso;
- **Contromisure:** Organizzare precedentemente ogni impegno non lavorativo e tempo di *slack*^[g] per evitare contrattempi.

- **Difficoltà di stima dei tempi previsti**

Con un progetto di così lunga durata e l'inesperienza che ci portiamo appresso è possibile che vengano fatti degli errori di valutazione in termini di tempistiche per lo svolgimento delle diverse attività pianificate.

- **Livello di rischio:** Medio;
- **Contromisure:** Rendere partecipi nella definizione del piano di lavoro persone esperte, come il tutor aziendale e il project manager.

3

Inquadramento delle attività di stage

3.1 Il progetto aziendale

Per riuscire ad inquadrare al meglio lo scopo del progetto di stage è importante capire a fondo il progetto generale che l'azienda porta avanti da ormai alcuni anni, poiché essi sono tra loro intrinsecamente legati. Il progetto generale mira a fornire un prodotto che possa far progredire il mercato del trasporto attraverso la digitalizzazione dello stesso, questo attraverso il superamento dei vecchi gestionali, utilizzati concretamente solo per la fatturazione, e l'introduzione di una piattaforma web attraverso cui gestire ogni passaggio nel rapporto cliente-operatore.

Il software permette di gestire la flotta di mezzi e organizzare gli ordini di carico-scarico commissionati dai clienti. Questa organizzazione può essere fatta manualmente da un operatore esperto oppure affidandosi ad algoritmi, questi organizzano gli ordini utilizzando i camion disponibili nel miglior modo possibile.

Il sistema dispone di un algoritmo per il *vehicle routing*^[9], a cui si può dire faccia capo l'intero progetto, questo si occupa di ottimizzare i viaggi dei camionisti tenendo conto di un numero incredibile di variabili, questo algoritmo ha subito diversi rilasci in un'ottica di miglioramento continuo, per renderlo sempre più efficiente, versatile e preciso.

In questo contesto avere un'euristica che fornisca la valutazione approssimativa

dello spazio occupato da un certo numero di merci è molto importante, questo perché è necessario sapere quanti mezzi siano necessari per trasportare le merci richieste e quanto spazio sia ancora disponibile per ordini futuri.

3.2 Il progetto di stage

Il progetto di stage dopo l'evento StageIt [12] ed alcune riunioni presso l'azienda è stato studiato dettagliatamente e riportato nel documento piano di lavoro, nello stesso sono riportati gli obiettivi e la pianificazione delle attività.

L'importanza di ottenere una valutazione reale dello spazio occupato da un certo numero di oggetti è fondamentale. Lo stage prevede la realizzazione di diversi modelli il cui sviluppo sarà incrementale in quanto ciascun modello eredita struttura dei vincoli dai precedenti.

Ogni modello per definizione permetterà di individuare la disposizione ottima dell'istanza corrente, questa potrà poi essere confrontata con quella dell'euristica. Dal confronto potremo individuare informazioni utili che permettano di valutare oggettivamente la bontà delle soluzioni fornite dall'euristica, allo scopo di capire se con il passare delle versioni rilasciate l'euristica stia migliorando e di quanto rispetto alle versioni passate.

3.3 Astrazione del problema

Una possibile astrazione del problema è considerare il container e le merci come se fossero parallelepipedi.

Questa astrazione è necessaria per poter approssimare il tutto come richiesto dal *Bin Packing Problem*, problema su cui si è studiato e discusso molto in ambito accademico portando alla realizzazione di algoritmi esatti ed euristiche.

Si consideri un insieme $I = \{1, \dots, n\}$ di oggetti aventi dimensioni w_i , d_i e h_i con $i \in I$, un insieme $J = \{1, \dots, m\}$ di contenitori di uguale dimensione W , D e H , ogni oggetto $i \in I$ ha la possibilità di essere ruotato di 90° rispetto la propria base e si da per ipotesi che $w_i \leq W$, $d_i \leq D$ e $h_i \leq H$. Esiste inoltre la possibilità che gli oggetti vengano sovrapposti. l'obiettivo è di utilizzare il minor numero di contenitori J che riescano a contenere tutti gli oggetti dell'insieme I . Nella

figura (3.1) vengono mostrati due contenitori di uguali dimensioni contenenti dei pacchi di diverse dimensioni.

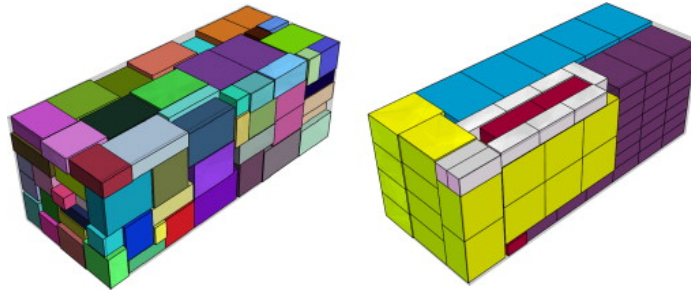


Figure 3.1: Due contenitori completamente pieni

3.4 Variante del Bin Packing

Una variante del problema del Bin Packing, il più famoso di una lunga serie di problemi di disposizione di oggetti in contenitori, è lo *Strip Packing Problem*. Le differenze principali dal precedente possono essere individuate di seguito:

- **Numero di contenitori:** viene utilizzato un singolo contenitore;
- **profondità:** la profondità del contenitore è infinita;
- **Oggetti:** come obiettivo si quello di occupare minor spazio lineare possibile rispetto la profondità del contenitore.

4

Modelli per il Bin Packing

In questo capitolo presenteremo prima una rapida introduzione ai modelli matematici, tratta dal libro [13], per poi analizzare nel dettaglio ciascun modello utilizzato spiegando ciascun vincolo e delineandone quindi l'idea di base.

4.1 Modelli di programmazione matematica

Quando si parla di problemi di ottimizzazione spesso si fa riferimento alla ricerca operativa e ai modelli matematici. Questi modelli producono una soluzione che massimizza o minimizza una funzione obiettivo su un certo dominio, questa funzione può rappresentare un aspetto di costo o guadagno. La soluzione restituita ha la proprietà di essere una soluzione ottima. La funzione obiettivo solitamente viene rappresentata come segue:

$$\max z = f(x) \text{ (oppure } \min z = f(x))$$

s.t.

$$g_i(x) = \begin{cases} \leq b_i \\ = b_i, & i = 1, \dots, m \\ \geq b_i \end{cases}$$

$$x = (x_1, \dots, x_n) \in X \subseteq \mathbb{R}^n$$

In un modello sono presenti:

- **Variabili decisionali:** sono variabili con un dominio prefissato, che vengono utilizzate per formulare tutti gli altri elementi del modello, agendo sui valori assunti dalle stesse si troverà la soluzione ottima;
- **Funzione obiettivo:** funzione che deve essere massimizzata o minimizzata in base agli altri elementi su un dominio dato;
- **Vincoli:** serie di vincoli che correlano tra loro le variabili decisionali e permettono di descrivere condizioni fisiche o requisiti particolari richiesti dalla soluzione.

4.2 Modelli di programmazione lineari

Particolari tipi di problemi di programmazione matematica sono quelli in cui la funzione obiettivo $f(x)$ e i vincoli $g_i(x)$ sono funzioni lineari, in tal caso si può parlare di modello di *programmazione lineare*, espresso nel seguente modo:

$$\max z = \sum_{j=1}^n c_j x_j$$

s.t.

$$\sum_{j=1}^n a_{ij} x_j = \begin{cases} \leq b_i \\ = b_i, & i = 1, \dots, m \\ \geq b_i \end{cases}$$

$$x = (x_1, \dots, x_n) \in X \subseteq \mathbb{R}^n$$

Nonostante i modelli non lineari siano a volte molto più compatti ed intuitivi da capire, i modelli lineari mantengono comunque semplicità e sono facilmente risolvibili, sono molti gli strumenti ad uso *commerciale*^[9] e *open source*^[9] disponibili per la loro risoluzione, quello utilizzato durante lo stage è il *solver*^[9] open source di programmazione lineare intera Cbc [14] scritto in C++ [10] interfacciato con

Python [3] grazie a Google Or-Tools [1]. La creazione di un modello lineare parte dall'osservazione di un problema reale, fissando un obiettivo e creando vincoli che descrivano i requisiti estratti dal problema astraendo il tutto ad un insieme di vincoli lineari. Ogni variabile può avere un dominio differente, continua ($x_j \in \mathbb{R}$), intera positiva ($x_j \in \mathbb{Z}^+$) o binaria ($x_j \in \{0, 1\}$) a seconda dell'utilizzo che se ne deve fare.

