Università degli Studi di Ferrara

DIPARTIMENTO DI MATEMATICA E INFORMATICA
Corso di Dottorato di Ricerca in Matematica e Informatica

Tesi di Dottorato di Ricerca

Assistenza sanitaria a domicilio: problemi multi-obiettivo d'instradamento di veicoli con bilanciamento di carico e fidelizzazione paziente-infermiera

Settore Scientifico-Disciplinare
MAT/09

Candidato:
Boccafoli Matteo

Coordinatore:
Prof.ssa Ruggiero Valeria

Supervisore della ricerca:
Prof.ssa Nonato Maddalena
Prof. Zanghirati Gaetano

Anno Accademico 2010–2012 — Ciclo XXV

Indice

Elenco delle tabelle

Elenco delle figure

Elenco degli algoritmi

Ringraziamenti

Si desidera ringraziare la Prof.ssa Maddalena Nonato e Prof. Gaetano Zanghirati per la guida, l'incoraggiamento e la supervisione. Il periodo di dottorato è stato supportato dalla borsa di studio ministeriale.

Questo lavoro ha tratto giovamento dai mesi trascorsi presso il *Centre interuniversitarie de recherche sur les réseaux d'entreprise, a logistique et le transport* (CIRRELT). In particolare, si ringrazia il Prof. Gilbert Laporte per l'ospitalità presso il CIRRELT e per la sua guida scientifica, a cui la bontà dell'ALNS per questo tipo di problema deve molto. Inoltre i dialoghi con il Dr. İbrahim Muter hanno incoraggiato l'esplorazione del problema rilassando i vincoli di routing.

Si ringraziano anche i controrelatori per i loro preziosi commenti, che hanno contribuito a migliorare considerevolmente questa tesi.

Introduzione

Motivazioni

La domanda di cure sanitarie a domicilio (*home health care*) offerte per anziani e malati cronici è costantemente in aumento. Le cause principali di questo aumento sono l'invecchiamento della popolazione ed il miglioramento della qualità della vita che i pazienti percepiscono stando a casa invece che in ospedale. Un altro fattore che influisce su tale aumento è la possibilità di avere servizi di alta qualità a costi contenuti: ciò è possibile coniugando la qualità del lavoro degli operatori con la riduzione dei costi assistenziali dovuta alla non ospedalizzazione del malato. Nei paesi industrializzati, i fornitori di servizi sanitari hanno investito molte risorse in questo tipo di servizio (Nesti et al., 2004).

Noi consideriamo il caso di un particolare servizio fornito nell'Italia del nord-est. Attualmente, la pianificazione di servizi è gestita manualmente da infermiere esperte, ma con il rapido aumento della domanda ha senso mettere a punto un sistema decisionale di supporto basato sull'ottimizzazione matematica. Nel nostro caso il servizio è attivato da un assistente medico a un paziente e le terapie sono decise con l'assistenza delle infermiere, che in particolare, determinano la natura e la frequenza delle cure. Quest'informazione permette al fornitore del servizio di impostare una pianificazione settimanale nella quale in ogni giorno della settimana le infermiere in turno avranno una sequenza di servizi da svolgere. Questa pianificazione può essere modificata durante la settimana, per la settimana successiva, all'evolversi delle situazioni come ad esempio il cambiamento della frequenza di visita, la richiesta di servizi a nuovi pazienti, l'assenza di un'infermiera o la domanda di servizi una tantum, come il prelievo del sangue.

In questo lavoro ci occupiamo di affrontare il problema del *master schedule settimanale*.

Home Health Care Problem

Lo scopo di questo lavoro è di modellare e risolvere il problema della domiciliazione di servizi sanitari, *Home Health Care Problem* (HHCP), definito come segue. Consideriamo un insieme di *servizi* S ed un *orizzonte temporale* H di una settimana, il quale è diviso in *giorni* h. Sono noti la *frequenza* f_s, cioè il numero di ripetizioni di un servizio s durante la settimana, ed il *tempo di lavoro* medio a_s stimato per compiere il servizio s. Ad ogni servizio s corrisponde un paziente p_s ed il relativo indirizzo di residenza. Per semplicità, s denota anche la residenza del

paziente. È dato il *tempo di viaggio* $d_{ss'}$ tra due residenze s e s', incluso il centro operativo 0, da dove tutte le infermiere partono e dove tutte devono ritornare.

Un tempo minimo d'attesa e_s, chiamato *intermezzo*, è imposto tra due ripetizioni consecutive del servizio s. Si pensi al caso nel quale un paziente richiede due tipologie di servizio differenti, alle quali sono associati una frequenza ed intermezzo differenti; per esempio il controllo della pressione due giorni alla settimana con un intermezzo di tre giorni, ed una medicazione complessa richiesta tre volte in un settimana con intermezzo due giorni. Accorpare i servizi del paziente per ogni giorno, equivale a risolvere una parte del problema. Pertanto, sarebbe un'approssimazione l'assegnazione di un unico servizio giornaliero, al quale sia associato un costo complessivo pari alla somma dei costi delle prestazioni prescelte per quel paziente in quel giorno.

Nei casi reali, i turni del personale godono di schemi modulari consolidati, che permettono la ciclicità del tipo di turno e gestiscono la malattia di un'infermiera mediante la reperibilità per sostituzione. Pertanto, la turnazione del personale è gestita in modo consolidato, mentre il problema dell'HHCP è principalmente la pianificazione della domiciliazione dei servizi sanitari.

Per *pianificazione dei servizi* si intende che i servizi devono essere distribuiti in un orizzonte temporale in accordo con le richieste di frequenza e devono essere assegnati alle infermiere. Per ogni infermiera j, viene determinata la sequenza ottimale di visita. Un obiettivo del processo di ottimizzazione è di favorire la *fidelizzazione* di un'infermiera ad un paziente, nel senso che sarebbe auspicabile che un paziente venisse trattato sempre dalla stessa infermiera. Un altro obiettivo è il *bilanciamento di carico* u tra le infermiere: esso quantifica il tempo complessivo di servizio (incluso il tempo di viaggio) e può considerare il tipo di servizi da svolgere, misurati secondo una classificazione che ne accredita la complessità e l'intensità. Un giusto bilanciamento di carico tra le infermiere ha il doppio effetto di migliore affiatamento del gruppo e di evitare il fenomeno di '*burnout*'.

Letteratura

Problemi simili all'HHCP sono stati studiati da numerosi autori (Cheng e Rich, 1998; Bertels e Fahle, 2006; Nickel et al., 2012; Eveborn et al., 2006; Lahrichi, 2008; Eveborn et al., 2009). L'HHCP combina due problemi noti: il *Periodic Vehicle Routing Problem with Time Windows* (PVRPTW) (Cordeau et al., 2001; Francis et al., 2006) e il *Nurse Rostering Problem* (Burke et al., 2004; Wren, 1996).

Molti contributi di Ottimizzazione all'HHCP si concentrano in specifiche applicazioni del problema. In aggiunta alla modellazione e alla proposta di algoritmi solutori del problema, tali contributi spesso descrivono una gestione della base dati per la pianificazione dei servizi. Il contesto d'applicazione varia da uno studio all'altro: diventa perciò difficile comparare i modelli ed i metodi adottati. La maggior parte dei contributi considera un servizio dove i pazienti specificano una finestra temporale (*time window*) durante la quale sono disponibili a ricevere il servizio. Questa caratteristica non è presente nel nostro problema, quindi il confronto diretto è molto difficile.

Uno dei primi articoli che trattava questa classe di problemi è di Cheng e Rich (1998). Nel problema si desidera assegnare servizi, i quali hanno associata una time windows, a lavoratori a tempo pieno e tempo parziale (*part-time*). La pianificazione per i lavoratori a tempo pieno include una pausa pranzo. L'obiettivo è minimizzare i costi del tempo di lavoro straordinario dei lavoratori a tempo pieno e parziale. Gli autori propongono un modello matematico e una semplice euristica per darne soluzione.

Laps Care (Eveborn et al., 2006) è un sistema adottato in Svezia per i servizi a domicilio (non necessariamente servizi sanitari), ed è capace di prendere in considerazione alcune delle questioni dell'HHCP. Il problema considerato dagli autori di Laps Care è formulato come un modello di partizionamento (*set partitioning model*) ed è risolto mediante un metodo iterativo. La soluzione iniziale propone per ogni servizio un percorso che lo contiene e i percorsi sono fusi fino a quando non è possibile ottenere un ulteriore miglioramento della soluzione. Per fuggire dall'ottimo locale, uno dei percorsi è diviso in dapprima uno dei percorsi è frazionato in tanti percorsi quanti sono i servizi, un percorso per servizio, dopodiché viene nuovamente attivata la fase di fusione fra percorsi (*merging*). Un altro sistema integrato è presentato in Begur et al. (1997a), dove l'attenzione è concentrata principalmente sulla componente GIS del sistema associato di supporto alle decisioni. Il problema è ridotto ad un instradamento di veicoli ed è risolto per mezzo di euristiche classiche.

Un sistema di domiciliazione di servizi sanitari simile a quello proposto nel presente lavoro è stato analizzato da Borsani et al. (2006). Uno degli obiettivi di quel sistema è bilanciare il carico di lavoro, mentre alcuni *soft constraints* di preferenze sono imposti all'assegnamento infermiera-paziente e ai giorni della settimana per le visite. Un altro obiettivo è la minimizzazione del numero di visite scoperte che devono essere esternalizzate. Gli autori propongono un semplice modello di assegnamento ignorando le componenti d'instradamento, poiché il tempo di visita, che include il tempo di viaggio, è deterministicamente fissato ad una costante. Quest'ipotesi può semplificare eccessivamente il problema se il servizio territoriale è ampio e il tempo di viaggio è altamente variabile, per esempio quando varia da cinque minuti a un'ora.

Rasmussen et al. (2012) considera uno specifico sottoproblema costituito dalla pianificazione giornaliera delle infermiere. In questo problema la funzione obiettivo rappresenta molti fattori prevalentemente connessi alla qualità del servizio (come i servizi scoperti o una funzione di priorità) invece che i costi operativi, che sono difficilmente applicabili ad un orizzonte temporale di un singolo giorno. Il problema è un'estensione dell'ampiamente studiato *Vehicle Routing Problem with Time Windows* (vedere per esempio Baldacci et al. (2012)). Gli autori propongono una formulazione set partitioning e un algoritmo esatto di branch-and-price, il quale sfrutta una speciale regola di branching, traendo vantaggio dalla struttura del problema.

Bertels e Fahle (2006) adottano un approccio ibrido combinando constraint programming, local search e programmazione matematica. Essi sfruttano la presenza di finestre temporali strette, le quali limitano il numero di soluzioni ammissibili.

Un altro lavoro che usa un approccio ibrido tra constraint programming e programmazione matematica è quello di Nickel et al. (2012). Qui gli autori considerano il problema di assegnare i pazienti alle infermiere su una base settimanale e di costruire i programmi giornalieri. L'obiet-

tivo è duplice: da un lato gli autori minimizzano il tempo di lavoro straordinario e la lunghezza dei percorsi; dall'altro lato rappresentano un fattore di fidelizzazione privilegiando l'assegnazione della stessa infermiera allo stesso paziente. Anche in questo caso, le finestre temporali sono presenti. Il problema è scomposto in un problema principale di pianificazione settimanale e in un sottoproblema di pianificazione giornaliera al fine di adeguare il servizio a cambiamenti dell'ultimo minuto, il che è molto comune nell'assistenza domiciliare sanitaria.

Un approccio completamente differente per gestire la variabilità delle richieste dell'ultimo minuto è quella di Koeleman et al. (2012), dove un processo di Markov cattura l'aspetto stocastico del problema. Un altro modello stocastico usato nella pianificazione a medio termine è proposto da Lanzarone et al. (2010).

Poiché non ci sono finestre temporali nell'HHCP, questo problema potrebbe essere considerato in linea di massima come un instradamento di veicoli con vincolo sulla distanza (*Distance-Constrained Vehicle Routing Problem*, DCVRP) (Laporte et al., 1984). Tuttavia, gli algoritmi sviluppati per il DCVRP non possono essere efficientemente adottati per l'HHCP, in particolare a causa della necessità di equilibrare il carico di lavoro tra le infermiere. Nell'HHCP, questa componente della funzione obiettivo si comporta come un collo di bottiglia (funzione min-max) ed è difficile da affrontare. Infine, il fattore di fidelizzazione è un'altra componente particolare della funzione obiettivo dell'HHCP, che lo rende significativamente differente da altri problemi.

Cattafi et al. (2012) adottano constraint logic programming per calcolare una soluzione di un sottoproblema di quello da noi affrontato in questa tesi, ottenuto fissando il giorno di esecuzione per ogni servizio; inoltre, il carico di lavoro è considerato settimanalmente e la matrice delle distanze è simmetrica. Infine, Rendl et al. (2012) utilizza la programmazione a vincoli per produrre una soluzione iniziale al problema relativo al singolo giorno in cui le infermiere possono utilizzare diversi mezzi di trasporto; diverse meta euristiche vengono confrontate per il miglioramento della soluzione iniziale.

Struttura del lavoro

L'HHCP modellato nella sua interezza ha una complessità difficilmente gestibile. Nei casi esplorati in letteratura, il problema reale è semplificato riconducendolo a problemi noti, o a loro varianti. Mantenere il più possibile le richieste del problema reale è una sfida colta dal presente lavoro. Esso contiene quattro contributi principali: la modellizzazione di un nuovo problema reale di instradamento, lo studio di metodi di programmazione matematica per la soluzione, l'analisi dei dati reali e la messa a punto di meta euristiche atte a risolvere l'HHCP.

Comprendere a priori se è possibile applicare un metodo di soluzione non è sempre possibile, e diviene particolarmente difficile se il problema è complesso e inesplorato in letteratura. La ricerca di un metodo per ottenere soluzioni ammissibili al problema è stata svolta su diversi livelli. In primo luogo usando la programmazione matematica, sia per ottenere una soluzione che un lower bound. Poiché i metodi esatti presi in esame si sono dimostrati inefficaci per gestire la complessità del HCCP nella sua completezza, abbiamo esplorato sottoproblemi, ad esempio il DCVRP. In secondo luogo abbiamo esplorato metodi euristici. Mentre nei classici problemi

di routing, quali CVRP e TSP, dopo aver fatto una mossa è immediato valutare l'ammissibilità della nuova soluzione, nel caso dell'HHCP il semplice spostamento di un servizio in un giorno diverso o a un'infermiera diversa richiede una fase di ripristino computazionalmente più oneroso per poter recuperare l'ammissibilità della soluzione. Sovente le euristiche hanno un insieme di mosse in comune: pertanto abbiamo astratto il concetto di mossa in modo da proporne di altre. Questo procedimento ha avuto una ricaduta positiva, permettendo di generalizzare una classica euristica costruttiva come la Clarke & Wright, in modo da essere applicata al problema HHC.

Verrà data una formulazione matematica dell'HHCP (capitolo 1), e sarà approfondito l'aspetto del bilanciamento di carico (capitolo 2). I metodi di programmazione matematica possono avere un 'tallone d'Achille' nella complessità computazionale degli algoritmi e nelle esose richieste di memoria per la rappresentazione di modelli complessi. Per mezzo di moderni sistemi paralleli da 2048 core è possibile rendere accettabili i tempi di soluzione di problemi NP-Hard complessi o di grandi dimensioni (Shinano et al., 2012; Applegate et al., 2006). Metodi di programmazione matematica, troppo costosi per gli elaboratori elettronici del tempo, sono recentemente applicati con successo in contesti reali, come per esempio nella decomposizione di Benders (Hooker, 2005; Contreras et al., 2011); inoltre, sono usati in sinergia con metodi euristici, dando luogo alle *matheuristic* (Boschetti et al., 2009). La prospettiva dell'aumento della potenza di calcolo, la disponibilità di macchine altamente parallele e la velocità nella soluzione di certi sottoproblemi, come il problema del commesso viaggiatore (*Traveling Salesman Problem*, TSP) (Roberti e Toth, 2012), hanno motivato lo studio dell'applicazione di tecniche di programmazione matematica all'HHCP: in questa tesi consideriamo la decomposizione di Benders (capitolo 3), la column generation (capitolo 4) e la ricerca di lower bound (capitolo 5).

In seguito saranno introdotte le meta euristiche basate sugl'intorni (*neighbourhood-based*) *variable neighbourhood search* (VNS), *adaptive large neighbourhood search* (ALNS) , ed un metodo per fornire una soluzione iniziale dalla quale partire (capitolo 6). La VNS e la ALNS, partendo da una data soluzione, si spostano mediante mosse ad un'altra soluzione: la messa a punto di mosse che colgano i diversi aspetti del problema è presentata nel capitolo 7. Verifiche di ammissibilità preventive e successive ad una mossa sono descritte nel capitolo 8. Avendo generalizzato il concetto di operatore, nel capitolo 9, saremo in grado di estendere l'euristica Clarke & Wright per l'HHCP.

Prima di descrivere come sono state generate le istanze per la sperimentazione numerica (capitolo 11), osserviamo le caratteristiche dello storico delle richieste reali fornito dall'AUSL di Ferrara (capitolo 10). Infine descriviamo la scelta dei parametri nel capitolo 12, per poi mostrare nel capitolo 13 i risultati delle computazioni.

Parte I

Home Health Care Problem

Sommario della parte I

L'obiettivo di questa prima parte della tesi è fornire gli strumenti fondamentali per la rappresentazione e la comprensione del problema allo studio. I capitoli che seguono introducono, da un lato, il formalismo matematico indispensabile per impostare in modo corretto e non ambiguo la descrizione del modello e, dall'altro lato, presentano una caratterizzazione accurata degli aspetti cruciali dell'assistenza domiciliare.

Fra questi ultimi riveste certamente un ruolo fondamentale la questione del bilanciamento del carico di lavoro.

Un elemento di novità di questa tesi sta nell'aver individuato un modo di quantificare tale carico che risulta idoneo al trattamento nel contesto dei problemi di HHC.

Come risulterà chiaro dal capitolo 2, la formulazione del bilanciamento di carico qui proposta coglie aspetti importanti per il caso reale, che altre modalità di formalizzazione non consentono di trattare. Un esempio significativo di questa debolezza di altri approcci si ha nel caso in cui si voglia quantificare il bilanciamento del carico di lavoro tra diverse infermiere, utilizzando la differenza tra il numero massimo ed il numero minimo delle ore settimanali lavorate: con quest'approccio si incorre inevitabilmente nell'inconveniente che la soluzione del problema rimane ammissibile anche quando si ha una distribuzione poco uniforme del carico tra le infermiere.

