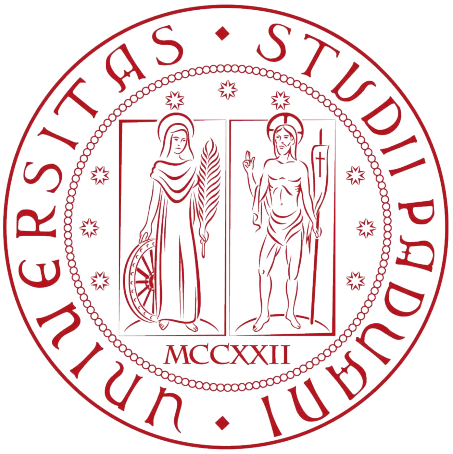
Università degli Studi di Padova

Dipartimento di Matematica "Tullio Levi-Civita"

Corso di Laurea Magistrale in Informatica



Modelli ed algoritmi di ottimizzazione "Data-driven" per la gestione del traffico aereo in Europa

Tesi di laurea magistrale

Relatore

Prof. Luigi De Giovanni

Laureando

Nicola Carlesso
Matricola: 1237782

Anno Accademico 2021-2022

Nicola Carlesso: *Modelli ed algoritmi di ottimizzazione "Data-driven" per la gestione del traffico aereo in Europa,* Tesi di laurea magistrale, Aprile 2022.

Sommario

La gestione del traffico aereo (*Air Traffic Flow Management*, ATFM) chiede di essere organizzata in modo sicuro ed ottimale, affinché lo spazio aereo non venga occupato da un numero di voli superiore a quello consentito dalle autorità e, al tempo stesso, i piani di voli incontrino il più possibile la preferenza delle compagnie aeree. L'approccio al problema considerato in questa tesi trova il suo fondamento nell'utilizzo di modelli matematici basati sui dati storici dei voli, nel rispetto dei requisisti indicati. Tuttavia, l'applicazione dei modelli proposti è difficilmente realizzabile in uno scenario realistico, a causa dell'eccessivo tempo computazionale richiesto. Questa tesi, a partire da studi presentati nella letteratura per la risoluzione del problema ATFM attraverso una serie di algoritmi euristici, ha l'obiettivo di limitare il tempo di computazione, raggiungendo il compromesso di ottenere una soluzione qualitativamente buona in un tempo compatibile con le prassi operative. L'approccio vede la combinazione di tecniche di ottimizzazione combinatoria, programmazione matematica, e *machine learning*.

Ringraziamenti

Desidero esprimere la mia gratitudine al Prof. Luigi De Giovanni, relatore della mia tesi, per il concreto aiuto datomi durante la stesura del lavoro e per avermi offerto costantemente il Suo prezioso sostegno.

Desidero altresì ringraziare la mia famiglia per l'incoraggiante appoggio e la sicura vicinanza negli anni di studio.

Un ringraziamento sentito è rivolto all'azienda in cui attualmente lavoro, per la comprensione e la disponibilità dimostratami durante la stesura della tesi.

Infine, il mio grazie va anche ai miei amici per il tempo vissuto insieme e le molte esperienze condivise.

Padova, Aprile 2022 Nicola Carlesso

Indice

Elenco delle figure

Elenco delle tabelle

Capitolo 1

Introduzione

Negli ultimi anni, i concetti di internazionalizzazione e globalizzazione sono entrati nella nostra quotidianità, favorendo una maggiore consuetudine del viaggio in aereo. Il numero di viaggiatori in aereo è pertanto vertiginosamente aumentato, in particolar modo per i voli low-cost, i cui prezzi sono oramai accessibili ai più.

Prendendo in esame solo l'Europa, e considerando i dati pre-pandemia COVID-19, nel 2019 hanno viaggiato in aereo 1.2 miliardi di persone, circa il 26.8% del traffico mondiale, registrando un incremento del numero di passeggeri del 6.6% rispetto all'anno precedente, come indicato in [31]. Sebbene i dati del biennio 2020-2021 invertano questa tendenza a causa della pandemia da COVID-19, si ritiene che i livelli del traffico aereo possano tornare ai livelli pre-pandemici con la fine dell'emergenza. Un altro interessante dato indica che, di questi voli, il 31% sono low-cost ed evidenzia che l'intensità di traffico aereo, soprattutto nelle zone popolose, è decisamente alta, come dimostra la Figura 1.1.

Il problema della gestione del flusso del traffico aereo, Air Traffic Flow Management (ATFM), assume dunque notevole importanza, anche in considerazione della possibilità di soddisfare la maggior parte delle richieste di diverse compagnie aeree, in termini di voli da effettuare, considerati i vincoli di capacità dello spazio aereo, secondo i criteri stabiliti dall'ente regolatore International Civil Aviation Organization (ICAO). Il compito delle autorità preposte consiste nel garantire la sicurezza dei voli. Pertanto, esse impongono il numero massimo di aerei che possono sorvolare un determinato spazio aereo in un certo lasso di tempo. Al fine di stabilire l'intero piano giornaliero dei voli, è presente una fase di "contrattazione" tra le compagnie aeree e gli enti regolatori dello spazio aereo: le prime richiedono di effettuare certi voli seguendo il loro business model, mentre le seconde devono assicurare il rispetto dei vincoli di carico degli spazi aerei, dovendo scegliere i voli da pianificare, considerando le richieste delle compagnie. Il problema ATFM ha l'obiettivo di automatizzare parte di questa fase velocizzandola, proponendo la pianificazione di determinati voli in modo tale che questi rispettino i vincoli imposti e massimizzino le preferenze delle compagnie.

L'enorme quantità di voli rende il problema ATFM particolarmente complesso e negli anni ha portato allo sviluppo di numerosi modelli matematici che hanno il compito di assegnare ad ogni volo la miglior traiettoria, al fine di massimizzare le preferenze delle compagnie aeree o di ridurre il ritardo complessivo dei voli. Le rotte utilizzate dai modelli possono essere generate analizzando lo spazio aereo, oppure scelte

Figura 1.1: I voli sul cielo europeo del 29 settembre 2021 [28]

da un insieme prestabilito di rotte.

Molti modelli presenti in letteratura per il problema ATFM, e in particolare quello utilizzato per questa tesi, applicano il secondo metodo. Nello specifico, il modello preso in considerazione ottiene, per ogni volo, un insieme di piani di volo[G], analizzando lo storico dei voli in un determinato periodo. Tali dati vengono elaborati attraverso tecniche di machine learning[G], in modo da selezionare, tra le diverse rotte, un significativo sottoinsieme che vada incontro alle preferenze delle compagnie aeree.

Studi applicati a scenari concreti mostrano che il costo computazionale per la soluzione esatta di questo problema è molto, talvolta troppo, oneroso, a causa dell'alto numero di voli e vincoli.

I metodi euristici[G] rappresentano un'utile alternativa, che non garantisce il raggiungimento della miglior soluzione, ma comporta un minor costo computazionale. Tali metodi sono dunque preferibili nell'applicazione di scenari reali, dato che il rapporto costi-benefici è più alto.

1.1 Contenuti e contributi della tesi

La tesi riprende i contenuti presenti in [14, 15, 19, 20], in particolare la procedura di generazione dei dati necessari per la generazione dei modelli matematici per l'ATFM e delle relative procedure di soluzione, con l'intento di migliorarne le performance.

Nel Capitolo 2 viene descritto il problema ATFM e i suoi elementi al fine di

far comprendere i successivi passaggi contenuti nella tesi. Inoltre, è riportata una panoramica dei modelli matematici per il problema ATFM presenti in letteratura.

Nel Capitolo 3 sono riportati gli strumenti e le tecniche applicati per lo sviluppo della tesi.

Nel Capitolo 4 sono descritti i metodi considerati in questa tesi, come introdotti in [14, 15, 19], per la risoluzione del problema ATFM, e il modello, presente in [2], su cui essi si basano: in particolare, sono descritti dei metodi costruttivi e un'euristica basata su generazione di colonne.

Nel Capitolo 5 è descritto il procedimento di creazione delle istanze, introdotto in [20], che costituiscono l'input per i metodi descritti nel capitolo precedente. Il capitolo descrive i miglioramenti apportati alla procedura del lavoro di tesi. Tali miglioramenti permettono di ridurre significativamente i tempi di esecuzione dell'algoritmo, in particolare nel caso di creazione multipla di istanze.

Nel Capitolo 6 sono riportati i risultati computazionali derivanti dall'implementazione originale dei modelli di programmazione matematica descritti nel Capitolo 2, e dalla calibrazione dei metodi presentati in letteratura, secondo quanto riportato in [10]. Le valutazioni sono fatte sulla base dei tempi ottenuti dalla risoluzione dei modelli in rapporto ai valori delle soluzioni ottenute. I risultati mostrano l'impraticabilità, allo stato attuale, del solver allo stato dell'arte per la soluzione esatta del modello completo, mentre l'adozione del metodo basato su generazione di colonne, se ben calibrato, rappresenta una via percorribile e adatta a contesti realistici.

Nel Capitolo 7 sono riportate le conclusioni del lavoro svolto e le indicazioni per possibili sviluppi futuri.

In Appendice A sono riportati nel dettagli o risultati delle misurazioni descritte al Capitolo 6.

Capitolo 2

Il problema della gestione del flusso del traffico aereo

Nel capitolo viene descritto nel dettaglio il problema ATFM e gli elementi utili alla comprensione della tesi.

2.1 Definizione del problema ATFM

L'agenzia ICAO definisce il problema ATFM come un servizio che ha l'obiettivo di garantire un flusso sicuro dei voli, assicurandosi di utilizzare nella massima misura possibile le capacità dello spazio aereo, compatibilmente con le restrizioni dichiarate dalle autorità competenti [22].

Per rendere effettiva tale definizione, è necessaria la collaborazione tra le compagnie aeree e le diverse unità Air Traffic Control (ATC), operanti nelle diverse regioni, che si avvalgono dell'ausilio di strumenti specifici. La difficoltà è data dalle numerose variabili in gioco, alcune più prevedibili, come il numero di passeggeri in alcuni mesi dell'anno, oppure la presenza di voli a diverse priorità. Altre variabili invece sono imprevedibili, come ad esempio la presenza di maltempo, che può essere causa di ritardi, cancellazioni o dirottamenti.

Per tale motivo, la pianificazione dei voli in una giornata viene suddivisa in tre fasi [25]:

- **Pianificazione strategica:** effettuata mesi prima, prende in considerazione le indicazioni dell'unità ATC per effettuare aggiustamenti sugli orari e sulle traiettorie dei voli, in modo da bilanciare il traffico aereo;
- **Pianificazione pre-tattica:** effettuata il giorno precedente, attua i necessari aggiustamenti alla fase strategica, prendendo in considerazione le informazioni meteorologiche del giorno successivo;
- **Pianificazione tattica (o operativa):** effettuata durante il giorno stabilito, monitora i voli in modo che essi possano rispettare le indicazioni pianificate, e le modifica nell'eventualità l'unità ATC dia nuove informazioni.

In questa tesi viene affrontato il problema ATFM nella fase pre-tattica.

2.2 Descrizione dello spazio aereo

Lo spazio aereo europeo è gestito dall'European Civil Aviation Conference (ECAC), la quale utilizza una base di dati chiamata Demand Data Repository 2 (DDR2), gestita da EUROCONTROL. L'azienda mette a disposizione le informazioni del traffico aereo attraverso il software NEST, organizza lo spazio aereo in maniera gerarchica e definisce i seguenti elementi [2]:

- **Airblock:** elemento base formato da un poligono chiuso in due dimensioni (2D) i cui vertici sono costituiti da coordinate geografiche. L'intero spazio aereo è suddiviso in airblock;
- **Elementary sector:** porzione di spazio in tre dimensioni (3D) definita da airblock e Flight Level (FL). Un Elementary sector è definito dalla tripletta (airblock, FL minimo, FL massimo), dunque ve ne possono essere più sovrapposti sullo stesso airblock;
- **Collapsed sector:** unione di più elementary sector contigui;
- **Airspace:** unione di più settori (elementary o collapsed) per definire una porzione generica di spazio. Esistono più tipi di airspace, anch'essi gerarchici, come ad esempio la Control Area (CTA) che rappresenta un'area omogenea per l'organizzazione e l'attribuzione delle capacità nei settori. Un esempio è presentato in Figura 2.1. Tali aree vengono gestite dall'Area Control Centre (ACC).

Per organizzare il traffico aereo, ogni CTA è strutturata attraverso una *configurazione*, cioè una tassellatura in 3D dello spazio, attraverso l'unione dei settori che le appartengono. Per rendere flessibile il flusso dei voli, durante la giornata, una CTA può adottare diverse configurazioni, tutte definite nell'*opening scheme* che indica per ogni ora del giorno, quale configurazione segue una determinata CTA, come mostrato in Figura 2.2. Dato un orario, i settori indicati nella configurazione attiva vengono definiti "abilitati".

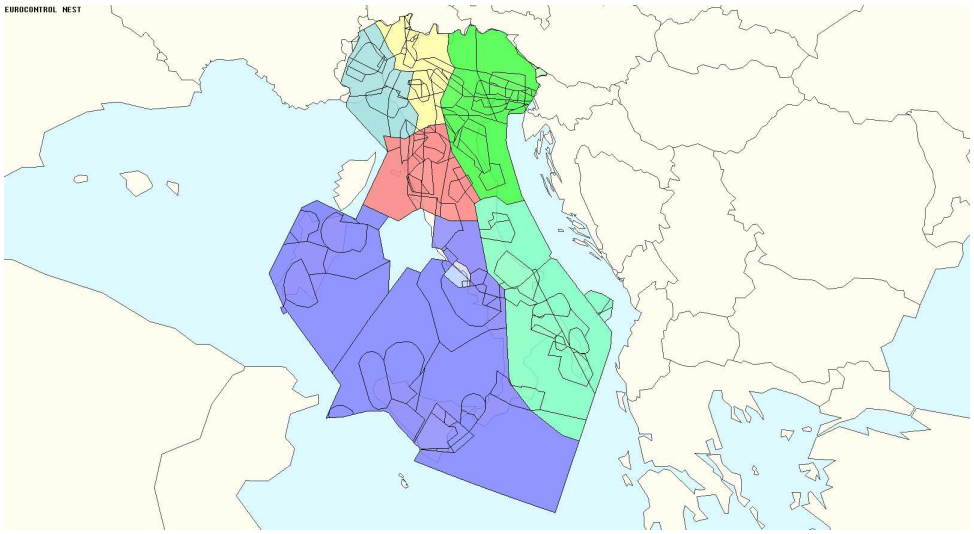


Figura 2.1: CTA presenti in Italia

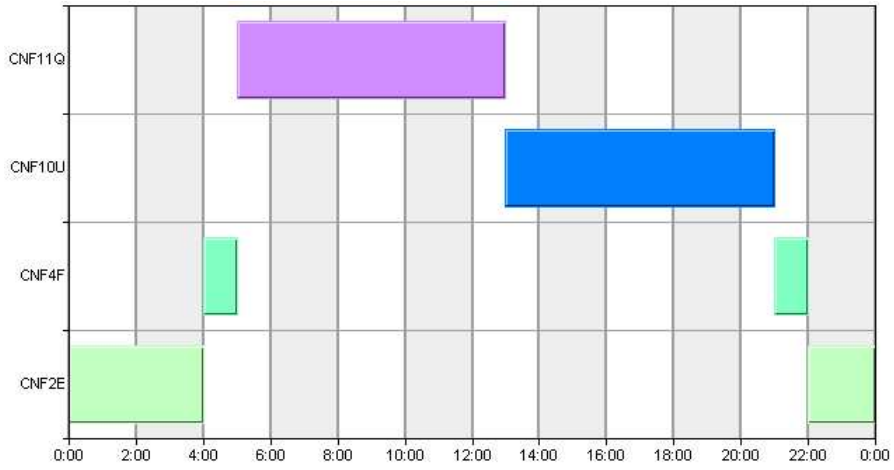


Figura 2.2: Opening Scheme della CTA italiana che comprende Veneto, Friuli, Trentino, parte dell'Emilia e delle Marche

2.3 Descrizione dei piani di volo

Un piano di volo corrisponde a una particolare traiettoria spazio-temporale scelta per uno specifico volo. In fase di pianificazione possono dunque essere proposti più piani di volo per un singolo volo, ed è compito del modello ATFM individuare il piano di volo che massimizzi le preferenze, nel rispetto delle restizioni di capacità degli airspace.

Come per la pianificazione, il piano di volo può essere definito in tre modi, a seconda della fase della pianificazione:

- **Initial:** il piano di volo assegnato durante la pianificazione pre-tattica;
- **Regulated:** il piano di volo approvato alla partenza, durante durante le operazioni tattiche;
- **Actual:** il piano di volo registrato al momento della sua conclusione.

In questa tesi si prendono in esame i piani di volo nella fase *initial*. Inoltre, l'orizzonte temporale di pianificazione viene discretizzato, cioè suddiviso in intervalli di tempo di durata stabilita (ad esempio, 5 minuti).

Un piano di volo può essere descritto in due modi, come mostrato in Figura 2.3:

- **Sequenza di punti geografici:** ogni punto del piano di volo viene descritto attraverso una tupla del tipo (*latitudine*, *longitudine*, *altitudine*, *numero degli intervalli di tempo dalla partenza*);
- **Sequenza di settori attraversati:** dato che i principali vincoli imposti sono la capienza massima dei settori aerei, in questo contesto è utile descrivere un piano di volo in termini di settori attraversati. Nel momento in cui un aereo entra in un settore, alla sua traiettoria viene aggiunta la tripletta (*ID del settore*, *numero degli intervalli di tempo dalla partenza*, *momento in cui l'aereo esce dal settore*).

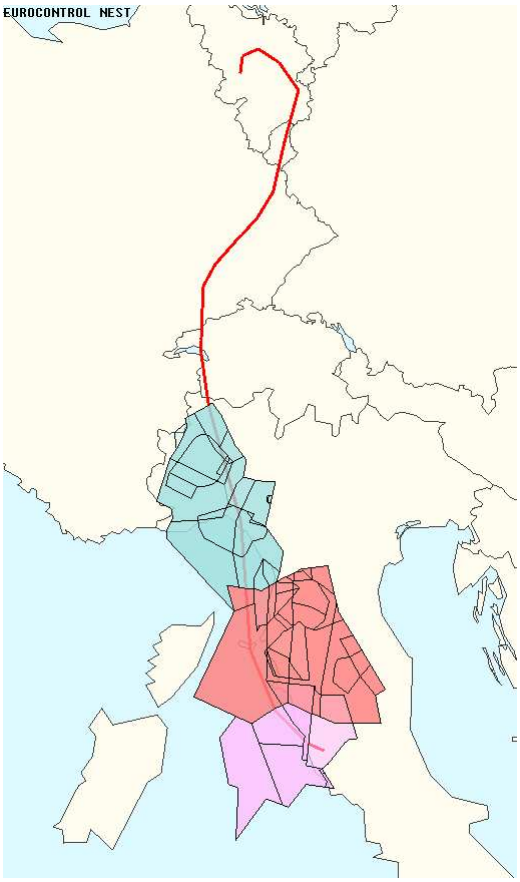


Figura 2.3: Traiettoria di un volo Roma-Bruxelles mostrato come sequenza di settori e di coordinate

2.4 Definizione del problema

Per questo lavoro di tesi viene utilizzato il modello descritto in [2]. Vengono date in input alcune informazioni dei voli da pianificare, come le generalità del volo e le loro possibili traiettorie descritte come sequenza di settori attraversati ed una descrizione dei settori che compongono lo spazio aereo; infine, un grado di preferenza per ogni coppia *volo-traiettoria*.

L'obiettivo principale del problema ATFM è la massimizzazione delle preferenze, assegnando ad ogni volo una traiettoria e rispettando i vincoli di capacità di ogni settore. Altro obiettivo considerato è la minimizzazione del ritardo di ogni volo.

Il problema fa riferimento alla fase di pianificazione *pre-tattica* e, quindi, i piani si riferiscono alla fase *initial*.

Come vedremo, la difficoltà principale per le soluzioni di questo modello è la notevole dimensione in termini di variabili decisionali. L'obiettivo della tesi è la riduzione delle risorse computazionali, in particolare i tempi, necessari per la predisposizione di alcuni modelli matematici proposti dalla letteratura e per la loro soluzione euristica. In particolare, verranno considerate le strategie che prevedono la riduzione del numero di variabili decisionali attraverso la selezione di coppie *volo-traiettoria* da considerare con tecniche euristiche e di machine learning.

2.5 Stato dell'arte

Nel corso degli anni sono stati proposti molti modelli per il problema ATFM, differenziantisi l'un l'altro soprattutto per il tipo di variabili considerate e per il tempo di computazione.

Inizialmente, non veniva considerata la congestione dello spazio aereo, ma l'affollamento degli aeroporti. Dunque i vincoli erano legati alla capacità di questi ultimi e i modelli prevedevano solo di ritardare la partenza di un volo (*ground holding*). Questi modelli, in un primo momento, furono pensati per un aeroporto singolo, Single-Airport Ground-Holding Ploblem (SAGHP) [1] e solo successivamente modificati per gestire il traffico di più aeroporti, Multi-Airport Ground-Holding Ploblem (MAGHP) [23].

In seguito, furono considerate anche le rotte come sequenza di settori aerei attraversati, dato l'incremento del traffico aereo. Questo introdusse una maggiore flessibilità nei modelli, permettendo così la gestione di ulteriori fasi, oltre che del ritardo a terra. Con questa prospettiva, il primo modello matematico realizzato fu quello di Odoni nel 1987 [21]. Il modello introduce delle variabili binarie che indicano l'attraversamento o meno di un volo in un determinato momento.

Successivamente, furono introdotte altri tipi di variabili, che prendevano in considerazione alcune tecniche della fase *pre-tattica*, come il cambiamento di traiettoria (*flight rerouting*), il ritardo nell'atterraggio restando in volo (*airborn holding*) e la modifica della velocità di crociera (*speed control*).

Sfruttando le prime due tecniche (*flight rerouting* e *airborn holding*), Helme nel 1992 propose un modello che affrontava la questione come un problema di flussi di costo minimo (*Multicodity Minimum Cost Flow Problem*) [16].

Sei anni più tardi, nel 1998, Bertsimas e Stock Patterson presentarono un modello

che agiva sulla *speed control*, infatti le variabili riguardavano l'orario di partenza del volo e la velocità di attraversamento dei settori incontrati lungo la traiettoria. Con tale modifica, un volo occupa un settore non più se vi è dentro in un preciso istante di tempo, ma se vi si immette entro quell'istante [4].

Gli stessi autori, nel 2000 proposero un aggiornamento del modello presentato precedentemente in cui introducevano la possibilità di effettuare un cambio di rotta dei voli, per evitare le congestioni di particolari settori aerei. Tale modello è però risultato computazionalmente troppo oneroso, data la grande mole di variabili [5].

Per velocizzare il modello precedente e renderlo più flessibile, nel 2011 Bertsimas, Lulli e Odoni proposero un modello basato sul *flight rerouting*, la *airborn holding*, la *speed control* e il *ground holding*. In questo modo era possibile avere più settori alternativi per il cambio di rotta del volo [3].