4.3 Modello 2D

Il modello per la versione 2D è stato preso dall'articolo [15], esso considera solo larghezza e profondità degli oggetti e non ne permette la rotazione, l'idea di base è quella di considerare solo la *visione aerea* del contenitore che viene così approssimata ad un rettangolo, questo rettangolo ha larghezza prefissata e profondità infinita, le due dimensioni posso essere considerate come assi di un piano cartesiano aventi origine nel punto di intersezione tra i due assi, individuato nella figura (4.1) dal pallino verde, per convenzione diciamo che l'asse x sarà quello della larghezza e l'asse y quello della profondità.

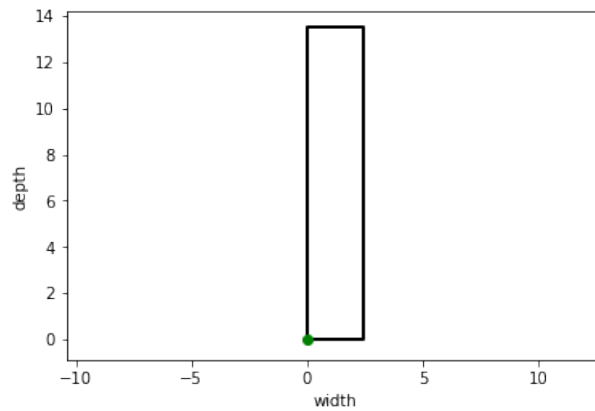


Figure 4.1: Veduta area del container

Il modello consiste delle seguenti variabili continue positive:

- la variabile continua x_i con $i \in I$ individua la coordinata sull'asse x del vertice in basso a sinistra dell'oggetto i;

- la variabile continua y_i con $i \in I$ individua la coordinata sull'asse y del vertice in basso a sinistra dell'oggetto i;
- la variabile continua D rappresenta i metri lineari rispetto la profondità e va minimizzata.

Il modello consiste delle seguenti variabili binarie:

- la variabile binaria l_{ij} con $i, j \in I$ assume il valore 1 se l'oggetto i è situato alla sinistra dell'oggetto j, altrimenti è 0;
- la variabile binaria b_{ij} con $i, j \in I$ assume il valore 1 se l'oggetto i è situato al di sotto dell'oggetto j, altrimenti è 0.

$$\min D \quad (4.1)$$

$$\text{s.t.} \quad l_{ij} + l_{ji} + b_{ij} + b_{ji} \geq 1 \quad i < j \quad i, j \in I \quad (4.2)$$

$$y_i - y_j + M_d b_{ij} \leq M_d - d_i \quad i, j \in I \quad (4.3)$$

$$x_i - x_j + M_w l_{ij} \leq M_w - w_i \quad i, j \in I \quad (4.4)$$

$$x_i + w_i \leq W \quad i \in I \quad (4.5)$$

$$y_i + d_i \leq D \quad i \in I \quad (4.6)$$

$$b_{ij}, l_{ij} \in \{0, 1\} \quad i \neq j \quad i, j \in I \quad (4.7)$$

$$x_i, y_i, w_i, d_i \in \mathbb{R}^+ \quad i \in I \quad (4.8)$$

Spieghiamo ora il significato di ciascun vincolo:

- il vincolo (4.1) la funzione obiettivo minimizza i metri lineari rispetto alla profondità;
- il vincolo (4.2) impone che presi due oggetti almeno uno dei due si trovi dietro o alla sinistra dell'altro;
- il vincolo (4.3) dice che se $b_{ij} = 1$ allora impone che l'oggetto i si trovi dietro l'oggetto j, altrimenti il vincolo viene disattivato.
- il vincolo (4.4) dice che se $l_{ij} = 1$ allora impone che l'oggetto i si trovi alla sinistra l'oggetto j, altrimenti il vincolo viene disattivato.

- il vincolo (4.5) impone che dato il valore x_i , corrispondente alla coordinata x dell'angolo sinistro più arretrato dell'oggetto, sommandoci la larghezza w_i dell'oggetto e ottenuta quindi la coordinata x dell'angolo destro più arretrato, questa sia contenuta nell'intervallo $[0, W]$;
- il vincolo (4.6) impone che dato il valore y_i , corrispondente alla coordinata y dell'angolo sinistro più arretrato dell'oggetto, sommandoci la profondità d_i dell'oggetto e ottenuta quindi la coordinata y dell'angolo sinistro più avanzato, questa sia contenuta nell'intervallo $[0, D]$;

4.4 Modello 2D con rotazione

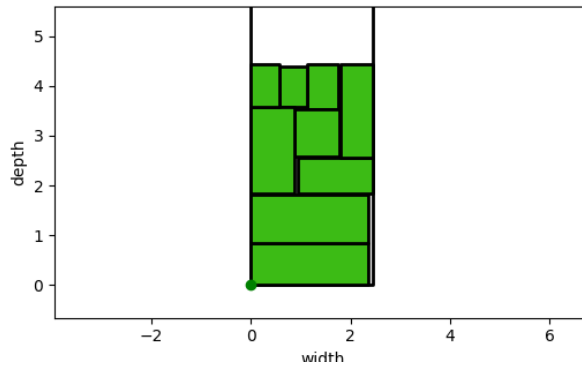


Figure 4.2: Veduta aerea di 9 pacchi, alcuni ruotati.

Il modello consiste delle seguenti variabili continue positive:

- la variabile continua x_i con $i \in I$ individua la coordinata sull'asse x del vertice in basso a sinistra dell'oggetto i ;
- la variabile continua y_i con $i \in I$ individua la coordinata sull'asse y del vertice in basso a sinistra dell'oggetto i ;
- la variabile continua D rappresenta i metri lineari rispetto la profondità e va minimizzata.
- la variabile continua Ω_i con $i \in I$ assume il valore d_i se l'oggetto è stato ruotato altrimenti w_i ;

- la variabile continua Δ_i con $i \in I$ assume il valore w_i se l'oggetto è stato ruotato altrimenti d_i ;

Il modello consiste delle seguenti variabili binarie:

- la variabile binaria l_{ij} con $i, j \in I$ assume il valore 1 se l'oggetto i è situato alla sinistra dell'oggetto j , altrimenti è 0;
- la variabile binaria b_{ij} con $i, j \in I$ assume il valore 1 se l'oggetto i è situato al di sotto dell'oggetto j , altrimenti è 0.
- la variabile binaria r_i con $i \in I$ assume il valore 1 se l'oggetto i è ruotato rispetto la propria base di 90° , ne risulta che w_i e d_i sono invertiti, altrimenti 0;

$$\min D \quad (4.9)$$

$$\text{s.t.} \quad l_{ij} + l_{ji} + b_{ij} + b_{ji} \geq 1 \quad i < j \quad i, j \in I \quad (4.10)$$

$$\Delta_i = d_i(1 - r_i) - w_i r_i \quad i, j \in I \quad (4.11)$$

$$\Omega_i = w_i(1 - r_i) - d_i r_i \quad i, j \in I \quad (4.12)$$

$$y_i - y_j + M_d b_{ij} \leq M_d - \Delta_i \quad i, j \in I \quad (4.13)$$

$$x_i - x_j + M_w l_{ij} \leq M_w - \Omega_i \quad i, j \in I \quad (4.14)$$

$$x_i + \Omega_i \leq W \quad i \in I \quad (4.15)$$

$$y_i + \Delta_i \leq D \quad i \in I \quad (4.16)$$

$$b_{ij}, l_{ij}, r_i \in \{0, 1\} \quad i \neq j \quad i, j \in I \quad (4.17)$$

$$x_i, y_i, w_i, d_i, \Delta_i, \Omega_i \in \mathbb{R}^+ \quad i \in I \quad (4.18)$$

Spieghiamo ora il significato di ciascun vincolo:

- il vincolo (4.10) impone che presi due oggetti almeno uno dei due si trovi dietro o alla sinistra dell'altro;
- il vincolo (4.11) impone che Δ_i corrisponda alla profondità corretta considerando la rotazione;
- il vincolo (4.12) impone che Ω_i corrisponda alla larghezza corretta considerando la rotazione;