Va comunque sottolineato che si tratta di una questione rilevante nella pratica, in quanto influisce pesantemente sulla qualità del servizio erogato e sulla qualità della vita dell'operatore.

Capitolo 1

Il problema dell'assistenza sanitaria domiciliare

Lo scopo principale di questo capitolo è definire la notazione per formalizzare il problema con un modello di programmazione matematica. Alla fine del capitolo saremo in grado di osservare come l'HHCP generalizzi problemi di instradamento di veicoli noti.

1.1 Notazione

La pianificazione dei turni equivale alla distribuzione del lavoro lungo un *orizzonte temporale*, e viene modellata con un insieme H di fascie orarie h, ad esempio la sequenza turno del mattino e turno del pomeriggio di ogni giorno di un mese. In seguito parleremo di *giorno* h, per sintetizzare il generico concetto di fascia oraria h. Analogamente, per sintesi e leggibilità l'insieme di fascie orarie H sarà chiamato *settimana*, pertanto composto di sette giorni, ossia $|H| = 7$. Denotiamo con il termine *servizio* la coppia $s = (\texttt{tipo}_s, p_s)$ in cui $\texttt{tipo}_s$ e p_s sono il tipo di trattamento e il paziente che lo richiede, rispettivamente. S è l'insieme di servizi. Per semplicità denotiamo con 0 (zero) la *centrale operativa*, luogo dove i viaggi delle infermiere hanno inizio e fine. Ad un servizio è associato un luogo che per leggibilità li denotiamo con lo stesso simbolo s usato per il servizio. Potendo usare in modo indistinguibile s per denotare il servizio ed il luogo ove deve essere svolto, analogamente possiamo usare la notazione $S^0 = S \cup \{0\}$, per indicare l'insieme dei luoghi, inclusa la centrale operativa. Ogni servizio s ha una frequenza f_s che è il numero di ripetizioni di tale servizio nell'orizzonte temporale H. Tra due ripetizioni consecutive del servizio s, un tempo minimo e_s deve intercorrere, il quale sarà chiamato *intermezzo*. Definiamo $S_+ = \{s \in S \mid e_s > 0\}$ l'insieme di servizi che devono essere ripetuti in giorni distinti. Denotiamo con P l'insieme di pazienti e con S_p l'insieme di servizi richiesti dal paziente p. Sia N l'insieme di infermiere e N_h quelle che sono in turno il giorno h. Conosciamo il numero di giorni b_j nei quali l'infermiera j è in servizio durante H.

Una sequenza di servizi T viene detta *turno giornaliero*, o *tour*. I tour possono essere accorpati in una famiglia di tour $\mathcal{T}$. Nella maggior parte dei casi, useremo la notazione T_i, per

indicare l'i-esimo tour della famiglia di tour $\mathcal{T}$, poiché non è noto il giorno di svolgimento e a quale infermiera è assegnato il tour i.

Quando si conosce la temporanea assegnazione di un tour all'infermiera j e al giorno h, denotiamo il tour con T^{jh}. I turni assegnati il giorno h sono contenuti nell'insieme denotato con $\mathcal{T}^h$, ed analogamente $\mathcal{T}^j$ contiene i turni assegnati all'infermiera j. Pertanto $\mathcal{T} = \bigcup_{h\in H} \mathcal{T}^h = \bigcup_{j\in N} \mathcal{T}^j$.

Diversamente dall'enumerazione dei tour appartenenti alla famiglia $\mathcal{T}$ data dalla coppia di indici (j, h), il primo stile di indicizzazione permette di enumerare famiglie di tour molto grandi, ossia $|\mathcal{T}| > |N \times H|$. Inoltre il secondo stile di indicizzazione può essere sempre ricondotto al primo usando l'inversione della funzione coppia di Cantor[1].

Un paziente potrebbe non essere sempre disponibile a domicilio, per esempio tre giorni alla settimana il paziente è in ospedale per sottoporsi a emodialisi, oppure un servizio potrebbe avere la necessità di essere richiesto solo in certi giorni, per esempio i giorni feriali o i giorni di apertura di un laboratorio analisi (eventi indipendenti dalla presenza in turno dell'infermiera). Pertanto, modelliamo la *disponibilità* del servizio mediante il parametro binario g_s^h così definito da:

$$g_s^h = \begin{cases} 1 & \text{se il servizio } s \text{ può essere compiuto nel giorno } h \\ 0 & \text{altrimenti} \end{cases} \tag{1.1}$$

La disponibilità settimanale del servizio in H è denotata dall'insieme G_s delle coppie (g_s^h, h): $G_s = \cup_{h\in H}\{g_s^h\} \times \{h\}$.

Denotiamo con $S^h = \{s \in S \mid g_s^h = 1\}$ l'insieme dei servizi che possono essere svolti nel giorno $h \in H$, e con $S_+^h = S^h \cap S_+$ i servizio che possono essere svolti il giorno h e che richiedono un intermezzo. Essendo i turni $T_i \in \mathcal{T}$ delle sequenze di servizi $s \in T_i$, possiamo definire l'insieme di giorni per il quale il turno può essere svolto come $\overline{G}_i = \cup_{h\in H}\{q_i^h\} \times \{h\}$, con q_i^h così definito:

$$q_i^h = \bigwedge_{s\in T_i} g_s^h \,. \tag{1.2}$$

Pertanto, se i servizi contenuti nel tour i possono essere svolti ogni giorno della settimana, cioè $\forall s \in T_i \, (\bigwedge_{k\in H} g_s^k = 1)$, allora anche il tour i è possibile svolgerlo in uno qualsiasi, cioè $\bigwedge_{k\in H} q_i^k = 1$. Diversamente, un giorno h della settimana sarà unico e prefissato, se $\bigvee_{k\in H\setminus\{h\}} q_i^k = 0$ e $q_i^h = 1$ cioè tutti i servizi hanno un solo giorno comune di svolgimento.

Il tipo di notazione definita è contestualizzata per un'applicazione sanitaria. Sebbene una generalizzazione aiuterebbe a porre in evidenza la duttilità del modello per contesti differenti, abbiamo optato per aiutare la leggibilità nei vari punti ove i dati tratti dal caso reale ne richiedono l'uso. Questo problema trova applicazione nei seguenti ambiti. Nel settore della ristorazione, dove sovente il rappresentante si occupa anche di consegne di alimentari, sono presenti inoltre la periodicità dei servizio, la differenza tra servizi di rappresentanza (mostrare nuovi prodotti) e scarico merci. Anche nel settore militare vi sono servizi, come ad esempio il trasporto di detenuti in ospedale per una visita e la verifica della presenza di detenuti agli arresti domiciliari, che possono presentare una periodicità e un diverso tempo di svolgimento.

[1] $i = \langle j, h\rangle = \frac{(j+h)(j+h+1)}{2} + h$. $w = \lceil(\sqrt{8i+1} - 1)/2\rceil$; $t = (w^2 + w)/2$; $h = i - t$; $j = w - h$.

1.2 Formulazione

Consideriamo una possibile formulazione suddivisa in due aspetti decisionali: *pianificazione* (*scheduling*) e *instradamento* (*routing*). La pianificazione si occupa di *stabilire* i giorni nei quali il servizio è ripetuto e di *conferire* il servizio ad una o più infermiere nei giorni stabiliti. Quindi la pianificazione è in grado di identificare i turni e le infermiere che li coprono. L'aspetto d'instradamento, invece, riguarda il sequenziamento dei servizi appartenenti a ciascun turno. La pianificazione è modellata dalle variabili:

$$y_s^{jh} = \begin{cases} 1 & \text{se il servizio } s \text{ è assegnato all'infermiera } j \text{ nel giorno } h \\ 0 & \text{altrimenti} \end{cases}$$

Al fine di modellare la sequenza di visita, per ogni coppia (j, h) con $h \in H$ $j \in N_h$ introduciamo le seguenti variabili:

$$x_{ss'}^{jh} = \begin{cases} 1 & \text{se l'infermiera } j \text{ compie consecutivamente} \\ & \text{i servizi } s \text{ ed } s' \text{ nel giorno } h \\ 0 & \text{altrimenti} \end{cases}$$

L'aspetto della fidelizzazione paziente-infermiera è modellato dal numero di infermiere diverse che servono un paziente p durante una settimana. Per questo conteremo le occorrenze non nulle delle seguenti variabili:

$$z_p^j = \begin{cases} 1 & \text{se l'infermiera } j \text{ serve il paziente } p \text{ almeno una volta} \\ 0 & \text{altrimenti} \end{cases}$$

Pertanto, $\sum_{j \in N} z_p^j$ è il numero di infermiere che servono il paziente p.

Un servizio s si conclude dopo un tempo di lavoro a_s. Nel caso della centrale operativa 0 il tempo di lavoro è nullo, ovvero $a_0 = 0$. Indichiamo con $d_{ss'}$ il tempo di spostamento necessario tra il servizio s e il servizio s'. Un turno svolto nel giorno h ed assegnato ad un'infermiera j ha un *tempo di lavoro* esprimibile come

$$\ell^{jh} = \sum_{s \in S_0^h} \sum_{s' \in S_0^h} (a_{s'} + d_{ss'}) x_{ss'}^{jh} \qquad \forall h \in H,\ \forall j \in N_h\,. \tag{1.3}$$

Desideriamo che ℓ^{jh} non ecceda la soglia predeterminata L, che denota il *massimo tempo di lavoro giornaliero*, altrimenti il *lavoro straordinario* w dovrà essere cospicuamente remunerato. La quantità massima W^{max} giornaliera di ore di straordinario concesse ad ogni infermiera nella pratica è finita: infatti, poiché ci sono al più 24 ore in una giorno, non è realistico impostare W^{max} ad un valore superiore a $24 - L$ ore. Riterremo pertanto inammissibili le soluzioni che superano di oltre W^{max} la soglia L. Tuttavia in seguito utilizzeremo la notazione $W^{max} = \infty$ per indicare i casi in cui vogliamo trascurare il limite superiore sulla corrispondente variabile w.

Misuriamo il *carico di lavoro* con la variabile u: essa denota il massimo tempo lavorativo medio giornaliero. Con le definizioni fin qui introdotte, possiamo ora formulare il problema generale dell'assistenza sanitaria domiciliare (HHC) come un problema vincolato di programmazione matematica:

$$\min \ \alpha_0 u + \alpha_1 \sum_{p\in P}\left(\sum_{j\in N} z_p^j - 1\right) + \alpha_2 w + \alpha_3 \sum_{h\in H}\sum_{j\in N_h} \ell^{jh} \tag{1.4}$$

$$\text{s. t.} \sum_{h\in H}\sum_{j\in N_h} y_s^{jh} = f_s \qquad \forall s \in S \tag{1.5}$$

$$\sum_{k=h}^{h+e_s-1}\sum_{j\in N_h} y_s^{jk} \le 1 \qquad \forall s \in S_+,\ \forall h \in \{1,\dots,|H|-e_s\} \tag{1.6}$$

$$\sum_{s'\in S_0^h\setminus\{s\}} x_{ss'}^{jh} = y_s^{jh} \qquad \forall h \in H,\ \forall s \in S^h,\ \forall j \in N_h \tag{1.7}$$

$$\sum_{s'\in S_0^h\setminus\{s\}} x_{s's}^{jh} = y_s^{jh} \qquad \forall h \in H,\ \forall s \in S^h,\ \forall j \in N_h \tag{1.8}$$

$$\sum_{s'\in S^h} x_{0s'}^{jh} = 1 \qquad \forall h \in H,\ \forall j \in N_h \tag{1.9}$$

$$\sum_{s'\in S^h} x_{s'0}^{jh} = 1 \qquad \forall h \in H,\ \forall j \in N_h \tag{1.10}$$

$$x_{ss'}^{jh} = 1 \Rightarrow t_{s'}^{jh} \ge t_s^{jh} + 1 \qquad \forall h \in H,\ \forall s \in S_0^h,\ \forall s' \in S^h \setminus \{s\},\ \forall j \in N_h \tag{1.11}$$

$$\ell^{jh} - L \le w \qquad \forall h \in H,\ \forall j \in N_h \tag{1.12}$$

$$u \ge \frac{1}{b_j}\sum_{h:j\in N_h} \ell^{jh} \qquad \forall j \in N \tag{1.13}$$

$$z_p^j \ge y_s^{jh} \qquad \forall h \in H,\ \forall j \in N_h,\ p \in P,\ s \in S_p \cap S^h \tag{1.14}$$

$$y_s^{jh} \le g_s^h \qquad \forall s \in S^0,\ \forall h \in H,\ \forall j \in N_h \tag{1.15}$$

$$w \le W^{max} \tag{1.16}$$

$$z_p^j \in \{0,1\} \qquad \forall j \in N,\ p \in P \tag{1.17}$$

$$x_{s's}^{jh} \in \{0,1\} \qquad \forall s' \in S^0,\ \forall s \in S^0,\ \forall h \in H,\ \forall j \in N \tag{1.18}$$

$$y_s^{jh} \in \{0,1\} \qquad \forall s \in S^0,\ \forall h \in H,\ \forall j \in N \tag{1.19}$$

$$t_s^{jh} \in \mathbb{R} \qquad \forall s \in S^0,\ \forall h \in H,\ \forall j \in N_h \tag{1.20}$$

$$u \in \mathbb{R}^+ \tag{1.21}$$

$$w \in \mathbb{R}^+ \tag{1.22}$$

Vediamo ora il significato delle varie componenti del problema di programmazione matematica. La funzione obiettivo (1.4) modella diversi aspetti del problema HHC, i quali possono essere mutuamente fissati impostando opportunamente i parametri $\alpha_i \quad i = 0,\dots,3$. Per la parte

d'instradamento si desiderano minimizzare il massimo carico di lavoro medio giornaliero u, i tempi di lavoro straordinario w, ed il tempo di lavoro totale,

$$\ell = \sum_{h \in H} \sum_{j \in N_h} \ell^{jh}, \tag{1.23}$$

con ℓ^{jh} definita dall'equazione (1.3). Per la parte di pianificazione, nella funzione obiettivo si minimizza, per ogni paziente, il numero di infermiere che lo servono. Noi consideriamo prioritario l'abbattimento dei tempi di lavoro straordinario, successivamente riteniamo importanti il bilanciamento di carico di lavoro medio e la fidelizzazione paziente-infermiere. Trascureremo il tempo totale di viaggio, sebbene sia utile averlo considerato per mostrare che il problema HHC generalizza altri problemi noti.

In (1.5) garantiamo che un servizio s sia servito f_s volte. I vincoli (1.6) assicurano che passi un intermezzo minimo e_s passi prima che la mansione sia ripetuta. I vincoli (1.7) e (1.8) uniscono la parte di pianificazione con la parte di instradamento e assicurano che per ogni servizio svolto esiste un solo predecessore e un solo successore in uno stesso giorno h. In (1.9) e (1.10) viene imposto che ogni infermiera abbia un solo turno al giorno.

I vincoli (1.11) per eliminare i sottocicli sono opportunamente linearizzati (Miller et al., 1960). Nei vincoli (1.12), per ogni giorno e per ogni infermiera, il tempo di lavoro straordinario w è misurato come eccedenza del massimo tempo di lavoro giornaliero L. In (1.14), se un'infermiera j compie un servizio s, allora deve risultare che l'infermiera j serve il paziente p associato a tale servizio (ricordiamo che per definizione $s = (\texttt{tipo}_s, p_s)$).

Un aspetto interessante è la possiblità di predefinire uno storico della fidelizzazione: questo si ottiene preimpostando $z_p^j = 1$ se nel passato l'infermiera j ha servito il paziente p (per esempio se nelle due settimane precedenti, il paziente p era stato servito dall'infermiera j. Questa parametrizzazione di alcuni valori di z_p^j spinge il metodo di soluzione ad associare quell'infermiera ai servizi di quel paziente, in quanto l'evento opposto (cioè la non associazione) viene penalizzato nella funzione obiettivo.

Considerando la disponibilità g dei servizi nei vincoli (1.15) possiamo predeterminare una riduzione del dominio delle possibili associazioni y_s^{jh}, e di conseguenza delle variabili x.

Il vincolo (1.16) limita superiormente il numero di ore di straordinario concesse alle infermiere.

Osserveremo che l'analisi dei dati storici delle richieste ha fatto emergere casi non contemplati dal precedente modello, ovvero il fatto che in un luogo viene erogato lo stesso servizio più volte nella stessa giornata. Per motivi di privacy, conosciamo solo l'indirizzo ove è stato svolto il servizio, pertanto non abbiamo modo di sapere se esso è stato erogato in due momenti della giornata ad uno stesso paziente, oppure se è stato erogato a due persone differenti. Non siamo a conoscenza della motivazione esatta di questo fenomeno: potrebbe essere che il servizio, ad esempio una medicazione complessa che richiede quasi un'ora, essendo erogato a due coniugi è riportato due volte. Quest'ultimo esempio può essere gestito dal modello precedente attuando la seguente trasformazione. Occorre gestire il caso in cui $e_s = 0$ e $f_s = k$, ossia il caso in cui in uno stesso giorno un paziente riceve più volte lo stesso servizio. Osserviamo che, avendo

per semplicità assunto nullo l'intermezzo, il vincolo sulla frequenza non influisce sulla loro successione giornaliera: per questo motivo possiamo dividere il servizio s in k servizi $s_1, \ldots, s_k$, i quali, essendo per lo stesso paziente p_s, nelle nostre ipotesi hanno distanza $d_{s_i s_j}$ nulla tra loro. Con questa scomposizione il vincolo della frequenza di servizio è gestito dalla suddivisione in k servizi ed il tempo di viaggio posto a zero ne incentiva la visita consecutiva, ossia nello stesso turno.

1.3 Sottoproblemi

In questa sezione mostriamo come il modello appena presentato rappresenta una generalizzazione di svariati problemi d'instradamento NP-Hard. Ne segue che l'HHCP eredita la notevole complessità computazionale necessaria per trovare soluzioni ammissibili, ottime o bound.

1.3.1 Multiple Traveling Salesman Problem

Visitare tutti i nodi di un grafo una e una sola volta è detto "problema del ciclo hamiltoniano". Se il grafo è pesato, ossia ad ogni arco corrisponde un peso, trovare tra tutti i cicli hamiltoniani quello di costo minore è detto "problema del commesso viaggiatore" (Traveling Salesman Problem, TSP). Se il nodo di partenza può essere visitato più di una volta, allora il problema è detto multiple Traveling Salesman Problem (m-TSP) (Bektas, 2006).