Utilizzando le tecniche precedentemente elencate, nel 2017, Fomeni, Lulli e Zografos proposero due modelli basati sul paradigma Trajectory Based Operations (TBO). Quest'ultimo prevede l'utilizzo delle informazioni dei diversi voli e le loro traiettorie definite come sequenza di punti geografici, come descritto in §2.3. Vengono pianificate delle traiettorie ideali in modo tale da minimizzare la differenza con le traiettorie effettive. Tali modelli puntano a trovare un compromesso tra gli enti ACC, che devono garantire la sicurezza dei voli e le performance dell'intero sistema. A partire da questo, viene poi considerata la preferenza delle compagnie aeree [13].

Nel 2020 e 2021 vengono sviluppati alcun studi di metodi euristici per ATFM [14, 19], che si basano sul modello proposto in [2]. Nello specifico, sono presentate una matheuristic[G] e un'euristica di ricerca locale, il cui vicinato è formato da un sottoinsieme di variabili ottenute da un apposito classificatore [14, 15]. Negli stessi anni, il lavoro [19] sviluppa un'euristica basata sulla soluzione esatta del rilassamento continuo degli stessi modelli di generazione di colonne, che permettono di ridurre l'alto numero di variabili presenti nel modello matematico in [2]. Tali lavori saranno oggetto di analisi approfondita nei Capitoli 4 e 6.

Capitolo 3

Tecniche e tecnologie

Nel capitolo vengono presentati le principali tecniche studiate e le tecnologie utilizzate per lo sviluppo di questa tesi.

3.1 Software e linguaggi di programmazione

In questa sezione vengono descritti i linguaggi di programmazione, i software e gli ambienti di sviluppo utilizzati durante il lavoro di tesi.

I diversi tool elencati di seguito sono stati utilizzati nelle macchine denominate *labnum* nel Dipartimento di Matematica dell'Università di Padova. Tali macchine possiedono un processore Intel® Core™ i7-8700 @ 3.20GHz 6 cores e 32 GB di RAM.

3.1.1 Python

Python è un linguaggio di programmazione multi-paradigma, esso è dunque un linguaggio orientato agli oggetti, possiede alcune caratteristiche dei linguaggi funzionali e della programmazione imperativa. La sua caratteristica di essere un linguaggio interpretato e non tipizzato ne facilita molto l'utilizzo e la rapidità di scrittura degli script. Tale proprietà dilata però i tempi di esecuzione e permette la rilevazione di alcune tipologie di errore solo a *run-time* [36].

Questa sua peculiarità ha permesso un fiorente sviluppo di librerie terze e l'utilizzo di questo linguaggio in un ampio numero di contesti, principalmente in ambito scientifico e accademico.

Tra le principali librerie utilizzate, si trova la libreria *NumPy*, la quale fornisce l'utilizzo di array multidimensionali ed una serie di funzioni matematiche ottimizzate per gli oggetti di questa libreria [38]. Esiste poi la libreria *Pandas*, che permette una facile gestione e manipolazione delle basi di dati, di grande aiuto quando si eseguono lavori che richiedono operazioni di machine learning che operano con ingenti quantità di dati [29]. Infine, per il lavoro di questa tesi, è stata utilizzata la libreria *Scikit-learn*, che mette a disposizione le principali tecniche di apprendimento supervisionato e non [30].

Per la sua facilità di apprendimento, il linguaggio python[G] viene utilizzato anche nelle scuole come strumento per l'insegnamento della programmazione. Essendo un linguaggio ad "alto livello", permette di sviluppare il pensiero computazionale senza dover entrare nei tecnicismi del funzionamento di un compilatore [18].

3.1.2 Jupyter Notebook

Jupyter è un'applicazione web open-source per la creazione di notebook, files che possono contenere contemporaneamente codice, risultati della sua esecuzione, parti testuali ed immagini. La sua versatilità permette di mostrare rapidamente i risultati di una computazione, e dunque è possibile effettuare un rapido debugging. Rende possibile inoltre l'esecuzione di singole parti di codice, che all'occorrenza possono essere isolate attraverso le "celle" [35].

Jupyter supporta diversi linguaggi di programmazione grazie l'installazione di specifici *kernel*: un programma che riceve i dati dal client che esegue il notebook, li elabora in base al linguaggio utilizzato e restituisce i risultati della computazione [33].

Per questa tesi, Jupyter è stato usato solo per scrivere script Python.

3.1.3 C++

C++[G] può essere considerato il successore del linguaggio C. È un linguaggio aperto, standardizzato ISO dal 1998. Grazie alla caratteristica di essere un linguaggio compilato e fortemente tipizzato, se il codice è ottimizzato bene, la sua esecuzione è molto veloce.

Per ottenere una buona ottimizzazione, il linguaggio consente di eseguire operazioni di basso livello in memoria, rendendo però il linguaggio "non sicuro". Si possono infatti modificare porzioni di memoria che intaccano l'intera esecuzione del programma. Ci si aspetta dunque che il programmatore sappia in tal senso usare il linguaggio in modo adeguato [24].

Utilizzato in misura significativa ancora oggi, ci sono molte librerie in C++ per la sua applicazione in diversi ambiti.

Proprio per le problematiche dei tempi di esecuzione citati in §2.4, il C++ si è rivelata la scelta migliore per la creazione dei metodi applicati in questa tesi.

3.1.4 NEST

Network Strategic Tool (NEST) è un software per le simulazioni e la pianificazione dello spazio aereo attraverso la gestione di risorse, come la capacità dei settori. È stato sviluppato da EUROCONTROL per la pianificazioni dei voli, sia per i servizi di aviazione (Air Navigation Service Provier - ANSP), sia per i centri di controllo degli Airspace (ACC), descritti in §2.2.

NEST permette la visione di questi dati attraverso specifici database forniti da EUROCONTROL chiamati AIRAC, che vengono rilasciati regolarmente ogni setimana.

Attraverso NEST, è possibile la visualizzazione di rotte in formato 2D o 3D, la modellazione dello spazio aereo, l'organizzazione del traffico attraverso gli *opening schema* e la simulazione di scenari alternativi [34].

Potendo esportare tali dati, NEST è stato utilizzato in questa tesi come sorgente delle basi di dati utili all'esecuzione degli algoritmi sviluppati.

3.1.5 SQLite

SQLite è una libreria scritta in C che implementa un database SQL. Le caratteristiche di questa libreria sono la velocità, la possibilità di incorporare in un unico file tabelle, query[G], trigger e funzioni. È multi piattaforma ed è facilmente utilizzabile con diversi linguaggi di programmazione.

Queste caratteristiche rendono SQLite la libreria per database più utilizzata al mondo [37]. Ad esempio, i segnalibri di Firefox sfruttano questa libreria.

In questa tesi, SQLite è stato utilizzato per creare un database relazionale a partire dai dati ricavati da NEST.

3.1.6 Visual Studio Code

Visual Studio Code è un editor di codice sorgente freeware rilasciato da Microsoft, disponibile per Windows, macOS e Linux. È tra gli editor più utilizzati grazie al gran numero di estensioni che i diversi utenti e la stessa Microsoft hanno sviluppato, permettendo l'utilizzo di questo editor per più linguaggi di programmazione. Il supporto che danno i plugin risulta fondamentale quando è richiesto un lavoro che un comune editor di codice non può eseguire, come il debugging o l'integrazione con applicazioni terze. Questa peculiarità permette a Visual Studio Code di essere più simile a un IDE, coi vantaggi però di un editor di codice, cioè più leggero e veloce.

Visual Studio Code permette inoltre di configurare in modo semplice i diversi workspace attraverso file JSON.

Queste caratteristiche rendono Visual Studio Code l'editor di testo attualmente più utilizzato [26].

3.2 Tecniche applicate

Lo sviluppo dei modelli matematici per la risoluzione al problema ATFM proposta in questa tesi hanno richiesto l'approfondimento e l'applicazione di alcune tecniche apprese nell'ambito della Ricerca Operativa e dell'Intelligenza Artificiale.

3.2.1 Programmazione lineare, CPLEX e generazione di colonne

Programmazione lineare

Un modello di programmazione lineare[G] è un particolare modello matematico che ha lo scopo di descrivere un problema di ottimizzazione. Consiste in una funzione obiettivo da minimizzare o massimizzare, le cui le variabili definiscono lo spazio delle soluzioni e in un insieme di vincoli che delimitano tale spazio. In un modello di programmazione lineare, tutti i vincoli e la funzione obiettivo sono lineari.

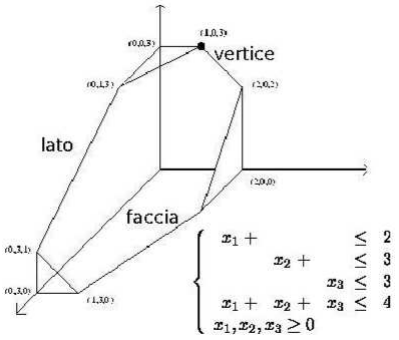


Figura 3.1: Poliedro associato a un modello di programmazione lineare [8]

Come mostrato in Figura 3.1, ogni variabile della funzione obiettivo definisce una dimensione dello spazio delle soluzioni e i vincoli del modello delimitano tale spazio, definendo dei piani che, intersecandosi, formano un poliedro. Lo strumento che verrà poi usato per risolvere il modello di programmazione lineare dovrà trovare un punto all'interno dello spazio delle soluzioni, che definisce la soluzione ottima. Le coordinate di tale punto sono i valori delle variabili che minimizzano la funzione obiettivo. Le soluzioni fuori dallo spazio delle soluzioni vengono definite *inammissibili*.

Dato che il compito del risolutore del modello consiste nell'esplorare le diverse soluzioni e trovare la soluzione ottima, è facile notare che, in prima approssimazione, più piccolo è lo spazio delle soluzioni, meno lungo sarà il tempo di risoluzione del modello.

La definizione estesa di un modello di programmazione lineare è (in forma standard):

$$\begin{array}{lll} \min & c_1 x_1 + \ldots + c_n x_n & \\ \text{s.t.} & a_{i,1} x_1 + \ldots + a_{i,n} x_n = b_i & (i = 1 \ldots m) \\ & x_i \in \mathbb{R}_+ & (i = 1 \ldots m) \end{array} \tag{3.1}$$

Questa definizione può essere ridotta in forma matriciale:

$$\min\{c^T x : Ax = b, x \geq 0\},$$

dove il vettore $x \in \mathbb{R}^n$ rappresenta le variabili decisionali, $c \in \mathbb{R}^n$ il vettore dei coefficienti delle variabili nella funzione obiettivo, $A \in \mathbb{R}^{m \times n}$ la matrice dei coefficienti dei vincoli e $b \in \mathbb{R}^m$ il vettore dei termini noti dei diversi vincoli. Viene infine definito il dominio delle variabili decisionali.

Tale dominio definisce la tipologia del modello di programmazione lineare e dunque anche i suoi metodi di risoluzione. Nel caso particolare in cui tutte le variabili decisionali sono intere ($x \in \mathbb{Z}^n$) si parla di modello di Programmazione Lineare Intera (PLI), mentre se una parte delle variabili può assumere valori reali ed altre valori interi, si parla di Programmazione Lineare Mista. Quando ad un modello di Programmazione Lineare Intera o Mista vengono rimossi i vincoli id interezza delle variabili, si esegue il rilassamento continuo del problema.

La maggiore "densità" di soluzioni in problemi con variabili reali permette di risolvere i modelli con algoritmi più veloci. Per questo motivo, una tecnica adottata

per la risoluzioni di problemi PLI consiste nel risolvere il rilassamento continuo del problema intero e poi applicare diverse tecniche di arrotondamento per soddisfare i vincoli di interezza.

Una delle tecniche per risolvere un modello di Programmazione Lineare è il metodo del Simplesso, che garantisce di trovare la soluzione ottima in un numero finito di iterazioni [8].

Uno dei metodi più utilizzati per risolvere problemi PLI, partendo dalla soluzione ottima del rilassamento continuo, è il Branch & Bound[G]. Questo metodo consiste nel costruire una struttura ad albero i cui nodi rappresentano diversi sottoinsiemi di soluzioni, con l'aggiunta iterativa di vincoli. Il nodo radice contiene tutte le soluzioni e i suoi nodi figli rappresentano sottoinsiemi differenti con l'aggiunta di vincoli diversi. Viene calcolata la soluzione col nuovo vincolo e successivamente vengono creati nuovi nodi figli, aggiungendo ricorsivamente vincoli. Questo metodo permette di suddividere il problema in sotto-problemi più piccoli, che esplorano diverse parti dello spazio delle soluzioni. Un buon albero è caratterizzato dalla sua capacità di esplorare il meno possibile aree dello spazio delle soluzioni e dalla creazione di sotto-problemi le cui aree esplorate si sovrappongono il meno possibile.

Un'altra tecnica addotta è quella dei piani di taglio (*cutting plane method*) e consiste nel risolvere una serie di problemi rilassati a cui vengono aggiunti iterativamente dei vincoli che tendono ad ottenere, attraverso la risoluzione rilassata del problema, dei valori interi per le variabili. Questa tecnica permette di mantenere la velocità di risoluzione dei problemi rilassati con l'aggiunta di vincoli ad-hoc. Esiste un metodo che combina il Cutting Plane col Branch & Bound, chiamato Branch & Cut [9]. È importante notare che l'aggiunta dei vincoli riduce lo spazio delle soluzioni ammissibili, escludendo però solo soluzioni che contengono valori non interi.

CPLEX

Uno dei risolutori utilizzati e che implementano algoritmi di ottimizzazione per PLI è la libreria CPLEX. IBM ILOG® CPLEX® Optimization Studio è una piattaforma che permette lo sviluppo e la risoluzione di problemi di ottimizzazione combinatoria attraverso la programmazione matematica e l'integrazione di vincoli [32]. L'ambiente comprende un IDE integrato che supporta Optimization Programming Language (OPL), e delle API che permettono di integrare CPLEX[G] con diversi linguaggi di programmazione. Nel caso di questa tesi, sono state utilizzate le API per C.

Generazione di colonne

La generazione di colonne (*Column Generation*) è un metodo per la risoluzione di modelli di programmazione lineare caratterizzati da un alto numero di variabili continue. Il suo funzionamento consiste nel partire dalla definizione di un modello formato da un ristretto sottoinsieme di variabili e semplice da risolvere, chiamato *Restricted master problem*. A questo modello vengono poi aggiunte iterativamente una o più variabili che portano un miglioramento della soluzione ammissibile [11].

Osservando il modello (3.1), è più semplice comprendere perché il metodo si chiami generazione di colonne. Quando si aggiunge una colonna per la variabile x, s'intende aggiungere ad ogni vincolo del modello la variabile x che precedentemente era esclusa dal

restricted master problem, ovvero si aggiunge una colonna nella matrice dei coefficienti dei vincoli.

Nello specifico, i passaggi dell'algoritmo sono:

1. Definizione di un modello iniziale facilmente risolvibile con un sottoinsieme di variabili;
2. Risoluzione del problema ottenendo anche la soluzione duale del problema;
3. Ricerca della variabile da aggiungere nel modello, ad esempio minimizzando il costo ridotto (*slave problem*), esprimibile attraverso la soluzione duale appena calcolata. Se la variabile è stata trovata e non è stato raggiunto il numero massimo di iterazioni, l'algoritmo riparte dal punto 2, altrimenti la soluzione corrente è quella ottima.

3.2.2 Metaeuristiche e Matheuristiche

Per la risoluzione dei modelli di Programmazione Lineare sono richieste ingenti risorse. A tal proposito, esistono metodi di risoluzione per trovare la soluzione ottima, chiamati metodi esatti, e metodi per la ricerca di soluzioni ammissibili del problema che restituiscono una soluzione non ottima, ma qualitativamente buona. Questi metodi vengono definiti euristici, perché si basano su assunzioni a priori a partire dalle caratteristiche della soluzione esatta. La particolarità di questi metodi, spesso impiegati in contesti reali, è la loro velocità di computazione e la poca memoria utilizzata.

Un metodo *metaeuristico* è definito per una categoria di problemi, mai per un caso specifico, definendo in generale l'algoritmo da seguire ad ogni iterazione. Tali metodi, per essere dunque applicati, devono essere sempre adattati al caso specifico. I metodi metaeuristici utilizzati per questa tesi sono principalmente basati sulla *ricerca locale* [12].

Per matheuristic si intendono invece metodi euristici[G] che sono basati su modelli matematici di programmazione matematica. Una caratteristica essenziale è lo sfruttamento, in alcune parti dell'algoritmo, del modello matematico utilizzato [6].

3.2.3 Metodo Greedy

Un algoritmo greedy[G] opera in modo iterativo partendo da un assunto che dovrebbe aiutare a raggiungere il più velocemente possibile una soluzione qualitativamente buona [12]. Il nome di questo metodo, che tradotto significa goloso, vuole motivare la caratteristica di questo algoritmo nel dirigersi verso la soluzione, al momento, migliore. Ad esempio, per il problema Traveling Salesman Problem (TSP), un semplice algoritmo greedy consiste nel muoversi all'interno del grafo verso il nodo più vicino non ancora esplorato. Un algoritmo greedy non garantisce l'ottimo, ma attraverso un'assunzione euristica, può fornire una soluzione qualitativamente buona in poco tempo.

3.2.4 Local search

Il metodo della ricerca locale (*Neighbourhood Search*) consiste nel partire da una soluzione ed iterativamente modificarla con l'obiettivo di migliorarla. Per comprendere

il metodo, si può immaginare una scacchiera dove ogni casella rappresenta una soluzione diversa e due caselle vicine rappresentano due soluzioni tra loro simili. Il metodo inizia partendo da una soluzione, cioè una casella, scelta arbitrariamente. Dalla posizione iniziale è possibile "vedere" solo i valori delle caselle adiacenti (il vicinato) e attraverso una funzione di esplorazione del vicinato ci si sposta verso una nuova casella che rappresenterà la nuova soluzione. Tale esplorazione può subire delle fasi di diversificazione[G] o intensificazione[G], cioè una ricerca orientata più a soluzioni diverse da quella corrente o a soluzioni simili.

Per definire il vicinato è utilizzata la funzione $N : X \to \mathcal{P}(X)$ con X che rappresenta l'insieme delle soluzioni ammissibili e $\mathcal{P}(X)$ l'insieme delle parti di X. Tale funzione definisce un insieme di soluzioni simili a quella passata come parametro, è poi la funzione di esplorazione del vicinato che stabilisce verso quale soluzione spostarsi.

Gli elementi fondamentali per implementare un algoritmo di ricerca locale sono [12]:

1. Il metodo di scelta della soluzione iniziale;
2. Il metodo per rappresentare una soluzione;
3. La funzione per generare un vicinato, data una soluzione;
4. La funzione che valuta il valore della soluzione vicina;
5. La strategia di ricerca del vicinato e il criterio di stop dell'algoritmo.

3.2.5 Simulated Annealing

Il *Simulated Annealing* è una tecnica metaeuristica utilizzata in contesti dove lo spazio delle delle soluzioni è ampio e ricco di minimi locali [12]. Il nome deriva dalla tecnica utilizzata in metallurgica per cambiare le proprietà di un metallo, mediante il riscaldamento e successivamente il raffreddamento lento.

L'algoritmo genera una soluzione iniziale che poi confronta con un'altra scelta randomicamente dal vicinato generato. Se questa è migliore, essa viene considerata come attuale soluzione ottima, altrimenti, in funzione crescente di un parametro chiamato temperatura, viene calcolata la probabilità con cui accettare la nuova soluzione non migliorante. Inizialmente, il parametro temperatura sarà alto, poi inizierà a diminuire lentamente durante le iterazioni [7].

In tal modo, in un primo momento l'algoritmo esplora un'ampia porzione dello spazio delle soluzioni, concentrandosi poi solo su soluzioni miglioranti.

3.2.6 Clustering

Le tecniche di apprendimento automatico si dividono in tecniche di apprendimento supervisionato e non supervisionato. Le prime, a cui appartengono anche gli alberi di decisione[G], ottengono i risultati analizzando dei dati già associati ad etichette (*labels*), apprendendo a quali etichette associare i nuovi dati che vengono analizzati.

Una delle differenze sostanziali tra i metodi supervisionati e non, sta nelle etichette associate ai dati: nel primo caso le labels vengono già stabilite ed utilizzate per

classificare i diversi dati, nel secondo invece vengono create in base ai valori delle features dei dati.

L'apprendimento non supervisionato consiste invece nell'analisi dei dati, in modo tale da comprendere le correlazioni e le somiglianze tra loro. Le tecniche di clustering[G] sono tecniche di apprendimento non supervisionato, che puntano a suddividere i dati in insiemi chiamati *cluster*. Per analizzare i dati, l'algoritmo mappa ogni elemento in uno spazio multidimensionale, nel quale il numero di dimensioni equivale al numero di *features* del dato, e infine valuta le distanze tra i diversi punti. È conseguente dunque che ogni feature debba essere definita con un valore numerico. Ad esempio, se volessi definire la somiglianza fisica tra due persone, il loro nome non può risultare una feature rilevante e utilizzabile.

Esistono gli algoritmi di clustering partizionali, che suddividono i dati in un numero predefinito di insiemi. Un esempio è l'algoritmo *k-means*. Altri algoritmi di clustering sono quelli gerarchici, che prevedono di partire da un unico cluster che contiene tutti i dati, per poi suddividerlo ricorsivamente (algoritmi divisivi). Diversamente, essi possono creare un cluster per ogni singolo dato ed iterativamente unirli (algoritmi aggregativi).

In questo lavoro di tesi, le tecniche di clustering utilizzate sono il k-means e il DBSCAN[G].

Il k-means consiste nel calcolare iterativamente k centroidi (punto centrale di un cluster) in modo da minimizzare la media della loro distanza dai punti appartenenti a quel cluster. DBSCAN invece analizza la densità di punti in una zona, ricevendo come parametro la distanza che definisce il vicinato di ogni punto e dunque anche il modo di definire i diversi cluster.

3.2.7 K-fold cross validation

Nel caso in cui si stia utilizzando una tecnica di apprendimento automatico supervisionato, in cui ogni esempio possiede la coppia features-label, è necessario eseguire due fasi principali: la fase di *training*, in cui il metodo apprende il modo migliore per assegnare un'etichetta a un esempio con determinate caratteristiche, e la fase di *testing*, dove, dato l'insieme delle features di un esempio, il metodo deve autonomamente assegnare una label. Viene verificato infine se l'assegnazione è corretta.

Per svolgere queste due fasi è necessario suddividere i dati raccolti in due insiemi chiamati comunemente `training_set` e `test_set`, solitamente con un rapporto 70-30.

Nel caso in cui i dati a disposizione non siano sufficienti per effettuare un buon addestramento del metodo, può essere applicata la tecnica del *k-cross validation*, detta anche validazione incrociata[G], la quale consiste nel suddividere i dati in k insiemi, denominati $F_1, \ldots, F_k$. Successivamente, il metodo di apprendimento automatico viene applicato k volte e, a ogni esecuzione, viene utilizzato come test_set uno degli insiemi, denominato F_n. Dunque, ad ogni esecuzione $training\_set = \{F_1, \ldots, F_k\} \setminus F_n,\ test\_set = F_n,\ n = [1, \ldots, k]$.