- il vincolo (4.13) dice che se $b_{ij} = 1$ allora impone che l'oggetto i si trovi dietro l'oggetto j , altrimenti il vincolo viene disattivato.
- il vincolo (4.14) dice che se $l_{ij} = 1$ allora impone che l'oggetto i si trovi alla sinistra l'oggetto j , altrimenti il vincolo viene disattivato.
- il vincolo (4.15) impone che dato il valore x_i , corrispondente alla coordinata x dell'angolo sinistro più arretrato dell'oggetto, sommandoci la larghezza w_i dell'oggetto e ottenendo quindi la coordinata x dell'angolo destro più arretrato, questa sia contenuta nell'intervallo $[0, W]$;
- il vincolo (4.16) impone che dato il valore y_i , corrispondente alla coordinata y dell'angolo sinistro più arretrato dell'oggetto, sommandoci la profondità d_i dell'oggetto e ottenendo quindi la coordinata y dell'angolo sinistro più avanzato, questa sia contenuta nell'intervallo $[0, D]$;

4.5 Modello 2D con rotazione e sequenza scarico

Con questo modello ci proponiamo di disporre nel miglior modo possibile gli oggetti sul pianale del camion tenendo conto della sequenza di scarico, un oggetto può essere scaricato lateralmente alla sua destra/sinistra oppure dal fondo del camion.

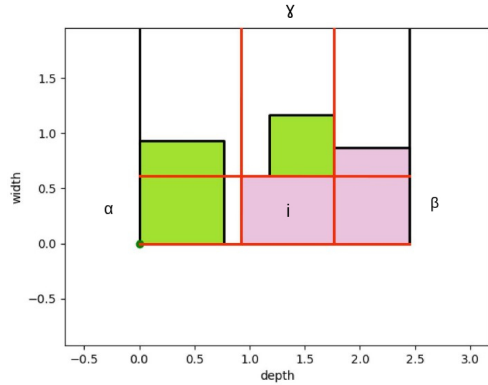


Figure 4.3: Vie di scarico pacco

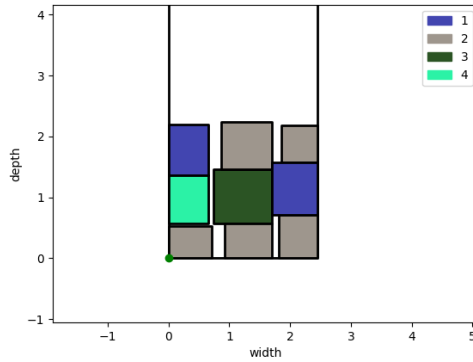


Figure 4.4: Sequenza corretta

Nel momento in cui l'oggetto verrà scaricato questo dovrà avere almeno una di queste vie libere affinché possa essere scaricato. Come si può notare nella figura 4.3, le vie attraverso cui può essere effettuato uno scarico vengono deciso attraverso l'indice $v \in \theta = \{\alpha, \beta, \gamma\}$ che individua rispettivamente:

- $v = \alpha$: via alla sinistra del pacco da cui può essere sfilato;
- $v = \beta$: via alla destra del pacco da cui può essere sfilato;
- $v = \gamma$: via di fronte al pacco da cui può essere sfilato;

Il modello introduce le seguenti variabili continue positive:

- la variabile continua x_i con $i \in I$ individua la coordinata sull'asse x del vertice in basso a sinistra dell'oggetto i ;
- la variabile continua y_i con $i \in I$ individua la coordinata sull'asse y del vertice in basso a sinistra dell'oggetto i ;
- la variabile continua D rappresenta i metri lineari rispetto la profondità e va minimizzata.

- la variabile continua Ω_i con $i \in I$ assume il valore d_i se l'oggetto è stato ruotato altrimenti w_i ;
- la variabile continua Δ_i con $i \in I$ assume il valore w_i se l'oggetto è stato ruotato altrimenti d_i ;

Il modello introduce le seguenti variabili binarie:

- la variabile binaria l_{ij} con $i, j \in I$ assume il valore 1 se l'oggetto i è situato alla sinistra dell'oggetto j , altrimenti è 0;
- la variabile binaria b_{ij} con $i, j \in I$ assume il valore 1 se l'oggetto i è situato al di sotto dell'oggetto j , altrimenti è 0.
- la variabile binaria r_i con $i \in I$ assume il valore 1 se l'oggetto i è ruotato su se stesso di 90° , ne risulta che w_i e d_i sono invertiti, altrimenti è 0;
- la variabile binaria α_{ij} con $i, j \in I$ assume il valore 1 se l'oggetto i ha alla sua sinistra l'oggetto j e questo gli impedisca di essere sfilato da sinistra, altrimenti è 0;
- la variabile binaria β_{ij} con $i, j \in I$ assume il valore 1 se l'oggetto i ha alla sua destra l'oggetto j e questo gli impedisca di essere sfilato da destra, altrimenti è 0;
- la variabile binaria γ_{ij} con $i, j \in I$ assume il valore 1 se l'oggetto i ha davanti a se l'oggetto j e questo gli impedisca di essere sfilato centralmente, altrimenti è 0;
- la variabile binaria s_{vi} con $i \in I$ e $v \in \theta$ assume il valore 1 se l'oggetto i ha nella direzione v almeno un oggetto che gli impedisca di essere sfilato nella rispettiva direzione, altrimenti è 0;
- la parametro binario o_{ij} assume il valore 1 se l'oggetto i appartiene ad un ordine che deve essere scaricato prima dell'ordine a cui appartiene l'oggetto j , altrimenti è 0;

$$\min D \quad (4.19)$$

$$\text{s.t.} \quad l_{ij} + l_{ji} + b_{ij} + b_{ji} \geq 1 \quad i < j \quad i, j \in I \quad (4.20)$$

$$y_i - y_j + M_d b_{ij} \leq M_d - \Delta_i \quad i, j \in I \quad (4.21)$$

$$x_i - x_j + M_w l_{ij} \leq M_w - \Omega_i \quad i, j \in I \quad (4.22)$$

$$x_i + \Omega_i \leq W \quad i, j \in I \quad (4.23)$$

$$y_i + \Delta_i \leq D \quad i, j \in I \quad (4.24)$$

$$\Delta_i = d_i(1 - r_i) - w_i r_i \quad i, j \in I \quad (4.25)$$

$$\Omega_i = w_i(1 - r_i) - d_i r_i \quad i, j \in I \quad (4.26)$$

$$\alpha_{ij} \leq l_{ij} \quad i, j \in I \quad (4.27)$$

$$\alpha_{ij} \leq 1 - (b_{ij} + b_{ji}) \quad i, j \in I \quad (4.28)$$

$$\alpha_{ij} \geq l_{ij} - M_\alpha(b_{ij} + b_{ji}) \quad i, j \in I \quad (4.29)$$

$$\beta_{ij} \leq l_{ji} \quad i, j \in I \quad (4.30)$$

$$\beta_{ij} \leq 1 - (b_{ij} + b_{ji}) \quad i, j \in I \quad (4.31)$$

$$\beta_{ij} \geq l_{ji} - M_\beta(b_{ij} + b_{ji}) \quad i, j \in I \quad (4.32)$$

$$\gamma_{ij} \leq b_{ij} \quad i, j \in I \quad (4.33)$$

$$\gamma_{ij} \leq 1 - (l_{ij} + l_{ji}) \quad i, j \in I \quad (4.34)$$

$$\gamma_{ij} \geq b_{ij} - M_\gamma(l_{ij} + l_{ji}) \quad i, j \in I \quad (4.35)$$

$$\alpha_{ij} o_{ij} \leq s_{\alpha i} \quad i, j \in I \quad (4.36)$$

$$\beta_{ij} o_{ij} \leq s_{\beta i} \quad i, j \in I \quad (4.37)$$

$$\gamma_{ij} o_{ij} \leq s_{\gamma i} \quad i, j \in I \quad (4.38)$$

$$\sum_{v \in \theta} s_{vi} \leq 2 \quad i \in I \quad (4.39)$$

$$b_{ij}, l_{ij}, \alpha_{ij}, \beta_{ij}, \gamma_{ij}, r_i \in \{0, 1\} \quad i \neq j \quad i, j \in I \quad (4.40)$$

$$x_i, y_i, w_i, d_i, \Delta_i, \Omega_i \in \mathbb{R}^+ \quad i \in I \quad (4.41)$$

$$s_{vi} \in \{0, 1\} \quad i \in I \wedge v \in \theta \quad (4.42)$$

Spieghiamo ora il significato di ciascun vincolo:

- il vincolo (4.20) impone che presi due oggetti almeno uno dei due si trovi dietro o alla sinistra dell'altro;
- il vincolo (4.21) dice che se $b_{ij} = 1$ allora impone che l'oggetto i si trovi dietro l'oggetto j , altrimenti il vincolo viene disattivato;
- il vincolo (4.22) dice che se $l_{ij} = 1$ allora impone che l'oggetto i si trovi alla sinistra l'oggetto j , altrimenti il vincolo viene disattivato;
- il vincolo (4.23) impone che dato il valore x_i , corrispondente alla coordinata x dell'angolo sinistro più arretrato dell'oggetto, sommandoci la larghezza w_i dell'oggetto e ottenendo quindi la coordinata x dell'angolo destro più arretrato, questa sia contenuta nell'intervallo $[0, W]$;
- il vincolo (4.24) impone che dato il valore y_i , corrispondente alla coordinata y dell'angolo sinistro più arretrato dell'oggetto, sommandoci la profondità d_i dell'oggetto e ottenendo quindi la coordinata y dell'angolo sinistro più avanzato, questa sia contenuta nell'intervallo $[0, D]$;
- il vincolo (4.25) impone che Δ_i corrisponda alla profondità corretta considerando la rotazione;
- il vincolo (4.26) impone che Ω_i corrisponda alla larghezza corretta considerando la rotazione;
- i vincoli (4.27), (4.28) e (4.29) impongono che se $l_{ij} = 1 \wedge t_{ij} + t_{ji} = 0 \Rightarrow \alpha_{ij} = 1$, che significa che se un oggetto i si trovi a destra di un oggetto j ma nessuno dei due sia dietro all'altro, allora l'oggetto i non potrà essere sfilato dalla sponda sinistra del camion;
- i vincoli (4.30), (4.31) e (4.32) impongono che se $l_{ji} = 1 \wedge t_{ij} + t_{ji} = 0 \Rightarrow \beta_{ij} = 1$, che significa che se un oggetto i si trovi a sinistra di un oggetto j ma nessuno dei due sia dietro all'altro, allora l'oggetto i non potrà essere sfilato dalla sponda destra del camion;
- i vincoli (4.33), (4.34) e (4.35) impongono che se $b_{ij} = 1 \wedge l_{ij} + l_{ji} = 0 \Rightarrow \gamma_{ij} = 1$, che significa che se un oggetto i si trovi dietro un oggetto j ma nessuno dei due sia alla sinistra dell'altro, allora l'oggetto i non potrà essere sfilato centralmente del camion;
- il vincolo (4.36) impone che se $s_{1i} = 1$ allora potranno esservi dei pacchi posizionati alla sua sinistra che ne blocchino lo scarico, se $s_{1i} = 0$ allora la via dovrà essere libera permettendo così lo scarico;

- il vincolo (4.37) impone che se $s_{2i} = 1$ allora potranno esservi dei pacchi posizionati alla sua destra che ne blocchino lo scarico, se $s_{2i} = 0$ allora la via dovrà essere libera permettendo così lo scarico;
- il vincolo (4.38) impone che se $s_{3i} = 1$ allora potranno esservi dei pacchi posizionati davanti a sé che ne blocchino lo scarico, se $s_{3i} = 0$ allora la via dovrà essere libera permettendo così lo scarico;
- il vincolo (4.39) impone che $\sum_{v \in \theta} s_{vi} \leq 2$ ossia che possano essere occupate al massimo 2 vie di scarico lasciandone libera almeno 1.

4.6 Modello 3D con rotazione e sovrapposizione

Con questo modello, evoluzione del modello 2D con rotazione, ci si propone di introdurre la sovrapposizione degli oggetti, considerando però che alcuni oggetti potranno avere come condizione quella di non poter avere altri oggetti al di sopra.

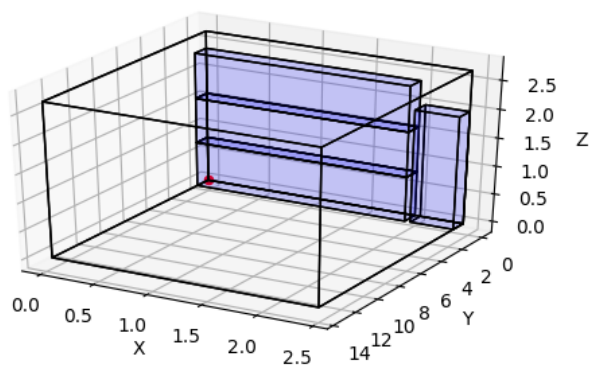


Figure 4.5: Veduta 3D di alcuni pacchi.

Un problema non indifferente inerente questo modello viene individuato nella stabilità degli oggetti, soluzioni che riportino oggetti sopraelevati, le cui aree di base non poggino completamente sugli oggetti sottostanti sono da ritenersi non valide. Per risolvere questo problema si è deciso di optare per una semplificazione, se un oggetto i si trova sopra ad un oggetto j , allora l'area di base dell'oggetto i dovrà essere interamente appoggiata alla faccia superiore dell'oggetto j .

Il modello 3D con rotazione e sovrapposizione introduce in più rispetto al precedente le seguenti variabili continue positive:

- la variabile continua x_i con $i \in I$ individua la coordinata sull'asse x del vertice in basso a sinistra dell'oggetto i;
- la variabile continua y_i con $i \in I$ individua la coordinata sull'asse y del vertice in basso a sinistra dell'oggetto i;
- la variabile continua z_i con $i \in I$ individua la coordinata sull'asse z del vertice in basso a sinistra dell'oggetto i;
- la variabile continua D rappresenta i metri lineari rispetto la profondità e va minimizzata.
- la variabile continua Ω_i con $i \in I$ assume il valore d_i se l'oggetto è stato ruotato altrimenti w_i ;
- la variabile continua Δ_i con $i \in I$ assume il valore w_i se l'oggetto è stato ruotato altrimenti d_i ;

Introduce in più rispetto al precedente anche le seguenti variabili binarie:

- la variabile binaria l_{ij} con $i, j \in I$ assume il valore 1 se l'oggetto i è situato alla sinistra dell'oggetto j, altrimenti è 0;
- la variabile binaria b_{ij} con $i, j \in I$ assume il valore 1 se l'oggetto i è situato al di sotto dell'oggetto j, altrimenti è 0.
- la variabile binaria r_i con $i \in I$ assume il valore 1 se l'oggetto i è ruotato su se stesso di 90° , ne risulta che w_i e d_i sono invertiti, altrimenti è 0;
- la variabile binaria t_{ij} assume il valore 1 se l'oggetto i ha sopra di sé l'oggetto j, altrimenti è 0;
- la variabile binaria f_{ij} assume il valore 1 se l'oggetto i ha sopra di sé l'oggetto j e la faccia superiore dell'oggetto i è a contatto con la faccia inferiore dell'oggetto j, altrimenti è 0;
- la variabile binaria k_{ij} con $i, j \in I$ assume il valore 1 se l'oggetto i si trova alla sinistra o/e dietro un oggetto j oppure al di sotto di esso.