Riduciamo l'HHCP ad un m-TSP impostando come segue i parametri: con $W^{max} = \infty$ non si limita il tempo di viaggio di un singolo turno; con $\alpha_0 = \alpha_1 = \alpha_2 = 0$ e $\alpha_3 = 1$ si desidera minimizzare solo il tempo totale di viaggio; $f_s = 1$ e $e_s = -1$ richiedono che un servizio è svolto una volta sola; per avere massima disponibilità, potendo svolgere sempre il servizio in qualsiasi giorno della settimana, $g_s^h = 1 \ \forall s \forall h$.

In letteratura, ci sono varie trasformazioni che permettono di ricondurre l'm-TSP al TSP nelle diverse varianti relative a diverse proprietà del grafo di partenza (Lenstra e Rinnooy Kan, 1975; Hong e Padberg, 1977; Rao, 1980; Jonker e Volgenant, 1988; Laporte e Nobert, 1980; Bellmore e Hong, 1974). Queste trasformazioni permettono di avvalersi di efficienti risolutori *ad hoc* per il TSP (Applegate et al., 2006).

1.3.2 Distance Constrained Vehicle Routing Problem

Il problema di instradamento di veicoli con capacità (Capacited Vehicle Routing Problem; CVRP), è un m-TSP ove ad ogni veicolo è associata una capacità che non deve essere superata (Toth e Vigo, 2001). Invece, quando abbiamo un m-TSP con un tempo limite L per tornare al deposito allora ci si riferisce al problema dell'instradamento di veicoli con distanza vincolata (Distance Constrained Vehicle Routing Problem; DVRP) (Laporte et al., 1987). Poiché il problema dell'instradamento dinamico di veicoli (Dynamic Vehicle Routing Problem), con omonima sigla, ha ricevuto maggiore attenzione in letteratura, è usuale riferirsi al DVRP come al problema d'instradamento con capacità e distanza vincolata (Distance Constrained Capacited Vehicle Routing Problem; DCVRP) (Laporte et al., 1985), rilassando il vincolo di capacità.

Uniformandoci alla letteratura, in questo lavoro al posto di DVRP useremo il termine DCVRP sottintendendo il rilassamento del vincolo di capacità. L'HHCP può essere ridotto al DCVRP con gli stessi parametri usati per l'm-TSP e con $W^{max} = 0$, per limitare i tempi di viaggio totali dei tour.

Il DCVRP è un problema dove ad ogni nodo corrisponde una stessa finestra temporale di visita $[0, L]$. La generalizzazione del DCVRP è data da un problema d'instradamento con finestre temporali (Vehicle Routing Problem with Time Windows; VRPTW) (Kolen et al., 1987), dove le finestre temporali relative ai nodi sono di dimensioni eterogenee. Riguardo alla ricerca di un lower bound per il DCVRP la letteratura è molto limitata. Al meglio della nostra conoscenza, non esiste un riscontro del successo dell'applicazione al DCVRP dei metodi ottimi, o di approssimazione, proposti per il VRPTW, sebbene quest'ultimo problema abbia ricevuto maggiore attenzione.

Come confermano altri autori (Mendoza, 2009), abbiamo constatato che, dal 1985 ad oggi, il DCVRP ha avuto scarsa attenzione in letteratura. Tuttavia, nella nostra esperienza (Boccafoli et al., 2011) il problema trova svariate applicazioni reali derivate dal rispetto dei limiti di tempo di lavoro. Sebbene esistano proposte che affrontano la soluzione del problema DCVRP mediante approcci euristici, è arduo comprendere a priori la bontà di un metodo, poiché l'insieme delle istanze di un caso reale raramente presenta le stesse caratteristiche di un altro caso reale.

Talvolta, il vincolo della massima durata di un turno è incluso in un modello con finestre temporali (Steeg, 2008). In questo caso, rispetto al DCVRP il dominio delle soluzioni diminuisce poiché il limite viene assorbito dalla finestra temporale $[l_v, u_v]$ del nodo v, in quanto la finestra temporale $[0, L]$ è l'elemento neutro dell'operazione di intersezione: $[l_v, u_v] \cap [0, L] = [l_v, u_v]$. In altre parole, un VRPTW con vincolo di massima durata del turno L mantiene la stessa complessità di un VRPTW. Diversamente, si osservi che nell'HHCP, fissando il giorno di ciascun servizio, per ogni giornata il sottoproblema è un DCVRP: pertanto, la disponibilità del giorno di servizio g_s^h, che condiziona il valore di y_s^{jh} nel vincolo (1.16), non riduce la complessità del sottoproblema di DCVRP.

1.3.3 Periodic Vehicle Routing Problem

Il problema d'instradamento periodico di veicoli (Periodic Vehicle Routing Problem; PVRP) è un problema di instradamento nel quale il servizio viene ripetuto più volte in un orizzonte temporale (Tan e Beasley, 1984; Mourgaya e Vanderbeck, 2006). Per ridurre l'HHCP al PVRP occorre impostare i parametri come segue: si minimizza solo il tempo totale di viaggio (ponendo $\alpha_0 = \alpha_1 = \alpha_2 = 0$ e $\alpha_3 = 1$), non si impone un tempo limite per turno ($W^{max} = \infty$), non si prevede alcun intermezzo prima di una ripetizione ($e_s = -1$). La disponibilità g_s^h è compresa in molti modelli del PVRP, tuttavia è usuale semplificare il modello scegliendo di svolgere sempre il servizio in qualsiasi giorno della settimana, cioè assume $g_s^h = 1\ \forall s \forall h$.

In due articoli recenti (Mourgaya e Vanderbeck, 2007; Francis et al., 2008), il metodo della generazione di colonne è stato proposto per risolvere in modo esatto il problema PVRP. Per la potenza di calcolo attualmente a disposizione, è improponibile applicare tale metodo al

sottoproblema del HHCP, perché qui le dimensioni delle istanze dei casi reali sono cinque o sei volte più grandi.

Sovente la generalizzazione dei problemi aggiunge vincoli e variabili, che la rendono inapplicabile per l'aumento sensibile delle dimensioni della matrice dei vincoli. Ad esempio, in Ribeiro e Ramalhinho Dias Lourenço (2001) modellano una generalizzazione del PVRP, il Multi Objectives Multi Period Distribution Management Problem. Gli autori documentano che non hanno ottenuto soluzioni ammissibili dopo due giorni di calcolo con un istanza minimale ($|H| = 4, |N| = 2, |P| = 5$). In altre circostanze, il metodo di programmazione matematica è inapplicabile al caso reale poiché le proprietà che lo caratterizzano, ad esempio disuguaglianza triangolare, non sono mantenute.

Oltre alle prime euristiche per il PVRP (Tan e Beasley, 1984; Christofides e Beasley, 1984), ne sono state proposte altre, tra le quali citiamo l'approccio multifase (Russell e Gribbin, 1991), il tabu search (Cordeau et al., 2001), il VNS (Hemmelmayr et al., 2009), l'ALNS e programmazione vincolata (Costraing Programming, CP) (Steeg, 2008).

Capitolo 2

Carico di lavoro

Nell'ambito di attività a contatto con il pubblico, l'aspetto dell'esaurimento da lavoro, conosciuto come fenomeno di *burnout*, richiede particolare attenzione. L'osservazione di tale fenomeno in psicologia ha trovato correlazione con varie cause e dà origine a svariate conseguenze. Particolarmente delicato è il contesto delle cure mediche, dove l'operatore è a contatto con persone e con malattie (Deckard et al., 1994). Tra le conseguenze del fenomeno c'è il degrado emotivo, che si riperquote in famiglia e sul lavoro. La quantità di persone incontrate, la quantità di tempo lavorativo medio settimanale e la rotazione del personale, sono tra le cause del fenomeno. La priorità data alla qualità delle vite dell'operatore e del paziente si traduce nell'avere come obiettivi primari la limitazione della quantità di ore di straordinario e il bilanciamento del carico di lavoro del personale.

Vi sono varie metriche per misurare il carico di lavoro (Hughes, 1999). Alcuni lavori (Procter, 1992; Vitacca et al., 2000) osservano la correlazione tra il carico di lavoro ed il tempo lavorativo, e motivano l'interesse della gestione di questo aspetto. In questa sezione mostriamo le formulazioni del carico di lavoro (*workload*) nei vari settori, osservando quali di esse sono opportune per l'HHCP.

2.1 Contesto generale

Da almeno tre decenni (Stecke, 1983), la pianificazione è l'ambito applicativo dove il carico di lavoro suscita maggiore interesse, come dimostra l'ampia letteratura. Il settore più astratto della pianificazione spazia tra problemi di base, ad esempio i problemi di zaino (Martello e Toth, 1990; Mathur, 1998), e problemi arricchiti, ad esempio on-line load (Azar, 1998). Le applicazioni in casi reali sono svariate, come la distribuzione del carico di lavoro in una rete di computer (Rajakumar et al., 2004), o la distribuzione dell'elettorato (Bozkaya et al., 2003). Non esiste una nomenclatura uniforme. Talvolta è possibile trovare lavori nei quali al posto di 'bilanciamento' vengono usati i termini: "distribuzione uniforme", "equiripartizione", "equilibrio" (Blais et al., 2003). Esistono proposte di gestione del carico di lavoro applicabili a settori differenti: ad esempio l'equilibrio della distribuzione dell'elettorato in un distretto (Bozkaya et al., 2003) è applicato all'equilibrio del carico di lavoro nel servizio di assistenza sanitaria a domicilio

(Blais et al., 2003). Una sintesi della varietà di modelli per bilanciare il carico di lavoro può essere riassunta nelle tecniche riportate in recenti lavori per l'identificazione e la descrizione di vincoli per la programmazione vincolata (Pesant e Régin, 2005; Monette et al., 2007; Schaus et al., 2007), alle quali si aggiunge un approccio stocastico (Lanzarone e Matta, 2011). Occorre tuttavia entrare nel dettaglio per valutare l'applicabilità di tali metodi all'HHCP.

Una prima indagine è stata svolta nel campo della pianificazione dei turni delle infermiere (*Nurse Rostering Problem*), nel quale non si tiene conto della parte di routing. L'articolo di Ibrahim et al. (2010) aggiunge il riferimento ad alcune euristiche relative al bilanciamento di carico (Burke et al., 2004). Spesso i metodi sono relativi alla turnazione di un servizio in strutture ospedaliere, pertanto non gestiscono i tempi di spostamento tra pazienti. Il carico di lavoro è inteso talvolta come quantità di turni svolti in un orizzonte temporale, talvolta come numero di pazienti assegnati (Lahrichi, 2008), altre volte come tempo di lavoro.

Il carico di lavoro per l'HHCP ha ricevuto attenzione sin dai primi articoli (Begur et al., 1997b). Tuttavia, la complessità di gestire la soluzione di due sottoproblemi NP-Hard richiede spesso un'approssimazione, come per esempio l'uniformità dei tempi di viaggio e di servizio (Borsani et al., 2006), che porta a gestire il bilanciamento come quantità di servizi svolti. Nel seguito di questa sezione, riportiamo alcune tecniche modellistiche usate per problemi d'instradamento, osservando la loro applicabilità all'HHCP.

2.2 Forbici sul numero dei servizi e sulla distanza

Ricordiamo che possiamo ricondurre il nostro modello ad un m-TSP , come argomentato in sezione 1.3.1. Sutcliffe e Boardman (1990) applicano ad un caso reale, una tecnica proposta da Okonjo-Adigwe (1989), nella quale si limita, mediante lower e upper bound, il numero di nodi serviti ((2.1) e (2.2))

$$\lambda \leq \sum_{s \in S} y_s^{jh} \qquad \forall h \in H,\ \forall j \in N \tag{2.1}$$

$$\sum_{s \in S} y_s^{jh} \leq \upsilon \qquad \forall h \in H,\ \forall j \in N\,, \tag{2.2}$$

e la distanza dei singoli tour ((2.3) e (2.4))

$$\tfrac{L}{2} < \ell^{jh} \qquad \forall h \in H,\ \forall j \in N \tag{2.3}$$

$$\ell^{jh} \leq L \qquad \forall h \in H,\ \forall j \in N\,. \tag{2.4}$$

Come si può verificare in figura 2.1 (c), questi vincoli restituiscono soluzioni che non minimizzano né il tempo totale di servizio, né il carico di lavoro u. Tali valori sono invece ottenuti se si eliminano i bound, come mostra la figura 2.1 (b).

2.3 Tecniche di minima differenza

In letteratura è più frequente l'uso della misurazione del bilanciamento di carico come differenza tra due quantità, come il numero di servizi al giorno per paziente oppure il massimo e minimo

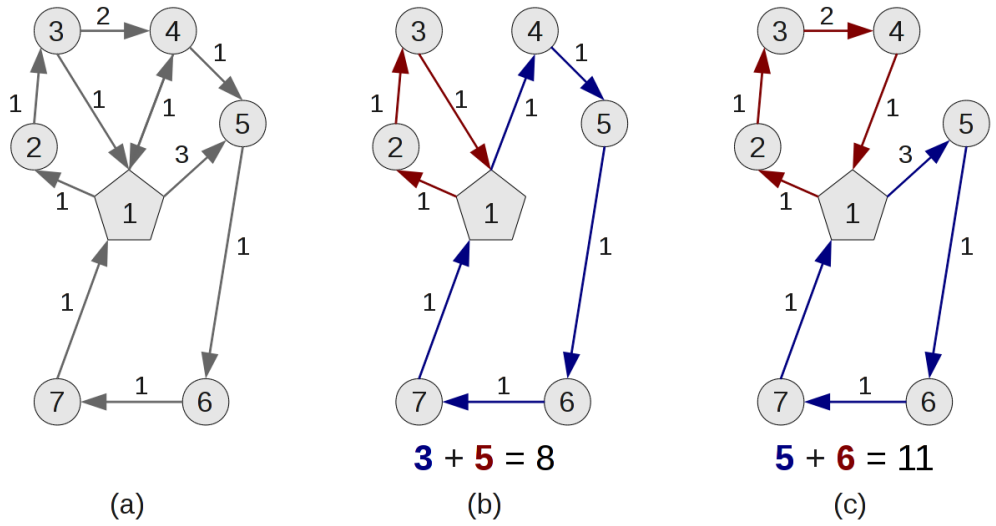


Figura 2.1: (a) grafo dell'istanza ove i nodi rotondi sono i servizi e il pentagono è il deposito, (b) m-TSP dell'istanza, (c) m-TSP con (2.1) *e* (2.2) *ove* $\lambda = 3$ *e* $v = 10$*, o anche m-TSP con* (2.3) *e* (2.4) *ove* $4 \leq \theta < 10$

tempo di lavoro. A seguire alcune proposte. Bredström e Rönnqvist (2008) considerano il problema di instradamento e pianificazione senza periodicità. Il metodo di bilanciamento proposto è la minimizzazione della massima differenza ζ tra tutti i possibili tour mediante i seguenti vincoli:

$$\ell^{jh} - \ell^{j'h} \leq \zeta \qquad \forall h \in H,\ \forall j \in N,\ \forall j' \in N \setminus \{j\} \tag{2.5}$$

Lee e Ueng (1999) modellano la gestione del bilanciamento di carico mediante la minimizzazione della differenza tra i costi di tutte le rotte e la rotta di carico più basso per il numero dei veicoli meno uno. Formalmente, questa idea può essere inserita nel nostro modello considerando il caso di un orizzonte temporale di un solo giorno, sostituendo nella funzione obiettivo $\alpha_0 u$ con (2.6) ed aggiungendo un ordinamento tale per cui il veicolo con indice più basso, ossia 1, abbia tempo totale più basso (2.7):

$$\min \left(\sum_{j \in N \setminus \{1\}} \ell^{j1} \right) - (v-1)\ell^{11} \tag{2.6}$$

$$\text{s. t. } \ell^{11} \leq \ell^{j1} \qquad \forall j \in N \setminus \{1\} \tag{2.7}$$

El-Sherbeny (2001) minimizza il carico di lavoro come differenza tra la rotta più lunga e quella più corta, rispettivamente denotate con indici $j = |N|$ e con $j = 1$. Lo stesso tipo di modellizzazione del bilanciamento di carico è usato da Jozefowiez et al. (2002) e Jozefowiez et al. (2007) per il l'instradamento di veicoli con bilanciamento di route (Vehicle Routing Problem with route Balancing, VRPB). Fissando un giorno h, un'equivalente formulazione è data dalla composizione dei vincoli (2.7) e (2.9) con funzione obiettivo (2.8):

$$\min \ell^{|N|1} - \ell^{11} \tag{2.8}$$

$$\text{s. t. } \ell^{|N|1} \geq \ell^{j1} \qquad \forall j \in N \setminus \{|N|\} \tag{2.9}$$

Il concetto è esteso su tutto l'orizzonte temporale da Blakeley et al. (2003), dando luogo a:

$$\min v^- - v^+ \tag{2.10}$$

$$\text{s. t. } v^- \leq \ell^{jh} \leq v^+ \qquad \forall h \in H,\ \forall j \in N\,. \tag{2.11}$$

Tutte queste proposte di modellare il bilanciamento di carico trovano una buona applicazione nell'omogeneità dei dati. Quando un'infermiera lavora nei giorni festivi e prefestivi ha un turno molto scarico rispetto ai giorni feriali. Definendo quindi il bilanciamento come differenza tra turno più lungo e turno più corto, avviene che la distanza rimane ampia. Se si desidera modellare il bilanciamento cercando una differenza tra la minima e la massima somma del tempo di lavoro settimanale, ci si scontra con il fatto che la soluzione possa prevedere che alcune infermiere lavorino sei giorni e altre solo quattro giorni.

2.4 Altri metodi

Chiarandini et al. (2000) descrivono il carico di lavoro come numero di 'turni di lavoro' per settimana. In un orizzonte temporale lungo si ha bilanciamento quando il numero di volte della settimana in cui un'infermiera svolge meno turni lavorativi rispetto alla media ed il numero di volte in cui svolge più turni lavorativi tendono entrambi ad essere nulli.