Con questa tecnica è possibile effettuare il training e il testing con un sottoinsieme diverso di dati, nonostante questi non siano molti.

3.2.8 Alberi di decisione

Per la creazione delle istanze e in supporto all'algoritmo di ricerca locale sono stati sviluppati alberi di decisione (*decision tree*), una tecnica di intelligenza artificiale utilizzata in questa tesi per il calcolo delle preferenze delle compagnie aeree e per la definizione del vicinato nell'algoritmo di ricerca locale.

Un albero di decisione valuta, per ogni elemento preso in esame, un insieme di caratteristiche, che possono essere determinanti per la classificazione[G] dell'elemento stesso. L'algoritmo crea inizialmente un nodo radice e valuta la caratteristica dell'elemento che può fornire il maggior "guadagno di informazione" per la classificazione dell'elemento. Ad esempio, se un albero di decisione dovesse scegliere con che mezzo una persona dovrebbe muoversi per raggiungere un posto, una discriminante importante potrebbe essere la distanza da percorrere o le condizioni meteo.

Il nodo radice riceve in input tutto il training_set, poi, in base alle features scelte, il database è suddiviso ricorsivamente tra i nodi figli, fino alla creazione delle "foglie", le quali sono raggiunte solo dagli input che possiedono un determinato valore per ogni feature. Creato l'albero di decisione, l'esempio del test_set passato in input esegue un percorso che inizia dal nodo radice, passa per i nodi figli (o interni) e raggiunge una foglia che rappresenta una label.

Per stabilire la bontà del classificatore nei metodi di apprendimento supervisionato come gli alberi di decisione, vengono analizzate diverse metriche. Le più importanti sono la precision[G] e la recall[G]: la precision indica quanti tra gli esempi che il classificatore ha dichiarato appartenere a una specifica classe lo siano veramente, e la recall indica quanti, tra tutti gli esempi appartenenti effettivamente a una classe, il classificatore riesce a classificare correttamente. Formalmente,

$$Precision = \frac{TP}{TP + FP} \qquad Recall = \frac{TP}{TP + FN}$$

dove:

TP: *true positive*, gli esempi classificati positivi (appartenenti alla label di riferimento) ed effettivamente positivi;

FP: *false positive*, gli esempi classificati positivi, ma che non lo sono;

TN: *true negative*, gli esempi classificati negativi (non appartenenti alla label di riferimento) ed effettivamente negativi;

FN: *false negative*, gli esempi classificati negativi, ma che sono positivi.

Capitolo 4

Soluzione del problema ATFM

Nel capitolo viene descritto il modello matematico adottato per affrontare il problema ATFM e in metodi sviluppati in precedenti lavori.

4.1 Un modello per ATFM

Il modello utilizzato in questa tesi per affrontare il problema ATFM, presentato in [2], è basato su variabili che selezionano la traiettoria per ogni volo all'interno delle pianificazioni definite da dati storici.

A monte dell'applicazione del modello, avviene la generazione di queste alternative attraverso le tecniche di apprendimento automatico descritte in §3.2. Come viene spiegato meglio al Capitolo 5, viene analizzato lo storico dei voli in un periodo particolarmente trafficato, analizzando le traiettorie dei diversi voli, suddividendoli per la tratta di appartenenza. Ogni variabile corrisponde dunque all'assegnazione di un volo a una determinata traiettoria seguita per percorrere una certa tratta, considerando anche il ritardo che quel volo può avere. Per ogni variabile decisionale, inoltre, viene predeterminato il livello di preferenza della traiettoria con un valore tra 0 e 1, come descritto in §5.4.

La scelta di costruire un modello su traiettorie già seguite in precedenza è motivata dalla loro, già verificata, fattibilità in contesti reali. Alcuni modelli presentati in §2.5 costruiscono direttamente le traiettorie, basandosi sulla rete dei settori aerei descritti in §2.2. In questo caso, le traiettorie possono risultare operativamente infattibili o non in linea con i vincoli imposti dalle compagnie aeree.

Viene di seguito riportato il modello per il problema ATFM proposto in [2].

Insiemi

F : insieme dei voli da pianificare espresso come enumerazione, non legata all'ID del volo. Ogni volo appartiene ad una certa tratta città-città;

T : enumerazione dei time unit, d'ora in avanti chiamati TU. Il numero dei TU dipende dal parametro di discretizzazione del tempo di pianificazione, a partire

dalla mezzanotte;

S : enumerazione dei settori dello spazio aereo nella configurazione aperta nel momento in viene eseguito il volo preso in considerazione;

$B(f)$: insieme dei possibili ritardi che può effettuare il volo $f \in F$, espresso come numero di TU contati dal momento della partenza pianificata del volo;

$P(f)$: insieme dei possibili piani di volo effettuabili dal volo $f \in F$. Dato che per una tratta vengono effettuati più voli, è possibile che questi seguano lo stesso piano di volo, chiamato anche traiettoria;

Parametri

G_p^f : livello di preferenza di una compagnia aerea per il piano di volo $p \in P(f)$ appartenente al volo $f \in F$. $G_p^f \in [0, 1]$;

STD^f : orario di partenza previsto per il volo $f \in F$ espresso come numero del TU che corrisponde all'orario di partenza del volo;

STA^f : orario di arrivo previsto per il volo $f \in F$ espresso come numero del TU che corrisponde all'orario di arrivo del volo;

D_p : durata del piano $p \in P(f)$ espressa come numero di TU. La durata di un piano di volo p appartenente al volo $f \in F$ può differire da $STA^f - STD^f$, dato che i piani di volo possono avere durata differente tra loro;

R : massimo budget di TU di ritardo cumulabile da tutti i voli;

$C_s(t)$: capacità del settore $s \in S$ nel TU $t \in T$ espresso in numero di voli che vi possono essere all'interno. Tale valore dipende anche dalla configurazione attiva al tempo t. La capacità di un settore può cambiare ogni unità di tempo (ogni ora solitamente). Se dunque i TU sono discretizzati con un tempo di 5 minuti, la capacità di un settore può cambiare al più ogni $k = 60/5 = 12$ TU (k = *"numero di TU che compongono un'unità di tempo"*);

$A_{ps}(t)$: parametro binario che vale 1, se il piano di volo $p \in P(f)$ "insiste" nel settore $s \in S$ al tempo $t \in T$, 0 altrimenti. Un piano di volo p insiste nel settore s al tempo t, se p entra nel settore s al tempo $u \in \{t, t+1, t+2, \ldots, t+k-1\}$. Questo procedimento viene realizzato perché in area ECAC un settore viene considerato occupato da un volo quando esso è in procinto di entrarci e quando vi rimane per un'unità di tempo;

Variabili decisionali

Le variabili decisionali sono tutte binarie e definiscono la pianificazione o meno di un determinato piano di volo con un specifico ritardo.

$$y_{pd}^f = \begin{cases} 1, & \text{se il volo } f \in F \text{ viene effettuato seguendo il piano di volo } p \in P \text{ con un} \\ & \text{ritardo a terra di } d \in B(f) \text{ TU} \\ 0, & \text{altrimenti} \end{cases}$$

Nel modello sono presenti anche altre variabili decisionali binarie, che stabiliscono se un volo $f \in F$ viene pianificato, al fine di ammettere soluzioni che non prevedono la pianificazione di tutti i voli. L'uso di questa variabile è utile nei metodi euristici utilizzati in [14, 15, 19].

$$z^f = \begin{cases} 1, & \text{se il volo } f \in F \text{ non viene pianificato} \\ 0, & \text{altrimenti} \end{cases}$$

Per indurre il risolutore a pianificare un volo, nella funzione obiettivo la variabile z ha un coefficiente M molto grande. Considerando $G = max_{f,p}\{G_p^f\}$ e $B = max_f\{B(f)\}$, si definisce $M = |F|\ max\{G, B\}$.

Il modello possiede due versioni, per poter rispondere a due diverse finalità:

- **Minimizzazione del ritardo:** il modello è impostato per ridurre al minimo il ritardo complessivo dei diversi voli pianificati, dove per ritardo s'intende di quanti TU il volo decolla dopo l'orario previsto (STD^f);
- **Massimizzazione delle preferenze:** il modello è impostato per massimizzare le preferenze attraverso il parametro G_p^f. In questa versione è presente un vincolo che impedisce di ottenere un ritardo complessivo dei voli superiore al parametro R.

L'utilizzo più comune per questa tesi delle due versioni del modello, consiste nel calcolare prima la minimizzazione del ritardo e utilizzare il valore ottimo della funzione obiettivo come valore del parametro R per la massimizzazione delle preferenze.

Il modello di programmazione lineare per il problema ATFM, presentato in [2] e successivamente esteso in [14] per la minimizzazione dei ritardi, è il seguente:

$$\min \quad \sum_{f \in F} \left(M z^f + \max \left\{ 0, \sum_{p \in P(f)} \sum_{d \in B(f)} (d + D_p) y_{pd}^f - (STA^f - STD^f) \right\} \right) \tag{4.1a}$$

$$\text{s.t.} \quad z^f + \sum_{p \in P(f)} \sum_{d \in B(f)} y_{pd}^f = 1, \qquad \forall f \in F \tag{4.1b}$$

$$\sum_{f \in F} \sum_{p \in P(f)} \sum_{d \in B(f)} A_{ps}(t - (STD^f + d)) y_{pd}^f \leq C_s(t), \quad \forall s \in S, t \in T \tag{4.1c}$$

$$y_{pd}^f \in \{0, 1\} \quad \forall f \in F, p \in P(f), d \in B(f) \tag{4.1d}$$

$$z^f \in \{0, 1\} \quad \forall f \in F \tag{4.1e}$$

Ogni vincolo del modello 4.1 ha il seguente significato:

(4.1a) la funzione obiettivo è spinta a pianificare più voli possibili. Per ogni volo $f \in F$ calcolo la differenza tra la durata effettiva del volo $(d + D_p)$ e la durata pianificata del volo $(STA^f - STD^f)$. Per il vincolo (4.1b), sappiamo che $\forall f \in F,\ \exists!\ y_{pd}^f = 1\ \forall p \in P(f), d \in B(f)$;

(4.1b) nel caso in cui il volo venga pianificato ($z^f = 0$), di questo va considerato un solo piano di volo ed un solo ritardo;

(4.1c) viene calcolata la partenza effettiva di ogni volo ($STD^f + d$) ed imposto che questi non eccedano sulle capacità dei diversi settori;

(4.1d, 4.1e) vengono definiti i domini delle variabili.

Nel caso invece della massimizzazione delle preferenze il modello è il seguente:

$$\max \quad \sum_{f \in F} \left(-M z^f + \sum_{p \in P(f)} \sum_{d \in B(f)} G_p^f y_{pd}^f \right) \tag{4.2a}$$

$$\text{s.t.} \quad z^f + \sum_{p \in P(f)} \sum_{d \in B(f)} y_{pd}^f = 1, \qquad \forall f \in F \tag{4.2b}$$

$$\sum_{f \in F} \sum_{p \in P(f)} \sum_{d \in B(f)} A_{ps}(t - (STD^f + d)) y_{pd}^f \leq C_s(t), \qquad \forall s \in S, t \in T \tag{4.2c}$$

$$\sum_{f \in F} \max \left\{ 0, \sum_{p \in P(f)} \sum_{d \in B(f)} (d + D_p) y_{pd}^f - (STA^f - STD^f) \right\} \leq R \tag{4.2d}$$

$$y_{pd}^f \in \{0, 1\} \quad \forall f \in F, p \in P(f), d \in B(f) \tag{4.2e}$$

$$z^f \in \{0, 1\} \quad \forall f \in F \tag{4.2f}$$

Ogni vincolo del modello (4.2) ha il seguente significato:

(4.2a) la funzione obiettivo è spinta a pianificare quanti più voli possibile, in particolar modo quelli con la preferenza massima;

(4.2b) medesima funzionalità di (4.1b);

(4.2c) medesima funzionalità di (4.1c);

(4.2d) simile a (4.1a), impone che la somma del ritardo accumulato non ecceda dal budget di ritardo massimo R;

(4.2e, 4.2f) vengono definiti i vincoli delle variabili.

Come si può notare dalla struttura del modello e dalla dichiarazione dei suoi insiemi, esiste una relazione "molti a molti" tra i voli ed i piani di volo:

- un volo ha a disposizione più traiettorie (dette anche piani di volo) per compiere la tratta a cui è assegnato;
- una traiettoria può essere utilizzata da più di un volo appartenente alla stessa tratta.

Non è scontato invece che ogni volo di una tratta possa effettuare ogni piano di volo appartenente alla medesima tratta, e viceversa.

Per ogni coppia volo-traiettoria è presente invece un numero fisso di possibili ritardi, stabiliti in base alla tipologia di volo, come illustrato in §5.3.

4.2 Soluzione esatta del modello

Per valutare l'impegno richiesto in termini di risorse computazionali dai calcolatori del laboratorio *Labnum* nel dipartimento di Matematica, è stato sviluppato, con le librerie messe a disposizione da CPLEX, il modello esatto descritto il §4.1.

Per valutare l'utilizzabilità del modello, la sua implementazione è stata eseguita con istanze di crescente dimensione, e analizzeremo i risultati in §5.6.

4.3 Soluzioni euristiche per il modello

A causa dell'alto numero di vincoli e variabili all'interno del modello, la risoluzione esatta con istanze relativamente grandi può diventare problematica per i risolutori, i quali non possiedono risorse sufficienti per il carico di lavoro richiesto.

Per questo motivo, l'approccio considerato in questa tesi e nei lavori precedenti correlati, consiste nello sviluppo di metodi euristici per il problema ATFM.

In questa sezione vengono presentate le tecniche euristiche per i modelli descritti in §4.1 sviluppate in [14, 15, 19].

4.3.1 Soluzione Greedy

L'algoritmo greedy sviluppato in [19] viene adoperato per ottenere la soluzione iniziale utilizzata dagli altri metodi. Trovare una soluzione qualitativamente buona consente a questi ultimi di ricevere in input una soluzione iniziale che permetta un minor tempo di esecuzione, rispetto ad una soluzione iniziale creata in modo randomico. Purtroppo, il problema preso in esame possiede molti vincoli rendendo molto alto il numero di soluzioni non ammissibili. Questo restringe lo spazio di ricerca dell'algoritmo greedy, il quale solitamente non riesce a restituire efficaci soluzioni iniziali.

Quando si vuole minimizzare il ritardo, l'algoritmo assegna ad ogni volo la traiettoria che attraversa i settori in quel momento più liberi, favorendo quelle con minor ritardo. Successivamente si verifica se, per ogni volo, sia disponibile una traiettoria che, a parità di ammissibilità, diminuisca il ritardo totale. In caso positivo, la si sostituisce con la precedente traiettoria. L'obiettivo prioritario è di pianificare più voli possibili, quello secondario di ridurre il ritardo.

Nel caso in cui si vogliano massimizzare le preferenze, si comincia innanzitutto ad assegnare ad ogni volo la traiettoria con meno ritardo. A parità di ritardo, quelle con la maggiore occupazione residua nei settori. Come in precedenza, nel caso sia presente per un volo una traiettoria che a parità di ammissibilità porti a un ritardo minore rispetto a quella precedente, questa viene sostituita. Dopodiché vengono aggiunti eventuali voli

non pianificati ed infine si verifica se siano presenti delle traiettorie che a parità di ammissibilità, aumentano il valore complessivo delle preferenze.

4.3.2 Generazione di colonne

Il metodo di generazione di colonne (vedi §3.2.1), per la soluzione del rilassamento continuo dei modelli presentati in §4.1, è stato sviluppato in [19].

Il restricted master problem iniziale viene creato dalle variabili proposte dalla soluzione dell'algoritmo greedy. Successivamente, ad ogni iterazione vengono aggiunte n colonne al restricted master problem, in base ad un parametro impostato all'avvio dell'algoritmo. Ad ogni iterazione, viene risolto il problema, ottenendo la coppia di soluzioni primale-duale, in modo da calcolare i costi ridotti di tutte le variabili escluse dal modello corrente e individuare quali colonne aggiungere per l'iterazione successiva.

Nel corso delle iterazioni, è importante contenere la dimensione del restricted master problem. È dunque presente un parametro, chiamato *aging* (invecchiamento) che stabilisce quante iterazioni consecutive una colonna può restare al massimo nel modello.

Operando su questi due parametri, il numero delle nuove colonne e l'invecchiamento, è possibile agire pesantemente sul modo di evolvere della soluzione e sui tempi di convergenza alla soluzione ottima.

Una volta ottenuta la soluzione del rilassamento continuo, l'ultimo restricted master problem viene risolto con variabili intere, portando ad una soluzione (euristica) del modello ATFM.

I tempi di esecuzione complessivi di questo metodo per istanze di media-grande dimensione possono raggiungere diverse ore.

4.3.3 Matheuristica

Il metodo presentato in [15] si basa su una matheuristica, che vede dunque la combinazione di metodi euristici e tecniche di programmazione matematica, come descritto in §3.2.1.

La prima versione di questo metodo si basa sul simulated annealing descritto in §3.2.5, dove ad ogni iterazione viene preso in considerazione un ristretto numero di variabili a causa del loro eccessivo numero in un'istanza completa. Il vicinato viene definito fissando alcune variabili della soluzione e dando la possibilità alle altre di essere cambiate. Per ogni insieme di variabili, viene poi risolto il modello descritto in §4.1. Tale metodologia permette di trovare buone soluzioni in tempi adeguati.

Una seconda versione del metodo consiste nell'imporre che ogni nuova soluzione non si discosti troppo da quella corrente, portando ad una diminuzione del tempo richiesto ad ogni iterazione, dato che si sta esplorando una porzione minore dello spazio delle soluzioni. Anche in questa versione, la ricerca locale segue le procedure del simulated annealing, inserendo però una soglia minima di miglioramento (per maggiori dettagli vedere [15]).

Tale metodo risulta più veloce del metodo con la generazione di colonne, fornendo però soluzioni solitamente meno buone. La soluzione della matheuristica può però

essere utilizzata come soluzione iniziale per la generazione di colonne, portando a tempi complessivamente migliori [15].

4.3.4 Ricerca locale

Tale metodo, sviluppato in [14], segue i passaggi di una ricerca locale, descritti in §3.2.4, applicati al problema ATFM. A causa dell'elevato numero di variabili in gioco, è stato deciso di considerarne solo una parte, andando a costruire il vicinato secondo due diversi criteri:

1. **Ricerca nello spazio degli assegnamenti:** data una soluzione in cui per ogni volo esiste un assegnamento $f \in F \rightarrow p \in P(f)$, il vicinato viene definito dai possibili piani di volo alternativi in $P(f)$. Con questa procedura si evita di togliere dalla pianificazione alcuni voli, assegnando un piano di volo alternativo. Per evitare di considerare come vicinato tutto lo spazio delle soluzioni, il numero delle sostituzioni dei piani di volo è limitato ad un parametro massimo k. La valutazione della soluzione avviene tramite la funzione obiettivo;

2. **Ricerca nello spazio degli insiemi di variabili decisionali:** con questo tipo di ricerca viene considerato un piccolo sottoinsieme delle variabili, purché questo comprenda le variabili poste ad 1 in una soluzione ammissibile. Tale decisione è stata presa perché per definire una soluzione solo circa il 2% delle variabili viene impostato ad 1, le restanti sono nulle. Il sottoinsieme di variabili viene generato vincolando a 0 tutte le variabili che non vi appartengono. Se consideriamo $\mathbb{P}(var)$ l'insieme delle parti di *Var*, che rappresenta l'insieme di tutte le variabili, il vicinato è composto da un sottoinsieme di $\mathbb{P}(Var)$, cioè un insieme di sottoinsiemi di variabili che si possono ottenere a partire da quello corrente. La valutazione del vicinato avviene risolvendo il modello in §4.1 con il sottoinsieme di variabili attraverso il risolutore di CPLEX.

Dopo determinate valutazioni in [14], è stato accertato che la seconda tipologia di ricerca fosse più fattibile.

A supporto di questa procedura, sono stati sviluppati metodi di apprendimento automatico (alberi di decisione) per migliorare la definizione dei sottoinsiemi di variabili attraverso un opportuno classificatore, che, in base ad alcune features prefissate, determina la capacità delle variabili di contribuire al miglioramento locale delle soluzioni.

Capitolo 5

Generazione delle istanze dai dati storici

Nel capitolo viene descritto il procedimento utilizzato per la generazione delle istanze di dati da passare come input ai diversi metodi descritti in §4.3 e §4.2. Il procedimento segue le indicazioni presenti in [2] ed è stato sviluppato nei diversi lavori in [14, 15, 19, 20], e presentiamo degli ulteriori miglioramenti.

Come è possibile notare dalla struttura del modello in §4.1, le diverse variabili fanno riferimento ad uno specifico insieme di voli e di piani di volo. Non è appropriato definire tali insiemi sulla sola base della disposizione dei settori aerei. In primo luogo, perché dover tracciare una traiettoria attraverso i settori è un ulteriore problema di ottimizzazione, che richiede aggiuntive risorse, successivamente perché tali rotte sarebbero ancora da verificare, per fattibilità, in un contesto reale.

Per questo motivo, dai data repository descritti in [2] è stato deciso di raccogliere lo storico dei voli di un determinato periodo ed utilizzare tali dati per determinare, attraverso tecniche di apprendimento automatico, le relazioni tra gli insiemi F e P, ed i valori di G_p^f nel modello.

5.1 Dati da NEST

Per collezionare i dati relativi ai voli è stato utilizzato il software NEST, il quale legge i database AIRAC dei periodi presi in esame. Per questa tesi, poiché l'obiettivo è processare quanti più dati possibile, sono stati analizzati i voli effettuati nel periodo dal 23/06/2016 al 14/09/2016.

Oltre che permettere la visualizzazione grafica dei voli e dei settori nel periodo impostato, come mostrato in Figura 2.3 e in Figura 2.1, NEST dà la possibilità di esportare i dati dei voli in diversi formati. Le esportazioni da effettuare sono di cinque tipologie di file differenti:

- **.xlsx:** file denominati `ENV_Original` per il programma Excel, i quali contengono la definizione dello spazio aereo in termini di settori e degli opening scheme delle diverse giornate;

- **.arp:** file con le coordinate dei diversi aeroporti coinvolti nel periodo preso in esame;
- **.exp2:** file contenenti le informazioni dei diversi voli. Queste consistono nell'ID di volo, le città di origine e destinazione, il modello di aereo, la data del volo ed il suo orario di partenza e di arrivo;
- **.t5:** file contenenti le descrizioni delle traiettorie dei voli in termini di settori attraversati. Dunque, ogni traiettoria è descritta come una sequenza di settori aerei, il loro orario di entrata, di uscita e l'altitudine;
- **.so6:** file contenti le descrizioni dei voli come sequenza di punti 4D nello spazio-tempo.