Inoltre la variabile binaria k_{ij} viene utilizzata per imporre che un oggetto possa trovarsi alla sinistra o dietro ad un altro oggetto ma non al di sotto di un altro e viceversa, imponendo ciò che segue:

$$\min \quad D \quad (4.43)$$

$$\text{s.t.} \quad k_{ij} \leq l_{ij} + l_{ji} + b_{ij} + b_{ji} \quad i < j \quad i, j \in I \quad (4.44)$$

$$2k_{ij} \geq l_{ij} + l_{ji} + b_{ij} + b_{ji} \quad i < j \quad i, j \in I \quad (4.45)$$

$$1 - k_{ij} = t_{ij} + t_{ji} \quad i < j \quad i, j \in I \quad (4.46)$$

$$x_i + \Omega_i \leq W \quad i, j \in I \quad (4.47)$$

$$y_i + \Delta_i \leq D \quad i, j \in I \quad (4.48)$$

$$z_i + h_i \leq H \quad i \in I \quad (4.49)$$

$$y_i - y_j + M_d b_{ij} \leq M_d - \Delta_i \quad i, j \in I \quad (4.50)$$

$$x_i - x_j + M_w l_{ij} \leq M_w - \Omega_i \quad i, j \in I \quad (4.51)$$

$$z_i - z_j + M_h t_{ij} \leq M_h - h_i \quad i, j \in I \quad (4.52)$$

$$z_i - z_j + M_h f_{ij} \geq -M_h - h_i \quad i, j \in I \quad (4.53)$$

$$x_i - x_j \leq M_w(1 - f_{ij}) \quad i, j \in I \quad (4.54)$$

$$y_i - y_j \leq M_d(1 - f_{ij}) \quad i, j \in I \quad (4.55)$$

$$x_i - x_j + \Omega_i - \Omega_j \geq -M_w(1 - f_{ij}) \quad i, j \in I \quad (4.56)$$

$$y_i - y_j + \Delta_i - \Delta_j \geq -M_d(1 - f_{ij}) \quad i, j \in I \quad (4.57)$$

$$f_{ij} \leq t_{ij} \quad i \in I \quad (4.58)$$

$$M_h \sum_{i \in I} f_{ij} \geq z_j \quad i \in I \quad (4.59)$$

$$\Delta_i = d_i(1 - r_i) - w_i r_i \quad i, j \in I \quad (4.60)$$

$$\Omega_i = w_i(1 - r_i) - d_i r_i \quad i, j \in I \quad (4.61)$$

$$b_{ij}, l_{ij}, t_{ij}, f_{ij}, k_{ij}, r_i \in \{0, 1\} \quad i \neq j \quad i, j \in I \quad (4.62)$$

$$x_i, y_i, z_i, w_i, d_i, \Delta_i, \Omega_i \in \mathbb{R}^+ \quad i \in I \quad (4.63)$$

Spieghiamo ora il significato di ciascun vincolo:

- i vincoli (4.44), (4.45) e (4.46) impongono che presi due oggetti uno sia dietro e/o alla sinistra dell'altro oppure sia sotto di esso;
- il vincolo (4.47) impone che dato il valore x_i , corrispondente alla coordinata x dell'angolo sinistro più arretrato dell'oggetto, sommandoci la larghezza w_i dell'oggetto e ottenendo quindi la coordinata x dell'angolo destro più arretrato, questa sia contenuta nell'intervallo $[0, W]$;
- il vincolo (4.48) impone che dato il valore y_i , corrispondente alla coordinata y dell'angolo sinistro più arretrato dell'oggetto, sommandoci la profondità d_i dell'oggetto e ottenendo quindi la coordinata y dell'angolo sinistro più avanzato, questa sia contenuta nell'intervallo $[0, D]$;
- il vincolo (4.49) impone che dato il valore z_i , corrispondente alla coordinata z dell'angolo sinistro più arretrato dell'oggetto, sommandoci l'altezza h_i dell'oggetto e ottenendo quindi la coordinata z dell'angolo sinistro più in alto, questa sia contenuta nell'intervallo $[0, H]$;
- il vincolo (4.50) dice che se $b_{ij} = 1$ allora impone che l'oggetto i si trovi dietro l'oggetto j, altrimenti il vincolo viene disattivato;
- il vincolo (4.51) dice che se $l_{ij} = 1$ allora impone che l'oggetto i si trovi alla sinistra l'oggetto j, altrimenti il vincolo viene disattivato;
- il vincolo (4.52) dice che se $t_{ij} = 1$ allora impone che l'oggetto i si trovi al di sotto dell'oggetto j, altrimenti il vincolo viene disattivato;
- il vincolo (4.53) dice che se $f_{ij} = 1$ allora impone che l'oggetto i si trovi al di sotto dell'oggetto j e a contatto con lo stesso, altrimenti il vincolo viene disattivato;
- i vincoli (4.54) e (4.55) impongono che se un oggetto i è sotto e a contatto con un oggetto j, allora la coordinata dell'angolo più vicino all'origine dovrà trovarsi all'interno del piano individuato dalla base superiore dell'oggetto i;
- i vincoli (4.56) e (4.57) impongono che se un oggetto i è sotto e a contatto con un oggetto j, allora la coordinata dell'angolo più lontano all'origine dovrà trovarsi all'interno del piano individuato dalla base superiore dell'oggetto i;
- il vincolo (4.58) impone che un oggetto i possa essere al di sotto e a contatto con un oggetto j solo se l'oggetto i è al di sotto dell'oggetto j;

- il vincolo (4.59) impone che tutti gli oggetti i che non abbiano al di sotto di sé nessun oggetto debbano avere coordinata $z_i = 0$
- il vincolo (4.60) impone che Δ_i corrisponda alla profondità corretta considerando la rotazione;
- il vincolo (4.61) impone che Ω_i corrisponda alla larghezza corretta considerando la rotazione;

5

Test e validazione

Una attività di fondamentale importanza è stata quella di testing dei modelli.

5.1 Istanze

Diamo prima una definizione di istanza:

Una **istanza** è un insieme formato dai pacchi da disporre nel contenitore.

Mentre invece la definizione di gruppo di istanze:

Un **gruppo di istanze** è un insieme di istanze accomunate tra loro dalle simili dimensioni o dal numero di oggetti.

La prassi generale nell'eseguire il testing dei modelli è stata quella di creare un centinaio di istanze casualmente, eseguire il modello sulle istanze generate e verificare la correttezza delle soluzioni fornite, con l'obiettivo di individuare imperfezioni o testare la tenuta dei vincoli, la verifica della correttezza delle soluzioni veniva fatto attraverso funzioni realizzate appositamente che verranno illustrate nelle sezioni successive. Questa procedura richiedeva circa una ventina di minuti di esecuzione ed è stata utilizzata solo dopo modifiche importanti che da un punto di vista teorico erano corrette, inoltre venivano eseguite su istanze con pochi oggetti per risparmiare tempo e individuare subito eventuali errori.

Dopo la realizzazione di tutti i modelli, sicuri delle soluzioni fornite grazie alla validazione effettuata, si è passati all'esecuzione di 100 test per ciascuna delle istanze riportate di seguito per ottenere maggiori informazioni in merito, il tutto eseguito per ciascun modello.

	width_a	width_b	depth_a	depth_b
0	0.5	2.45	0.5	2.45
1	0.5	1.50	0.5	4.00
2	1.5	2.45	0.5	4.00
3	0.5	1.50	3.0	4.00
4	1.5	2.45	3.0	4.00
5	0.1	1.00	0.1	1.00
6	0.1	1.00	3.0	4.00
7	2.0	2.45	3.0	4.00
8	2.0	2.45	2.0	2.45
9	0.1	1.00	0.1	4.00

Table 5.1: Le dimensioni dei gruppi di istanze eseguite

La tabella riporta gli intervalli entro cui si è deciso casualmente le dimensioni dei pacchi, ogni gruppo di istanze aveva dimensioni particolari entro cui dovevano trovarsi i pacchi:

- **Larghezza:** [width_a, width_b]
- **Profondità:** [depth_a, depth_b]

5.2 Numero di oggetti

Fin dal modello 2D abbiamo capito che il numero di pacchi per ogni istanza sarebbe dovuto essere limitato in quanto già con 7 pacchi i tempi di esecuzione aumentavano vertiginosamente, abbiamo quindi deciso di scegliere in modo casuale in un intervallo [3,10] il numero di pacchi da assegnare a ciascuna istanza, imponendo un *time limit*^[9] di 300 secondi, questo portava ad ottenere soluzioni

ottime e soluzioni *best bound*^[9], l'analisi dei risultati sarà divisa in queste due categorie infatti.