Ribeiro e Ramalhinho Dias Lourenço (2001) gestiscono il multi-periodo. Essi definiscono il carico di lavoro come quantità μ_j di nodi visitati dal veicolo j (eq. (2.13)), modellando il bilanciamento di carico come minimizzazione della deviazione standard del carico delle rotte (2.12):

$$\min \sqrt{\frac{\sum_{j\in N} \mu_j^2}{m} - \left(\frac{\sum_{j\in N} \mu_j}{m}\right)^2} \tag{2.12}$$

$$\text{s. t. } \mu_j = \sum_{h\in H} \sum_{s\in S} y_s^{jh} \qquad \forall j \in N \tag{2.13}$$

Tutti gli approcci che pongono l'enfasi del bilanciamento del carico di lavoro sulla quantità di servizi sono difficilmente applicabili al contesto reale, nel quale i costi di viaggio non sono omogenei. Infatti, spesso sono applicati nel contesto della turnazione del personale, dove non ci sono tempi di viaggio, oppure in modelli che approssimano il tempo di viaggio come Lahrichi (2008) e Borsani et al. (2006). È facile comprendere l'inopportunità di applicare questo modello di bilanciamento quando si gestisce un contesto in cui le distanze tra i servizi sono eterogenee. Questo è il caso, ad esempio, quando servizi in zone periferiche richiedono tempi di viaggio pari al minimo tempo di svolgimento di due o più servizi, mentre i servizi in città hanno tempi di viaggio pari ad una frazione del minimo tempo di un servizio. In una situazione reale si potrebbe giungere ad una soluzione nella quale le infermiere hanno lo stesso numero di servizi al giorno, ma in cui le infermiere con servizi in periferia abbiano i servizi con tempo di lavoro massimo, mentre le infermiere con servizi in centro cittadino hanno i servizi con tempo di lavoro minimo:

in questo contesto, chiaramente, le infermiere che andranno in periferia saranno notevolmente più cariche, in termini di tempo, sebbene il bilanciamento sia ottimo dal punto di vista del numero di servizi.

2.5 Formulazione min-max

Un modo per modellare il bilanciamento di carico è minimizzare la massima durata media di lavoro giornaliero (vincoli (1.13)). Questo stile di modellare il bilanciamento di carico è simile a quello presentato nel settore del Flexible Manufacturing Systems (FMS) (Stecke, 1983; Wilson, 1992; Arbib et al., 1991). Si presti attenzione al fatto che l'aspetto d'instradamento descritto nell'FMS ha complessità polinomiale, poiché è un cammino minimo tra due macchine, mentre il sottoproblema d'instradamento nell'HHCP è NP-Hard .

Consideriamo la sola parte di instradamento dell'HHCP priva della periodicità, cioè fissando un giorno. Al meglio della nostra conoscenza, si trovano in letteratura alcuni lavori relativi a problemi simili. Per esempio, il Selective Shortest Longest Route Problem è una variante del m-TSP in cui, minimizzando il tour di lunghezza più grande, si richiede di visitare tutti i nodi e ogni veicolo parte da un diverso deposito di partenza (Glaab, 2002; Valle et al., 2009). Gli articoli documentano il fatto che la natura min-max della funzione obiettivo aumenta la difficoltà anche del solo ottenere il rilassamento lineare.

Sebbene con questo tipo di formulazione, per gestire il bilanciamento del carico di lavoro, sia possibile modellare correttamente tutti quegli aspetti del problema per il quali l'eterogenità dei dati rendere inadeguati gli altri metodi, il che motiva la formulazione proposta, si ha purtroppo lo svantaggio dovuto alla difficoltà notevolmente maggiore di determinare una soluzione del problema.

2.6 Conclusioni

L'attività di assistenza sanitaria a domicilio richiede qualità umane del personale per poter dare un sollievo al paziente, non solo con cure mediche, ma anche con la presenza umana. Tali qualità umane vengono meno quando si presenta il fenomeno di esaurimento da lavoro. Vari studi hanno osservato la relazione tra questo fenomeno e la distribuzione del lavoro in termini di orario lavorativo.

In questo capitolo abbiamo esplorato il tema del bilanciamento del tempo di lavoro, il quale ha avuto interesse in diversi settori della letteratura.

Dall'approfondimento svolto, è emerso che molte proposte non coglierebbero il problema reale da noi trattato ed abbiamo argomentato mostrando i casi nei quali l'eterogeneità di alcuni dati porterebbe a soluzioni sbilanciate per l'HHCP.

Lo stile min-max per modellare il bilanciamento di carico, applicato al modello proposto in questa tesi, è risultato ideale per cogliere l'eterogeneità dei dati: i tempi di spostamento, i tempi di servizio, la quantità di infermiere in turno e la quantità di richieste giornaliere.

Riepilogo della parte I

I capitoli precedenti hanno dunque introdotto il formalismo matematico necessario per la descrizione del problema HHC, requisito irrinunciabile per quanto verrà sviluppato nei capitoli successivi. Inoltre, dall'esposizione del problema del bilanciamento del carico, presentato nel capitolo 2, è emerso che la formulazione proposta gestisce aspetti dell'HHCP di rilievo nei casi reali, che fino ad ora non era possibile affrontare in modo soddisfacente senza ricorrere a semplificazioni eccessivamente limitanti.

Dalla formulazione del modello emergono due osservazioni importanti:

- per particolari scelte dei parametri si ottengono, come casi speciali, altri problemi di ottimizzazione ben noti in letteratura;
- nella sua globalità, il problema si decompone in un problema di scheduling ed un problema di routing.

In particolare, l'HHCP generalizza il DCVRP, problema poco indagato in letteratura, sebbene abbia interesse in molti casi pratici. Questi aspetti saranno approfonditi nei capitoli successivi.

Parte II

Approcci di programmazione lineare intera mista

Sommario della parte II

Come i capitoli precedenti hanno evidenziato, cogliere tutti gli aspetti del problema HHC porta ad un modello corposo e difficile da risolvere. In questi casi, molto spesso l'unica strada percorribile è utilizzare tecniche di decomposizione del problema in sottoproblemi, per abbassarne la complessità: fra queste, la decomposizione di Benders, la column generation ed il rilassamento lagrangiano sono tra le più note.

Abbiamo inoltre visto nella parte precedente che uno dei sottoproblemi dell'HHCP è il TSP, la cui risoluzione, com'è ben noto (Applegate et al., 2006; Roberti e Toth, 2012), è veloce (da pochi secondi a pochi minuti) per istanze di moderata dimensione (fino a circa 500 nodi) e diventa molto veloce per istanze di piccole dimensioni (meno di 50 nodi).

Pertanto, è naturale indagare se la decomposizione di Benders, la column generation e il rilassamento lagrangiano possono essere applicate nella ricerca di soluzioni esatte o lower bound dell'HHCP, sfruttando il TSP come sottoproblema nella decomposizione. I prossimi capitoli introducono quindi queste tre tecniche, analizzandole nel contesto del problema HHC.

Capitolo 3

Decomposizione di Benders

In questo capitolo valutiamo la fattibilità di usare il metodo di decomposizione di Benders (Benders, 1962) alla soluzione dell'HHCP.

3.1 Introduzione

Il metodo è applicato in un contesto nel quale ci sia un sottinsieme di variabili che renda il problema difficile. L'ambito usuale di applicazione è la programmazione lineare intera mista (Mixed Integer Linear Programming, MILP), poiché una parte facile è costituita delle variabili continue $\mathbf{x} \in \mathbb{R}^m$ e la parte difficile dalle variabili discrete $\mathbf{y} \in \mathbb{Z}_+^n$.

Un problema generico di MILP può essere formulato come segue:

$$\min\{z(\mathbf{x}, \mathbf{y}) = \mathbf{c}^T\mathbf{x} + \mathbf{d}^T\mathbf{y} \mid \mathbf{A}\mathbf{x} + \mathbf{D}\mathbf{y} \leq \mathbf{b}, \mathbf{x} \geq 0, \mathbf{y} \in \mathbb{Z}_+^n\}\,, \tag{3.1}$$

con $\mathbf{c} \in \mathbb{R}^m$, $\mathbf{d} \in \mathbb{R}^n$, $\mathbf{A} \in \mathbb{R}^{p\times m}$, $\mathbf{D} \in \mathbb{R}^{p\times n}$, $\mathbf{b} \in \mathbb{R}^p$. Fissando un valore $\tilde{\mathbf{y}} \in \mathbb{Z}_+^n$, il problema (3.1) può essere scritto come:

$$\mathbf{d}^T\tilde{\mathbf{y}} + \min\{\mathbf{c}^T\mathbf{x} \mid \mathbf{A}\mathbf{x} \leq \mathbf{b} - \mathbf{D}\tilde{\mathbf{y}}, \mathbf{x} \geq 0\}\,. \tag{3.2}$$

Se $\mathbf{x}$ è una soluzione del sistema (3.2), allora $(\mathbf{x}, \tilde{\mathbf{y}})$ è una soluzione del sistema (3.1) ed il vettore $\tilde{\mathbf{y}}$ prescelto è detto ammissibile. Se non esiste $\tilde{\mathbf{y}} \in \mathbb{Z}_+^n$ ammissibile, allora (3.1) non ammette soluzione. Possiamo scrivere il duale del problema (3.2) come:

$$v(\tilde{\mathbf{y}}) = \max\{v(\mathbf{u}, \tilde{\mathbf{y}}) = \mathbf{d}\tilde{\mathbf{y}} + \mathbf{u}(\mathbf{D}\tilde{\mathbf{y}} - \mathbf{b}) \mid -\mathbf{u}\mathbf{A} \leq \mathbf{c}, \mathbf{u} \geq 0\}\,, \tag{3.3}$$

con $\mathbf{u} \in \mathbb{R}^p$. Se il duale (3.3) è inammissibile, il problema primale (3.2) è inferiormente illimitato per ogni $\tilde{\mathbf{y}} \in \mathbb{Z}_+^n$, ovvero il problema di partenza (3.1) non ha soluzione finita, essendo esso o inammissibile o inferiormente illimitato.

Assumendo ammissibile il problema (3.3), allora per ogni $\tilde{\mathbf{y}} \in \mathbb{Z}_+^n$ il valore ottimo sarà lo stesso di (3.2), pertanto il problema (3.1) è equivalente a

$$\min\{v(\mathbf{y}) \mid \mathbf{y} \in \mathbb{Z}_+^n\}\,. \tag{3.4}$$

Assunto (3.3) ammissibile, $v(\mathbf{y})$ è illimitata superiormente se e solo se (3.2) non ammette soluzione, ovvero se $\mathbf{y}$ è inammissibile. In questo caso esiste almeno una soluzione $\boldsymbol{\mu}$ di (3.3) tale che $\boldsymbol{\mu}^T(\mathbf{D}\mathbf{y}-\mathbf{b}) > 0$. Siano $\boldsymbol{\mu}^1, \ldots, \boldsymbol{\mu}^p$ le soluzioni di (3.3): $\tilde{\mathbf{y}} \in \mathbb{Z}^n_+$ è ammissibile se e solo se

$$(\boldsymbol{\mu}^j)^T(\mathbf{D}\mathbf{y}-\mathbf{b}) \leq 0 \qquad \forall j = 1, \ldots, p\,. \tag{3.5}$$

Siano $\boldsymbol{\pi}^1, \ldots, \boldsymbol{\pi}^q$ tutte le soluzioni di base del problema duale (3.3). Quando le (3.5) sono soddisfatte abbiamo:

$$v(\mathbf{y}) = \max\{\mathbf{d}\mathbf{y} + (\boldsymbol{\pi}^k)^T(\mathbf{D}\mathbf{y}-\mathbf{b}) \mid k = 1, \ldots, q\}\,. \tag{3.6}$$

algoritmo 3.1 Benders

Require: $\mathbf{c}, \mathbf{d}, \mathbf{D}, \mathbf{A}, \mathbf{b}, \Pi, M$

while true **do**
  $(P) \leftarrow$ problema (3.7) limitato a $\{\boldsymbol{\mu}^j\}_{j\in M}$ e $\{\boldsymbol{\pi}^k\}_{k\in\Pi}$
  if (P) è inammissibile **then**
    return nessuna soluzione
  else
    $(\hat{z}_0, \hat{\mathbf{y}}) \leftarrow$ soluzione di (P)
  end if
  $(Q) \leftarrow$ problema (3.3) con $\mathbf{y} \leftarrow \hat{\mathbf{y}}$
  if (Q) illimitato **then**
    $\boldsymbol{\mu}^{\hat{j}}$ soluzione omogenea corrispondente alla semiretta lungo la quale $v(\mathbf{u}, \hat{\mathbf{y}})$ diverge a $+\infty$
    $M \leftarrow M \cup \{\hat{j}\}$
  else
    $\boldsymbol{\pi}^{\hat{k}}$ soluzione ottima di (Q) e $\hat{v} \leftarrow v(\boldsymbol{\pi}^{\hat{k}}, \hat{\mathbf{y}})$
    if $\hat{v} > \hat{z}_0$ **then**
      $\Pi \leftarrow \Pi \cup \{\hat{k}\}$
    else
      return $(\hat{\mathbf{x}}, \hat{\mathbf{y}}) \leftarrow$ soluzione di (3.2) con $\mathbf{y} \leftarrow \hat{\mathbf{y}}$
    end if
  end if
end while

Da quanto detto abbiamo che il problema seguente è equivalente al problema di partenza (3.1):

$$\begin{aligned} \min\{z_0 \quad | \quad & z_0 - \mathbf{d}^T\mathbf{y} - \boldsymbol{\pi}^k(\mathbf{D}\mathbf{y}-\mathbf{b}) \geq 0, k = 1, \ldots, q, \\ & (\boldsymbol{\mu}^j)^T(\mathbf{D}\mathbf{y}-\mathbf{b}) \leq 0, j = 1, \ldots, p, \mathbf{y} \in \mathbb{Z}^n_+\}\,. \end{aligned} \tag{3.7}$$

Se la coppia di valori $(z_0^*, \mathbf{y}^*)$ è un ottimo del problema (3.7), allora abbiamo che z_0^* equivale al valore ottimo dato dalla soluzione del problema (3.1). Ponendo $\tilde{\mathbf{y}} = \mathbf{y}^*$ nel problema (3.3), otteniamo una sua soluzione ottima $\mathbf{x}^*$, da cui $(\mathbf{x}^*, \mathbf{y}^*)$ è soluzione ottima di (3.1).

Spesso nei problemi reali il numero dei vincoli è enorme, pertanto si usa selezionare un sottinsieme: siano dunque $\Pi \subseteq \{1, \ldots, k\}$, $M \subseteq \{1, \ldots, j\}$ tali sottinisiemi nella procedura descritta nell'algoritmo 3.1.

Nell'HHCP l'insieme di variabili discrete è molto ampio. È noto che i tempi computazionali per il raggiungimento di alcune soluzioni ottime di problemi d'instradamento sono talvolta rapidi (Applegate et al., 2006; Laporte, 2009). È quindi sensato esplorare la possibilità di decomporre il modello, assumendo alcune variabili decisionali del problema di instradamento come "variabili facili", cioè le $\mathbf{x}$. La forma del nostro problema sembra suggerire una decomposizione che porta ad avere la soluzione di una serie di TSP per ogni coppia (infermiera,giorno).

3.2 TSP come sottoproblema

Assumiamo di fissare il servizio s per ogni infermiera j e per ogni giorno h, ossia di conoscere y_s^{jh}. Ne segue che, per ogni coppia (j, h), abbiamo un insieme di servizi $S^{jh} = \{s \in S \mid y_s^{jh} = 1\}$, e $S_0^{jh} = \{s \in S_0 \mid y_s^{jh} = 1\}$. In questo caso, i valori di y_s^{jh} permettono di ricavare z_p^j, quindi di avere il valore della fidelizzazione paziente-infermiera, il quale sarà indicato con il valore costante $\Lambda = \sum_{p \in P} \left(\sum_{j \in N} z_p^j - 1 \right)$.

Se il tempo massimo L è superato si ha un tempo straordinario w. D'ora in poi, assumeremo che il decisore abbia scelto di ammettere tempo straordinario, ossia $W^{max} = \infty$: allora, per ogni coppia (j, h) il modello dell'HHCP si riduce a

$$\min \alpha_0 u + \alpha_2 w \tag{3.8}$$

$$\text{s. t.} \quad \sum_{s' \in S_0^{jh}} x_{ss'} = 1 \qquad \forall s \in S^{jh} \tag{3.9}$$

$$\sum_{s' \in S_0^{jh}} x_{s's} = 1 \qquad \forall s \in S^{jh} \tag{3.10}$$

$$\sum_{s' \in S^{jh}} x_{0s'} = 1 \tag{3.11}$$

$$\sum_{s' \in S^{jh}} x_{s'0} = 1 \tag{3.12}$$

$$x_{ss'} = 1 \Rightarrow t_{s'} \geq t_s + 1 \qquad \forall s \in S_0^{jh},\ \forall s' \in S^{jh} \setminus \{s\} \tag{3.13}$$

$$\ell^{jh} \leq L + w \tag{3.14}$$

$$u \geq \frac{1}{b_j} \ell^{jh} \tag{3.15}$$

$$\ell^{jh} = \sum_{s \in S_0^h} \sum_{s' \in S_0^h} (a_{s'} + d_{ss'}) x_{ss'} \tag{3.16}$$

$$x_{s's} \in \{0, 1\} \qquad \forall s' \in S_0^{jh},\ \forall s \in S_0^{jh} \tag{3.17}$$

$$t_s \in \mathbb{R}^+ \qquad \forall s \in S_0^{jh} \tag{3.18}$$

$$u, w \in \mathbb{R}^+ \tag{3.19}$$

Proposizione 3.2.1. *Il problema descritto dai vincoli* (3.8)–(3.19) *è equivalente al problema TSP.*

Dimostrazione. Per verificare l'affermazione mostriamo che la funzione obbiettivo (3.8) è ricavabile dalla soluzione di un TSP.

Assumiamo di risolvere un sistema $\min\{\ell^{jh} \mid (3.9)\text{–}(3.13)\ \}$, questo equivale a risolvere un TSP. Poiché nella funzione obiettivo (3.8) desideriamo minimizzare u, dal vincoli (3.15) $u \geq \ell^{jh}/b_j$ ricaviamo che $u = \ell^{jh}/b_j$. Analogamente per w, nella funzione obiettivo (3.8) desideriamo minimizzare w, pertanto dal vincolo (3.14) $w = 0$ se $\ell^{jh} \leq L$, ed è $(\ell^{jh} - L)$ se $\ell^{jh} > L$.

Risolvendo un TSP abbiamo che tutti i vincoli del problema (3.8)–(3.19) sono rispettati, e minimizzando la distanza totale ℓ^{jh} si minimizzano anche i valori minimi di w e u. La funzione obiettivo (3.8) è quindi ricavabile dalla minimizzazione del costo totale di viaggio ℓ^{jh} come segue:

$$\alpha_0 u + \alpha_2 w = \phi(\ell^{jh}) \stackrel{def}{=} \begin{cases} \alpha_0 \frac{\ell^{jh}}{b_j} + \alpha_2(\ell^{jh} - L) & \ell^{jh} > L \\ \alpha_0 \frac{\ell^{jh}}{b_j} & \ell^{jh} \leq L \end{cases} . \tag{3.20}$$

□

Proposizione 3.2.2. *Non è possibile usare il metodo di decomposizione di Benders, se si ha come sottoproblema* (3.8)–(3.19).