5.2 Creazione del database

I dati esportati dal software NEST devono poi essere formattati in modo tale da poterli elaborare con la procedura di creazione delle istanze.

Per tale operazione è stata sviluppata la struttura di un database in SQL adattata per ricevere i dati dei file sopra descritti e uno script in Python che popola il database tramite i file esportati da NEST.

Dopo la popolazione del database SQL, che per contenere i dati del periodo desiderato pesa circa 40 GB, questo viene convertito in formato SQLite.

5.3 Selezione delle traiettorie

Analizzando le traiettorie, si è visto come buona parte di quelle appartenenti alla stessa tratta si somigliassero molto. Per ridurre dunque il numero di variabili da utilizzare nel modello, è stato effettuato un cluster per raggruppare tutte le traiettorie simili.

Un primo metodo di clusterizzazione utilizzato è il DBSCAN, nel quale sono impostati due parametri principali: la distanza minima tra due dati nello stesso cluster (densità) e il numero di elementi in un cluster. Un altro metodo di clusterizzazione utilizzato è il k-means, che utilizza il parametro esogeno k (numero di cluster). I metodi sono descritti in §3.2.6.

Dato che nella descrizione delle traiettorie viene indicato l'aeroporto di partenza e di arrivo, e che in una città possono essere presenti più aeroporti, prima di effettuare il raggruppamento delle traiettorie vengono raggruppati gli aeroporti appartenenti alla stessa città.

La clusterizzazione può avvenire seguendo due possibili descrizioni delle traiettorie:

1. **Descrizione geografica:** ogni punto della traiettoria viene descritto attraverso la tupla *(latitudine, longitudine, altezza da terra, tempo)*;
2. **Descrizione per settori:** la traiettoria è descritta come sequenza di settori attraversati e ogni settore è indicato con la tupla *(ID del settore, orario di ingresso nel settore, orario di uscita dal settore, altezza da terra).*

I passaggi di questo primo cluster sono i seguenti [2]:

- **Ricampionamento delle traiettorie:** i punti geografici che descrivono le traiettorie vengono linearmente interpolati per descriverle con 50 punti equidistanti temporalmente tra loro;
- **Esclusione del decollo e dell'atterraggio:** per escludere le manovre di decollo e di atterraggio, vengono ignorati il 10% dei punti iniziali ed il 5% del punti finali della rotta;
- **Analisi PCA:** per migliorare il clustering, viene utilizzata una Principal Component Alalysis (PCA) che permette di individuare delle caratteristiche artificiali delle traiettorie che meglio riescono a raggrupparle;
- **Esecuzione del DBSCAN:** esecuzione dell'algoritmo DBSCAN per la definizione dei cluster e l'eliminazione degli *outsider*. È possibile stabilire che tipo di cluster eseguire
  - *Cluster geometrico:* due traiettorie sono ritenute simili quando sono geograficamente simili tra loro;
  - *Cluster discretizzato:* due traiettorie sono simili quando attraversano gli stessi settori.

Dopo la definizione dei cluster, come mostrato in Figura 5.1, viene selezionata una traiettoria di riferimento per ogni cluster. Esse diventano le diverse alternative per ogni volo appartenente alla medesima tratta. È presente un parametro che indichi come scegliere i rappresentanti dei cluster, ad esempio la tratta più veloce o quella più vicina al centroide.

Per stabilire quante alternative di piani di volo assegnare ad ogni volo, le rotte vengono suddivise per tipologia, ed ognuna di queste prevede un numero preciso di alternative. Il procedimento prevede che se vengono individuati k cluster per una rotta e servono n piani di volo alternativi, con $n \leq k$, vengono scelte le rotte rappresentanti di n cluster, una per ciascuno. Le tratte vengono suddivise in:

- **Main:** viene eseguita una classificazione delle città più trafficate e tra queste ne vengono scelte n, in base ad un parametro fissato. Per questo tipo di rotte viene effettuato un secondo clustering *k-means* che individua $k = \sqrt{n/2}$ centri, con n il numero totale delle traiettorie. Di questi centri viene scelto un rappresentante per cluster;
- **Eur:** delle restanti rotte che hanno origine e destinazione in Europa, viene scelto un numero fisso di alternative, ora impostato a 4;
- **Default:** delle rotte che entrano o escono dall'Europa viene scelta una sola traiettoria di riferimento.

Con questa prima fase si definiscono gli insiemi F e $P(f)$ del modello e le relazioni tra i due.

Per la definizione dei possibili ritardi da assegnare ad ogni volo, è possibile impostare quanto discretizzare la durata dei voli. Ad esempio, le analisi di questa tesi utilizzano tutte istanze con TU da 5 minuti. In base alla tipologia di volo, è stato deciso inizialmente di permettere un preciso numero di ritardi possibili, in particolare:

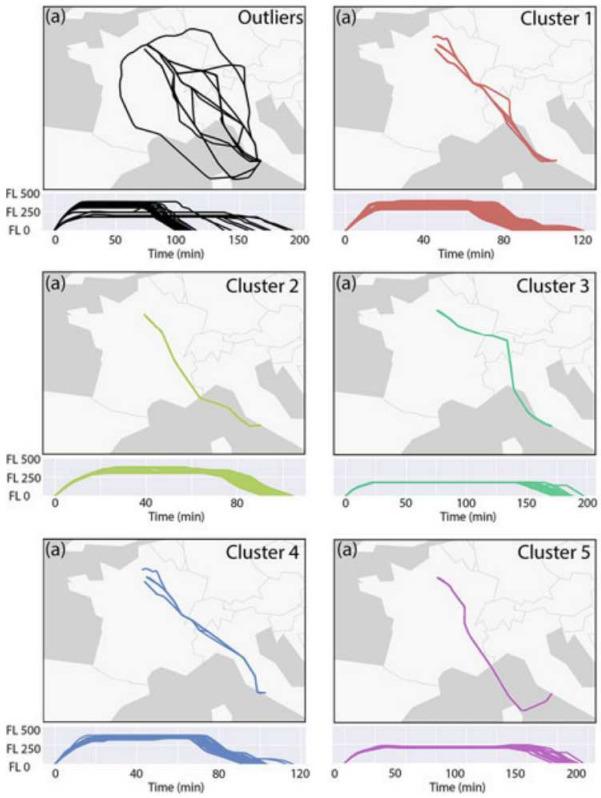


Figura 5.1: Risultati del clustering delle traiettorie nella rotta Roma-Parigi [17]

- **Main:** viene concesso il ritardo per ogni TU, per un ritardo massimo di 45 minuti. Dunque sono concessi $45/5 = 9$ possibili ritardi;
- **Eur:** i ritardi vengono permessi ogni 3 TU, per un massimo di 90 minuti di ritardo. Dunque sono concessi $75/15 = 6$ possibili ritardi;
- **Default:** viene concesso un ritardo ogni 30 minuti, per un massimo di 90. Dunque sono concessi $90/30 = 3$ possibili ritardi.

La scelta è stata fatta per poter dare al modello la possibilità di avere più "elasticità" coi voli principali, in modo da garantire maggior probabilità di essere inclusi nella soluzione del modello.

5.4 Classificazione dei voli

Per predire il grado di preferenza G_p^f che le diverse compagnie aeree hanno per effettuare il volo f tramite il piano di volo p, è stato utilizzato un albero di decisione per ogni singola tratta. In questa fase, le features considerate non sono più i punti delle traiettorie dei diversi voli, bensì altre caratteristiche del volo, quali: il giorno della settimana, il periodo dell'anno, in che parte del giorno è stato effettuato il volo, la compagnia aerea del volo, il tipo di aeromobile ed il tipo di volo.

Essendo quello usato un metodo di apprendimento supervisionato, viene effettuata una fase di *training* e una di *testing*. Infine, i valori di precision e recall vengono confrontati coi valori ottenuti da un secondo classificatore di supporto.

L'obiettivo dell'albero di decisione è quello di poter ricevere in input un volo e, in base alle features che possiede, stabilire in percentuale quale piano di volo sarebbe meglio seguire, come mostrato in Figura 5.2, sulla base di precedenti voli simili a quello preso in esame. I passaggi per la costruzione dei voli sono i seguenti [17]:

- **Definizione delle foglie:** le foglie consistono nei diversi cluster trovati nella fase precedente. Più foglie possono essere associate allo stesso cluster, perché voli con caratteristiche diverse possono seguire la stessa tipologia di traiettoria. Dunque il numero di foglie può superare quello dei cluster;
- **Costruzione dell'albero:** i rami dell'albero vengono definiti attraverso più fasi di training e testing, seguendo il metodo *k-fold cross validation*. Le label degli esempi consistono nel cluster a cui la traiettoria del volo preso in esame appartiene;
- **Definizione dei valori delle preferenze:** a questo stadio dello sviluppo, quando inserisco in input un volo, esso raggiunge una foglia che rappresenta uno specifico cluster/traiettoria. L'obiettivo però è di fornire per ogni volo una percentuale di preferenza riferita a tutte le possibili traiettorie alternative. A tal fine, viene definito n_i^c come il numero di voli che etichettati col cluster c raggiungono la foglia i, la quale, si ricorda, rappresentava anch'essa un cluster. Se dunque il classificatore è stato allenato correttamente, ci si aspetta che $n_k^c \geq n_i^c \quad | \quad i, k \in \{$ "foglie dell'albero" $\}, \quad k =$ "foglia associata al cluster c". Si definisce poi la percentuale di tale valore con $\overline{n_i^c} = n_i^c / n$. Viene definito infine $G_p^f = \overline{n_{i(f)}^{c(p)}}$ dove $c(p) =$ "il cluster a cui appartiene il piano di volo $p \in P(f)$", è $i(f) =$ "la foglia dell'albero raggiunta dal volo f". In tal modo è possibile ritrovarsi con valori $G_p^f \geq 0$ anche quando p non appartiene allo stesso cluster a cui appartiene la traiettoria di f. Alla fine di questo processo, ogni foglia dell'albero di decisione riporta un livello di preferenza per ogni cluster; dunque, l'informazione iniziale in cui ogni foglia era associata ad un singolo cluster, diventa superflua.

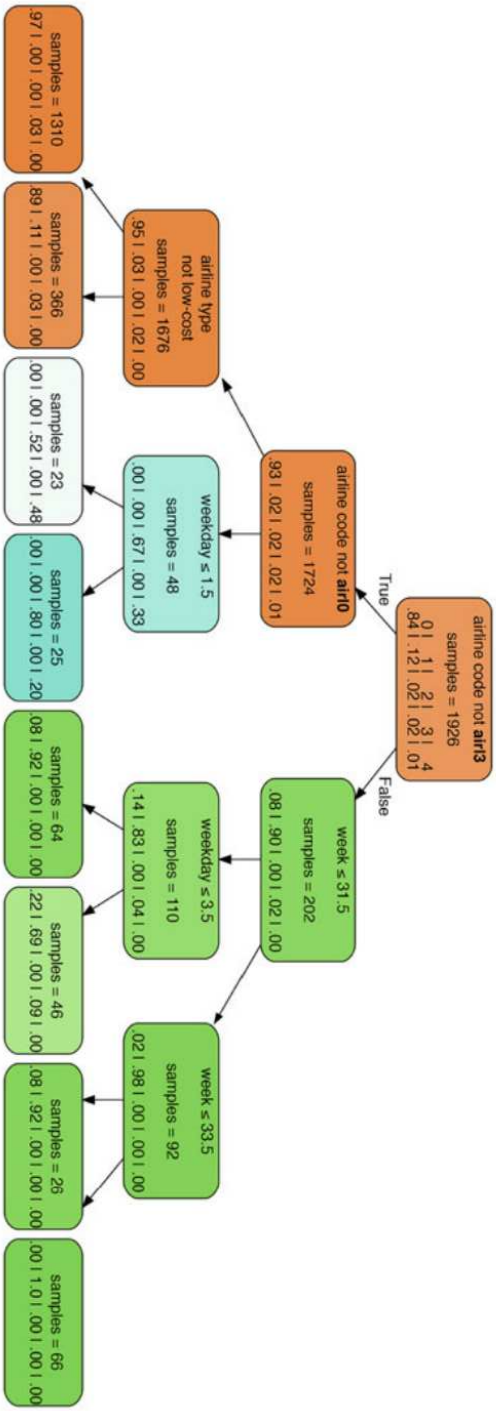


Figura 5.2: Esempio di albero di decisione [17]

5.5 Memorizzazione risultati parziali

Il processo di generazione delle istanze può richiedere diverse ore. Per ridurre tale tempistica sono stati adottati due principali stratagemmi [15]: l'utilizzo massiccio di algoritmi parallelizzabili e il salvataggio di risultati parziali.

L'utilizzo della parallelizzazione permette di sfruttare tutti i core del processore della macchina utilizzata, consentendo di ridurre il tempo di esecuzione anche di 6 volte, come il numero dei core del processore utilizzato.

Nel momento in cui vengono cambiati i parametri in input e rimane invariato il database, dopo una seconda o successiva esecuzione esistono alcune operazioni, anche lunghe, che vengono ripetute. Considerando solo le operazioni computazionalmente pesanti che vengono parallelizzate, di alcune viene salvato il risultato parziale:

- **Lettura del database:** i risultati delle query che leggono i dati dal database vengono salvati in tre distinti file per ogni tratta. I tre file rispecchiano il contenuto dei file di NEST descritti in §5.1, in cui il primo descrive le caratteristiche dei voli, il secondo la descrizione geografica delle traiettorie ed il terzo la descrizione delle traiettorie come sequenza di settori aerei. Questo tipo di dati può essere calcolato una-tantum, se richiesto, ed essere riutilizzato nelle successive esecuzioni dell'algoritmo. I tre file per ogni traiettoria vengono creati solo nel momento in cui viene generata un'istanza che possiede tale tratta, pertanto, per la creazione delle prime istanze sono molteplici le tratte di cui bisogna calcolare i dati. Le ultime, dunque, richiedono la creazione di meno file e terminano più velocemente;
- **Clustering:** poiché il database non viene cambiato, anche i risultati del processo di clusterizzazione delle traiettorie rimangono invariati. Anche questi dati vengono salvati per le esecuzioni successive in appositi file;
- **Alberi di decisione:** come nella creazione dei cluster, per ogni tratta esiste un albero di decisione, che che cambia solo al variare del database. La sua struttura è salvata in un file per evitare di ricostruirla ad ogni esecuzione;
- **Classificazione dei voli:** i risultati di questi calcoli variano in base ai parametri inseriti e vengono perciò ricalcolati ogni volta che questi cambiano. I risultati finali vengono salvati in appositi file per rendere quasi immediata l'esecuzione dell'algoritmo quando i parametri in input restano invariati.

Si fa notare che le procedure implementate in [15], dove venivano memorizzati solo una parte dei dati ricavati dal database, i risultati del clustering e i dati finali, sono state migliorate per avere la memorizzazione di tutti i dati parziali ritenuti rilevanti, come sopra descritta.

Per semplificare l'esecuzione dell'algoritmo, è presente un file per i parametri in formato `.yaml`. Con questa metodologia è possibile eseguire in parallelo la creazione di più istanze, anche se si sconsiglia tale procedura per un possibile conflitto tra i file dei risultati parziali. Avere diversi file di configurazione è utile invece per automatizzare la creazione sequenziale di istanze differenti.

Il parametro principale da impostare è il giorno a cui deve fare riferimento l'istanza. L'algoritmo utilizza l'intero database SQLite per la creazione e l'allenamento dei cluster e degli alberi di decisione. Dopodiché, se si possiedono i dati dei voli di uno specifico

giorno, è possibile inserirli nel database seguendo la stessa procedura descritta nei passi precedenti ed infine classificare i voli.

5.6 Miglioramenti

Rispetto alla precedente versione dell'algoritmo per la creazione delle istanze, sono stati apportati alcuni miglioramenti in termini di performance, e sono stati sistemati dei bug minori. Nello specifico:

- **Funzione per la clusterizzazione:** precedentemente erano presenti due versioni della funzione per la clusterizzazione e creazione degli alberi di decisione. La prima creata da [19] e la seconda, più lineare, creata da [14]. La seconda funzione, preferibile alla prima in termini di performance, aveva il difetto di interrompersi durante la creazione di specifiche istanze. Il problema consisteva nel fatto che non venivano ignorate quelle tratte che nel giorno indicato non avevano voli programmati, mandando in errore il processo di clusterizzazione. Tale problema è stato risolto;

- **Velocizzazione della fase di clusterizzazine:** effettuando alcuni test sulle performance della funzione per la clusterizzazione, è risultato che il 90% del tempo era occupato per le query al database SQLite. Dato che le stesse chiamate vengono eseguite durante la lettura del database, è stato deciso di salvare i risultati delle chiamate in appositi file. In tal modo, la funzione non deve eseguire una query che analizza quasi 2 milioni di records, ma semplicemente leggere un file con i risultati calcolati una-tantum. Un metodo che velocizzava ulteriormente la lettura dei dati per la clusterizzazione, consisteva nel creare delle tabelle direttamente nel database SQLite coi risultati delle query usate più volte, dato che il processo di lettura dei dati da un database è più veloce della lettura da file. Purtroppo l'opzione è stata scartata, perché i database SQLite non permettono la scrittura in parallelo, togliendo la capacità di parallelizzare alla procedura di creazione delle istanze. Se nella prima versione dell'algoritmo erano richiesti circa 20 secondi a istanza (i test venivano eseguito senza parallelizzazione), con tale metodo servivano 0.1 secondi, dunque lo 0.5% del tempo impiegato;

- **Organizzazione delle funzioni:** precedentemente, la fase di clusterizzazione e classificazione dei dati erano uniti in un'unica funzione. Come spiegato sopra, queste due fasi devono essere invece trattate separatamente, perché la clusterizzazione può essere eseguita solo una volta quando si calcolano le nuove tratte, mentre la classificazione ogni volta che vengono cambiati i parametri in input. Le due fasi sono state dunque separate in funzioni differenti;

- **Debug della normalizzazione:** nella fase di normalizzazione delle traiettorie in 50 punti equidistanti nel tempo, era presente un bug nel calcolo dei punti, che risultavano talvolta erroneamente troppo distanti tra loro. Attraverso l'utilizzo di una libreria apposita il problema è stato sistemato;

- **Compressione dei file temporanei:** considerato che per ogni tratta presa in esame vengono creati 5 file, e che durante le date prese in esame sono state percorse circa 35000 tratte, vengono generati in tutto circa $5 \cdot 35000 = 175000$ file. Per quanto questi possano essere leggeri, il loro numero impone di utilizzare un

Fase algoritmo	Old Version (ore)	New Version (ore)
Analisi Tratte	4.5/5	5.5
Cluster e Classificazione	10	1
Altre Fasi	0.25	0.25
Totale	14.75	6.75

Tabella 5.1: Confronto dei tempi di esecuzione per la creazione di un'istanza

algoritmo di compressione per ridurre il loro peso complessivo. A tal fine, è stato scelto di utilizzare l'algoritmo di compressione *lzma* che, come riportato in [27], è tra gli algoritmi col più alto rapporto di compressione, ma anche tra quelli che richiede più tempo per essere decompresso. Per questo motivo, i file temporanei che vengono letti poche volte vengono compressi con l'algoritmo *lzma*, gli altri con l'algoritmo *gzip*;

- **Organizzazione dei file:** l'algoritmo di generazione delle istanze è stato rimaneggiato più volte. Inizialmente è stato sviluppato in un unico notebook Jupyter in Python, poi è stato organizzato in più file, ed infine fu creata una copia del progetto in un unico script Python per permettere un veloce debug. Avendo più versioni dello stesso algoritmo da utilizzare, la fase di sviluppo procedeva lenta, dunque è stato deciso di convertire completamente i notebook Jupyter in file Python, mantenendo una separazione tra le diverse funzioni e permettendo una più rapida fase di debugging;
- **Analisi dei giorni più trafficati:** per la creazione di istanze significative, è stato aggiunto uno script Python chiamato `busyDay.py` per ottenere una classifica dei giorni più trafficati nel periodo preso in esame.

5.6.1 Tempi

È stato eseguito un confronto tra la precedente e la presente versione dell'algoritmo per la creazione delle istanze, al fine di verificare la bontà delle modifiche.

Nella Tabella 5.1 sono mostrati i tempi della creazione di un'istanza senza la presenza di risultati parziali, prima e dopo i miglioramenti apportati. Tutti i valori delle seguenti tabelle riportano i risultati della creazione di istanze che hanno un tempo di discretizzazione di 5 minuti e possiedono 20 città di tipo Main.

Dalla Figura 5.3 e dalla Tabella 5.2 è possibile osservare come i file parziali, man mano che vengono generate nuove istanze (nella Tabella 5.2 le righe delle istanze sono in ordine cronologico di creazione), aiutino a ridurre drasticamente i tempi di esecuzione. Infatti, se per ogni istanza il numero di voli resta costante a circa 30 000, il numero delle nuove tratte da analizzare diminuisce linearmente con i tempi di generazione dell'istanza.

Il più importante miglioramento è avvenuto con la modifica delle chiamate al database SQLite.

È stato notato come invece nella creazione di istanze di prova che contenevano circa un centinaio di tratte, verosimilmente circa 400 traiettorie, i tempi di classificazione dei voli aumentavano vertiginosamente, anche di 400 volte (50 secondi a tratta rispetto ai soliti 0.1). Tale difetto si dimostra però poco rilevante ai fini del lavoro finale. Si lascia una più approfondita analisi del problema in lavori futuri.

Data	Tempi (min)	N° Nuove Traiettorie	N° Voli
08/07/2016	371	14173	32484
13/08/2016	121	5334	28850
26/08/2016	81	2133	32752
27/08/2016	72	1361	28866
28/08/2016	90	2648	30443
29/08/2016	82	2018	32023
30/08/2016	77	1739	31754
31/08/2016	78	1668	32027
01/09/2016	76	1609	32260
02/09/2016	74	1248	32882
09/09/2016	75	1303	32831

Tabella 5.2: Tempi di esecuzione della generazione progressiva delle istanze

Data	N° Voli Main	N° Voli Eur	N° Voli Def	N° Tot Voli
07/08/2016	3823	21227	6775	31825
13/08/2016	3088	18700	6716	28504
26/08/2016	3540	21595	6872	32007
27/08/2016	3113	18490	6905	28508
28/08/2016	3348	19605	7137	30090
29/08/2016	3692	20805	6788	31285
30/08/2016	3692	20490	6807	30989
31/08/2016	3760	20699	6821	31280
01/09/2016	3871	20703	6914	31488
02/09/2016	3821	21501	6804	32126
09/09/2016	3932	21489	6632	32053
Media	3607	28482	6834	30923

Tabella 5.3: Numero di voli in ogni istanza suddivisi per tipologia

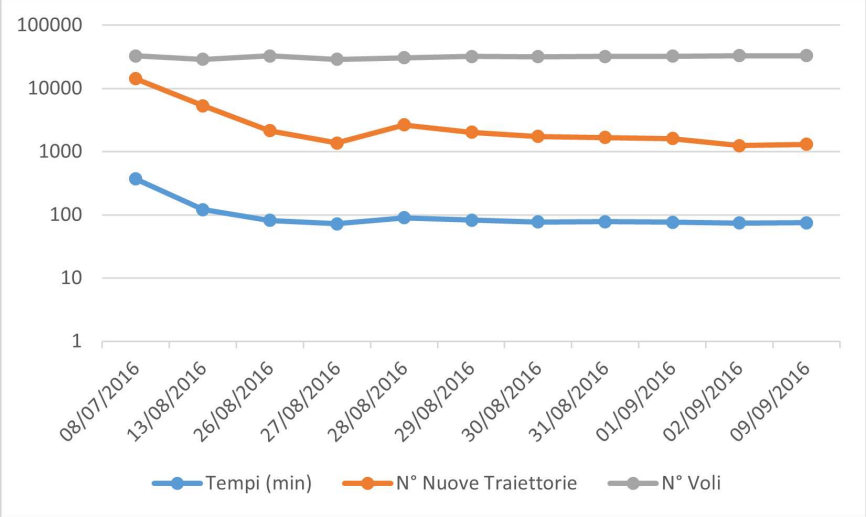


Figura 5.3: Tempi di generazione progressiva delle istanze

5.7 Caratteristiche delle istanze

Durante la creazione delle istanze, sono state effettuate misurazioni per analizzare le caratteristiche dei voli effettuati nelle giornate considerate.