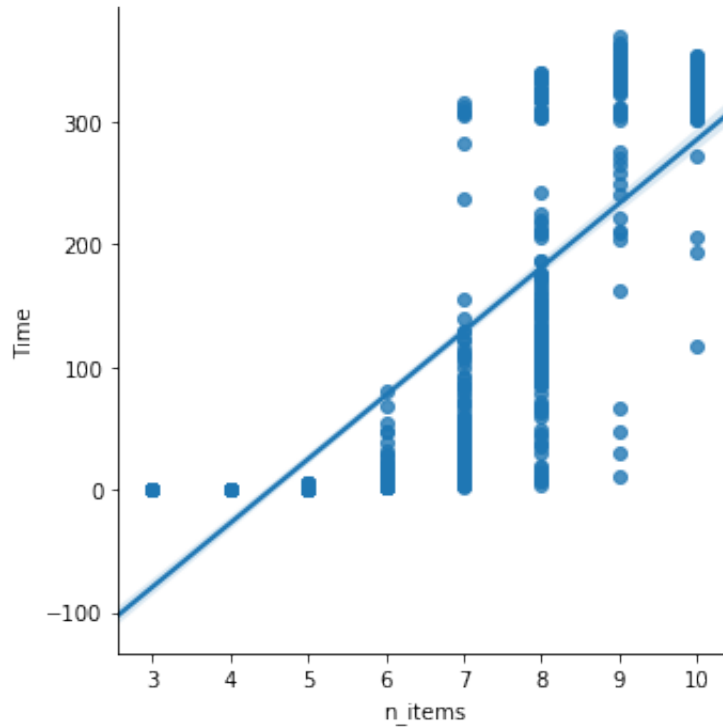


Figure 5.1: Grafico tempi esecuzione modello con diversi oggetti

Nel grafico sopra riportato (5.1) viene riportato come ordinata il numero di oggetti e sull'ascissa il tempo espresso in secondi, come si può vedere man mano che gli oggetti aumentano il tempo aumenta notevolmente passando da pochi secondi a minuti, risultano molto interessanti quelle istanze che nonostante abbiano molti oggetti richiedano meno tempo per essere eseguiti, forse questo dovuto alle particolari dimensioni degli oggetti.

5.3 Validazione soluzioni fornite

Di fondamentale importanza per il confronto è stato che le soluzioni fornite dal modello fossero corrette, questo ha richiesto lo sviluppo di alcune funzione che

controllassero la validità delle stesse. Per fare questo sono state realizzate le seguenti funzioni:

- **CheckFeasible:** utilizzata per controllare che non vi fossero pacchi che si intersecassero con altri;
- **CheckSequence:** realizzata in due versioni:
 - **2D:** considerando che tutti gli oggetti fossero alla stessa altezza controllava che ciascun pacco avesse almeno una via attraverso cui essere scaricato.
 - **3D:** considerava anche la sovrapposizione tra pacchi e oltre a controllare di avere almeno una delle tre direzioni libere rispetto l'altezza a cui si trovava controllava anche se sopra ciascun pacco al momento dello scarico non vi fossero altri pacchi.
- **CheckStable:** utilizzata per controllare la stabilità degli oggetti, questo verificava che l'area di base inferiore poggiasse completamente sulle basi superiori degli oggetti sottostanti;
- **ProjectionCommonArea:** utilizzata per verificare se due parallelepipedi avessero intersezioni tra le loro proiezioni rispetto a 2 assi.

Inoltre per natura stessa dei modelli essi ricercano la soluzione ottima, quindi si può dire che se le soluzioni trovate vengono individuate entro il tempo limite e vengono verificate dalle funzioni precedentemente riportate allora queste possono dirsi effettivamente ottime e reali, perfette candidate per un confronto con la soluzione corrispondente fornita dal modello.

6

Tecnologie e strumenti

Questo capitolo riporta le tecnologie e gli strumenti utilizzati durante il corso dello stage.

6.1 Tecnologie

6.1.1 Python



Figure 6.1: Logo Python

Python [3] è un linguaggio di programmazione tra i più famosi attualmente disponibili, è un linguaggio ad oggetto, sintatticamente semplice e intuitivo che permette di avere un approccio ai problemi molto più operativo. Python [3] è stata la base tecnologica su cui si è basato lo stage e gli strumenti utilizzati. Oltre ad aver usato questo linguaggio abbiamo utilizzato alcune librerie fornite dalla libreria standard.

6.1.2 Google Or-Tools

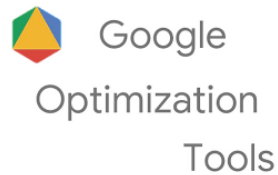


Figure 6.2: Logo Or-Tools

Google Or-Tools [1] è una libreria open source sviluppata da Google che permette di risolvere problemi di ottimizzazione potendo scegliere se utilizzare solver commerciali come CPLEX [16] o Gurobi [17] oppure open source come SCIP [18] o Cbc [14]. Inoltre viene fornita la libreria per diversi linguaggi come:

- C++
- Java
- Python
- C#

Nel nostro caso dato si è deciso di utilizzare la libreria per il linguaggio Python [3], utilizzandola per realizzare dei modelli di programmazione lineare intera. La libreria è stata imposta dall'azienda dopo il primo periodo di studio degli strumenti e delle tecnologie.

6.1.3 Cbc



Figure 6.3: Logo Cbc

Cbc [14] è un solver di programmazione lineare intera scritto in C++ [10], il motivo per cui si è scelto questo è la compatibilità con la libreria Google Or-Tools e la sua natura open source.

6.1.4 Boost



Figure 6.4: Logo Boost

Boost [9] è un insieme di librerie C++ [10] open source che permettono di aumentare le capacità applicative del linguaggio. Viene utilizzato per progetti open source e commerciali. Il suo utilizzo si è reso necessario in quanto le librerie fornite dall'azienda dipendevano da esso, in particolare grazie alla capacità di esporre classi C++ [10] al linguaggio Python [3].

6.1.5 Pandas



Figure 6.5: Logo Pandas

Pandas [11] è una libreria open source utilizzata con il linguaggio Python [3], essa è fondamentale per eseguire analisi dei dati, inoltre offre delle funzionalità per la lettura dei dati da file `xlsx`, `csv`, `txt` e la manipolazione degli stessi. Inoltre fornisce dei comodi metodi di scrittura su file anche verso \LaTeX , libreria consigliata dall'azienda.

6.1.6 Matplotlib



Figure 6.6: Logo Matplotlib

La libreria Matplotlib [19] è stata utilizzata durante lo stage per la realizzazione dei grafici e su di essa si basano le immagini 2D e 3D dei pacchi precedentemente mostrate, è stato fondamentale per poter eseguire una verifica qualitativa delle soluzioni. Inoltre è stata utilizzata per realizzare grafici importanti grazie ai quali spiegare in modo efficace alcuni concetti.

6.1.7 Multiprocessing



Figure 6.7: Logo Multiprocessing

La libreria Multiprocessing [20] è stata utilizzata per seguire un numero maggiore di test in modo concorrente superando gli impedimenti del *GIL*^[9]. Consigliata dall'azienda in quanto i tempi di esecuzione dei test non utilizzando la concorrenza si attestavano alle 10 ore circa, con la suddetta libreria ci attestiamo circa sulle 2 ore.

6.2 Strumenti

6.2.1 Dropbox



Figure 6.8: Logo Dropbox

Dropbox [2] è un servizio di archiviazione condivisa in cloud molto famoso e utilizzato nel mondo, permette di condividere file e documenti di vario genere con altri collaboratori, utile per la documentazione. Durante lo stage è stato utilizzato per condividere il diario e le presentazioni con i colleghi.

6.2.2 GitHub



Figure 6.9: Logo GitHub

GitHub [7] è un servizio di versionamento basato su repository git [8]. Questo servizio permette di creare repository pubbliche e private con cui versionare il proprio progetto, aspetto improntante nello sviluppo software.

6.2.3 Visual Studio Code



Figure 6.10: Logo Visual Studio Code

Visual Studio Code [21] è un editor open source che grazie alle estensioni installabili permette di utilizzare lo stesso editor per moltissimi linguaggi di programmazione. Inoltre grazie all'estensione per git [8] è possibile versionare direttamente dall'editor, le estensioni permettono anche di eseguire una verifica della sintassi del codice su cui si lavora a livello statico in modo da individuare eventuali errori ed è integrato in esso il terminale con cui lanciare gli script Python [3].

6.2.4 Jupyter Notebook



Figure 6.11: Logo Jupyter Notebook

Jupyter notebook [4] è applicazione web open source che permette di creare e condividere documenti che condividono del codice. Supporta molti linguaggi di programmazione tra i quali Python [3] e permette una programmazione interattiva generalizzata dal tool IPython. Fondamentale per semplicità di utilizzo e per la possibilità di esportare gli script in molti formati.

6.2.5 PIP



Figure 6.12: Logo Pip

PIP [22] è un gestore di pacchetti Python [3] fondamentale per l'installazione di moduli non presenti nella libreria standard, di facile utilizzo ha permesso di installare velocemente moduli indispensabili per il proseguimento dello stage.

6.2.6 Taiga



Figure 6.13: Logo Taiga

Taiga [5] è una piattaforma di project management utilizzata durante il progetto per la gestione delle task da soddisfare in un certo periodo di tempo.