Dimostrazione. Assumiamo di avere (3.8)–(3.19) come sottoproblema (3.2) del metodo di Benders. Abbiamo visto che risolvere il problema (3.8)–(3.19) equivale a risolvere il TSP. Nell'introduzione teorica abbiamo asserito che al valore ottimo del sottoproblema (3.2) corrisponde il valore ottimo del sottoproblema duale $v(\boldsymbol{\pi}^k, \mathbf{y})$, pertanto nell'HHCP $v(\boldsymbol{\pi}^k, \mathbf{y})$ corrisponde al valore della soluzione del sottoproblema TSP. Quando $W^{max} = \infty$, non essendoci limiti su ℓ^{jh} e avendo un grafo che ammette cicli hamiltoniani, si ha sempre una soluzione del TSP. Ne segue che il sottoproblema (3.2) del metodo di decomposizione di Benders ha sempre soluzione, ossia il problema (3.7) sarà sempre nella forma:

$$\min\{z_0 \mid z_0 \overbrace{-\mathbf{dy} - \boldsymbol{\pi}^k(\mathbf{Dy} - \mathbf{b})}^{-v(\boldsymbol{\pi}^k, \mathbf{y})} \geq 0 \ \forall k = 1, \ldots, p,\ \mathbf{y} \in \mathbb{Z}^n_+\} . \tag{3.21}$$

Senza perdere di generalità, assumiamo che b_j, il numero di giorni in cui l'infermiera j è in turno, sia uguale per ogni j. Ne segue che ogni k il valore $v(\boldsymbol{\pi}^k, \mathbf{y})$ non dipenderà da b_j, ovvero non dipenderà dall'infermiera j.

Possiamo scrivere (3.21) come:

$$\min\{z_0 \mid z_0 - \phi(\ell^k) \geq 0 \ \forall k = 1, \ldots, p,\ \mathbf{y} \in \mathbb{Z}^n_+\} . \tag{3.22}$$

Possiamo affermare che z_0 dipenderà dal tour con lunghezza più grande, poiché il minimo z_0 sarà grande almeno quanto il tour di lunghezza massima (scalato da b_j), difatti in (3.22) $\min\{z_0 \mid z_0 - \phi(\ell^k) \geq 0\}$. Assumiamo che per un istanza esista una soluzione data da un'euristica tale che i tempi di straordinario w siano abbattuti, il tempo di carico di lavoro medio giornaliero u sia

sotto L, ed la fidelizzazione paziente-infermiera sia ottima, ossia ogni paziente è servito da una sola infermiera. Per questa istanza, del quale conosciamo la soluzione ammissibile, applichiamo l'algoritmo 3.1. Assumiamo che durante l'esecuzione dell'algoritmo 3.1, $\tilde{\mathbf{y}}$ identifichi un insieme di servizi S_0^{jh} tali che il valore del TSP ecceda il tempo limite L, allora otteniamo un valore z_0 più grande di L; per la natura del problema (3.22) non possiamo più abbassarlo, ossia il metodo non converge. □

3.3 Conclusione

Poiché la soluzione del TSP è affrontata in tempi rapidi, è stato sensato esplorare la fattibilità del metodo di decomposizione di Benders mediante un sottoproblema che si riducesse al problema del commesso viaggiatore. Sebbene il risultato della proposizione 3.2.2 scoraggi l'uso della decomposizione descritta in sezione 3.1 con il sottoproblema (3.8)–(3.19), è possibile che varianti del metodo o altri sottoproblemi permettano un utilizzo efficacie delle tecniche di programmazione matematica.

Capitolo 4

Column generation

Il metodo branch-and-price permette, mediante l'uso della generazione di colonne per la soluzione del rilassamento lineare del problema intero, di risolvere un problema di programmazione lineare intera. Il metodo della generazione di colonne, particolarmente indicato per problemi lineari con tante variabili, aggiunge nuove colonne ad un problema ridotto. Le colonne sono individuate mediante la soluzione di un sottoproblema di pricing. In questo capitolo mostriamo, rimodellando l'HHCP e proponendo un sottoproblema di pricing, come è possibile applicare un metodo branch-and-price.

4.1 Introduzione

Alcuni problemi di programmazione lineare sono caratterizzati da un elevato numero di variabili e/o di vincoli, che corrispondono effettivamente rispettivamente alle colonne e alle righe della matrice dei vincoli associata al problema. La struttura di alcuni problemi permette la decomposizione in sottoproblemi di più facile soluzione, risolvibili in tempi rapidi. Nel metodo *generazione di colonne* ('*column generation*', CG) il problema di partenza ha un insieme ridotto di variabili e ad ogni passo del metodo sono aggiunte opportune colonne allo scopo di migliorare il valore della funzione obiettivo.

Per descrivere questo metodo, chiamiamo *problema principale* (Master Problem, MP) il seguente problema di programmazione lineare:

$$z^*_{MP} = \min \sum_{j \in J} c_j x_j \tag{4.1}$$

$$\text{s. t.} \sum_{j \in J} \mathbf{a}_j x_j \geq \mathbf{b}$$

$$x_j \geq 0 \qquad \forall j \in J\,. \tag{4.2}$$

Affrontiamo la risoluzione di (4.1) con il metodo del simplesso e, ad ogni iterazione del metodo del simplesso, cerchiamo un'opportuna variabile non in base per farla entrare in base: per questo, prendiamo un vettore non negativo $\boldsymbol{\pi}$ di variabili duali ottime, si desidera trovare

j che massimizza $\overline{c}_j = c_j - \boldsymbol{\pi}^t \mathbf{a}_j$. Come è ben noto, questo sottoproblema, chiamato *pricing subproblem*, è troppo costoso quando $|J|$ è molto grande. L'idea è dunque di lavorare con un sottoinsieme $J' \subseteq J$ ragionevolmente piccolo di colonne, ottenendo un nuovo problema che prende il nome di *problema principale ristretto* (Restricted Master Problem, RMP) nel quale i costi ridotti sono ottenuti mediante enumerazione implicita. Siano $\mathbf{x}$ e $\boldsymbol{\pi}$ le rispettive variabili primali e duali della soluzione ottima dell'attuale RMP. Quando le colonne $\mathbf{a}_j$, $j \in J$, sono date come elementi di un insieme A, e abbiamo il costo c_j allora il sottoproblema di pricing è:

$$\overline{c}^* = \min\{c_j - \boldsymbol{\pi}^T \mathbf{a}_j \mid \mathbf{a}_j \in A,\ j \in J\}$$

Se $\overline{c}^* \geq 0$, nessun c_j è negativo, $j \in J$, e la soluzione risolve il MP. Altrimenti, si aggiunge all'RMP la colonna ottenuta dalla soluzione ottima del sottoproblema, e si itera risolvendo l'RMP (Desrosiers e Lübbecke, 2005).

Nel caso di problemi di programmazione lineare, la generazione di colonne determina una soluzione ottima del problema. Nel caso della programmazione lineare intera, otteniamo un bound risolvendo il rilassamento lineare del problema con la generazione di colonne. Si parla di *branch-and-price*, riferendosi ad algoritmi di branch-and-bound nei quali il bound è calcolato usando la generazione di colonne.

Al meglio della nostra conoscenza, non sono descritti in letteratura problemi di complessità pari all'HHCP, nel quale si ha l'unione di due sottoproblemi NP-Hard risolti con il metodo della column generation. Il *Periodic VRPTW* (PVRPTW) è un sottoproblema dell'HHCP, che si presenta quando la finestra temporale è $[0, L]$, per il quale recentemente sono stati riportati riscontri positivi sull'uso del metodo della generazione di colonne (Francis et al., 2006; Francis et al., 2008; Mourgaya e Vanderbeck, 2007). La CG è stata usata come strumento per la soluzione di varianti del VRPTW (Kallehauge, 2006; Kallehauge, 2008); sebbene ci si riduca al DCVRP quando la finestra temporale è $[0, L]$, non sono stati trovati riscontri positivi sull'uso della column generation per il DCVRP. La CG è stata applicata alle varianti VRPTW e PVRPTW, tuttavia non ci sono riscontri positivi sulla soluzione degli stessi sottoproblemi di pricing nel DCVRP (Bertels e Fahle, 2006). Questo è dovuto a due aspetti: da un lato il sottoproblema di pricing con l'aggiunta del vincolo di massima durata passa da una complessità di tipo polinomiale ad essere un problema NP-Hard (talvolta risolvibile con algoritmi di complessità pseudo-polinomiale), dall'altro l'eterogeneneità delle finestre temporali aiuta la fase di presoluzione a ridurre la cardinalità del problema. Due aspetti rendono proibitivo l'impiego del PVRPTW come sottoproblema: il numero di nodi delle istanze risolte in letteratura (Mourgaya e Vanderbeck, 2007) è significativamente inferiore (80 nodi) ai casi reali trattati in questo lavoro ($\geq$ 700 nodi), ed il tempo di calcolo per ottenere la soluzione vicino all'ottimo è molto alto ($\geq$ 22 ore).

Abbiamo dettoche la CG è un metodo basato sulla decomposizione del problema, nel quale desideriamo inserire delle colonne con il criterio dei costi ridotti negativi. Le colonne sono in relazione ad un sottoproblema, che può essere risolto in tempi accettabili. In letteratura si è a conoscenza che per determinati tipi di istanze di grafi il problema del PCTSP (Stevanato et al., 2012) diviene risolvibile in tempo polinomiale (Balas, 1999; Balas e Simonetti, 2001).

Inoltre mediante la trasformazione dal PCTSP al TSP (Dell'Amico et al., 1995; Feillet et al., 2005) unitamente all'uso di un solutore per TSP, il problema può essere computazionalmente risolubile in tempi accettabili. Questo ha motivato l'esplorazione di fattibilità dell'uso della CG con il PCTSP come sottoproblema.

4.2 Modello

In aggiunta alla notazione precedentemente usata, definiamo un insieme di turni $\mathcal{T}$ che rispettano il vincolo di massima durata di un turno, cioè ogni turno $T_i \in \mathcal{T}$ ha la distanza totale ℓ_i che non supera il limite L. Definiamo una variabile booleana x_i^j, la quale assume valore di 1 se il turno $T_i \in \mathcal{T}$ è associato all'infermiera $j \in N$, 0 altrimenti. Possiamo modellare l'HHCP come segue:

$$\min\ u + \alpha \sum_{p \in P} \left(\sum_{j \in N} z_p^j - 1 \right) \tag{4.3}$$

$$\text{s. t.} \quad \sum_{i:s \in T_i} \sum_{j \in N} x_i^j = f_s \qquad \forall s \in S \tag{4.4}$$

$$\sum_{\substack{i:s \in T_i \\ \hbar(T_i) \in [h, h+e_s-1]}} \sum_{j \in N} x_i^j \leq 1 \qquad \forall s \in S,\ \forall h \in \{1, \ldots, |H| - e_s\} \tag{4.5}$$

$$u \geq \frac{1}{b_j} \sum_{i=1}^{|\mathcal{T}|} \ell_i x_i^j \qquad \forall j \in N \tag{4.6}$$

$$z_p^j \geq x_i^j \qquad \forall j \in N,\ \forall p \in P,\ \forall i : S_p \cap T_i \neq \emptyset \tag{4.7}$$

$$z_p^j \in \{0, 1\} \qquad \forall j \in N,\ p \in P \tag{4.8}$$

$$x_i^j \in \{0, 1\} \qquad \forall i \in \{1, \ldots, |\mathcal{T}|\},\ \forall j \in N \tag{4.9}$$

$$u \in \mathbb{R}^+ \tag{4.10}$$

In (4.4) è richiesto che non esistano meno di f_s turni contenenti un servizio s. Il modello presentato assume lo stesso valore della funzione obiettivo anche quando al vincolo (4.4) si sostituisce con il seguente vincolo:

$$\sum_{i:s \in T_i} \sum_{j \in N} x_i^j \geq f_s \qquad \forall s \in S. \tag{4.11}$$

I vincoli (4.5) garantiscono che non ci siano turni contenenti servizi s che violano l'intermezzo. Il valore del carico di lavoro u è ottenuto dai vincoli (4.6). Con i vincoli (4.7) ricaviamo l'informazione z_p^j, ossia se durante la settimana l'infermiera j visita il paziente p. Nella funzione obiettivo vogliamo minimizzare il carico di lavoro u ed il numero di infermiere che visitano un paziente.

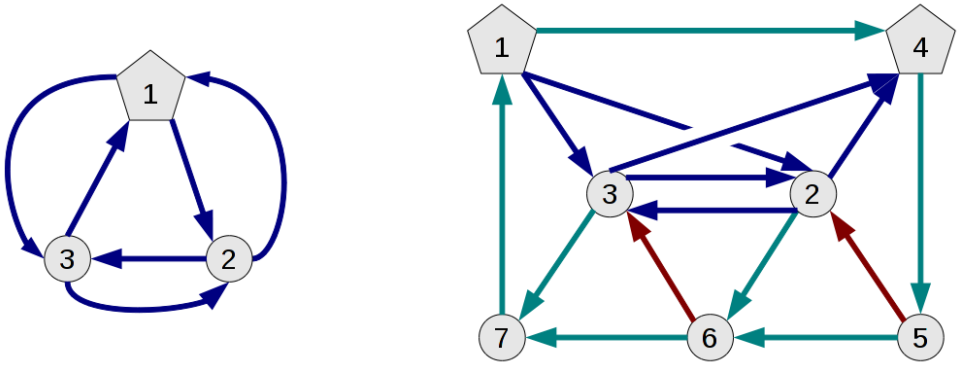


Figura 4.1: Trasformazione PCTSP a TSP

4.3 Metodo

Nel modello (4.3)–(4.10), consideriamo il rilassamento continuo delle variabili x_i^j e z_p^j, ottenendo:

$$\min \quad u + \alpha \sum_{p \in P} \left(\sum_{j \in N} z_p^j - 1 \right) \tag{4.12}$$

$$\text{s. t.} \quad \sum_{i:s \in T_i} \sum_{j \in N} x_i^j \geq f_s \qquad \forall s \in S \tag{4.13}$$

$$\sum_{\substack{i:s \in T_i \\ \hbar(T_i) \in [h, h+e_s-1]}} \sum_{j \in N} x_i^j \leq 1 \qquad \forall s \in S,\ \forall h \in \{1, \ldots, |H| - e_s\} \tag{4.14}$$

$$u \geq \frac{1}{b_j} \sum_{i=1}^{|\mathcal{T}|} \ell_i x_i^j \qquad \forall j \in N \tag{4.15}$$

$$z_p^j \geq x_i^j \qquad \forall j \in N,\ \forall p \in P,\ \forall i \in S_p \cap T_i \tag{4.16}$$

$$x_i^j, z_p^j \geq 0\,. \tag{4.17}$$

Siano λ_s, σ_{sh}, π_j e ρ_{jpi} le variabili duali associate ai vincoli nel primale. La formulazione duale è:

$$\max \quad \sum_{s \in S} \lambda_s f_s + \sum_{s \in S} \sum_{h=1}^{e_s} \sigma_{sh} \tag{4.18}$$

$$\text{s. t.} \quad \sum_{s \in T_i} \lambda_s + \sum_{s \in T_i} \sum_{h \in [\overline{h}-e_s, \overline{h}]} \sigma_{sh} + \frac{\ell_i}{b_j} \pi_j - \sum_{p: S_p \cap T_i \neq \emptyset} \rho_{jpi} \leq 0 \qquad i = 1, \ldots, |\mathcal{T}|,$$

$$\forall \overline{h} \in \hbar(T_i),\ \forall j \in N \tag{4.19}$$

$$-\sum_{j \in N} \pi_j \leq 1 \tag{4.20}$$

$$\sum_{i=1}^{|\mathcal{T}|} \rho_{jpi} \leq \alpha \qquad \forall j \in N,\ \forall p \in P \tag{4.21}$$

$$\lambda_s \geq 0,\ \rho_{jpi} \geq 0,\ \sigma_{sh} \leq 0,\ \pi_j \leq 0\,. \tag{4.22}$$

Nel contesto della CG, consideriamo un sottoinsieme di variabili x_i^j, ossia un sottoinsieme $\mathcal{T}' \subset \mathcal{T}$ di turni, e risolviamo (4.12)–(4.17) ottenendo una soluzione duale. Segue la generazione di nuovi turni ammissibili, i quali sono aggiunti dinamicamente in $\mathcal{T}$. La determinazione di nuovi turni, cioè di colonne del problema, è ottenuta risolvendo un '*pricing subproblem*'. Il pricing subproblem consiste nel determinare un turno T_i, un giorno $\overline{h}$ e un'infermiera j che violano il vincolo (4.19). Se tale turno esiste, aggiungerlo a $\mathcal{T}'$ porta un miglioramento del primale.

Imponiamo $\rho_{jpi} = 0$ per ogni coppia (j, p) e cerchiamo il turno T_i massimizzante:

$$\sum_{s \in T_i} \lambda_s + \sum_{s \in T_i} \sum_{h \in [\overline{h} - e_s, \overline{h}]} \sigma_{sh} + \frac{\ell_i}{b_j} \pi_j \,. \tag{4.23}$$

Si ottiene una sequenza di servizi che viola il vincolo (4.19). La reale violazione dipende dal valori di ρ_{jpi} da noi posti a zero. Si osservi che se non esistono turni T_i tali che l'equazione (4.23) sia positiva, allora possiamo affermare che nessun turno viola (4.19). Consideriamo la soluzione duale, λ_s, σ_{sh}, π_j, un'infermiera j e un giorno $\overline{h}$, tali che il corrispondente turno massimizzi (4.23). Questa soluzione duale equivale a una variante del PCTSP in un appropriato grafo $\mathcal{G}_{\overline{h}j} = (V, A)$. L'insieme di nodi V include un nodo rappresentante il centro di servizio e un nodo per ogni servizio. In nodi relativi ad un servizio s hanno peso $\gamma_s = \lambda_s + \sum_{h \in [\overline{h} - e_s, \overline{h}]} \sigma_{sh} \geq 0$. Ci sono archi tra tutti i servizi, e tra i servizi e il centro operativo. Il costo dell'arco (u, v) è $c_{uv} = -d_{uv}/(-b_j)\pi_j \geq 0$, con d_{uv} la distanza tra il nodo u e il nodo v.