I successivi dati sono stati prelevati durante la creazione di istanze, dove viene utilizzato un TU da 5 minuti, 20 città Main, le altre 2125 città considerate come secondarie e i ritardi diversificati per tipologia di volo, come descritto in §5.3.

Innanzitutto, è stato analizzato il numero di voli suddivisi per tipologia, visibili alla Tabella 5.3, con lo scopo di verificare quanto ogni tipologia di volo incide sul numero di variabili nel modello finale.

Sono poi stati misurati il numero medio, minimo e massimo di piani di volo alternativi che ogni volo possiede nel modello. I risultati sono presenti nella Tabella 5.4, Tabella 5.5 e Tabella 5.6. Ovviamente, per i voli di tipo Default, tali valori sono tutti ad 1, perché ad essi viene concessa una sola traiettoria, mentre per i voli di tipo Eur le alternative sono 4. Come si può osservare in Tabella 5.4, il classificatore tende ad assegnare più alternative possibili ad ogni volo.

Considerando i ritardi diversificati per tipologia di volo e prendendo in considerazione il numero medio di alternative per volo, è stato calcolato quanto ogni tipologia di volo pesi sul numero di variabili del modello finale: per il calcolo, bisogna moltiplicare i valori della Tabella 5.3 con la Tabella 5.4 ed i rispettivi ritardi. I risultati sono presenti alla Tabella 5.7.

È stato infine verificato da quanti punti fosse descritta una traiettoria di un volo Main prima della fase di normalizzazione in 50 punti geografici equidistanti nel tempo. I risultati mostrano che: in media una traiettoria è descritta da 21 punti, toccando un massimo di 124 punti ed un minimo di 1. Analizzando questi dati è possibile vedere come spesso le traiettorie, dopo la fase di normalizzazione, vengono descritte da un numero eccessivo di punti. Si lascia ad un approfondimento successivo lo studio su come il variare del numero di punti dopo la fase di normalizzazione possa influire sulle performance del modello esatto e dei metodi euristici sviluppati.

Date	AVG Traj Main	AVG Traj Eur	AVG Traj Def
07/08/2016	17.72	3.83	1
13/08/2016	17.61	3.84	1
26/08/2016	17.34	3.80	1
27/08/2016	17.55	3.84	1
28/08/2016	16.94	3.86	1
29/08/2016	17.62	3.84	1
30/08/2016	17.62	3.85	1
31/08/2016	17.66	3.84	1
01/09/2016	17.62	3.82	1
02/09/2016	17.64	3.83	1
09/09/2016	17.68	3.81	1
Media	17.55	3.83	1

Tabella 5.4: Numero medio di alternative di voli a tratta

Date	Min Traj Main	Min Traj Eur	Min Traj Def
08/07/2016	3	1	1
13/08/2016	2	1	1
26/08/2016	5	1	1
27/08/2016	2	1	1
28/08/2016	2	1	1
29/08/2016	3	1	1
30/08/2016	3	1	1
31/08/2016	3	1	1
01/09/2016	3	1	1
02/09/2016	3	1	1
09/09/2016	3	1	1
Media	2.91	1	1

Tabella 5.5: Numero minimo di alternative di voli effettuati a tratta

Date	Max Traj Main	Max Traj Eur	Max Traj Def
08/07/2016	47	4	1
13/08/2016	47	4	1
26/08/2016	44	4	1
27/08/2016	47	4	1
28/08/2016	47	4	1
29/08/2016	44	4	1
30/08/2016	44	4	1
31/08/2016	47	4	1
01/09/2016	47	4	1
02/09/2016	44	4	1
09/09/2016	44	4	1
Media	45.64	4	1

Tabella 5.6: Numero massimo di alternative di voli effettuati a tratta

Date	Weight Main	Weight Eur	Weight Default
08/07/2016	58.84%	39.20%	1.96%
13/08/2016	56.33%	41.35%	2.32%
26/08/2016	56.16%	41.75%	2.10%
27/08/2016	56.71%	40.90%	2.39%
28/08/2016	56.11%	41.54%	2.35%
29/08/2016	58.22%	39.76%	2.02%
30/08/2016	58.53%	39.42%	2.04%
31/08/2016	58.81%	39.17%	2.01%
01/09/2016	59.60%	38.39%	2.01%
02/09/2016	58.38%	39.65%	1.96%
09/09/2016	59.29%	38.82%	1.89%
Media	57.90%	40.00%	2.10%

Tabella 5.7: Percentuale dei tipi di volo nel numero finale di variabili nel modello

Capitolo 6

Prove computazionali

Nel capitolo vengono mostrati i risultati ottenuti dal confronto dei metodi euristici sviluppati per il problema ATFM col modello esatto presentato in §4.1.

Il lavoro principale, nella fase iniziale di questa tesi, ha riguardato l'analisi e il miglioramento dei lavori precedentemente sviluppati in [14, 15, 19, 20], i quali, prendendo in considerazione metodi diversi, risultano in parte separati l'uno dall'altro. In un primo periodo è stato dunque approfondito lo studio del codice dei diversi lavori, al fine di unificarli in un unico progetto organico, iniziando a sistemare le diverse cartelle che in origine erano nominate col nome degli autori dei lavori (Echerle, Loss, ...). Al momento, il lavoro è organizzato in un albero di cartelle, che descrive i metodi sviluppati (ricerca locale, generazione di colonne, ...). I diversi file di codice e le cartelle sono poi state documentate e commentate adeguatamente, in modo da facilitare il lavoro di comprensione ai prossimi sviluppatori.

Successivamente, il lavoro di tesi ha visto l'implementazione completa dei modelli PLI per il problema ATFM presentati in §4.1. Questo ha permesso il confronto delle prestazioni dei metodi euristici e dei solver PLI off-the-shelf.

Tutti i risultati presentati in questo capitolo prendono in esame le date ed i parametri descritti in §5.6.1, dunque un tempo di discretizzazione di 5 minuti e la presenza di 20 città di tipo Main.

6.1 Analisi del metodo basato su generazione di colonne

Dopo lo sviluppo e l'analisi dei diversi metodi descritti in §4.3, si è scelto di ragionare sulla loro efficienza ed efficacia, prima analizzando i risultati al variare dei parametri degli algoritmi, e poi confrontandoli coi risultati ottenuti dal modello esatto del problema.

I parametri considerati

Lo scopo di questa analisi è di migliorare le performance dell'algoritmo di generazione di colonne rispetto ai dati raccolti in [14], modificando tre parametri fondamentali:

- **Newcols:** è possibile stabilire attraverso questo parametro quante colonne aggiungere ad ogni iterazione. Newcols imposta la "velocità" di convergenza dell'algoritmo, perché, aggiungendo molte colonne ad ogni iterazione, si permette ad esso di raggiungere con una più alta probabilità l'ottimo. Viceversa, dall'altro lato, un modello con molte variabili richiede, ad ogni iterazione, più risorse per essere risolto;
- **Aging:** questo parametro stabilisce il numero massimo di iterazioni di permanenza di una variabile nel modello ristretto. Superata tale soglia, la colonna corrispondente viene eliminata. Questo parametro permette al modello di crescere in maniera controllata e che esso si diversifichi tanto o poco dalle iterazioni precedenti;
- **Ritardo:** questo parametro stabilisce se applicare la diversificazione dei ritardi descritta in §5.3 o meno. Per permettere al modello più elasticità nel programmare anche i voli di tipo EUR e DEFAULT, tra le diverse prove, è stato deciso di impostare a tutti i voli lo stesso numero di possibili ritardi, nello specifico 24 TU.

Descrizione delle tabelle

Le misurazioni eseguite in [10] verificano inizialmente il tempo di esecuzione delle diverse fasi dell'algoritmo, che si dividono principalmente in: ottenimento della soluzione rilassata nel modello costruito con la generazione di colonne (CG) e la fase di Branch & Cut (BC) per l'ottenimento di una soluzione intera ammissibile. Dopodiché, sono state misurate le soluzioni del rilassamento continuo ottenute nelle due diverse esecuzioni, chiamate rispettivamente *Upper Buond* (per la massimizzazione delle preferenze) o *Lower Bound* (per la minimizzazione del ritardo) e le rispettive soluzioni intere, chiamate entrambe *Object*.

Per verificare con quale velocità e precisione l'algoritmo raggiunge l'Upper/Lower Bound, sono state eseguite le misurazioni con due diversi livelli di tolleranza, per la precisione, del 3% e dell'1%. La tolleranza permette che l'algoritmo di Branch & Cut si interrompa quando il numero ottenuto da $|\text{Soluzione rilassata} - \text{Object}|/(1e{-}10 + |\text{Object}|)$ raggiunge il valore impostato.

Come accennato in §4.1, per ottenere il budget di ritardo (il parametro R) usato per il calcolo della massimizzazione delle preferenze, viene calcolata prima la minimizzazione del ritardo.

Le intestazioni delle righe delle tabelle riportano i parametri utilizzati (Newcols,Aging). Le altre diciture sono:

- **CG:** dati riguardanti la risoluzione del problema rilassato durante la fase di generazione di colonne;
- **BC3%:** dati riguardanti la fase di Branch & Cut con una tolleranza del 3%;
- **BC1%:** dati riguardanti la fase di Branch & Cut con una tolleranza del 1%;

- **Tot3%:** dati riguardanti le tempistiche totali dell'algoritmo, considerando la tolleranza al 3%;
- **Tot1%:** dati riguardanti le tempistiche totali dell'algoritmo, considerando la tolleranza al 1%;
- **T(m):** tempo totale richiesto dall'algoritmo espresso in minuti;
- **W:** peso di computazione (weight) che tale fase ha sull'intero algoritmo in termini di tempistiche;
- **W1%:** peso di computazione (weight) che ha la fase CG con una tolleranza al 3% in termini di tempistiche;
- **W1%:** peso di computazione (weight) che ha la fase CG con una tolleranza al 1% in termini di tempistiche;
- **Budget:** il ritardo massimo consentito per il calcolo della massimizzare delle preferenze;
- **N° Var:** numero di variabili che possedeva il modello al termine dell'esecuzione;
- **LB:** soluzione rilassata del problema di minimizzazione (lower bound). Espressa in TU;
- **UB:** soluzione rilassata del problema di massimizzazione (upper bound). Espressa in TU;
- **Objective 3%:** soluzione intera del problema con la tolleranza al 3%. Espressa in TU per la minimizzazione;
- **Objective 1%:** soluzione intera del problema con la tolleranza al 1%. Espressa in TU la minimizzazione;
- **Gap 3%:** errore relativo con una tolleranza al 3%;
- **Gap 1%:** errore relativo con una tolleranza al 1%.

Tutti i tempi calcolati sono espressi in minuti (m), i ritardi in TU, le preferenze ottenute come semplici valori numerici e infine i gap e l'impatto di ogni fase in percentuale. Per motivi di spazio nell'impaginazione, solo per alcuni valori le tabelle riportano l'unità di misura.

Per considerare i dati ottenuti rilevanti, è stato impostato un *time limit* che impone alla risoluzione del modello con variabili intere, ma non all'intero algoritmo, di non poter superare tale parametro in termini di tempo di esecuzione. In tutte le prove effettuate si ha *time limit* = 120 minuti (2 ore), eccetto che per la coppia di parametri (40,10) che riguarda risultati ottenuti in [14], il quale non riporta per la minimizzazione del ritardo i tempi di esecuzione delle diverse fasi e i risultati con la tolleranza al 3%.

	CG			BC3%		Tot3%	BC1%		Tot1%
(N,A)	T(m)	W3%	W1%	T(m)	W	T(m)	T(m)	W	T(m)
(40,10)	-	-	-	-	-	-	-	-	470.52
(750,7)	13.87	87%	67%	3.39	13%	15.41	12.01	33%	24.03
(1000,2)	7.73	83%	72%	13.60	17%	20.33	19.73	28%	26.46
(1000,3)	8.58	82%	71%	3.64	18%	12.22	9.41	29%	17.99
(1000,4)	9.93	88%	56%	1.31	12%	9.87	10.18	44%	18.74
(1000,5)	12.03	91%	71%	1.14	9%	11.25	5.80	29%	14.84
(1500,2)	13.30	95%	91%	2.72	5%	16.02	6.45	9%	19.76
(1500,3)	9.60	88%	76%	2.80	12%	12.40	5.46	24%	13.61
(2000,2)	8.46	86%	71%	4.86	14%	13.31	5.81	29%	13.12

I dati non presenti in [14] sono riportati con "-".

Tabella 6.1: Tempi della minimizzazione del ritardo sull'euristica basata su generazione di colonne

(N,A)	LB (TU)	Object 3% (TU)	Gap 3%	Object 1%	Gap 1%
(40,10)	6320.91	-	-	6359.20	0.60%
(750,7)	6320.91	6263.56	1.83%	6169.44	0.33%
(1000,2)	6320.91	6253.78	1.68%	6174.11	0.40%
(1000,3)	6320.91	6435.20	1.73%	6308.00	0.48%
(1000,4)	6320.91	6279.56	2.06%	6171.78	0.37%
(1000,5)	6320.91	6255.50	1.89%	6079.33	0.39%
(1500,2)	6034.49	6134.10	1.62%	6059.70	0.39%
(1500,3)	6320.91	6448.70	1.94%	6100.25	0.37%
(2000,2)	6320.91	6450.40	1.95%	6180.78	0.51%

I dati non presenti in [14] sono riportati con "-".

Tabella 6.2: Risultati della minimizzazione del ritardo sull'euristica basata su generazione di colonne

Analisi dei risultati

Le seguenti tabelle riportano la media dei risultati di quelle presenti in Appendice A, ottenute da [10]. È dunque normale che dalle Tabelle 6.2, 6.4 risulti che alcuni valori di UB siano più bassi dei valori Object, o viceversa, che valori di LB siano più alti. Inoltre, l'esecuzione dell'istanza del 27/08/2016 non è quasi mai stata risolta nel tempo limite e nel caso in cui questo non fosse impostato, restituiva valori anomali.

I risultati della generazione di colonne coi parametri (40,10) sono riportati da [14].

Il primo dato che si nota osservando le Tabelle 6.1 e 6.3, è la differenza in termini di tempo tra l'esecuzione con una bassa Newcols ed una più alta. Come indicato precedentemente, tale parametro permette di esplorare più o meno velocemente lo spazio delle soluzioni.

Seguendo l'ipotesi di voler generare un modello che potesse cambiare velocemente e spesso nel corso dell'esecuzione, oltre ad alzare di molto NewCols, è stato abbassato il parametro Aging, mostrando un sensibile miglioramento delle performance, come si può notare in Tabella 6.2 alla voce Gap 1%.

Dai risultati emerge che, tra i valori scelti di Newcols, restituisce i migliori risultati l'esecuzione col valore 1500, oltre il quale questi peggiorano, senza avere una significativo

	CG			BC3%		Tot3%	BC1%		Tot1%
(N,A)	T(m)	W3%	W1%	T(m)	W	T(m)	T(m)	W	T(m)
(40,10)	104.5	95.3%	77%	10.9	4.7%	115.5	60.7	23%	165.2
(100,4)	43.4	93.1%	74%	5.9	6.9%	49.4	29.7	26%	65.2
(500,4)	29.3	99.5%	69%	0.1	0.5%	29.4	27.5	31%	51.4
(1000,3)	25.9	99.3%	65%	0.2	0.7%	26.0	23.3	35%	42.9
(1500,2)	22.4	99.2%	66%	0.2	0.8%	22.6	31.9	34%	50.3
(2000,2)	23.1	99.0%	72%	0.2	1.0%	22.5	17.1	28%	35.0

Tabella 6.3: Tempi della massimizzazione delle preferenze sull'euristica basata su generazione di colonne

(N,A)	Budget	N° Var	UB	Object3%	Gap3%	Object1%	Gap1%
(40,10)	6995	51474	23623*	23258.2	1.57%	23506.5	0.50%
(100,4)	7006	52033	23630	23230.7	1.73%	23791.7	0.50%
(500,4)	7006	60535	23630	23178.2	1.96%	23791.2	0.50%
(1000,3)	7006	65620	23630	23156.8	2.05%	23945.9	0.49%
(1500,2)	7006	68679	23630	23006.0	2.81%	23792.3	0.50%
(2000,2)	7006	71771	23630	23274.1	1.98%	23554.8	0.78%

* Il valore è diverso perché riferito ad un diverso budget di ritardo.

Tabella 6.4: Risultati della massimizzazione delle preferenze sull'euristica basata su generazione di colonne

miglioramento delle tempistiche.

Un discorso diverso avviene invece quando si calcola la massimizzazione delle preferenze, in Tabella 6.3, che in genere richiede più tempo per essere eseguita rispetto alla massimizzazione.

L'errore relativo Gap 1% in Tabella 6.4 mostra come sia più stabile il risultato rispetto ai valori in Tabella 6.2, nonostante l'aumento del numero di variabili ad ogni iterazione all'aumentare di Newcols. Perciò, la strategia che consiste nell'alzare molto Newcols e diminuire Aging, in questo caso non risulta migliorativa in termini di performance. È necessario considerare il trade-off dei tempi di esecuzione, che rende l'utilizzo di parametri con alti Newcols e bassi per Aging decisamente preferibili.

Confrontando i risultati ottenuti in Gap 3% e Gap 1% nelle rispettive Tabelle 6.2 e 6.4, con i tempi di esecuzione BC 3% e BC 1% nelle Tabelle 6.1 e 6.3, si osserva un miglioramento relativo dell'1-1.5% per un 20-30% del tempo in più. Un investimento in termini di tempo ragionevole, considerando che si è nell'ordine dei minuti.

Da queste analisi concludiamo che la coppia di parametri (1500,3) con una tolleranza dell'1% sono i migliori per la minimizzazione del ritardo, bilanciando risultati e tempi di esecuzione, mentre per la massimizzazione delle preferenze la coppia di parametri (500,4) con una tolleranza dell'1%.

Come è possibile vedere all'Appendice A, i risultati delle singole istanze variano non poco tra loro. Si lascia dunque a sviluppi futuri individuare le ragioni per cui la conformazione di certe istanze richieda più tempo di altre nell'essere risolte.

6.2 Analisi del modello esatto

Come prevedibile, l'esecuzione dell'algoritmo per la risoluzione del modello descritto in §4.1, non ha mai, eccetto una volta, restituito una soluzione completa, a causa dell'eccessiva domanda di risorse che l'algoritmo richiede.

Nella maggior parte dei casi, l'algoritmo restituisce una soluzione ammissibile, ma certamente non la soluzione ottima, anzi, molto discostata dai risultati degli altri metodi.

Per analizzare le prestazioni dell'algoritmo, sono state eseguite diverse prove, cercando di modificare i possibili ritardi che ogni tipo di volo può avere. Le valutazioni sono state effettuate per tre configurazioni:

1. **Configurazione normale:** il modello viene eseguito senza alcun cambiamento dei valori di default dei parametri. Dunque, l'insieme dei ritardi rimane diversificato per i diversi tipi di volo e il budget per la massimizzazione delle preferenze è il risultato della minimizzazione dei ritardi;

2. **Configurazione con un budget di ritardo maggiorato:** la massimizzazione delle preferenze considera i valori di budget provenienti dalla minimizzazione del ritardo con l'euristica basata sulla generazione di colonne coi parametri (1500,2) aumentati del 10%;

3. **Configurazione con i ritardi unificati:** oltre ai cambiamenti apportati al punto precedente, viene considerato lo stesso numero di possibili ritardi per ogni tipologia di volo, e cioè 24 TU (2 ore) a intervalli di 1 TU.

Descrizione delle tabelle

Come per la sezione precedente, è stato impostato un *time limit* di 120 minuti. I risultati evidenziati in giallo indicano il superamento di tale soglia o l'insufficienza della memoria RAM. Essi non sono dunque da considerare nell'analisi d'insieme. I dati che non è stato possibile ottenere sono segnati con un "-".

Per questa analisi, è stata utilizzata una tolleranza del 1%, soglia comunque mai raggiunta, dato che l'algoritmo veniva interrotto prima a causa del time limit.

Poiché il valore Gap 1% viene registrato come il valore soglia (in questo caso dell'1%), quando l'esecuzione non terminava correttamente non è stato possibile avere un'analisi qualitativa dei risultati, come avvenuto invece in §6.1. I valori raccolti sono però significativi per il confronto col metodo di generazione di colonne.

Le diciture delle tabelle indicano:

- **ConsT:** tempo in minuti impiegato per la costruzione del modello;
- **SolvT:** tempo in minuti impiegato per la risoluzione del modello;
- **TotT:** tempo in minuti impiegato per l'esecuzione dell'intero algoritmo, esso comprende le due fasi precedenti, più la fase di impostazione dei parametri;
- **N° Var:** numero di variabili presenti nel modello. Valore espresso in milioni;

Data	ConsT(m)	SolvT(m)	TotT(m)	N° Var(Mln)	Obj1%	RAM(GB)
08/07	0.50	122.05	124.70	1.45	10550	31.88
26/08	0.42	122.72	125.45	1.35	36759	31.95
27/08	0.38	123.03	125.44	1.22	34158	31.82
28/08	0.42	122.73	125.20	1.28	11338	31.81
29/08	0.42	122.85	125.60	1.37	10689	31.93
30/08	0.45	123.20	125.80	1.37	10459	31.77
31/08	0.45	121.30	123.94	1.42	9719	31.81
01/09	0.45	122.83	125.58	1.45	35968	31.95
02/09	0.47	122.98	126.15	1.42	9929	31.91
09/09	0.45	123.32	126.04	1.45	36790	31.77
Media	0.44	122.70	125.39	1.38	20635.9	31.86

Tabella 6.5: Minimizzazione del ritardo per la configurazione normale del modello esatto

Data	ConsT(m)	SolvT(m)	TotT(m)	N° Var(Mln)	Obj1%	RAM(GB)
08/07	0.50	122.62	125.34	1.45	10550	31.94
26/08	0.42	122.80	125.53	1.35	36759	31.91
27/08	0.38	123.12	125.49	1.22	34158	31.75
28/08	3.37	-	16.41	1.28	-	30.01
29/08	0.42	123.08	125.68	1.37	10689	31.92
30/08	0.43	123.15	125.74	1.37	10459	31.82
31/08	0.45	123.18	125.77	1.42	9719	31.88
01/09	0.45	123.30	126.03	1.45	35968	31.91
02/09	0.47	123.50	126.26	1.42	9929	31.79
09/09	0.45	123.42	126.06	1.45	36790	31.86
Media	0.73	123.13	125.77	1.38	21669	31.68

Le misurazioni di cui non è stato possibile registrare il valore per il superamento del *time limit* sono riportate col simbolo "-".