7

Risultati

Nel seguente capitolo riporteremo i risultati ottenuti dai confronti tra le soluzioni fornite dai modelli e le corrispondenti fornite dall'euristica.

Per ciascun modello verranno riportate due tabelle con i risultati ottenuti, queste verranno divise nelle due categorie di soluzioni ottenute:

- ottime;
- best bound.

Le tabelle riporteranno il numero di istanze per gruppo, errore relativo ϵ_r nel rapporto tra i metri lineari ottenute, errore assoluto ϵ_a espresso in metri e il tempo medio di esecuzione di quel gruppo di istanze. Successivamente verranno riportate alcune osservazioni sui risultati di alcuni gruppi di istanze e un rapporto generale sulla bontà delle soluzioni dei modelli aldilà dei gruppi di istanze, ricordiamo che sia il modello che l'euristica venivano eseguite sulla stessa istanze e con le stesse caratteristiche, questo permette un confronto oggettivo.

7.1 Errori

Sia modello che euristica restituiscono il valore minimizzato dalla funzione obiettivo ossia i metri lineari occupati. La nomenclatura sarà la seguente:

- Metri lineare modello: Obj_m
- Metri lineare euristica: Obj_h

Gli errori che verranno riportati successivamente vengono calcolati come segue:

$$\epsilon_a = Obj_h - Obj_m$$
$$\epsilon_r = \frac{100(Obj_h - Obj_m)}{Obj_m}$$

7.2 Risultati modello 2DR

Di seguito verranno riportati i risultati ottenuti dal confronto tra soluzioni ottime fornite da euristica e modello 2DR:

	#ist	ϵ_r	ϵ_a	Time		#ist	ϵ_r	ϵ_a	Time
0	64.0	3.89	0.23	40.95	0	36.0	7.35	0.59	318.86
1	73.0	11.90	0.81	31.51	1	27.0	15.98	1.48	314.78
2	76.0	0.94	0.10	19.76	2	24.0	0.91	0.17	316.74
3	84.0	12.29	1.26	19.79	3	16.0	17.26	2.62	308.85
4	75.0	0.00	0.00	27.69	4	25.0	0.00	0.00	316.80
5	73.0	14.17	0.11	12.58	5	27.0	16.16	0.21	311.99
6	78.0	6.60	0.47	20.95	6	22.0	19.40	1.70	326.03
7	76.0	0.00	0.00	36.62	7	24.0	0.00	0.00	326.78
8	81.0	0.00	0.00	23.70	8	19.0	0.00	0.00	319.78
9	81.0	10.34	0.45	10.60	9	19.0	20.58	1.10	307.55

Table 7.1: Risultati 2DR ottimi

Table 7.2: Risultati 2DR best bound

Le prime osservazioni che mi sento di dare sono sul perché gruppi di istanze come il 4,7,8 riportino errori ϵ_r e ϵ_a uguali a 0, perché mediamente i pacchi di tali gruppi di istanze avevano dimensioni che non permettevano di essere affiancati tra loro, per questo ogni istanza il modello ed euristica si sono ridotti ad allinearli uno davanti l'altro portando ed ottenendo sempre lo stesso risultato. Sembra interessante anche il gruppo di istanze 9 che riporta mediamente un tempo di esecuzione minore, questo gruppo di istanze è un formata da mix di pacchi stretti e corti e stretti e lunghi, questo porta gli errori ϵ_r per quanto riguarda le soluzioni ottime ad un valore del 10%, mentre invece con le soluzioni best bound questo aumenta fino al 20% aumentando così considerevolmente, altra considerazione interessante è come ci si aspetterebbe di veder diminuire gli errori con le soluzioni best bound, questo non accade ed anzi aumenti il distacco dalle soluzioni fornite dall'euristica, segno che le soluzione best bound fornite dal modello sono comunque soluzione valide per il confronto.

7.3 Risultati modello 2DRS

Di seguito verranno riportati i risultati ottenuti dal confronto tra soluzioni ottime fornite da euristica e modello 2DRS:

	#ist	ϵ_r	ϵ_a	Time
0	76.0	4.74	0.26	41.72
1	75.0	12.44	0.83	27.96
2	82.0	0.56	0.07	20.97
3	72.0	14.59	1.62	29.67
4	81.0	0.00	0.00	28.00
5	75.0	14.87	0.12	17.89
6	78.0	9.18	0.68	17.22
7	85.0	0.00	0.00	13.99
8	79.0	0.00	0.00	25.25
9	89.0	7.69	0.32	12.87

Table 7.3: Risultati 2DRS ottimi

	#ist	ϵ_r	ϵ_a
0	24.0	5.84	0.52
1	25.0	14.26	1.29
2	18.0	0.99	0.18
3	28.0	15.82	2.40
4	19.0	0.00	0.00
5	25.0	16.20	0.20
6	22.0	20.83	1.97
7	15.0	0.00	0.00
8	21.0	0.00	0.00
9	11.0	23.35	1.06

Table 7.4: Risultati 2DRS best bound

Le prime osservazioni da riportare è che l'euristica nel disporre i pacchi rispettando la corretta sequenza di ordine, li inserisce ordinando i pacchi in base all'ordine di scarico così da essere sicuro che ciascun pacco non abbia davanti a se pacchi di ordini successivi, questo però preclude la possibilità di avere quelle soluzioni, questo però porta ad avere anche soluzioni in cui però non sia garantita la stabilità generale degli oggetti. Possiamo osservare come un aumento della complessità del problema non abbia portato però ad un aumento degli errori generalizzato ma come in alcuni gruppi di istanze sia diminuito mentre i tempi d'esecuzione siano aumentati generalmente. Le istanze con gli errori maggiori restano sempre le stesse del precedente modello, in particolare salta all'occhio come l'errore relativo del gruppo di istanze 3 si attesti al 20%, questo gruppo è formato da pacchi stretti e lunghi, valore importante che porta ad uno scarto tra modello di quasi 3 metri.

7.4 Risultati modello 3D

Di seguito verranno riportati i risultati ottenuti dal confronto tra soluzioni ottime fornite da euristica e modello 3D:

	#ist	ϵ_r	ϵ_a	Time
0	76.0	5.05	0.25	39.15
1	77.0	11.11	0.67	31.87
2	84.0	0.84	0.10	33.15
3	72.0	9.83	0.96	34.07
4	79.0	0.00	0.00	36.59
5	65.0	9.90	0.08	13.53
6	76.0	6.33	0.45	26.50
7	76.0	0.00	0.00	39.09
8	78.0	0.00	0.00	25.45
9	76.0	10.73	0.37	20.24

Table 7.5: Risultati 3D ottimi

	#ist	ϵ_r	ϵ_a
0	24.0	5.24	0.49
1	23.0	12.87	1.22
2	16.0	0.80	0.11
3	28.0	21.01	2.95
4	21.0	0.00	0.00
5	35.0	14.47	0.17
6	24.0	16.85	1.55
7	24.0	0.00	0.00
8	22.0	0.00	0.00
9	24.0	13.96	0.59

Table 7.6: Risultati 3D best bound

Le istanze che sono state generate non contenevano solo oggetti *stackable*^[9], ma anche oggetti *fragili* che sopra di sé non potevano avere nessun altro, quindi nell'ordinare gli oggetti l'euristica inseriva nel container prima gli oggetti *fragili* comportando così che sopra di essi non potesse esservi altri oggetti. Dobbiamo Gli errori in generale comunque si sono mantenuti simili a quelli precedentemente riportati, nonostante con il modello in questione fosse possibile avere sotto di sé solo un oggetto mentre per l'euristica fosse possibile avere più di un oggetto.

8

Conclusione

Nel seguente capitolo verranno riportate le conclusioni che si è potuto trarre alla fine di questo stage.

8.1 Consuntivo finale

Prima di iniziare lo stage era stato concordato insieme con l'azienda la pianificazione dello stesso, definendo obiettivi da raggiungere e calendarizzandoli sulle 320 ore disponibili al fine di poterli soddisfare appieno. La pianificazione iniziale è stata rispettata nella sua visione generale andando a modificarne invece le scadenze orarie, la formazione è stata più breve del previsto visto che la scelta del software di modellazione algebrica è ricaduto sullo stesso Google Or-Tools [1], per la realizzazione dei modelli in generale la prototipazione e la scrittura del modello in Google Or-Tools [1] è risultata meno dispendiosa del previsto, questo ha permesso di realizzare un modello non preventivato su consiglio dell'azienda e del tutor esterno. Si è deciso di eseguire i test di confronto con l'euristica nella parte finale dello stage invece che durante la loro realizzazione, questo perché l'intero sistema usato per testare i modelli fosse unico, concentrandosi di più nell'esecuzione di test di validazione che permettessero di verificare le soluzioni fornite dai modelli.