Proposizione 4.3.1. *Massimizzare* (4.23) *equivale a trovare un tour in* $\mathcal{G}_{\overline{h}j}$*, che parte dal centro di servizio e passa da ogni nodo al più una volta massimizzando la differenza tra la somma dei pesi dei nodi (price) e la somma dei costi degl'archi.*

Dimostrazione. Prendiamo la funzione obiettivo del PCTSP:

$$\min \sum_{s \in S} \gamma_s (1 - y_s) + \sum_{e \in A(\mathcal{G}_{\overline{h}j})} c_e x_e \,. \tag{4.24}$$

Se un turno T_i è soluzione del PCTSP, abbiamo $y_s = 1$ per ogni servizio $s \in T_i$, pertanto:

$$\min \overbrace{\sum_{s \in S} \gamma_s - \sum_{s \in T_i} \gamma_s}^{\sum_{s \notin T_i} \gamma_s} + \sum_{e \in A(T_i)} c_e x_e \,. \tag{4.25}$$

Avendo il tour T_i, possiamo scrivere

$$\sum_{e \in T_i} c_e x_e = \sum_{e \in A(T_i)} \left(-d_e \frac{\pi_j}{b_j} \right) x_e = -\ell_i \frac{\pi_j}{b_j} \tag{4.26}$$

dalla quale possiamo ricavare una forma equivalente di (4.24):

$$\min \sum_{s \in S} \gamma_s - \sum_{s \in T_i} \gamma_s - \ell_i \frac{\pi_j}{b_j} \tag{4.27}$$

o equivalentemente:

$$\min \sum_{s \in S} \gamma_s - \left(\sum_{s \in T_i} \gamma_s + \ell_i \frac{\pi_j}{b_j} \right) . \tag{4.28}$$

Si osservi che, sostituendo γ_s, il termine tra partentesi di quest'ultima equazione equivale all'equazione (4.23). I termini γ_s sono noti, poiché sono ricavati dalla soluzione duale precedentemente ottenuta. Segue che l'equazione (4.28) raggiunge il minimo quando il termine tra parentesi è massimizzato.

□

algoritmo 4.1 Column Generation

1: $k \leftarrow 0$
2: $\mathcal{T} \leftarrow$ insieme iniziale di tour
3: $T_k \leftarrow \{0, s, 0\}$
4: **repeat**
5: $\quad \mathcal{T} \leftarrow \mathcal{T} \cup T_k$
6: $\quad k \leftarrow k + 1$
7: $\quad (\boldsymbol{\lambda}^k, \boldsymbol{\rho}^k, \boldsymbol{\sigma}^k, \boldsymbol{\pi}^k) \leftarrow$ solve { (4.18)–(4.22) }
8: $\quad T_k \leftarrow$ solve $\{PCTSP(\boldsymbol{\lambda}^k, \boldsymbol{\sigma}^k, \boldsymbol{\pi}^k)\}$
9: **until** $\sum_{s \in T_k} \gamma_s + \frac{\ell_k}{b_j} \pi_j \leq 0$
10: **return** solve { (4.18)–(4.22) }

Possiamo riassumere il rilassamento lineare con la generazione di colonne nell'algoritmo 4.1. Il metodo parte con un insieme iniziale $\mathcal{T}$ di tour (linea 3), finché è trovato un tour T_k non positivo (linea 9) viene aggiunto all'insieme $\mathcal{T}$ (linea 5). Quando il tour ottenuto è positivo allora abbiamo detto che nessun turno viola (4.19), il metodo esce restituendo la soluzione del sistema (linea 10).

4.4 Conclusioni

In questo capitolo abbiamo presentato un possibile sottoproblema di pricing per risolvere il problema HHC all'ottimo, mediante branch-and-price, o per ottenerne un lower bound. Nell'implementazione del metodo ci si è scontrati con la complessità di risolvere in tempi accettabili un PCTSP o un TSP descritto da una matrice asimmetrica di costi ottenuta dalla trasformazione PCTSP–TSP.

Capitolo 5

Lower Bound

Il rilassamento lineare del modello proposto nel capitolo 1 deve superare due ostacoli, le esose richieste di memoria per rappresentare tutti i vincoli e variabili, e la maggiore richiesta di tempo computazionale per risolvere il rilassamento di un problema di min-max. Nel primo caso troviamo riscontro in letteratura con il lavoro di (Ribeiro e Ramalhinho Dias Lourenço, 2001), nel secondo caso con il lavoro di (Valle et al., 2009). Occorre quindi ricercare un lower bound in un sottoproblema che approssimi per difetto l'HHCP: nella prima sezione studiamo il rilassamento lagrangiano per il DCVRP; nella seconda, consideriamo un approssimazione per difetto dell'HHCP nella quale tutti i tempi di viaggio sono nulli.

5.1 Instradamento

Per il DCVRP, uno dei sottoproblemi d'instradamento dell'HHCP, è arduo restituire un lower bound vicino all'ottimo. Abbiamo riscontrato che il risolutore di programmazione matematica non alzavano il valore del lower bound dopo una settimana di calcolo parallelo in una macchina con 16 thread, sebbene l'istanza era di piccole dimensioni e configurando il risolutore per aggiungere o meno i vari tagli a disposizione, come il taglio di Gomori. Al meglio della nostra conoscenza, non esistono lavori che esplorino il rilassamento lagrangiano per il DCVRP. Questo ha motivato l'esplorazione del rilassamento lagrangiano (Geoffrion, 1974), il quale può essere così riassunto.

5.1.1 Rilassamento lagrangiano

Assumiamo

$$X = \{\mathbf{x} \in Y,\ g_i(\mathbf{x}) \geq 0,\ i = 1, \ldots, r\} \tag{5.1}$$

ove determinare $\mathbf{x} \in Y$ è un problema "facile", ossia polinomiale o computabile in tempi accettabili, e il rispettare i vincoli $g_i(\mathbf{x}) \geq 0$ renda il problema "difficile". Si consideri il *lagrangiano*

$$\mathcal{L}(\boldsymbol{\lambda}, \mathbf{x}) = w(\mathbf{x}) - \sum_{i=1}^{r} \lambda^i g_i(\mathbf{x}) \tag{5.2}$$

si considera il problema

$$\min_{\substack{\mathbf{x}\in Y \\ \boldsymbol{\lambda}\geq\mathbf{0}}} \mathcal{L}(\boldsymbol{\lambda}, \mathbf{x}) = \mathcal{L}_0(\boldsymbol{\lambda}) \tag{5.3}$$

dove $\boldsymbol{\lambda} = \{\lambda^1, \ldots, \lambda^r\}$ è il vettore dei *moltiplicatori di Lagrange* di (5.2). Per ogni vettore λ abbiamo il lower bound $\mathcal{L}_0(\boldsymbol{\lambda}) \leq w_0$. Per essere più vicini all'ottimo w_0 si risolve il *lagrangiano duale*

$$\max_{\boldsymbol{\lambda}\geq\mathbf{0}} \mathcal{L}_0(\boldsymbol{\lambda}) = \mathcal{L}_0(\overline{\boldsymbol{\lambda}})\,. \tag{5.4}$$

Assumiamo di ordinare le soluzioni ammissibili, scrivendo

$$Y = \{\mathbf{x}_t,\, t = 1, \ldots, \tau\} \tag{5.5}$$

possiamo scrivere il problema (5.4) come

$$\max_{\boldsymbol{\lambda}\geq\mathbf{0}}\{\mathcal{L}_0 : \mathcal{L}_0 \leq w(\mathbf{x}_t) - \sum_{i=1}^{r} \lambda^i g_i(\mathbf{x}_t),\ t = 1, \ldots, \tau\}\,. \tag{5.6}$$

Un vettore $\mathbf{z}$ è detto *sottogradiente* di $\mathcal{L}_0(\boldsymbol{\lambda})$ nel punto $\overline{\boldsymbol{\lambda}}$ se si ha che per ogni vettore $\boldsymbol{\lambda}$

$$\mathcal{L}_0(\boldsymbol{\lambda}) \leq \mathcal{L}_0(\overline{\boldsymbol{\lambda}}) + \mathbf{z}(\boldsymbol{\lambda} - \overline{\boldsymbol{\lambda}})\,. \tag{5.7}$$

Supponiamo che per un certo $\boldsymbol{\lambda}$ il problema (5.3) viene risolto dalla t-esima soluzione ammissibile $\mathbf{x}_t$, ovvero che

$$\mathcal{L}_0(\boldsymbol{\lambda}) = \mathcal{L}_0(\mathbf{x}_t, \boldsymbol{\lambda}) \tag{5.8}$$

allora, dalla (5.2)

$$\mathbf{z} = -(g_1(\mathbf{x}_t), \ldots, g_r(\mathbf{x}_t)) \tag{5.9}$$

è un sottogradiente di $\mathcal{L}_0(\boldsymbol{\lambda})$.

5.1.2 Metodo del sottogradiente

Per ottenere un valore di (5.4) vi sono vari metodi; tra questi, il *metodo del sottogradiente* (Fisher, 2004) cerca di avvicinarsi all'ottimo per passi successivi, cercando di minimizzare la distanza tra $\boldsymbol{\lambda}$ e $\overline{\boldsymbol{\lambda}}$, in pseudocodice nell'algoritmo 5.1. La sequenza di vettori $\boldsymbol{\lambda}_t$ è ottenuta in linea 5, dove $\mathbf{z}$ è il vettore di (5.9), μ_t è uno scalare (linea 4), e $\mathbf{x}_k$ un vettore che risolve (5.3) (linea 3). Lo scalare μ_t indica la grandezza dal passo, può essere scelto con differenti regole, la più semplice è con $\mu_t = \epsilon_t$; dove ϵ_t è una variabile che viene dimezzata ogni volta che non si aumenta $\mathcal{L}_0(\boldsymbol{\lambda})$ dopo un numero prefissato di iterazioni (linee 6–10).

5.1.3 Rilassamento lagrangiano del DCVRP

Abbiamo applicato il metodo descritto al DCVRP usando diversi sottoproblemi "facili". Quando il sottoproblema "facile" era polinomiale o computazionalmente accessibile, non abbiamo ottenuto dei valori iniziali promettenti ed il metodo non ha mostrato un miglioramento in tempi accettabili. Sottoproblemi, usuali e polinomiali per problemi privi del vincolo di massima

algoritmo 5.1 sottogradiente

$t \leftarrow 0$; $\epsilon_t \leftarrow 1$; $\boldsymbol{\lambda}_t \leftarrow \mathbf{0}$ ..... # $t = 0 \rightarrow \boldsymbol{\lambda}_t = \boldsymbol{\lambda}_0$
while $\neg end()$ **do**
  $\mathbf{x}_t = \mathbf{x}(\boldsymbol{\lambda}_t) \leftarrow$ solve $\left(\min_{\substack{\mathbf{x}\in Y \\ \boldsymbol{\lambda}\geq 0}} \mathcal{L}(\boldsymbol{\lambda}_t, \mathbf{x})\right)$ ..... # $\mathcal{L}_0(\boldsymbol{\lambda}_t)$
  $\mu_t \leftarrow \epsilon_t \frac{(\underline{w} - \mathcal{L}_0(\boldsymbol{\lambda}_t))}{\|\mathbf{z}(\mathbf{x}_t)\|_2}$ ..... # or other rules
  $\boldsymbol{\lambda}_{t+1} \leftarrow \max\{\boldsymbol{\lambda}_t + \mu_t \mathbf{z}(\mathbf{x}_t), \mathbf{0}\}$
  if too much time same value **then**
    $\epsilon_{t+1} \leftarrow \frac{\epsilon_t}{2}$
  else
    $\epsilon_{t+1} \leftarrow \epsilon_t$
  end if
  $t \leftarrow t + 1$
  $\boldsymbol{\lambda} \leftarrow \boldsymbol{\lambda}_t$
end while
return $\{\boldsymbol{\lambda}\}$

lunghezza, ad esempio il minimum spanning tree problem (Held e Karp, 1970), con l'aggiunta del vincolo di massima lunghezza divengono NP-Hard, come il constrained minimum spanning tree problem (Ravi e Goemans, 1996).

Riportiamo ora la formulazione di uno dei sottoproblemi il quale ha mostrato alcune caratteristiche positive. La formulazione del DCVRP dalla quale si parte è:

$$\min \quad \sum_{(u,v)\in A} c_{uv} \sum_{k=1}^{m} x_{uv}^k \tag{5.10}$$

$$\text{s. t.} \quad \sum_{v\in\delta^+(u)\setminus\{0\}} \sum_{k=1}^{m} x_{uv}^k = 1 \qquad \forall u \in V \setminus \{0\} \tag{5.11}$$

$$\sum_{v\in\delta^+(0)} \sum_{k=1}^{m} x_{0v}^k = m \tag{5.12}$$

$$\sum_{u\in\delta^-(w)} x_{uw}^k = \sum_{v\in\delta^+(w)} x_{wv}^k \qquad \forall w \in V,\ \forall k \in \{1,\dots,m\} \tag{5.13}$$

$$\sum_{v\in V} y_{uv} - \sum_{v\in V} y_{vu} = 1 \qquad \forall u \in V \setminus \{0\} \tag{5.14}$$

$$y_{uv} \leq (n-m) \sum_{k=1}^{m} x_{uv}^k \qquad \forall (u,v) \in A \tag{5.15}$$

$$\sum_{(u,v)\in A} c_{uv} x_{uv}^k \leq L \qquad \forall k \in \{1,\dots,m\} \tag{5.16}$$

$$x_{uv}^k \in \{0,1\},\ y_{uv} \in \mathbb{R}^+ \qquad \forall (u,v) \in A,\ \forall k \in \{1,\dots,m\}\,. \tag{5.17}$$

Proposizione 5.1.1. *Al modello del DCVRP possiamo aggiungere il seguente vincolo:*

$$\sum_{(u,v)\in A} c_{uv} \sum_{k=1}^{m} x_{uv}^{k} \leq mL\,. \tag{5.18}$$

Dimostrazione. Ci sono due modi per proseguire. Il primo modo è sommando gli m vincoli (5.16), si ottiene un vincolo (5.18) che risulta rispettato se sono rispettati i vincoli (5.16).

Un altro modo, pone in relazione il nuovo vincolo con i vincoli surrogati (Glover, 1968; Greenberg e Pierskalla, 1970). Date m variabili non negative ω^k, possiamo rimpiazzare gli m vincoli (5.16) con il vincolo surrogato:

$$\sum_{k=1}^{m} \omega^k \sum_{(u,v)\in A} c_{uv} x_{uv}^{k} \leq \sum_{k=1}^{m} \omega^k L\,. \tag{5.19}$$

Quando per ogni k la variabile ω^k è una costante positiva α otteniamo il vincolo (5.18). □

Abbiamo ricavato il vincoli (5.18) usando una formulazione ove i vincoli di eliminazione dei sottocicli (5.14)–(5.15) rappresentano la conservazione del flusso, si osservi che può essere usato anche in formulazioni che usano le disuguaglianze Miller-Tucker-Zemlin per l'eliminazione dei sottocicli proposte da (Kara et al., 2004; Miller et al., 1960).

Introduciamo un vincolo che ordina i tour per lunghezza:

$$\sum_{(u,v)\in A} c_{uv} x_{uv}^{k_1} \leq \sum_{(u,v)\in A} c_{uv} x_{uv}^{k_2} \qquad \forall k_1, k_2 \in \{1,\ldots,m\} : k_1 < k_2\,. \tag{5.20}$$

Ora possiamo rilassare il vincolo relativo al tour m-esimo, ovvero quello di massima lunghezza:

$$\min \quad \sum_{(u,v)\in A} c_{uv} \sum_{k=1}^{m} x_{uv}^{k} - \lambda \left(L - \sum_{(u,v)\in A} c_{uv} x_{uv}^{m} \right) \tag{5.21}$$

$$\text{s. t.} \quad \sum_{(u,v)\in A} c_{uv} x_{uv}^{k} \leq L \qquad \forall k \in \{1,\ldots,m-1\} \tag{5.22}$$

Il problema lagrangiano $\mathcal{L}(\lambda, \mathbf{x})$ è composto dalla funzione obiettivo (5.21) e soggetto ai vincoli (5.11)–(5.15), (5.18), (5.20), (5.22) e (5.17).

5.2 Pianificazione

Ora cerchiamo di rilassare i vincoli relativi all'instradamento, osservando se è possibile ottenere un lower bound. Assumiamo di avere tutte le distanze di viaggio $d_{ss'}$ nulle, in questo modo tutti i vincoli relativi alla sequenza di visita possono essere scaricati. Ridefiniamo la variabile x_s^{jh}, che assume valore 1 se il servizio s è assegnato all'infermiera j nel giorno h, 0 altrimenti. Ottenendo un nuovo problema:

$$\min\ \alpha_0 u + \alpha_1 l \tag{5.23}$$

$$\text{s. t.} \quad \sum_{h\in H} \sum_{j\in N_h} x_s^{jh} \geq f_s \qquad \forall s \in S \tag{5.24}$$

$$\sum_{k=h}^{h+e_s-1} \sum_{j \in N_h} x_s^{jk} \leq 1 \qquad \forall s \in S_+,\ \forall h \in H \tag{5.25}$$

$$\sum_{s \in S} \sum_{h \in H} \frac{c_s}{b_j} x_s^{jh} \leq u \qquad \forall j \in N \tag{5.26}$$

$$\sum_{s \in S} c_s x_s^{jh} \leq L \qquad \forall h \in H,\ \forall j \in N \tag{5.27}$$

$$\sum_{p \in P} \left(\sum_{j \in N} z_p^j - 1 \right) = l \tag{5.28}$$

$$z_p^j \geq x_s^{jh} \qquad \forall h \in H,\ \forall j \in N_h,\ \forall p \in P,\ s \in S_p \tag{5.29}$$

$$x_s^{jh} \in \{0,1\} \qquad \forall h \in H,\ \forall j \in N,\ s \in S \tag{5.30}$$

$$z_p^j \in \{0,1\} \qquad \forall j \in N,\ p \in P\,. \tag{5.31}$$

Un servizio deve essere ripetuto f_s volte in una settimana, vincoli (5.24). Un intermezzo e_s deve trascorrere tra due ripetizioni di servizio, vincoli (5.25). Il carico medio giornaliero u è ricavato dai vincoli (5.26). Il tempo massimo di un turno è imposto dai vincoli (5.27). Ed i vincoli (5.28) e (5.29) ricavano il valore di fidelizzazione paziente-infermiera l.

Poiché le distanze sono nulle, i costi sono una stima per difetto, ne segue che il valore ottenuto per questo nuovo problema è una stima per difetto del valore ottimo del problema di partenza.

Vedremo in sezione 13.1, che quando si pone $\alpha_1 = 0$ si riesce a calcolare un valore in tempi rapidi, diversamente quando $\alpha_1 > 0$ la soluzione ottima è raggiunta in tempi non accettabili.

5.3 Conclusioni

In questo capitolo abbiamo presentato due approcci per ricavare un lower bound per l'HHCP. La prima parte del capitolo è stata dedicata al rilassamento lagrangiano del sottoproblema DCVRP, mentre nella seconda parte il bound è ottenuto rilassando i vincoli di routing.