Tabella 6.6: Minimizzazione del ritardo con la configurazione col budget maggiorato del modello esatto

- **Budget:** ritardo massimo consentito per la massimizzazione delle preferenze;
- **Obj1%:** valore della miglior soluzione ammissibile trovata al termine dell'algoritmo. Valore espresso in TU per la minimizzazione;
- **RAM:** memoria RAM utilizzata per l'esecuzione dell'algoritmo. Si ricorda che le macchine su cui girava l'algoritmo possiedono 32 GB di RAM.

Analisi dei risultati

Tutti i valori presenti alla dicitura Obj1%, nella Tabelle 6.5, 6.6, 6.7, 6.8, 6.9 e 6.10, derivano da un calcolo interrotto dopo 2 ore di esecuzione, dunque rappresentano una soluzione parziale. L'unica esecuzione conclusasi correttamente è evidenziata in verde e corrisponde alla massimizzazione delle preferenze con il budget maggiorato dell'istanza del 09/09/2016, ottenendo un Gap dello 0.31%.

L'esecuzione del modello esatto ha dimostrato che i requisiti necessari in termini di risorse computazionali sono al momento troppo alti, infatti è possibile notare nelle

Data	ConsT(m)	SolvT(m)	TotT(m)	N° Var(Mln)	Obj1%	RAM(GB)
08/07	1.45	-	122.26	4.41	-	31.92
26/08	1.28	-	130.30	4.19	-	31.87
27/08	1.12	198.45	201.65	3.76	41173	31.74
28/08	1.22	-	102.31	3.95	-	31.84
29/08	1.32	184.65	188.63	4.21	41685	31.93
30/08	1.33	-	111.48	4.20	-	31.90
31/08	1.35	-	109.11	4.33	-	31.81
01/09	1.35	-	107.34	4.39	-	31.82
02/09	1.37	11.22	14.01	4.36	-	31.74
09/09	1.33	11.77	14.60	4.42	-	31.79
Media	1.31	191.55	195.14	4.22	41429	31.84

Le misurazioni di cui non è stato possibile registrare il valore per il superamento del *time limit* sono riportate col simbolo "-".

Tabella 6.7: Minimizzazione del ritardo con la configurazione con i ritardi unificati del modello esatto

Data	ConsT(m)	SolvT(m)	TotT(m)	N° Var	Budget	Obj1%	RAM
08/07	0.50	122.05	124.68	1.45	10550	23109.4	31.92
26/08	0.42	123.82	126.50	1.35	36759	11909	31.92
27/08	0.38	123.85	126.22	1.22	34158	10621.4	31.85
28/08	0.42	122.30	124.72	1.28	11338	21579.9	31.86
29/08	0.42	123.08	125.72	1.37	10689	12664.8	31.92
30/08	0.43	122.68	125.35	1.37	10459	22425.2	31.81
31/08	0.47	123.12	126.22	1.42	9719	12717.7	31.85
01/09	0.45	124.03	126.60	1.45	35968	11768	31.90
02/09	0.47	123.38	126.23	1.42	9929	12905.9	31.91
09/09	0.45	124.05	126.71	1.45	36790	11629.8	31.83
Media	0.44	123.24	125.89	1.38	20635.9	15133.1	31.88

Tabella 6.8: Massimizzazione delle preferenze per la configurazione normale del modello esatto

Data	ConsT(m)	SolvT(m)	TotT(m)	N° Var	Budget	Obj1%	RAM
08/07	0.50	121.57	124.20	1.45	6376	22963.3	31.89
26/08	0.42	122.62	125.26	1.35	6524	13260	31.93
27/08	0.38	122.25	124.61	1.22	8336	19799.8	31.83
28/08	0.42	122.55	124.99	1.28	6993	12391.7	31.84
29/08	0.42	122.25	124.82	1.37	6939	22598.4	31.89
30/08	0.43	122.25	124.85	1.37	5677	22305.2	31.79
31/08	0.47	122.48	125.03	1.42	6652	22479	31.84
01/09	0.45	123.08	125.78	1.45	6437	13172.2	31.92
02/09	0.47	123.40	126.27	1.42	6472	23073.7	31.84
09/09	0.45	115.45	118.27	1.45	6251	23746.4	31.89
Media	0.44	121.79	124.41	1.38	6665.7	19578.97	31.87

Tabella 6.9: Massimizzazione delle preferenze con la configurazione col budget maggiorato del modello esatto

Data	ConsT(m)	SolvT(m)	TotT(m)	N° Var	Budget	Obj1%	RAM
08/07	1.45	15.06667	17.907	4.41	6376	-	31.95
26/08	1.28	-	107.633	4.19	6524	-	31.90
27/08	1.12	-	225.3723	3.76	8336	-	31.78
28/08	1.22	-	103.0048	3.95	6993	-	31.84
29/08	1.32	-	108.0883	4.21	6939	-	31.94
30/08	1.30	-	108.1363	4.20	5677	-	31.91
31/08	1.35	-	99.89417	4.33	6652	-	31.81
01/09	1.33	-	112.064	4.39	6437	-	31.81
02/09	1.37	9.866667	12.41767	4.36	6472	-	31.80
09/09	1.35	10.36667	12.92	4.42	6251	-	31.80
Media	1.31	11.76667	90.74377	4.22	6665.7	-	31.85

Le misurazioni di cui non è stato possibile registrare il valore per il superamento del *time limit* sono riportate col simbolo "-".

Tabella 6.10: Massimizzazione delle preferenze con la configurazione con i ritardi unificati del modello esatto

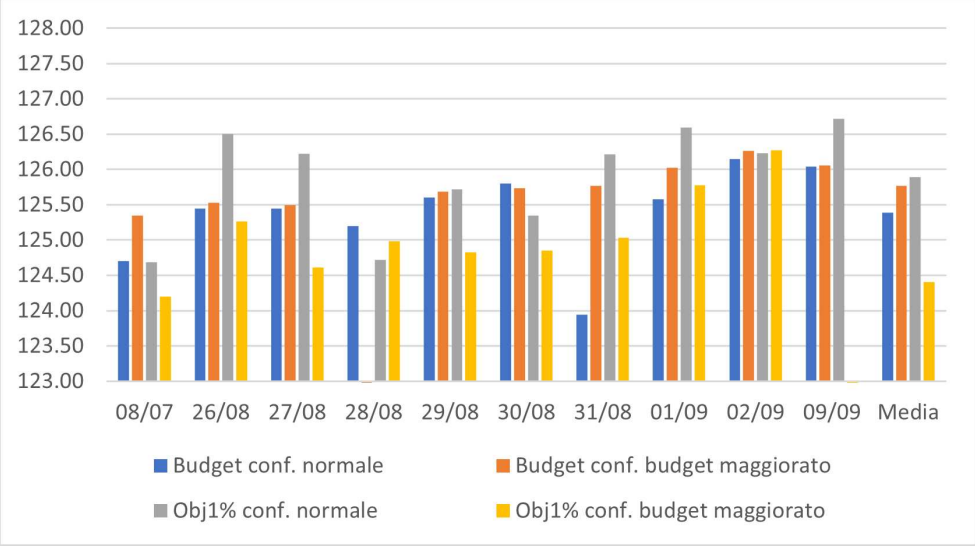


Figura 6.1: Confronto dei tempi della massimizzazione delle preferenze tra la configurazione normale e quella col budget maggiorato del modello esatto

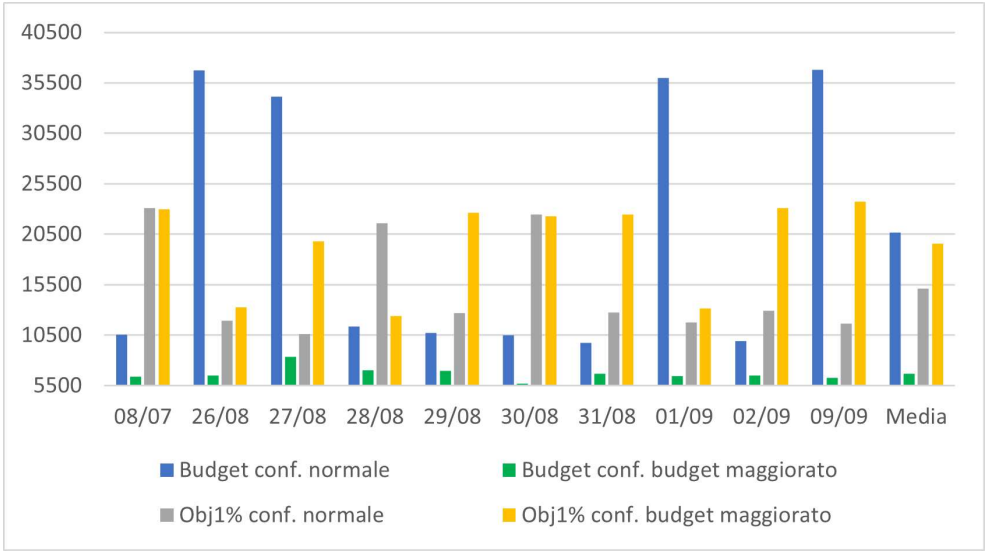


Figura 6.2: Confronto dei risultati della massimizzazione delle preferenze tra la configurazione normale e quella col budget maggiorato del modello esatto

diverse tabelle come si raggiunga sempre il time limit, o talvolta ci sia un'interruzione per l'esaurimento della RAM disponibile in meno di 2 ore, come nel caso del ritardo unificato.

L'ipotesi dietro la scelta di assegnare ad ogni tipo di volo lo stesso numero di possibili ritardi nasce dalla volontà di permettere al modello maggiore libertà per l'assegnamento dei voli alle diverse traiettorie. I risultati purtroppo mostrano che il numero di variabili da considerare è troppo alto (più del doppio delle altre esecuzioni), come si vede nelle Tabelle 6.7 e 6.10, portando invece ad un'anomala interruzione dell'algoritmo.

È possibile vedere, dalle Tabelle 6.5, 6.6, 6.8 e 6.9, come il risolutore CPLEX lavori diversamente in base al valore, in questo caso, del budget, nonostante il fattore più influente dovrebbe essere il numero di variabili, che rimane invariato nelle diverse prove, come si può vedere nelle Tabelle 6.8 e 6.9. Osservando infatti la Figura 6.2, che si concentra su alcuni dati nelle Tabelle 6.8 e 6.9, è evidente che il budget nella configurazione normale (in blu) sia decisamente maggiore del budget nella configurazione col budget maggiorato (in verde). Questo si è verificato perché l'esecuzione della minimizzazione del ritardo con la configurazione normale non viene completata, e dunque il risultato, passato come budget alla massimizzazione delle preferenze, è parziale. Invece, dall'altra parte, il budget della massimizzazione delle preferenze con la configurazione col budget maggiorato, proviene dall'esecuzione dalla minimizzazione del ritardo calcolato con la generazione di colonne, che, al contrario, termina. Considerando che il modello con un budget maggiore è meno vincolato, i valori Obj1% nella configurazione col budget maggiorato risultano comunque più alti (in giallo), e dunque migliori, rispetto a quelli nella configurazione normale (in grigio). A livello di tempistiche, mostrate in Figura 6.1, dato che la differenza è sull'ordine dei secondi, il confronto è irrilevante.

Analisi del modello esatto senza cancellazioni

Poiché la maggior discriminante che condiziona la conclusione o meno dell'esecuzione del metodo esatto è il numero di variabili, sono stati eseguiti dei test con una versione differente del modello in §4.1, presente in [19, 15]. Tale versione non prevede la presenza della variabile decisionale z, la cui funzione è di modellare la possibilità di cancellare i voli o, in altre parole, di evitare la pianificazione obbligatoria di tutti i voli. La variabile, introdotta inizialmente per consentire sempre una soluzione ammissibile dei problemi rilassati ristretti durante la procedura di generazione di colonne, non è invece necessaria per l'applicazione dei risolutori.

Considerato che in §6.2 i migliori risultati sono stati ottenuti dall'esecuzione della massimizzazione delle preferenze con la configurazione col budget maggiorato, è stato deciso di mantenere gli stessi parametri per l'esecuzione della massimizzazione delle preferenze con la variante del modello, vale a dire, mantenere i ritardi diversificati per tipologia di volo e il budget di ritardo equivalente ai risultati dell'esecuzione dell'euristica basata sulla generazione di colonne coi parametri (1500,2) aumentati del 10%.

Siccome il numero di variabili nella nuova versione del modello è ridotto solo del 2.3%, i risultati non sono variati rispetto ai valori ottenuti della massimizzazione delle preferenze con la configurazione col budget maggiorato.

Nelle specifico, il numero di variabili in questa versione del modello equivale a:

$$\textit{“N° di variabili nel modello in §4.1”} - \textit{“N° di voli”} - 1$$

Riteniamo dunque che, per il risolutore esatto, l'utilizzo di questa variante del modello non alteri in modo significativo i risultati finali.

6.3 Confronto del modello esatto con l'euristica basata su generazione di colonne

Descrizione delle tabelle

Le Tabelle 6.11, 6.12 e 6.13 riassumono i risultati dell'euristica della generazione di colonne, presenti in §6.1, e le medie dei risultati ottenuti dal modello esatto, presenti in §6.2. Le diverse configurazioni del modello esatto sono riportate con le seguenti sigle:

- **Normal:** configurazione normale;
- **MaxBudget:** configurazione col budget maggiorato;
- **UniqueDelay:** configurazione con i ritardi unificati.

Analisi dei risultati

Confrontando i valori alle Tabelle 6.11 e 6.12, è indubbio che il modello esatto, nella seconda parte delle tabelle, non possa essere eseguito nelle macchine odierne, nonostante si provi a modificare i parametri per migliorarne le performance.

(N,A)	TotT 1% (m)	Object 1% (TU)	LB (TU)	Gap 1%
(40,10)	470.52	6359.20	6320.91	0.60%
(750,7)	24.03	6169.44	6320.91	0.33%
(1000,2)	26.46	6174.11	6320.91	0.40%
(1000,3)	17.99	6308.00	6320.91	0.48%
(1000,4)	18.74	6171.78	6320.91	0.37%
(1000,5)	14.84	6079.33	6320.91	0.39%
(1500,2)	19.76	6059.70	6034.49	0.39%
(1500,3)	13.61	6100.25	6320.91	0.37%
(2000,2)	13.12	6180.78	6320.91	0.51%
Configurazione				
Normal	125.39	20635.90	6320.91	69.37%
MaxBudget	125.77	21669.00	6320.91	70.83%
UniqueDelay	195.14	41429.00	6320.91	84.74%

Tabella 6.11: Confronto della minimizzazione del ritardo tra il modello esatto e l'euristica basata su generazione di colonne

(N,A)	N° Var	Budget	TotT(m)	Object 1%	UB	Gap1%
(40,10)	51474	6995	165.1813	23506.45	23623	0.50%
(100,4)	52033	7006	65.16265	23791.69	23630	0.50%
(500,4)	60535	7006	51.36952	23792.22	23630	0.50%
(1000,3)	65620	7006	42.91792	23945.94	23630	0.49%
(1500,2)	68679	7006	50.33207	23792.33	23630	0.50%
(2000,2)	71771	7006	35.03167	23554.8	23630	0.78%
Configurazione						
Normal	1.38E+06	20635.90	125.89	15133.11	23630	56.15%
MaxBudget	1.38E+06	6665.70	124.41	19578.97	23630	20.69%
UniqueDelay	4.22E+06	6665.70	90.74	-	23630	-

Le misurazioni di cui non è stato possibile registrare il valore per il superamento del *time limit* sono riportate col simbolo "-".

Tabella 6.12: Confronto della massimizzazione delle preferenze tra il modello esatto e l'euristica basata su generazione di colonne

Se si osserva invece il confronto per l'unica esecuzione del modello esatto conclusasi correttamente, in Tabella 6.13, ovvero quella con l'istanza del 09/09/2016, si può notare un'anomalia tra i valori di UB e Object. Infatti, tali valori del modello esatto dovrebbero essere più alti rispetto agli altri. Si rimanda dunque a un'analisi futura il motivo di tale questione.

Dunque per il problema ATMF, allo stato attuale delle cose, i metodi euristici sono la miglior soluzione.

(N,A)	N° Var	Budget	TotT(m)	Object 1%	UB	Gap1%
(40,10)	53724	6565	78.3	24065.3	24203.1	0.57%
(100,4)	53552	6505	31.8	24009.8	24162.4	0.64%
(500,4)	63074	6505	21.0	24023.6	24162.4	0.58%
(1000,3)	68601	6505	29.7	24139.6	24162.4	0.09%
(1500,2)	72494	6505	14.3	24027.1	24162.4	0.56%
(2000,2)	76851	6505	54.0	24136.9	24162.4	0.11%
Configurazione						
MaxBudget	1.45E+06	6251	118.27	23746.4	23820.5	0.31%

Tabella 6.13: Confronto dei risultati del 09/09/2016 tra il modello esatto e l'euristica basata su generazione di colonne

Capitolo 7

Conclusioni

Nel capitolo viene riassunto il lavoro svolto durante la tesi, i risultati ottenuti e i contributi con indicazioni di possibili sviluppi futuri.

La tesi ha sviluppato e migliorato alcuni precedenti lavori sulla messa a punto di modelli e metodi per il problema ATFM basati su matheuristiche e machine learning. Le principali attività sono state relative allo studio del lavoro precedentemente svolto e all'analisi di una ristrutturazione e razionalizzazione, in prospettiva di progetti futuri. Inoltre, la tesi propone l'implementazione di un metodo esatto basato sulla soluzione diretta di modelli matematici di PLI con solver off-the-shelf, permettendo di valutarne i limiti e di confrontare le performance dei metodi euristici disponibili.

Un importante contributo della tesi è il miglioramento del processo di creazione delle istanze, i cui risultati sono stati decisamente positivi. L'analisi delle tempistiche delle diverse fasi dell'algoritmo ha permesso un loro miglioramento mirato, in particolar modo individuando e risolvendo il "collo di bottiglia" che rendeva decisamente lunghi alcuni processi, quali ad esempio quelli relativi alla fase di clusterizzazione. La soluzione adottata consiste nel gestire al meglio le chiamate al database attraverso il salvataggio di risultati parziali, che però richiedono 7.7 GB di ulteriore spazio nel disco, che si aggiungono ai precedenti 44 GB: il compromesso è però decisamente a vantaggio della nuova implementazione. Un ulteriore miglioramento è derivato dall'analisi e dalla razionalizzazione delle operazioni effettuate, che ha portato ad eliminare alcune elaborazioni superflue: ad esempio, la creazione degli alberi di decisione per l'attribuzione delle preferenze, inizialmente, veniva fatta ad ogni esecuzione dell'algoritmo, ora invece solo all'occorrenza, evitando elaborazioni ridondanti.

La seconda parte del lavoro di tesi è dedicata alla verifica computazionale di metodi esatti basati sull'implementazione dei modelli PLI e sulla loro risoluzione con solver allo stato dell'arte. In questo ambito, il contributo della tesi è l'implementazione dei modelli presentati in letteratura tramite le API di CPLEX, un solver per PLI allo stato dell'arte. Purtroppo, i risultati hanno evidenziato la difficoltà di ottenere una soluzione ottima, o anche solo accettabile, per il problema ATFM, anche se alcune esecuzioni dimostrano che l'eccezione è possibile.

Dal lavoro di questa tesi sono emerse alcune indicazioni per possibili futuri approfondimenti:

- **Sistemazione dell'algoritmo di generazione delle colonne:** il codice sorgente per l'esecuzione di questo metodo è quasi tutto incentrato in un unico file e poche funzioni. Rispettando i principi appresi ad Ingegneria del Software, sarebbe buona norma riorganizzare la struttura dei file sorgente e documentarla nel modo più appropriato. Questo aprirebbe anche alla possibilità di ottimizzare il codice e implementare alcune funzionalità in modo più efficiente;

- **Studio della struttura delle istanze:** alcune istanze sono risultate più difficili da risolvere di altre. La motivazione probabilmente risiede nella loro struttura: il numero dei voli e delle loro traiettorie, l'esistenza di zone più trafficate o meno percorribili. Lo studio di questi fattori potrebbe favorire la comprensione dell'efficacia di certi metodi su determinati tipi di istanze, oppure delle specifiche strategie in base alla loro struttura;

- **Parallelizzazione delle operazioni di accesso al database:** come spiegato in §5.6, attualmente non è possibile eseguire più scritture in parallelo nel database, anche quando esse non interferiscono. Trovare una strategia per implementare questa caratteristica gioverebbe in termini di tempo e memoria del disco per la generazione delle istanze;

- **Numero di punti nella fase di normalizzazione delle traiettorie:** in prospettiva, si potrebbe verificare se il numero di punti con cui viene descritta ogni traiettoria può essere cambiato per migliorare eventualmente l'efficienza del processo di creazione delle istanze, dei metodi euristici e del metodo esatto.

Appendice A

Risultati dell'euristica basata su generazione di colonne

Le tabelle mostrate in questa appendice riportano nel dettaglio i risultati presentati in [10] e mostrati sinteticamente in §6.1, riguardanti le esecuzioni dell'euristica su generazione di colonne con diversi parametri. Come già detto, tali parametri sono: il numero di nuove colonne aggiunte ad ogni iterazione (*Newcols* o *NCols*) e il numero di interazioni in cui una variabile può essere considerata nel modello (Aging). L'algoritmo è stato eseguito in una macchina con Intel® Xeon E-2176G™ @ 3.7 GHz 6 cores e 16 GB di RAM.

Le diciture nelle tabelle sono le medesime riportate in §6.1.

Quando le esecuzioni che entro il tempo limite di due ore non hanno trovato una soluzioni ammissibile, come per l'istanza del 27/08/2016, non sono segnate (caso evidenziato dal simbolo "-"), o, se sono stati trovati valori che superano di poco il time limit, essi sono segnati in giallo e comunque non conteggiati per il calcolo del valore medio del dato.

A.1 Minimizzazione del ritardo

Di seguito vengono riportati i valori di riferimento dei parametri nelle diverse tabelle:

Tabella A.1: NCols = 40, Aging = 10;

Tabella A.2 e A.3: NCols = 750, Aging = 7;

Tabella A.4 e A.5: NCols = 1000, Aging = 2;

Tabella A.6 e eA.7: NCols = 1000, Aging = 3;

Tabella A.8 e A.9: NCols = 1000, Aging = 4;

Tabella A.10 e eA.11: NCols = 1500, Aging = 5;

Tabella A.12 e A.13: NCols = 1500, Aging = 2;

Tabella A.14 e A.15: NCols = 1500, Aging = 3;

Tabella A.16 e A.17: NCols = 2000, Aging = 2.