Durata in ore		Descrizione dell'attività
32		A: Formazione
	8	• Ricerca <i>framework</i> di modellazione algebrica
	24	• Studio Google OR - Tools
64		B: Versione algoritmo 2D
	8	• Individuazione ed analisi <i>constraints</i>
	8	• Prototipazione modello
	24	• Traduzione in Python - Google OR - Tools
	24	• Test validazione
64		C: Versione algoritmo 2DR
	8	• Individuazione ed analisi <i>constraints</i>
	8	• Prototipazione modello
	24	• Traduzione in Python - Google OR - Tools
	24	• Test validazione
72		C: Versione algoritmo 2DRS
	8	• Individuazione ed analisi <i>constraints</i>
	16	• Prototipazione modello
	16	• Traduzione in Python - Google OR - Tools
	32	• Test validazione
72		D: Versione algoritmo 3D
	8	• Individuazione ed analisi <i>constraints</i>
	16	• Prototipazione modello
	16	• Traduzione in Python - Google OR - Tools
	32	• Test validazione
16		E: Test confronto modello ed euristica
	4	• Confronto modello 2D-euristica
	4	• Confronto modello 2DR-euristica
	4	• Confronto modello 2DRS-euristica
	4	• Confronto modello 3D-euristica
Totale: 320		

Table 8.1: Ripartizione reale delle ore di stage

8.2 Raggiungimento degli obiettivi

Gli obiettivi decisi prima dell'inizio dello stage e riportati nel capitolo 2, questi prevedevano il soddisfacimento di 8 obiettivi obbligatori, 4 obiettivi desiderabili e 1 opzionale, sono stati tutti soddisfatti appieno e anzi è stato realizzato un modello non preventivato.

L'obiettivo opzionale qui di seguito:

op01: evoluzione euristica, fornita dall'azienda, con nuove funzionalità

Non è stato inserito nella pianificazione in quanto si è svolto nel mentre che si realizzavano i modelli, le nuove funzionalità sono le funzioni di verifica con cui validare le soluzioni, riportate nel capitolo 5.

8.3 Conoscenze acquisite

Le conoscenze acquisite durante il corso dello stage sono state le seguenti:

- Linguaggio Python [3];
- Framework Google Or-Tools [1];
- Jupyter notebook [4];
- Librerie Matplotlib [19];
- Librerie Pandas [11];
- Modulo Multiprocessing [20];
- Package management attraverso PIP [22];

8.4 Valutazione personale

La valutazione dello stage svolto presso Trans-Cel non può che dirsi positiva, ho trovato un ambiente accogliente e dei colleghi che sono poi diventati amici. Tecnicamente parlando affrontare un problema così complesso è stato formativo,

poiché ho imparato ad usare strumenti utili ed un linguaggio molto popolare ed usato al giorno d'oggi, anche i lunghi confronti con il tutor interno sono stati utili per capire le meccaniche dell'euristica. Lo stage è stato la conclusione di un percorso di tre anni molto intensi che credo si possa dire abbia coronato gli sforzi fatti sia dal punto di vista tecnico sia umano.

Glossario

- **Agile:** modello di ciclo di vita nato per sopperire alla rigidità dei modelli precedenti, caratterizzato da rilasci rapidi e incrementali, usato per rispondere velocemente alle richieste dei clienti in termini di nuovi requisiti;
- **Best Bound:** è una soluzione fornita da un modello dopo che si è raggiunto il tempo limite di esecuzione, questa soluzione non è ottima ma può essere considerata come un buon risultato;
- **Brainstorming:** incontri di gruppo creativi utili per risolvere problemi o individuare nuove idee. Servono più di due partecipanti in modo che vi sia una discussione arbitraria che è utile;
- **Ciclo Di Vita:** insieme degli stati che il prodotto assume dal concepimento al ritiro;
- **Commerciale:** prodotto realizzato allo scopo di trarne un profitto o che è adatto a scopi commerciali;
- **Euristica:** algoritmo progettato per risolvere un problema più velocemente, spesso una strada obbligata per risolvere problemi molto difficili;
- **GIL:** meccanismo usato dagli interpreti dei linguaggi di programmazione per sincronizzare i thread in modo che ve ne sia in esecuzione in coda;
- **Incremento:** particolare tipo di iterazione dove il successo è garantito;
- **Modello Incrementale:** modello di ciclo di vita che prevede rilasci multipli e successivi, dove ciascuno di essi realizza un incremento di funzionalità. I requisiti vengono trattati per importanza, prima quelli di maggior importanza in modo che possano stabilizzarsi con il rilascio delle versioni fino a quelli minori;
- **Milestone:** punto nel tempo al quale associamo un insieme di stati di avanzamento;
- **Open Source:** termine utilizzato per riferirsi ad un software di cui i detentori dei diritti sullo stesso ne rendono pubblico il codice sorgente;

- **Project Management:** gestione delle attività di analisi, progettazione, pianificazione e realizzazione degli obiettivi di un progetto, compito svolto dal project manager di un'azienda attraverso anche strumenti idonei;
- **Slack:** tempo aggiuntivo ad un'attività che ha lo scopo di evitare ritardi nella produzione del prodotto;
- **Soluzioni:** termine usato per indicare l'insieme di contenitore e pacchi da disporre con le relative posizioni rispetto il contenitore;
- **Solver:** software commerciale o open source che permette di risolvere problemi di programmazione lineare;
- **Stackable:** termine utilizzato per indicare se un pacco può avere sopra di sé altri pacchi;
- **Time Limit:** termine usato per indicare un tempo limite entro il quale può essere ricercata una soluzione dal solver;
- **Vehicle Routing:** famiglia di problemi che trattano tutti gli aspetti della gestione di una flotta di veicoli nell'ambito della logistica.

Lista degli acronimi

2D	Modello 2D
2DR	Modello 2D con rotazione
2DRS	Modello 2D con rotazione e sequenza di scarico
3D	Modello 3D
CBC	Coin-or branch and cut
GIL	Global Interpreter Lock

Bibliografia

- [1] Google or-tools. [Online]. Available: <https://developers.google.com/optimization/>
- [2] Dropbox. [Online]. Available: <https://dropbox.com/it/>
- [3] Python. [Online]. Available: <https://www.python.org/>
- [4] Jupyter notebook. [Online]. Available: <http://jupyter.org/>
- [5] Taiga. [Online]. Available: <https://taiga.io/>
- [6] Telegram. [Online]. Available: <https://telegram.org/>
- [7] Github. [Online]. Available: <https://www.github.com/>
- [8] Git. [Online]. Available: <https://git-scm.com/>
- [9] Boost. [Online]. Available: <https://www.boost.org/>
- [10] C++. [Online]. Available: <http://www.cplusplus.org/>
- [11] Pandas. [Online]. Available: <https://pandas.pydata.org/>
- [12] Stageit. [Online]. Available: <http://informatica.math.unipd.it/laurea/stageit.html>
- [13] M. Ghirardi, A. Grosso, and G. Perboli, *Esercizi svolti di ricerca operativa*. Esculapio, 2017.
- [14] Cbc. [Online]. Available: <https://projects.coin-or.org/Cbc>
- [15] C. Blum and V. Schmid, “Solving the 2d bin packing problem by means of a hybrid evolutionary algorithm. (English),” *International Conference on Computational Science*, 2013.

- [16] Cplex. [Online]. Available: <https://www.ibm.com/analytics/cplex-optimizer>
- [17] Gurobi. [Online]. Available: <http://www.gurobi.com/>
- [18] Scip. [Online]. Available: <https://scip.zib.de/>
- [19] Matplotlib. [Online]. Available: <https://matplotlib.org/>
- [20] Multiprocessing. [Online]. Available: <https://docs.python.org/2/library/multiprocessing.html>
- [21] Visual studio code. [Online]. Available: <https://code.visualstudio.com/>
- [22] Pip. [Online]. Available: <https://pypi.org/project/pip/>