Riepilogo della parte II

In questa parte abbiamo indagato tre importanti tecniche di decomposizione del problema HHC, tutte basate sulla risoluzione del TSP come sottoproblema.

È stato possibile stabilire che per HHCP la decomposizione di Benders non è utilizzabile con il sottoproblema proposto in questa tesi, mentre la column generation, pur essendo applicabile in linea di principio, risulta poco utile nella pratica, in quanto computazionalmente troppo onerosa. In tal caso, infatti, ci si scontra con il problema di dover risolvere un PCTSP ad ogni iterazione, per il quale gli algoritmi sono ancora inefficienti.

Per quanto riguarda i lower bound ottenibili mediante rilassamento lagrangiano o dei vincoli di routing, la strada è computazionalmente percorribile per istanze di "qualsiasi" dimensione solo nel secondo caso, mentre nel primo lo è solo per istanze piccole. Vederemo però più avanti che anche la strada del rilassamento risulta poco utile, perché i bound che si ottengono sono eccessivamente laschi.

È pertanto indispensabile esplorare una via diversa per cercare la soluzione del problema HHC: una possibile alternativa è di utilizzare le meta euristiche, di cui si occupano i prossimi capitoli.

Parte III

Approcci euristici

Sommario della parte III

Abbiamo visto che le strategie di decomposizione indagate nella parte precedente non sono gli strumenti adeguati per risolvere il problema HHC come formulato in questa tesi. La strada alternativa che è possibile seguire per cercare una soluzione di HHCP è quella delle meta euristiche.

È da notare che il lavoro dei capitoli precedenti è importante: il rilassamento dei vincoli di routing, infatti, sarà la base per individuare rapidamente un punto iniziale per le meta euristiche presentate in questa parte.

Come abbiamo visto, sono più d'uno i livelli decisionali che caratterizzano l'HHCP ed anche i sottoproblemi nei quali esso può essere decomposto.

Nel capitolo 6 presentiamo due euristiche, VNS e ALNS, che forniscono due diverse strategie per selezionare opportune regioni del dominio (dette "intorni"), nelle quali ricercare soluzioni di HHCP.

Successivamente, il capitolo 7 descrive alcuni operatori: `move`, `swap`, k`swap`, `group` e `merge`. Essi sono trasformazioni di una soluzione in un'altra soluzione, con lo scopo di migliorare il valore della funzione obiettivo. Di questi operatori, oltre alla descrizione, vedremo il costo computazionale e l'efficacia nel contesto HHCP.

Un punto fondamentale è la verifica di ammissibilità di una soluzione ottenuta con ciascuna di queste mosse: a quest'aspetto è dedicato il capitolo 8. In esso verrà anche mostrato che tale verifica è computazionalmente molto costosa, al contrario di quanto accade per altri operatori in problemi più semplici (come ad esempio la '*chain relocation*' nel TSP). Ciò implica che implementazioni naïf dell'algoritmo di verifica dell'ammissibilità non sono sufficientemente efficienti e dunque, per affrontare casi reali, occorre mettere a punto strategie dedicate. Gli esperimenti dell'ultima parte mostreranno su un caso specifico la rilevanza di questi accorgimenti.

Il capitolo 9 si occupa in dettaglio dell'euristica Clarke & Wright nota come "algoritmo basato sui risparmi". Essa consente di raggiungere un duplice obiettivo: osservare come la generalizzazione del concetto di operatore consenta di progettare facilmente nuove euristiche, o estenderne di note, e avere una soluzione iniziale alternativa a quella dalla quale partono VNS e ALNS.

Capitolo 6

Euristiche neighbourhood-based

L'HHCP ha diversi livelli decisionali (carico di lavoro e fidelizzazione paziente-infermiera) e diversi sottoproblemi (pianificazione ed instradamento), da cui nascono intorni diversi che colgono alcuni aspetti del problema. In questo capitolo, dopo aver spiegato il concetto di intorno, presentiamo due euristiche, entrambe esplorano più intorni nella stessa procedura ma differiscono per la strategia con la quale è scelto l'intorno da esplorare. Infine presentiamo come ricavare la soluzione iniziale dalla quale entrambe partono.

6.1 Introduzione

Una *soluzione* ammissibile del problema σ equivale ad un *punto* nello *spazio delle soluzioni* $\mathcal{N}$. Ad una soluzione σ possiamo applicare un *operatore* $\mathtt{op}$; il risultato dell'applicazione, o *operazione*, è un'altra soluzione σ', ossia $\sigma' = \mathtt{op}(\sigma)$. Poiché l'operazione applicata ad un punto porta ad un nuovo punto, questo cambiamento è associato ad uno spostamento nello spazio delle soluzioni, ed è chiamato anche *mossa*. Diciamo che una mossa, o operazione, è *migliorativa* se la nuova soluzione ha un miglior valore della funzione obiettivo, ossia $f(\sigma') < f(\sigma)$ per un problema di minimo.

Per alcuni problemi, come per il TSP, è semplice costruire soluzioni ammissibili da soluzioni ammissibili, ad esempio $\mathtt{op} : \mathcal{N} \to \mathcal{N}$ è lo scambio della posizione di due nodi nella sequenza di visita. In problemi multi obiettivo, è usuale avere operatori $\mathtt{op}_k : \mathcal{N} \to \mathcal{N}_k$ con $\mathcal{N}_k \subset \mathcal{N}$, per esempio operazioni che portano a soluzioni che migliorano un solo obiettivo. Infatti, un problema multi obiettivo può avere aspetti decisionali in conflitto tra di loro, come nell'HHCP, nel quale si deve minimizzare la fidelizzazione paziente-infermiera e il carico di lavoro. Quando un operatore $\mathtt{op}_a$ pone l'enfasi per ottenere soluzioni che migliorino un obiettivo, quale il carico di lavoro, ed un altro operatore $\mathtt{op}_b$ non considera come migliorative le soluzioni in $\mathcal{N}_a$, per esempio le soluzioni di $\mathcal{N}_a$ peggiorerebbero la fidelizzazione paziente-infermiera, si ha che $(\mathcal{N}_a \cup \mathcal{N}_b) \setminus (\mathcal{N}_a \cap \mathcal{N}_b) \neq \emptyset$.

Il *neighbourhood* (*intorno*) di una soluzione σ, denotato con $\mathcal{N}_k(\sigma)$, è l'insieme delle possibili soluzioni ottenute partendo dal punto σ applicando l'operatore $\mathtt{op}_k$. Assumiamo sia $\mathtt{op}_k$ l'operatore che scambia l'assegnamento delle infermiere ai turni di uno stesso giorno, se nel giorno

h i turni in σ sono diversi dai turni σ' l'applicazione di $\mathtt{op}_k(\sigma)$ darà luogo a soluzioni diverse dall'applicazione di $\mathtt{op}_k(\sigma')$, pertanto $\mathcal{N}_k(\sigma) \neq \mathcal{N}_k(\sigma')$.

Dicendo che esploriamo un neighbourhood $\mathcal{N}_k(\sigma)$, intendiamo che consideriamo alcune delle soluzioni appartenenti a $\mathcal{N}_k(\sigma)$. Non è sempre possibile esplorare esaustivamente $\mathcal{N}_k(\sigma)$, pertanto l'esplorazione è spesso limitata nel tempo o nella quantità di soluzioni considerate. Quando identifichiamo una soluzione $\sigma' \in \mathcal{N}_k(\sigma)$, ci riferiamo ad una soluzione presa in modo deterministico o casuale a seconda dell'operatore.

Neighbourhood search (NS), o *local search* (LS), è un algoritmo impiegato per cercare un minimo locale partendo da un dato punto nello spazio delle soluzioni. Il neighbourhood del punto corrente è esplorato per trovare punti migliori e se trovato, questo punto migliore è preso come nuovo punto di partenza. Il processo è ripetuto fintanto che un criterio d'arresto è verificato, per esempio fintanto che un ottimo locale è trovato.

Vi sono varie meta euristiche basate sulla esplorazione di intorni, tra queste alcune sono applicate con successo nel contesto del routing e scheduling quali: la *variable neighbourhood search* (VNS) e l'*adaptive large neighbourhood search* (ALNS).

La VNS è una meta euristica di ricerca locale (*local search*) introdotta da Mladenović e Hansen (1997). In (Hansen et al., 2010) è mostrata la discendenza dal *variable metric method* proposto da Davidon (1991) e Fletcher e Powell (1963). È stata applicata nell'ottimizzazione continua (Mladenović et al., 2008), e in svariati contesti di ottimizzazione discreta, tra i quali rostering problem (Burke et al., 2008), job shop scheduling problem (Sevkli e Aydin, 2006; Amiri et al., 2010), leather nesting problem (Alves et al., 2012), one-commodity pickup-and-delivery travelling salesman problem (Mladenović et al., 2012) e vehicle routing problem with time windows (Bräysy, 2003).

Basata su *large neighborhood search* (*LNS*) (Shaw, 1997; Shaw, 1998), adaptive large neighbourhood search (ALNS) è stata introdotta da Pisinger e Ropke (2007) nel contesto del vehicle routing problem. Da allora è stato applicato a una varietà di altri contesti nell'ambito dei problemi di routing e scheduling come multiple depot vehicle scheduling problem (Pepin et al., 2009), cumulative capacitated vehicle routing problem (Ribeiro e Laporte, 2012), capacitated arc routing problems with stochastic demands (Laporte et al., 2010), fixed-charge network flow optimization (Hewitt et al., 2010), service technician routing and scheduling (Kovacs et al., 2012), and pickup and delivery problems with cross-docking (Petersen e Ropke, 2011).

Le differenza principale tra le due meta euristiche è data dalla scelta della mossa: VNS sceglie sistematicamente l'intorno da esplorare, mentre nell'ALNS sono ripetute le mosse che sono considerate di 'moda'. In questo capitolo presentiamo lo schema generico delle meta euristiche applicato all'HHCP. A seguire le sezioni gli aspetti comuni alle due meta euristiche: la soluzione iniziale, gli operatori, la verifica di ammissibilità.

6.2 Variable neighbourhood search

Questa meta euristica esplora lo spazio delle soluzioni mediante l'applicazione sistematica di neighbourhood predefiniti. Molto spesso, ma non sempre, i neighbourhood sono di complessità

crescente e incorporati gli uni negli altri. Ogni neighbourhood è esplorato localmente accettando solo mosse miglioranti. Quando non sono possibili miglioramenti, il processo di ricerca passa al neighbourhood successivo nella sequenza. Dopo che l'ultimo neighbourhood è stato esplorato localmente, la ricerca può ripartire con il primo neighbourhood. La sequenza di neighbourhood può essere ciclata fintanto che un criterio di fermata non è soddisfatto.

algoritmo 6.1 VNS(σ)

1: **repeat**
2: $k \leftarrow 1$
3: **repeat**
4: $\sigma' \leftarrow$ `Shake`(σ, k) ..... # $\sigma' \in \mathcal{N}_k(\sigma)$
5: $\sigma'' \leftarrow$ `FirstImprovement`(σ', k)
6: `NeighbourHoodChange`(σ, σ'', k)
7: **until** $k \neq k_{max}$
8: **until** `run time` $<$ `run time`$_{max}$

Ora mostriamo come applicare la VNS all'HHCP. L'algoritmo 6.1 parte da una soluzione iniziale σ. In linea 4, la procedura `Shake` è eseguita, ossia la soluzione σ' è generata scegliendo a caso una soluzione nel k-esimo neighbourhood di σ. La ricerca locale `FirstImprovement` è applicata in linea 5, alla quale segue la procedura `NeighbourHoodChange` che determina se passare al neighbourhood successivo o ricominciare dal primo (si veda algoritmo 6.2).

algoritmo 6.2 `NeighbourHoodChange`(σ, σ', k)

1: **if** $f(\sigma') < f(\sigma)$ **then**
2: $\sigma \leftarrow \sigma';\ k \leftarrow 1$
3: **else**
4: $k \leftarrow k + 1$
5: **end if**

La ricerca locale `FirstImprovement` (algoritmo 6.4) equivale a un $\overline{\kappa}$`Improvement` richiedendo un solo miglioramento, ovvero $\overline{\kappa} = 1$. Nella $\overline{\kappa}$`Improvement` (algoritmo 6.3) una soluzione iniziale σ è data, ed usando il neighbourhood $\mathcal{N}_k$ c'è uno spostamento alla soluzione σ'. Se la nuova soluzione ha un migliore valore della funzione obiettivo $f(\sigma') < f(\sigma)$, memorizziamo la nuova soluzione come miglior soluzione locale e aggiorniamo il contatore dei miglioramenti $\underline{\kappa}$ (linea 5). Lo stesso avviene se la soluzione trovata è buona tanto quanto la precedente $f(\sigma') = f(\sigma)$ (linea 8). Talvolta, in particolare nel caso della `FirstImprovement`, si preferiscono soluzioni strettamente migliori $f(\sigma') < f(\sigma)$ ed è al tempo stesso opportuno non avere, come punto di partenza per l'esplorazione dell'intorno $\mathcal{N}_k$, una soluzione σ' con stesso valore della funzione obiettivo dato da σ, ossia $f(\sigma') = f(\sigma)$. Pertanto, si può impostare una probabilità Q per accettare o meno la nuova soluzione σ' come migliorativa.

Non è possibile svolgere una ricerca esaustiva dell'intorno, pertanto occorre incrementare il numero dei tentativi di ricerca senza miglioramenti `step` (linea 10). La procedura termina

quando ha ottenuto $\overline{\kappa}$ miglioramenti o quando sono state esplorate consecutivamente $\overline{\kappa}$ soluzioni senza miglioramento (linea 2).

algoritmo 6.3 $\overline{\kappa}$`Improvement`$(\overline{\kappa}, \sigma, k, Q \in [0,1], \texttt{step}_{max})$

```
1: κ ← 0; step ← 0
2: while step < step_max ∧ κ ≠ κ̄ do
3:    σ' ∈ N_k(σ)
4:    if f(σ') < f(σ) then
5:       σ ← σ'; step ← 0; κ ← κ + 1
6:    else
7:       if f(σ') == f(σ) ∧ rand([0,1]) ≤ Q then
8:          σ ← σ'; step ← 0; κ ← κ + 1
9:       else
10:         step ← step + 1
11:      end if
12:   end if
13: end while
```

algoritmo 6.4 `FirstImprovement`$(\sigma, k, Q \in [0,1], \texttt{step}_{max})$

1: $\overline{\kappa}$`Improvement`$(1, \sigma, k, Q \in [0,1], \texttt{step}_{max})$

6.3 Adaptive large neighbourhood search

Questa meta euristica lavora con svariati operatori di distruzione e riparazione predefiniti. Nei problemi di routing gli operatori di distruzione e riparazione sono generalmente applicati insieme, figurando come una singola operazione. Ad ogni iterazione, l'algoritmo sceglie casualmente coppie di operatori di distruzione e di riparazione, e li applica alla soluzione corrente. Sono accettate mosse peggiorative secondo una legge probabilistica, come in simulated annealing. Inizialmente, a tutti gli operatori è associato un punteggio (*score*), per esempio il rapporto tra il numero dei successi di un operazione ed il numero di tentativi. Per successo di un operazione si può intendere o il fatto che ha restituito una soluzione ammissibile, o il fatto che ha portato ad una miglior soluzione. Nella nostra implementazione si opta per questo secondo caso. La scelta di un operatore ad una certa iterazione è ottenuta in accordo con il principio della *roulette-wheel*. Più precisamente, se ς_j denota lo score dell'operatore $\texttt{op}_j$ $\forall j$, allora l'operatore $\texttt{op}_i$ è selezionato con probabilità $\pi_i = \varsigma_i / \sum_j \varsigma_j$. Si può pensare allo score come ad un indice della bontà dell'operatore dato dalla memoria dei successi delle operazioni.

La nostra implementazione è descritta nell'algoritmo 6.5. Una soluzione iniziale σ è data, e un vettore $\overline{\varsigma}$ di score è inizializzato. A ogni iterazione del ciclo principale, usando le probabilità π_i, la procedura `Choice`$(\overline{\varsigma})$ seleziona l'operatore k. Una nuova soluzione σ' è ottenuta applicando l'operatore k (line 5). Nel framework standard dell'ALNS, la soluzione locale σ è usata come

algoritmo 6.5 ALNS(σ)

1: $\overline{\varsigma} \leftarrow \{\varsigma_1, \ldots, \varsigma_k, \ldots, \varsigma_m\}$
2: $\sigma^* \leftarrow \sigma$
3: **repeat**
4: $k \leftarrow$ `Choice`($\overline{\varsigma}$)
5: $\sigma' \in \mathcal{N}_k(\sigma^*)$
6: **if** σ' is feasible **then**
7: $\sigma \leftarrow \sigma'$
8: **end if**
9: `Update`($\overline{\varsigma}$)
10: **if** $f(\sigma) < f(\sigma^*)$ **then**
11: $\sigma^* \leftarrow \sigma$
12: **end if**
13: **until** `run time` $<$ `run time`$_{max}$

soluzione iniziale dell'operatore k; nella nostra implementazione, noi applichiamo l'operatore alla migliore soluzione corrente σ^*. Dopo la verifica di ammissibilità (line 6), la procedura `Update`($\overline{\varsigma}$) incrementa il valore dello score. Infine, la migliore soluzione è memorizzata in σ^*. L'euristica termina quando il tempo limite `run time`$_{max}$ è raggiunto.

Le soluzioni iniziali hanno tempi molto elevati di lavoro straordinario, il nostro obiettivo primario è azzerarli. Pertanto, nella nostra implementazione dell'ALNS consideriamo inammissibili soluzioni peggiorative ed accettiamo sempre soluzioni migliorative, a differenza della proposta originale dell'algoritmo che prevede di accettare peggiori soluzioni con una legge probabilistica.

Vedremo nel capitolo 8 che questa scelta di accettare sempre migliori soluzioni permette di evitare preventivamente la verifica della nuova soluzione nelle caratteristiche relative alla parte di scheduling, come frequenza e intermezzo.

6.4 Soluzione iniziale

Gli algoritmi VNS e ALNS richiedono una soluzione iniziale dalla quale partire, a tal fine abbiamo messo a punto una procedura parametrizzabile, basata sulla risoluzione del modello di programmazione matematica in seguito definito. Introduciamo ora la seguente notazione. Dato $\tilde{d}_s = \sum_{s' \in S \setminus \{s\}} d_{ss'}/(|S|-1)$ il *tempo medio di viaggio da un servizio s verso un altro servizio s'*, definiamo il *costo medio* come somma del tempo medio di viaggio e tempo di servizio $\tilde{c}_s = \tilde{d}_s + a_s$. Definiamo $\hat{c}_s = \hat{d}_s + a_s$, con $\hat{d}_s$ il primo quartile del campione ordinato dei tempi di percorrenza $d_{ss_{\pi(1)}}, \ldots, d_{ss_{\pi(|S|)}}$, dove $\pi(\eta)$ è una funzione di ordinamento crescente.