Date	LB (TU)	Objective 1% (TU)	Gap 1%	Time(m)
08/07	6030.2	6093	1.03%	63.3
26/08	6253.1	6269	0.25%	55.8
27/08	7868.6	7942	0.92%	3969.1
28/08	6721.9	6770	0.71%	165.3
29/08	6527.1	6557	0.46%	64.3
30/08	5365.5	5400	0.64%	67.1
31/08	6293.8	6307	0.21%	67.6
01/09	6106.0	6145	0.63%	57.2
02/09	6134.3	6141	0.11%	55.7
09/09	5908.5	5968	1.00%	139.9
Media	6320.9	6359	0.60%	470.5

Tabella A.1: Risultati della minimizzazione del ritardo in [14] con NCols = 40 e Aging = 10

	CG			BC3%		Tot3%	BC1%		Tot1%
Date	T(m)	W3%	W1%	T(m)	W	T(m)	T(m)	W	T(m)
08/07	15.05	89%	23%	1.83	11%	16.88	49.61	77%	64.66
26/08	12.12	80%	78%	3.07	20%	15.19	3.45	22%	15.56
27/08	30.52	-	-	-	-	-	-	-	-
28/08	17.88	45%	37%	21.85	55%	39.74	30.77	63%	48.65
29/08	12.15	92%	92%	1.04	8%	13.19	1.10	8%	13.25
30/08	11.75	94%	59%	0.74	6%	12.49	8.10	41%	19.85
31/08	10.35	96%	67%	0.45	4%	10.80	5.03	33%	15.38
01/09	10.70	96%	54%	0.48	4%	11.18	9.03	46%	19.73
02/09	7.63	95%	95%	0.43	5%	8.07	0.43	5%	8.07
09/09	10.53	95%	95%	0.60	5%	11.13	0.60	5%	11.13
Media	13.87	87%	67%	3.39	13%	15.41	12.01	33%	24.03

Le misurazioni di cui non è stato possibile registrare il valore per il superamento del *time limit* sono riportate col simbolo "-".

Tabella A.2: Tempi della minimizzazione del ritardo con NCols = 750 e Aging = 7

Date	LB (TU)	Object 3% (TU)	Gap 3%	Object 1% (TU)	Gap 1%
08/07	6030.2	6179	2.41%	6040	0.16%
26/08	6253.1	6405	2.37%	6262	0.14%
27/08	7869.6	-	-	-	-
28/08	6722.9	6800	1.15%	6788	0.97%
29/08	6527.1	6687	2.39%	6541	0.21%
30/08	5366.5	5511	2.64%	5399	0.62%
31/08	6294.8	6471	2.74%	6305	0.18%
01/09	6106.0	6252	2.34%	6123	0.28%
02/09	6134.3	6152	0.29%	6152	0.29%
09/09	5909.5	5915	0.11%	5915	0.11%
Media	6320.9	6263.56	1.83%	6169.44	0.33%

Le misurazioni di cui non è stato possibile registrare il valore per il superamento del *time limit* sono riportate col simbolo "-".

Tabella A.3: Risultati della minimizzazione del ritardo con NCols = 750 e Aging = 7

	CG			BC3%		Tot3%	BC1%		Tot1%
Date	T(m)	W3%	W1%	T(m)	W	T(m)	T(m)	W	T(m)
08/07	9.27	90%	89%	1.05	10%	10.32	1.18	11%	10.44
26/08	6.68	89%	88%	0.85	11%	7.53	0.92	12%	7.60
27/08	16.75	-	-	-	-	-	-	-	-
28/08	7.13	6%	4%	118.21	94%	125.34	156.61	96%	163.74
29/08	7.35	94%	92%	0.51	6%	7.86	0.60	8%	7.95
30/08	6.53	93%	34%	0.49	7%	7.02	12.90	66%	19.43
31/08	5.77	94%	93%	0.38	6%	6.15	0.44	7%	6.21
01/09	6.88	95%	61%	0.39	5%	7.27	4.39	39%	11.28
02/09	5.07	94%	94%	0.30	6%	5.37	0.30	6%	5.37
09/09	5.83	96%	96%	0.24	4%	6.08	0.24	4%	6.08
Media	7.73	83%	72%	13.60	17%	20.33	19.73	28%	26.46

Le misurazioni di cui non è stato possibile registrare il valore per il superamento del *time limit* sono riportate col simbolo "-".

Tabella A.4: Tempi della minimizzazione del ritardo con NCols = 1000 e Aging = 2

Date	LB (TU)	Object 3% (TU)	Gap 3%	Object 1% (TU)	Gap 1%
08/07	6030.2	6179	2.41%	6053	0.38%
26/08	6253.1	6395	2.22%	6316	1.00%
27/08	7868.6	-	-	-	-
28/08	6721.9	6805	1.22%	6789	0.99%
29/08	6527.1	6679	2.27%	6532	0.07%
30/08	5365.5	5515	2.71%	5389	0.44%
31/08	6293.8	6426	2.06%	6306	0.19%
01/09	6106.0	6235	2.07%	6132	0.42%
02/09	6134.3	6140	0.09%	6140	0.09%
09/09	5908.5	5910	0.03%	5910	0.03%
Media	6320.9	6253.78	1.68%	6174.11	0.40%

Le misurazioni di cui non è stato possibile registrare il valore per il superamento del *time limit* sono riportate col simbolo "-".

Tabella A.5: Risultati della minimizzazione del ritardo con NCols = 1000 e Aging = 2

	CG			BC3%		Tot3%	BC1%		Tot1%
Date	T(m)	W3%	W1%	T(m)	W	T(m)	T(m)	W	T(m)
08/07	9.52	84%	48%	1.76	16%	11.28	10.38	52%	19.90
26/08	8.38	87%	86%	1.27	13%	9.65	1.40	14%	9.79
27/08	16.78	41%	22%	23.92	59%	40.70	60.02	78%	76.80
28/08	8.48	86%	-	1.42	14%	9.90	-	-	-
29/08	8.42	94%	93%	0.57	6%	8.99	0.62	7%	9.04
30/08	8.35	93%	67%	0.63	7%	8.98	4.17	33%	12.52
31/08	6.68	54%	54%	5.76	46%	12.44	5.76	46%	12.44
01/09	6.62	94%	80%	0.42	6%	7.04	1.64	20%	8.26
02/09	5.70	94%	94%	0.36	6%	6.06	0.36	6%	6.06
09/09	6.82	96%	96%	0.31	4%	7.13	0.31	4%	7.13
Media	8.58	82%	71%	3.64	18%	12.22	9.41	29%	17.99

Le misurazioni di cui non è stato possibile registrare il valore per il superamento del *time limit* sono riportate col simbolo "-".

Tabella A.6: Tempi della minimizzazione del ritardo con NCols = 1000 e Aging = 3

Date	LB (TU)	Object 3% (TU)	Gap 3%	Object 1% (TU)	Gap 1%
08/07	6030.2	6175	2.34%	6044	0.23%
26/08	6253.1	6420	2.60%	6265	0.19%
27/08	7868.6	8104	2.90%	7943	0.94%
28/08	6721.9	6916	2.81%	-	-
29/08	6527.1	6661	2.01%	6592	0.98%
30/08	5365.5	5502	2.48%	5399	0.62%
31/08	6293.8	6313	0.30%	6313	0.30%
01/09	6106.0	6213	1.72%	6168	1.01%
02/09	6134.3	6135	0.01%	6135	0.01%
09/09	5908.5	5913	0.08%	5913	0.08%
Media	6320.9	6435.20	1.73%	6308.00	0.48%

Le misurazioni di cui non è stato possibile registrare il valore per il superamento del *time limit* sono riportate col simbolo "-".

Tabella A.7: Risultati della minimizzazione del ritardo con NCols = 1000 e Aging = 3

	CG			BC3%		Tot3%	BC1%		Tot1%
Date	T(m)	W3%	W1%	T(m)	W	T(m)	T(m)	W	T(m)
08/07	10.83	87%	85%	1.58	13%	12.42	1.85	15%	12.68
26/08	8.58	91%	35%	0.88	9%	9.47	15.76	65%	24.34
27/08	22.23	-	-	-	-	-	-	-	-
28/08	10.03	87%	22%	1.50	13%	11.53	35.35	78%	45.38
29/08	8.57	94%	38%	0.58	6%	9.15	13.81	62%	22.38
30/08	9.83	88%	69%	1.36	12%	11.19	4.36	31%	14.19
31/08	7.03	94%	57%	0.45	6%	7.49	5.32	43%	12.35
01/09	7.32	92%	41%	0.61	8%	7.93	10.34	59%	17.66
02/09	8.18	96%	96%	0.36	4%	8.54	0.36	4%	8.54
09/09	6.67	60%	60%	4.49	40%	11.15	4.49	40%	11.15
Media	9.93	88%	56%	1.31	12%	9.87	10.18	44%	18.74

Le misurazioni di cui non è stato possibile registrare il valore per il superamento del *time limit* sono riportate col simbolo "-".

Tabella A.8: Tempi della minimizzazione del ritardo con NCols = 1000 e Aging = 4

Date	LB (TU)	Object 3% (TU)	Gap 3%	Object 1% (TU)	Gap 1%
08/07	6030.2	6192	2.61%	6091	1.00%
26/08	6253.1	6410	2.45%	6263	0.16%
27/08	7868.6	-	-	-	-
28/08	6721.9	6932	3.03%	6775	0.78%
29/08	6527.1	6705	2.65%	6534	0.11%
30/08	5365.5	5530	2.97%	5389	0.44%
31/08	6293.8	6436	2.21%	6327	0.53%
01/09	6106.0	6264	2.52%	6120	0.23%
02/09	6134.3	6135	0.01%	6135	0.01%
09/09	5908.5	5912	0.06%	5912	0.06%
Media	6320.9	6279.56	2.06%	6171.78	0.37%

Le misurazioni di cui non è stato possibile registrare il valore per il superamento del *time limit* sono riportate col simbolo "-".

Tabella A.9: Risultati della minimizzazione del ritardo con NCols = 1000 e Aging = 4

	CG			BC3%		Tot3%	BC1%		Tot1%
Date	T(m)	W3%	W1%	T(m)	W	T(m)	T(m)	W	T(m)
08/07	13.35	83%	-	2.65	17%	16.00	-	-	-
26/08	10.68	-	-	-	-	-	-	-	-
27/08	28.75	-	-	-	-	-	-	-	-
28/08	13.28	87%	-	2.00	13%	15.28	-	-	-
29/08	11.05	95%	43%	0.61	5%	11.66	14.71	57%	25.76
30/08	10.67	94%	46%	0.73	6%	11.39	12.43	54%	23.10
31/08	8.62	85%	85%	1.56	15%	10.18	1.56	15%	10.18
01/09	8.68	94%	63%	0.51	6%	9.20	5.02	37%	13.71
02/09	7.17	93%	93%	0.57	7%	7.74	0.57	7%	7.74
09/09	8.07	94%	94%	0.49	6%	8.56	0.49	6%	8.56
Media	12.03	91%	71%	1.14	9%	11.25	5.80	29%	14.84

Le misurazioni di cui non è stato possibile registrare il valore per il superamento del *time limit* sono riportate col simbolo "-".

Tabella A.10: Tempi della minimizzazione del ritardo con NCols = 1000 e Aging = 5

Date	LB (TU)	Object 3% (TU)	Gap 3%	Object 1% (TU)	Gap 1%
08/07	6030.2	6197	2.69%	-	-
26/08	6253.1	-	-	-	-
27/08	7868.6	-	-	-	-
28/08	6721.9	6924	2.92%	-	-
29/08	6527.1	6709	2.71%	6532	0.07%
30/08	5365.5	5524	2.87%	5392	0.49%
31/08	6293.8	6355	0.96%	6355	0.96%
01/09	6106.0	6265	2.54%	6127	0.34%
02/09	6134.3	6153	0.30%	6153	0.30%
09/09	5908.5	5917	0.14%	5917	0.14%
Media	6320.9	6255.50	1.89%	6079.33	0.39%

Le misurazioni di cui non è stato possibile registrare il valore per il superamento del *time limit* sono riportate col simbolo "-".

Tabella A.11: Risultati della minimizzazione del ritardo con NCols = 1000 e Aging = 5

	CG			BC3%		Tot3%	BC1%		Tot1%
Date	T(m)	W3%	W1%	T(m)	W	T(m)	T(m)	W	T(m)
08/07	13.08	99%	99%	0.07	1%	13.16	0.18	1%	13.27
26/08	10.25	99%	98%	0.08	1%	10.33	0.18	2%	10.43
27/08	33.88	56%	36%	26.22	44%	60.10	60.08	64%	93.97
28/08	19.13	99%	88%	0.25	1%	19.38	2.65	12%	21.78
29/08	11.42	99%	97%	0.09	1%	11.51	0.32	3%	11.73
30/08	10.70	99%	97%	0.15	1%	10.85	0.37	3%	11.07
31/08	9.15	99%	98%	0.09	1%	9.24	0.22	2%	9.37
01/09	10.38	99%	98%	0.09	1%	10.47	0.22	2%	10.60
02/09	6.72	99%	98%	0.08	1%	6.80	0.17	2%	6.88
09/09	8.32	99%	98%	0.06	1%	8.37	0.15	2%	8.47
Media	13.30	95%	91%	2.72	5%	16.02	6.45	9%	19.76

Tabella A.12: Tempi della minimizzazione del ritardo con NCols = 1500 e Aging = 2

Date	LB (TU)	Object 3% (TU)	Gap 3%	Object 1%	Gap 1%
08/07	6030.2	5900	2.18%	5796	0.43%
26/08	6253.1	6055	2.13%	5931	0.08%
27/08	7868.6	7589	1.37%	7578	1.22%
28/08	6721.9	6435	2.20%	6357	1.00%
29/08	6527.1	6430	2.13%	6308	0.24%
30/08	5365.5	5250	2.37%	5161	0.68%
31/08	6293.8	6161	1.88%	6047	0.03%
01/09	6106.0	5954	1.86%	5852	0.15%
02/09	6134.3	5884	0.01%	5884	0.01%
09/09	5908.5	5683	0.08%	5683	0.08%
Media	6320.9	6134.10	1.62%	6059.70	0.39%

Tabella A.13: Risultati della minimizzazione del ritardo con NCols = 1500 e Aging = 2

	CG			BC3%		Tot3%	BC1%		Tot1%
Date	T(m)	W3%	W1%	T(m)	W	T(m)	T(m)	W	T(m)
08/07	9.72	90%	26%	1.13	10%	10.84	27.40	74%	37.12
26/08	9.55	96%	96%	0.44	4%	9.99	0.44	4%	9.99
27/08	20.42	49%	-	21.26	51%	41.68	-	-	-
28/08	10.48	86%	-	1.76	14%	12.24	-	-	-
29/08	9.00	90%	89%	0.95	10%	9.95	1.09	11%	10.09
30/08	10.60	93%	57%	0.75	7%	11.35	7.99	43%	18.59
31/08	6.63	95%	94%	0.37	5%	7.00	0.40	6%	7.04
01/09	6.88	93%	55%	0.53	7%	7.41	5.59	45%	12.47
02/09	5.55	92%	92%	0.51	8%	6.06	0.51	8%	6.06
09/09	7.20	96%	96%	0.29	4%	7.49	0.29	4%	7.49
Media	9.60	88%	76%	2.80	12%	12.40	5.46	24%	13.61

Le misurazioni di cui non è stato possibile registrare il valore per il superamento del *time limit* sono riportate col simbolo "-".

Tabella A.14: Tempi della minimizzazione del ritardo con NCols = 1500 e Aging = 3

Date	LB (TU)	Object 3% (TU)	Gap 3%	Object 1% (TU)	Gap 1%
08/07	6030.2	6193	2.63%	6036	0.10%
26/08	6253.1	6275	0.35%	6275	0.35%
27/08	7868.6	8112	3.00%	-	-
28/08	6721.9	6921	2.88%	-	-
29/08	6527.1	6699	2.57%	6587	0.91%
30/08	5365.5	5529	2.96%	5390	0.45%
31/08	6293.8	6438	2.24%	6331	0.59%
01/09	6106.0	6262	2.49%	6125	0.31%
02/09	6134.3	6145	0.17%	6145	0.17%
09/09	5908.5	5913	0.08%	5913	0.08%
Media	6320.9	6448.70	1.94%	6100.25	0.37%

Le misurazioni di cui non è stato possibile registrare il valore per il superamento del *time limit* sono riportate col simbolo "-".

Tabella A.15: Risultati della minimizzazione del ritardo con NCols = 1500 e Aging = 3

	CG			BC3%		Tot3%	BC1%		Tot1%
Date	T(m)	W3%	W1%	T(m)	W	T(m)	T(m)	W	T(m)
08/07	9.70	93%	93%	0.73	7%	10.43	0.73	7%	10.43
26/08	7.52	89%	42%	0.92	11%	8.43	10.30	58%	17.82
27/08	18.75	30%	-	43.09	70%	61.84	-	-	-
28/08	8.18	88%	24%	1.10	12%	9.29	26.31	76%	34.49
29/08	8.10	94%	94%	0.49	6%	8.59	0.56	6%	8.66
30/08	7.03	92%	46%	0.62	8%	7.65	8.38	54%	15.42
31/08	6.67	94%	93%	0.41	6%	7.07	0.48	7%	7.15
01/09	6.12	91%	56%	0.57	9%	6.69	4.87	44%	10.99
02/09	6.33	94%	94%	0.38	6%	6.72	0.38	6%	6.72
09/09	6.18	96%	96%	0.25	4%	6.43	0.25	4%	6.43
Media	8.46	86%	71%	4.86	14%	13.31	5.81	29%	13.12

Le misurazioni di cui non è stato possibile registrare il valore per il superamento del *time limit* sono riportate col simbolo "-".

Tabella A.16: Tempi della minimizzazione del ritardo con NCols = 2000 e Aging = 2

Date	LB (TU)	Object 3% (TU)	Gap 3%	Object 1% (TU)	Gap 1%
08/07	6030.2	6048	0.29%	6048	0.29%
26/08	6253.1	6434	2.81%	6302	0.78%
27/08	7868.6	8111	2.99%	-	-
28/08	6721.9	6926	2.95%	6781	0.87%
29/08	6527.1	6689	2.42%	6592	0.98%
30/08	5365.5	5526	2.90%	5397	0.58%
31/08	6293.8	6452	2.45%	6327	0.53%
01/09	6106.0	6267	2.57%	6129	0.38%
02/09	6134.3	6135	0.01%	6135	0.01%
09/09	5908.5	5916	0.13%	5916	0.13%
Media	6320.9	6450.40	1.95%	6180.78	0.51%

Le misurazioni di cui non è stato possibile registrare il valore per il superamento del *time limit* sono riportate col simbolo "-".

Tabella A.17: Risultati della minimizzazione del ritardo con NCols = 2000 e Aging = 2

A.2 Massimizzazione delle preferenze

Il budget di ritardo per la massimizzazione delle preferenze corrisponde al minimo ritardo ottenuto con l'esecuzione dell'algoritmo basato su generazione di colonne coi parametri Newcols = 1000, Aging = 4 e Tolleranza = 1%, maggiorato del 10%, dunque $\lceil Object1\% \rceil \cdot 1.10$.

Di seguito vengono riportati i valori di riferimento dei parametri nelle diverse tabelle:

Tabella A.18 e A.19: NCols = 40, Aging = 10;

Tabella A.20 e A.21: NCols = 100, Aging = 4;

Tabella A.22 e A.23: NCols = 500, Aging = 4;

Tabella A.24 e A.25: NCols = 1000, Aging = 3;

Tabella A.26 e A.27: NCols = 1500, Aging = 2;

Tabella A.28 e A.29: NCols = 2000, Aging = 2;

	CG			BC3%		Tot3%	BC1%		Tot1%
Date	T(m)	W3%	W1%	T(m)	W	T(m)	T(m)	W	T(m)
08/07	150.4	99.6%	83%	0.6	0.4%	151.0	31.0	17%	181.3
26/08	99.5	99.8%	66%	0.2	0.2%	99.6	50.5	34%	150.0
27/08	163.7	99.9%	33%	0.1	0.1%	163.8	328.6	67%	492.3
28/08	130.9	55.0%	55%	107.3	45.0%	238.2	107.3	45%	238.2
29/08	92.2	99.8%	97%	0.2	0.2%	92.3	2.5	3%	94.7
30/08	85.2	99.9%	55%	0.1	0.1%	85.3	69.7	45%	154.9
31/08	82.5	99.9%	98%	0.1	0.1%	82.7	1.3	2%	83.9
01/09	76.1	99.8%	98%	0.1	0.2%	76.2	1.6	2%	77.6
02/09	88.1	99.4%	88%	0.5	0.6%	88.6	12.5	12%	100.6
09/09	76.7	99.9%	98%	0.1	0.1%	76.8	1.6	2%	78.3
Media	104.5	95.3%	77%	10.9	4.7%	115.5	60.7	23%	165.2

Tabella A.18: Tempi della massimizzazione delle preferenze in [14] con NCols = 40 e Aging = 10

Date	Budget	N° Var	UB*	Object3%	Gap3%	Objec1%	Gap1%
08/07	6702	53356	24316.3	23943.7	1.56%	24270.1	0.19%
26/08	6896	52445	24355.7	23960.0	1.65%	24305.9	0.20%
27/08	8736	45936	21095.5	20635.8	2.23%	20923.0	0.82%
28/08	7447	48836	22605.4	22482.9	0.55%	22482.9	0.55%
29/08	7213	51629	23760.6	23382.1	1.62%	23582.7	0.75%
30/08	5940	51162	23420.1	23033.4	1.68%	23363.1	0.24%
31/08	6938	51486	23841.0	23450.2	1.67%	23676.2	0.70%
01/09	6760	52612	24152.6	23785.2	1.54%	23998.4	0.64%
02/09	6755	53557	24484.4	24113.9	1.54%	24397.0	0.36%
09/09	6565	53724	24203.1	23795.2	1.71%	24065.3	0.57%
Media	6995.2	51474.3	23623.5	23258.2	1.57%	23506.5	0.50%

* I valori di UB sono diversi rispetto alle altre tabelle perché sono ottenuti con un budget di ritardo diverso.

Tabella A.19: Risultati della massimizzazione delle preferenze in [14] con NCols = 40 e Aging = 10

	CG			BC3%		Tot3%	BC1%		Tot1%
Date	T(m)	W3%	W1%	T(m)	W	T(m)	T(m)	W	T(m)
08/07	47.9	99.8%	96%	0.1	0.2%	48.0	2.1	4%	49.9
26/08	37.2	99.7%	96%	0.1	0.3%	37.3	1.6	4%	38.8
27/08	115.1	99.9%	-	0.1	0.1%	115.2	-	-	-
28/08	54.5	99.8%	32%	0.1	0.2%	54.6	116.3	68%	170.7
29/08	32.6	99.7%	96%	0.1	0.3%	32.7	1.5	4%	34.1
30/08	32.5	99.7%	28%	0.1	0.3%	32.6	82.5	72%	115.1
31/08	29.7	99.7%	95%	0.1	0.3%	29.8	1.5	5%	31.1
01/09	29.0	33.1%	33%	58.5	66.9%	87.5	59.2	67%	88.2
02/09	25.2	99.6%	95%	0.1	0.4%	25.3	1.4	5%	26.7
09/09	30.8	99.7%	97%	0.1	0.3%	30.9	1.0	3%	31.8
Media	43.4	93.1%	74%	5.9	6.9%	49.4	29.7	26%	65.2

Le misurazioni di cui non è stato possibile registrare il valore per il superamento del *time limit* sono riportate col simbolo "-".

Tabella A.20: Tempi della massimizzazione delle preferenze con NCols = 100 e Aging = 4 [14]

Date	Budget	N° Var	UB	Object3%	Gap3%	Object1%	Gap1%
08/07	6649	53450	24281.5	23868.4	1.73%	24114.1	0.69%
26/08	6892	52696	24352.8	23914.3	1.83%	24160.3	0.80%
27/08	8738	47229	21096.7	20605.6	2.38%	20605.6	2.38%
28/08	7608	49667	22706.5	22248.5	2.06%	22634.3	0.32%
29/08	7252	52578	23785.5	23335.9	1.93%	23602.1	0.78%
30/08	5939	52147	23419.3	22960.0	2.00%	23371.9	0.20%
31/08	6945	52345	23845.5	23440.7	1.73%	23676.1	0.72%
01/09	6785	52619	24168.5	24095.9	0.30%	24095.9	0.30%
02/09	6749	54043	24480.3	24094.5	1.60%	24460.9	0.08%
09/09	6505	53552	24162.4	23743.2	1.77%	24009.8	0.64%
Media	7006.2	52032.6	23629.9	23230.7	1.73%	23791.7	0.50%

Tabella A.21: Risultati della massimizzazione delle preferenze con NCols = 100 e Aging = 4 [14]

	CG			BC3%		Tot3%	BC1%		Tot1%
Date	T(m)	W3%	W1%	T(m)	W	T(m)	T(m)	W	T(m)
08/07	34.8	99.6%	60%	0.1	0.4%	35.0	23.5	40%	58.3
26/08	24.3	99.5%	94%	0.1	0.5%	24.4	1.7	6%	25.9
27/08	78.2	99.8%	-	0.2	0.2%	78.4	-	-	-
28/08	34.9	99.5%	23%	0.2	0.5%	35.0	114.9	77%	149.8
29/08	23.2	99.5%	94%	0.1	0.5%	23.3	1.6	6%	24.8
30/08	22.4	99.5%	20%	0.1	0.5%	22.5	89.9	80%	112.3
31/08	20.3	99.4%	93%	0.1	0.6%	20.5	1.5	7%	21.8
01/09	20.1	99.4%	91%	0.1	0.6%	20.2	2.0	9%	22.1
02/09	15.1	99.2%	57%	0.1	0.8%	15.2	11.3	43%	26.4
09/09	19.6	99.4%	93%	0.1	0.6%	19.7	1.4	7%	21.0
Media	29.3	99.5%	69%	0.1	0.5%	29.4	27.5	31%	51.4

Le misurazioni di cui non è stato possibile registrare il valore per il superamento del *time limit* sono riportate col simbolo "-".

Tabella A.22: Tempi della massimizzazione delle preferenze con NCols = 500 e Aging = 4

Date	Budget	N° Var	UB	Object3%	Gap3%	Object1%	Gap1%
08/07	6649	62078	24281.5	23852.7	1.80%	24219.1	0.26%
26/08	6892	60844	24352.8	23893.4	1.92%	24173.1	0.74%
27/08	8738	53398	21096.7	20568.2	2.57%	20568.2	2.57%
28/08	7608	56635	22706.5	22220.2	2.19%	22625.4	0.36%
29/08	7252	62894	23785.5	23321.9	1.99%	23624.9	0.68%
30/08	5939	60640	23419.3	22969.2	1.96%	23361.9	0.25%
31/08	6945	61573	23845.5	23414.2	1.84%	23652.2	0.82%
01/09	6785	61432	24168.5	23728.2	1.86%	23991.7	0.74%
02/09	6749	62786	24480.3	24083.1	1.65%	24458.1	0.09%
09/09	6505	63074	24162.4	23731.1	1.82%	24023.6	0.58%
Media	7006.2	60535.4	23629.9	23178.2	1.96%	23791.2	0.50%

Tabella A.23: Risultati della massimizzazione delle preferenze con NCols = 500 e Aging = 5

	CG			BC3%		Tot3%	BC1%		Tot1%
Date	T(m)	W3%	W1%	T(m)	W	T(m)	T(m)	W	T(m)
08/07	29.3	99.5%	25%	0.2	0.5%	29.4	86.1	75%	115.4
26/08	22.9	99.4%	92%	0.1	0.6%	23.0	2.1	8%	24.9
27/08	67.2	99.7%	-	0.2	0.3%	67.4	-	-	-
28/08	34.7	99.6%	-	0.1	0.4%	34.8	-	-	-
29/08	18.7	99.2%	92%	0.1	0.8%	18.8	1.6	8%	20.3
30/08	20.5	99.3%	26%	0.1	0.7%	20.7	58.2	74%	78.7
31/08	17.9	99.2%	93%	0.2	0.8%	18.1	1.4	7%	19.3
01/09	16.7	99.1%	42%	0.1	0.9%	16.8	22.7	58%	39.3
02/09	14.2	99.0%	90%	0.1	1.0%	14.3	1.5	10%	15.7
09/09	16.6	99.1%	56%	0.1	0.9%	16.8	13.0	44%	29.7
Media	25.9	99.3%	65%	0.2	0.7%	26.0	23.3	35%	42.9

Le misurazioni di cui non è stato possibile registrare il valore per il superamento del *time limit* sono riportate col simbolo "-".

Tabella A.24: Tempi della massimizzazione delle preferenze con NCols = 1000 e Aging = 3

Date	Budget	N° Var	UB	Object3%	Gap3%	Object1%	Gap1%
08/07	6649	66252	24281.5	23829.7	1.90%	24236.5	0.19%
26/08	6892	65281	24352.8	23881.1	1.98%	24157.7	0.81%
27/08	8738	57523	21096.7	20546.7	2.68%	20546.7	2.68%
28/08	7608	61101	22706.5	22169.3	2.42%	22169.3	2.42%
29/08	7252	66457	23785.5	23327.1	1.97%	23601.9	0.78%
30/08	5939	64960	23419.3	22926.1	2.15%	23347.0	0.31%
31/08	6945	68186	23845.5	23376.2	2.01%	23646.4	0.84%
01/09	6785	68289	24168.5	23743.9	1.79%	24099.5	0.29%
02/09	6749	69549	24480.3	24051.3	1.78%	24338.9	0.58%
09/09	6505	68601	24162.4	23716.8	1.88%	24139.6	0.09%
Media	7006.2	65619.9	23629.9	23156.8	2.05%	23945.9	0.49%

Tabella A.25: Risultati della massimizzazione delle preferenze con NCols = 1000 e Aging = 3

	CG			BC3%		Tot3%	BC1%		Tot1%
Date	T(m)	W3%	W1%	T(m)	W	T(m)	T(m)	W	T(m)
08/07	25.8	99.4%	21%	0.2	0.6%	26.0	95.3	79%	121.2
26/08	20.9	99.3%	77%	0.1	0.7%	21.0	6.1	23%	26.9
27/08	58.5	99.7%	-	0.2	0.3%	58.7	-	-	-
28/08	31.7	99.5%	21%	0.2	0.5%	31.9	119.2	79%	150.9
29/08	16.7	99.1%	22%	0.2	0.9%	16.8	59.8	78%	76.5
30/08	17.2	99.1%	93%	0.1	0.9%	17.3	1.2	7%	18.4
31/08	14.0	98.9%	91%	0.1	1.1%	14.1	1.4	9%	15.3
01/09	16.2	99.1%	91%	0.2	0.9%	16.3	1.7	9%	17.8
02/09	10.4	98.6%	89%	0.1	1.4%	10.5	1.3	11%	11.7
09/09	12.9	98.9%	90%	0.1	1.1%	13.0	1.4	10%	14.3
Media	22.4	99.2%	66%	0.2	0.8%	22.6	31.9	34%	50.3

Le misurazioni di cui non è stato possibile registrare il valore per il superamento del *time limit* sono riportate col simbolo "-".

Tabella A.26: Tempi della massimizzazione delle preferenze con NCols = 1500 e Aging = 2

Date	Budget	N° Var	UB	Object3%	Gap3%	Object1%	Gap1%
08/07	6649	68936	24281.5	23809.2	1.98%	24211.6	0.29%
26/08	6892	69623	24352.8	23875.7	2.00%	24305.7	0.19%
27/08	8738	60348	21096.7	20585.2	2.48%	20585.2	2.48%
28/08	7608	62352	22706.5	20592.7	10.3%	22613.0	0.41%
29/08	7252	70013	23785.5	23341.7	1.90%	23752.8	0.14%
30/08	5939	68650	23419.3	22950.5	2.04%	23237.5	0.78%
31/08	6945	70774	23845.5	23390.6	1.94%	23672.3	0.73%
01/09	6785	71010	24168.5	23725.8	1.87%	23990.0	0.74%
02/09	6749	72592	24480.3	24060.1	1.75%	24321.1	0.65%
09/09	6505	72494	24162.4	23728.8	1.83%	24027.1	0.56%
Media	7006.2	68679.2	23629.9	23006.0	2.81%	23792.3	0.50%

Tabella A.27: Risultati della massimizzazione delle preferenze con NCols = 1500 e Aging = 2

	CG			BC3%		Tot3%	BC1%		Tot1%
Date	T(m)	W3%	W1%	T(m)	W	T(m)	T(m)	W	T(m)
08/07	26.5	99.3%	92%	0.2	0.7%	26.7	2.2	8%	28.7
26/08	20.1	99.2%	91%	0.2	0.8%	20.3	1.9	9%	22.0
27/08	57.0	99.6%	-	0.2	0.4%	57.2	-	-	-
28/08	30.8	-	-	-	-	-	-	-	-
29/08	16.6	99.0%	91%	0.2	1.0%	16.8	1.7	9%	18.3
30/08	17.6	99.1%	17%	0.2	0.9%	17.8	83.8	83%	101.4
31/08	17.0	99.1%	92%	0.2	0.9%	17.1	1.5	8%	18.5
01/09	17.3	99.0%	91%	0.2	1.0%	17.5	1.8	9%	19.1
02/09	13.7	98.9%	75%	0.2	1.1%	13.8	4.5	25%	18.2
09/09	14.8	97.8%	27%	0.3	2.2%	15.1	39.2	73%	54.0
Media	23.1	99.0%	72%	0.2	1.0%	22.5	17.1	28%	35.0

Le misurazioni di cui non è stato possibile registrare il valore per il superamento del *time limit* sono riportate col simbolo "-".

Tabella A.28: Tempi della massimizzazione delle preferenze con NCols = 2000 e Aging = 2

Date	Budget	N° Var	UB	Object3%	Gap3%	Object1%	Gap1%
08/07	6649	72631	24281.5	23831.2	1.89%	24110.0	0.71%
26/08	6892	73465	24352.8	23863.2	2.05%	24140.6	0.88%
27/08	8738	62343	21096.7	20562.4	2.60%	20562.4	2.60%
28/08	7608	65307	22706.5	-	-	-	-
29/08	7252	73490	23785.5	23328.3	1.96%	23588.1	0.84%
30/08	5939	70831	23419.3	22950.1	2.04%	23351.0	0.29%
31/08	6945	72363	23845.5	23383.5	1.98%	23645.7	0.84%
01/09	6785	75144	24168.5	23744.8	1.78%	24002.2	0.69%
02/09	6749	75284	24480.3	24052.5	1.78%	24456.2	0.10%
09/09	6505	76851	24162.4	23751.3	1.73%	24136.9	0.11%
Media	7006.2	71770.9	23629.9	23274.1	1.98%	23554.8	0.78%

Le misurazioni di cui non è stato possibile registrare il valore per il superamento del *time limit* sono riportate col simbolo "-".

Tabella A.29: Risultati della massimizzazione delle preferenze con NCols = 2000 e Aging = 2

Acronimi

ACC Area Control Centre.

ANSP Air Navigation Service Provier.

API Application Program Interface.

ATC Air Traffic Control.

ATFM Air Traffic Flow Management.

CTA Control Area.

DDR2 Demand Data Repository 2.

ECAC European Civil Aviation Conference.

FL Flight Level.

ICAO International Civil Aviation Organization.

IDE Integrated development environment.

JSON JavaScript Object Notation.

MAGHP Multi-Airport Ground-Holding Ploblem.

NEST Network Strategic Tool.

OPL Optimization Programming Language.

PCA Principal Component Alalysis.

SAGHP Single-Airport Ground-Holding Ploblem.

SQL Structured Query Language.

TBO Trajectory Based Operations.

TSP Traveling Salesman Problem.

TU time unit.

Glossario

albero di decisione metodo di apprendimento supervisionato per classificare i dati in input.

algoritmo greedy algoritmo che raggiunge la soluzione seguendo, ad ogni passo, la strada più promettente in quel momento.

Branch & Bound tecnica per rendere la soluzione di un problema di Programmazione Lineare Intera attraverso la suddivisione del problema principale in sottoproblemi.

C++ linguaggio di programmazione che permette elevate prestazioni.

classificazione fase dell'apprendimento automatico in cui si associa un determinato esempio alla sua etichetta/classe.

clustering metodo di apprendimento non supervisionato che suddivide i dati in base alle loro caratteristiche simili.

CPLEX risolutore di modelli di programmazione lineare fornito da IBM ILOG® CPLEX® Optimization Studio.

DBSCAN algoritmo di clustering basato sulla densità dei punti.

diversificazione durante una ricerca locale, la fase che spinge a scegliere soluzioni molto differenti a quella corrente.

intensificazione durante una ricerca locale, la fase che spinge a scegliere soluzioni simili a quella corrente.

machine learning branchia dell'intelligenza artificiale che studia metodi che riescano a riconoscere determinati pattern (apprendere) dai dati in input.

matheuristic metodologia che prevede l'uso ibrido di metodi metaeuristici applicati a un modello matematico.

metodo euristico metodo per l'ottimizzazione combinatoria che attraverso la presenza di alcune assunzioni permette di raggiungere una soluzione qualitativamente molto buona, seppur non ottima, con un costo computazionale ridotto.

PCA (Principal Component Alalysis) tecnica applicata per diminuire la dimensionalità dei dati.

piano di volo indicazione di una traiettoria per un determinato volo e del suo orario di partenza e di arrivo, allora è stato effettuato un piano di volo.

precision metrica per l'apprendimento supervisionato. Indica il rapporto tra il numero di elementi veri positivi e gli elementi classificati positivamente.

programmazione lineare metodologia per la formulazione e la soluzione di modelli di ottimizzazione descritti da una funzione obiettivo lineare da massimizzare o minimizzare e da un sistema di vincoli lineari.

python linguaggio ad alto livello, molto utilizzato per la sua semplice sintassi.

query interrogazione ad un database.

recall metrica per l'apprendimento supervisionato. Indica il rapporto tra il numero dei veri positivi e gli elementi veramente positivi.

validazione incrociata tecnica per l'apprendimento supervisionato usata quando non si possiedono molti dati. I training set e i test set sono una ricombinazione dell'intero insieme di dati.

Riferimenti bibliografici

[1] G. Andreatta, A. R. Odoni e O. Richetta. «Models for the Ground Holding Problem». In: *Large scale computation and information processing in air traffic control*. A cura di L. Bianco e A. R. Odoni. Berlin ; Hong Kong: Springer-Verlag, 1993, p. 125.

[2] G. Andreatta et al. *Rapporto sul progetto di ricerca Follow-up di Optiframe*. Rapporto tecnico. CFR (Consorzio Futuro di Ricerca), 2019.

[3] D. Bertsimas, G. Lulli e A. Odoni. «An Integer Optimization Approach to Large-Scale Air Traffic Flow Management». *Operations Research*, 59.1 (2011), pp. 211-227.

[4] D. Bertsimas e S. Patterson. «The Air Traffic Flow Management Problem with Enroute Capacities». In: *Massachusetts Institute of Technology (MIT), Sloan School of Management, Working papers* 46.3 (1994), pp. 406–422.

[5] D. Bertsimas e S. Patterson. «The Traffic Flow Management Rerouting Problem in Air Traffic Control: A Dynamic Network Flow Approach». *Transportation Science* 34.3 (2000), pp. 239-255. 2000.

[6] M. A. Boschetti et al. «Matheuristics: Optimization, Simulation and Control». In: *Hybrid Metaheuristics*. A cura di M. J. Blesa et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 171–177.

[7] L. De Giovanni. *Heuristics for Combinatorial Optimization*. Slides 31-32. Università degli Studi di Padova. 2019. URL: `https://www.math.unipd.it/~luigi/courses/metmodoc1920/m02.meta.beamer.en.pdf`.

[8] L. De Giovanni. *Note su Programmazione Lineare e Metodo del Simplesso*. Università degli Studi di Padova. 2019. URL: `https://www.math.unipd.it/~luigi/courses/ricop1920/m02.PLsim.01.pdf`.

[9] L. De Giovanni, M. Di Summa e G. Zambelli. *Solution Methods for Integer Linear Programming*. Università degli Studi di Padova. 2019. URL: `https://www.math.unipd.it/~luigi/courses/metmodoc1920/m06.pli.en.pdf`.

[10] L. De Giovanni, G. Lulli e R. Lancia. *Data-driven optimization for trajectory based Air Traffic Flow Management*. Rapporto tecnico. Università degli Studi di Padova, 2022.

[11] L. De Giovanni e G. Zambelli. *Metodi basati su generazione di colonne*. Università degli Studi di Padova. 2019. URL: `https://www.math.unipd.it/~luigi/courses/metmodoc1920/m04.01.gencol.pdf`.

[12] L. De Giovanni. *Heuristics for Combinatorial Optimization*. Università degli Studi di Padova. 2019. URL: https://www.math.unipd.it/~luigi/courses/metmodoc1920/m02.meta.en.partial01.pdf.

[13] F. Djeumou Fomeni, G. Lulli e K. Zografos. «An optimization model for assigning 4D-trajectories to flights under the TBO concept». In: Twelfth USA/Europe ATM R&D Seminar. 2017, pp. 1–10.

[14] M. Echerle. «Algoritmi euristici di ottimizzazione per la gestione del traffico aereo basati su modelli di programmazione matematica e machine learning». Tesi di Laurea Magistrale. Università degli Studi di Padova, 2020.

[15] A. Gelmi. «Data analytics e matheuristics per la gestione del traffico aereo con un'applicazione allo spazio aereo europeo». Tesi di Laurea Magistrale. Università degli Studi di Padova, 2020.

[16] M. Helme. «Reducing air traffic delay in a space-time network». In: *Proceedings 1992 IEEE International Conference on Systems, Man, and Cybernetics*. Vol. 1. 1992, pp. 236–242.

[17] C. Lancia, L. De Giovanni e G. Lulli. «Data Analytics for Trajectory Selection and Preference-Model Extrapolation in the European Airspace». In: *Operations Research Proceedings 2018*. A cura di B. Fortz e M. Labbé. Cham: Springer International Publishing, 2019, pp. 563–570.

[18] M. Lodi. «Introducing computational thinking in k-12 education: historical, epistemological, pedagogical, cognitive and affective aspects». Tesi di Dottorato. Università di Bologna, 2020.

[19] T. Loss. «Gestione del traffico aereo con metodi basati su data analytics e column generation: un'applicazione allo spazio aereo europeo». Tesi di Laurea Magistrale. Università degli Studi di Padova, 2020.

[20] D. Meneghetti. «Analisi dati e modelli di programmazione intera per il problema della gestione del traffico aereo». Tesi di Laurea Magistrale. Università degli Studi di Padova, 2019.

[21] A. R. Odoni. «The Flow Management Problem in Air Traffic Control». In: *Flow Control of Congested Networks*. A cura di A. R. Odoni, L. Bianco e G. Szegö. Berlin, Heidelberg: Springer Berlin Heidelberg, 1987, pp. 269–288.

[22] *Procedures for Air Navigation Services (PANS) - Air Traffic Management (Doc 4444)*. 16ª ed. ICAO. 2016.

[23] P. Vranas, D. Bertsimas e A. Odoni. «The Multi-Airport Ground-Holding Problem in Air Trafic Control». In: *Massachusetts Institute of Technology (MIT), Sloan School of Management, Working papers* 42 (1992), pp. 249–261.

Riferimenti sitografici

[24] *A Description - C++*. URL: https://www.cplusplus.com/info/description/ (visitato il 18/02/2022).

[25] *Air Traffic Flow Management (ATFM)*. SKYbrary. 30 Dic. 2020. URL: https://www.skybrary.aero/index.php/Air_Traffic_Flow_Management_(ATFM) (visitato il 30/09/2021).

[26] *Developer Survey 2021*. URL: https://insights.stackoverflow.com/survey/2021#integrated-development-environment (visitato il 20/02/2022).

[27] A. Farruggia. *Data Compression Benchmark*. Università degli Studi di Pisa. URL: http://pages.di.unipi.it/farruggia/dcb/ (visitato il 22/03/2022).

[28] Flightradar24. URL: https://www.flightradar24.com/ (visitato il 29/09/2021).

[29] *Getting started - Pandas documentation*. URL: https://pandas.pydata.org/docs/getting_started/index.html (visitato il 17/02/2022).

[30] *Getting started - Scikit-learn documentation*. URL: https://scikit-learn.org/stable/getting_started.html (visitato il 17/02/2022).

[31] ICAO. *The World of Air Transport in 2019*. URL: https://www.icao.int/annual-report-2019/Pages/the-world-of-air-transport-in-2019.aspx (visitato il 29/09/2021).

[32] *ILOG CPLEX Optimization Studio*. URL: https://www.ibm.com/it-it/products/ilog-cplex-optimization-studio (visitato il 22/02/2022).

[33] *Kernel - Jupyter*. URL: https://docs.jupyter.org/en/latest/projects/kernels.html (visitato il 18/02/2022).

[34] *Network Strategic Tool (NEST)*. URL: https://www.eurocontrol.int/model/network-strategic-modelling-tool (visitato il 18/02/2022).

[35] *Project Jupyter*. URL: https://jupyter.org/ (visitato il 18/02/2022).

[36] *Python - Overview*. URL: https://wiki.python.org/moin/BeginnersGuide/Overview (visitato il 06/04/2022).

[37] *SQLite*. URL: https://www.sqlite.org/index.html (visitato il 20/02/2022).

[38] *What is NumPy?* URL: https://numpy.org/doc/stable/user/whatisnumpy.html (visitato il 17/02/2022).