Abbiamo introdotto nel capitolo 5 il modello di programmazione matematica (5.23)–(5.31), nel quale i vincoli d'instradamento sono rilassati dando luogo a costi di servizio dipendenti solo dal costo di lavoro a_s. Il problema è NP-Hard, tuttavia il solver può essere impostato per ottenere una soluzione ammissibile in tempi rapidi, senza giungere alla soluzione ottima. La soluzione ammissibile ottenuta può essere usata come soluzione iniziale e può avere turni che

superino il massimo tempo di lavoro giornaliero L, Al fine di avere soluzioni ammissibili più conformi alle esigenze, espandiamo il modello come segue:

$$\min \alpha_0 u + \alpha_1 l + \alpha_2 w + \alpha_3 v \tag{6.1}$$

$$\text{s. t.} \sum_{h\in H}\sum_{j\in N_h} y_s^{jh} \geq f_s \qquad \forall s \in S \tag{6.2}$$

$$\sum_{k=h}^{h+e_s-1}\sum_{j\in N_h} y_s^{jk} \leq 1 \qquad \forall h \in H,\ \forall s \in S_h \tag{6.3}$$

$$\sum_{s\in S}\sum_{h\in H} \frac{c_s}{b_j} y_s^{jh} \leq u \qquad \forall j \in N \tag{6.4}$$

$$\sum_{s\in S} c_s y_s^{jh} \leq L - w \qquad \forall h \in H,\ \forall j \in N_h \tag{6.5}$$

$$\Lambda \leq \sum_{s\in S} y_s^{jh} \qquad \forall h \in H,\ \forall j \in N_h \tag{6.6}$$

$$\sum_{s\in S} y_s^{jh} \leq v \qquad \forall h \in H,\ \forall j \in N_h \tag{6.7}$$

$$v \leq \Upsilon \tag{6.8}$$

$$\zeta \leq \sum_{s\in S} c_s y_s^{jh} \qquad \forall h \in H,\ \forall j \in N_h \tag{6.9}$$

$$\sum_{p\in P}\left(\sum_{j\in N} z_p^j - 1\right) = l \tag{6.10}$$

$$z_p^j \geq y_s^{jh} \qquad \forall h \in H,\ \forall j \in N_h,\ \forall p \in P,\ s \in S_p \cap S_h \tag{6.11}$$

$$w \leq W^{max} \tag{6.12}$$

$$w \in \mathbb{R}_+ \tag{6.13}$$

$$y_s^{jh} \in \{0,1\} \qquad \forall h \in H,\ \forall j \in N_h,\ s \in S_h \tag{6.14}$$

$$z_p^j \in \{0,1\} \qquad \forall j \in N,\ p \in P\,. \tag{6.15}$$

Quando si desidera avere la prima soluzione ammissibile dal solver MILP del sistema (5.23)–(5.31) si ha una soluzione nella quale tutti i servizi che possono essere svolti il giorno h, e rispettano frequenza e intermezzo, sono in uno stesso tour. Per arginare il fenomeno, in aggiunta al modello (5.23)–(5.31), in funzione obiettivo desideriamo anche minimizzare il massimo numero di servizi svolti in una giornata denotato variabile v, e modellato dal vincoli (6.7). Il massimo numero di servizi v è compreso tra Λ e Υ, rispettivamente vincoli (6.6) e vincolo (6.8). Nei vincoli (6.9) si desidera che il tempo di servizio giornaliero di ogni infermiera sia maggiore del parametro ζ.

La soluzione del modello proposto porta ad assegnare ad ogni infermiera i servizi giornalieri nell'arco di tutta la settimana, ove la quantità di servizi per infermiera dovrebbe essere più bilanciata. È anche possibile migliorare la qualità della soluzione iniziale calcolando la sequenza di visita ottima risolvendo il TSP. Ora mostriamo i dettagli nell'algoritmo 6.6. Desideriamo avere una soluzione ammissibile in tempi rapidi, pertanto impostiamo i parametri del solver

per non eccedere il tempo limite di calcolo $\texttt{timelimit}_{max}$, e per terminare il processo di ricerca della soluzione ottima dopo aver trovato `numSolutions` soluzioni ammissibili.

Nelle righe 1–12, cerchiamo una soluzione ammissibile iniziando non ammettendo il tempo di lavoro straordinario ($W^{max} = 0$) ed avendo il costo di viaggio dato dal quartile ($c_s = \hat{d}_s + a_s$). Se non è restituita una soluzione si risolve annullando il costo di viaggio ($c_s = 0 + a_s$). Infine, se ancora non è ottenuta una soluzione si ammettono i costi di tempo di lavoro straordinario, impostando $W^{max} = \infty$, ed avendo il costo di viaggio dato dal quartile ($c_s = \hat{d}_s + a_s$). Se ancora non è stata ottenuta una soluzione ammissibile, allora reiteriamo il processo incrementando il tempo limite (linea 11).

Avendo una soluzione ammissibile, è possibile ottimizzare la sequenza di visita giornaliera come segue. Una soluzione ammissibile associa ad ogni infermiera j, in ogni giorno della settimana $h \in H$, un insieme di servizi $\{s \in S \mid y_s^{jh} = 1\}$ (linea 15). Per ognuno di questi insiemi di servizi, la sequenza ottima di servizi è ottenuta computando il TSP (linea 16).

algoritmo 6.6 `SoluzioneIniziale`($\texttt{timelimit}_{max}$, `numSolutions`)

1: `timelimit` $\leftarrow 120$ s
2: $\mathbf{x} \leftarrow$ infeasible
3: **while** $\mathbf{x}$ is infeasible $\wedge$ `timelimit` $\leq \texttt{timelimit}_{max}$ **do**
4: $\quad\mathbf{x} \leftarrow \text{solve}(\{(6.2) - (6.15)\}, W^{max} = 0, d_s = \hat{d}_s, \texttt{timelimit}, \texttt{numSolutions})$
5: $\quad$**if** $\mathbf{x}$ is infeasible **then**
6: $\quad\quad\mathbf{x} \leftarrow \text{solve}(\{(6.2) - (6.15)\}, W^{max} = 0, d_s = 0, \texttt{timelimit}, \texttt{numSolutions})$
7: $\quad$**end if**
8: $\quad$**if** $\mathbf{x}$ is infeasible **then**
9: $\quad\quad\mathbf{x} \leftarrow \text{solve}(\{(6.2) - (6.15)\}, W^{max} = \infty, d_s = \hat{d}_s, \texttt{timelimit}, \texttt{numSolutions})$
10: $\quad$**end if**
11: $\quad$`timelimit` $\leftarrow$ `timelimit` $+ 120$
12: **end while**
13: **for all** $h \in H$ **do**
14: $\quad$**for all** $j \in N$ **do**
15: $\quad\quad T_i \leftarrow \{s \in S \mid y_s^{jh} = 1\}$
16: $\quad\quad T_i \leftarrow \text{TSP}(T_i)$
17: $\quad\quad \mathcal{T} \leftarrow \mathcal{T} \cup T_i$
18: $\quad\quad i \leftarrow i + 1$
19: $\quad$**end for**
20: **end for**

Quando la soluzione iniziale presenta tour con molti servizi e tour con pochi servizi, sarà opportuno bilanciare il dislivello spostando i servizi mediante l'operatore `move` che tratteremo nel prossimo capitolo.

6.5 Conclusioni

In questo capitolo, dopo aver spiegato il concetto di intorno, abbiamo presentato due euristiche: VNS e ALNS. Entrambe esplorano più intorni nella stessa procedura, ma differiscono per la strategia con la quale è scelto l'intorno da esplorare. Infine abbiamo presentato come ricavare la soluzione iniziale dalla quale entrambe partono.

Capitolo 7

Lista operatori

Nel precedente capitolo abbiamo descritto le meta euristiche VNS e ALNS; abbiamo detto che tra i loro aspetti comuni c'è la soluzione iniziale e gli operatori che presentiamo in questo capitolo.

Il fine dell'operatore è ottenere una nuova soluzione con migliore valore della funzione obiettivo. Prendiamo l'operatore `move`, che sposta un servizio da un tour ad un altro tour, la mossa ha complessità $O(1)$ ed è possibile avere un intorno che descrive lo spazio di tutte le soluzioni ammissibili. Non possiamo svolgere un'esplorazione esaustiva del precedente intorno, occorre selezionare a priori su quali elementi (tour, servizio) applicare una mossa.

Quando selezioniamo alcuni elementi è perché ci aspettiamo di avere un miglioramento di alcuni degli aspetti del problema, ad esempio azzerare i tempi di lavoro straordinario. Nelle seguenti sezioni descriviamo gli operatori, gli aspetti del problema che ci si aspetta siano presi in considerazione ed il costo computazionale della ricerca degli elementi sui quali applicare l'operatore.

7.1 `move`

La mossa `move` consiste nello spostare un servizio $s \in T_i$ in coda ai servizi di un tour $T_{i'}$, con $i \neq i'$, e in una rigenerazione dei valori della distanza totale ℓ dei rispettivi tour (si veda algoritmo 7.1). Questa operazione di spostamento del servizio da un tour ad un altro, equivale

algoritmo 7.1 `move`$(s, T_i, T_{i'})$

1: $T_i \leftarrow T_i \setminus \{s\}$ ..... $\#\ T_i = \{0, s_a, \ldots, s_c, s, s_d, \ldots, s_b, 0\} \rightarrow \{0, s_a, \ldots, s_c, s_d, \ldots, s_b, 0\}$
2: $T_{i'} \leftarrow T_{i'} \cup \{s\}$ ..... $\#\ T_{i'} = \{0, s_e, \ldots, s_f, 0\} \rightarrow \{0, s_e, \ldots, s_f, s, 0\}$
3: $\ell_i \leftarrow \text{regenerate}(T_i)$
4: $\ell_{i'} \leftarrow \text{regenerate}(T_{i'})$

a una ricerca, un'eliminazione ed un inserimento. Si potrebbero selezionare strutture dati che hanno un basso costo computazionale per tali operazioni, per esempio $O(log|T|)$. Tuttavia, la costante moltiplicativa, nascosta dall'andamento asintotico del calcolo computazionale, non è trascurabile nel caso in cui i tour sono composti di un basso numero di elementi, cioè $|T| \leq 26$. Nella pratica la velocità delle operazioni di modifica di una lista, o un vettore, ammortizzano

il costo non logaritmico di ricerca. La rigenerazione del tour consiste in una sommatoria delle distanze. Possiamo considerare il costo dell'operazione come un costo costante.

In seguito presentiamo differenti modi di selezionare i due tour, T_i e $T_{i'}$, ed il servizio s. Se non diversamente descritto, il servizio è scelto casualmente. Talvolta, in particolare se i costi di viaggio e di servizio tendono ad essere omogenei, potrebbero esistere più coppie di tour con le stesse caratteristiche indicate per la selezione; in questo caso, è scelta la prima coppia di tour con riscontro positivo.

M1: casuale

I due tour T_i e $T_{i'}$, sono scelti casualmente. Nella scelta non viene svolta nessuna verifica di compatibilità di giorni. La complessità computazionale di M1 è data dalla generatore dei numeri pseudocasuali, ovvero è $O(1)$.

M2: da tour lungo a tour corto

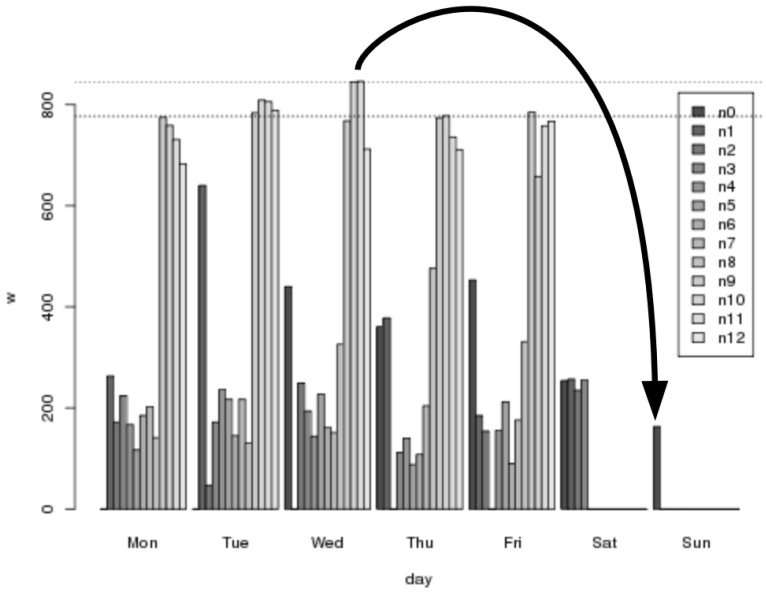


Figura 7.1: Nell'asse delle x i tour sono raggruppati per giorno della settimana, e per ogni giorno i tour sono ordinati per l'infermiera n_j alla quale il tour è assegnato. Sull'asse delle y è riportata la durata totale di un tour ℓ. Una freccia indica un possibile spostamento di un servizio tra due tour con infermiera e giorno di servizio differenti.

Si è osservato, vedi figura 7.1, che uno degli elementi di sbilanciamento delle soluzioni iniziali è dato dalla differenza di lunghezza tra due tour $\ell_i - \ell_{i'}$. In particolare, quando si vuole massimizzare questa differenza si ha che si sposta un servizio da un tour con molto carico di lavoro ad uno con basso carico di lavoro. Ovvero si aiuta a raggiungere il tempo limite di percorrenza L, tendendo ad azzerare il costo $\alpha_2 w$ della violazione del vincolo. Si richiede che i tour contengano servizi compatibili, ossia che dopo l'inserimento di $s \in T_i$ nel tour $T_{i'}$ esista

un giorno h tale che $g_s^h = 1$ e $q_{i'}^h = 1$:

$$\arg\max_{(i,i')}\{\ell_i - \ell_{i'} \mid T_i, T_{i'} \in \mathcal{T} \wedge \exists h \in H : q_i^h \wedge q_{i'}^h = 1\}\,. \tag{7.1}$$

La complessità della mossa della mossa è $O(|\mathcal{T}|^2)$, che per definizione (vedi capitolo 1) di $\mathcal{T}$ è $O((|N| \cdot |H|)^2)$.

M3: da tour lungo a tour corto, nello stesso giorno

Nella precedente mossa è possibile avere $q_i^h = q_i^{h'} = q_i^{h''} = 1$, ossia tutti i servizi contenuti nel tour T_i possono essere svolti nei tre giorni $\{h, h', h''\}$. Inoltre, assumiamo che sia assegnato il giorno h al tour T_i e sia assegnato il giorno h'' al tour $T_{i''}$. Si osservi che è possibile applicare l'operatore M1, poiché esiste h'' che verifica 7.1. In questo contesto, può accadere che lo spostamento del servizio s dal tour T_i al tour $T_{i''}$ porti alla violazione del vincolo (1.6), relativo al rispetto del tempo d'attesa e_s. Pertanto, non si richiede la verifica dell'ammissibilità delle parte di scheduling. Poiché computazionalmente onerosa, possiamo evitare questa verifica spostando il servizio s in un tour $T_{i'}$ al quale è assegnato il giorno h. In sintesi, la mossa M3 equivale alla mossa M2 con il giorno h predeterminato con un generatore pseudo casuale (vedi figura 7.2).

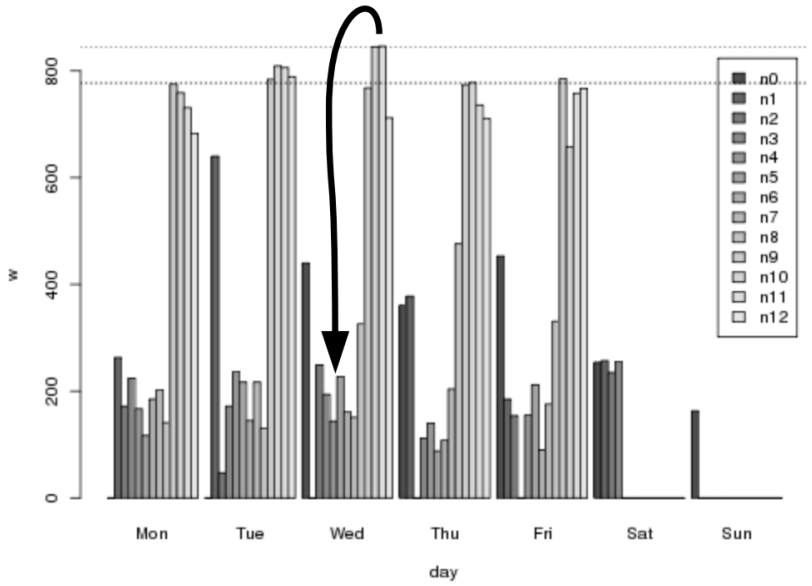


Figura 7.2: Nell'asse delle x i tour sono raggruppati per giorno della settimana, e per ogni giorno i tour sono ordinati per l'infermiera n_j alla quale il tour è assegnato. Sull'asse delle y è riportata la durata totale di un tour ℓ. In ogni giorno della settimana, molti tour hanno un tempo ℓ molto distante dal valore medio di quella giornata. Una freccia indica un possibile spostamento di un servizio dal tour assegnato all'infermiera n_{10} al tour assegnato all'infermiera n_4 in uno stesso giorno della settimana.

$$\arg\max_{(i,i')}\{\ell_i - \ell_{i'} \mid T_i, T_{i'} \in \mathcal{T}^h\} \tag{7.2}$$

Oltre a evitare la verifica della parte di scheduling, questa mossa porta a equilibrare il carico di lavoro giornaliero u^h. Per definizione (vedi capitolo 1) di $\mathcal{T}^h$, la complessità computazionale è relativa al numero di infermiere in turno nel giorno h, cioè $O((|N^h|)^2) = O(|\mathcal{T}^h|^2)$.

M4: da tour di un'infermiera sovraccarica a tour di un'infermiera sotto carico

Un caso particolarmente frequente nella soluzione iniziale è di avere un'infermiera j con un carico di lavoro u^j elevato, e un'infermiera j' con un carico di lavoro $u^{j'}$ basso (vedi figura 7.3). Poiché u è il massimo carico di lavoro di tutte le infermiere, ossia $u = \max\{u^1, \ldots, u^{|N|}\}$, ne segue che lo spostamento del servizio dall'infermiera più sovraccarica potrebbe abbassare il valore di u.

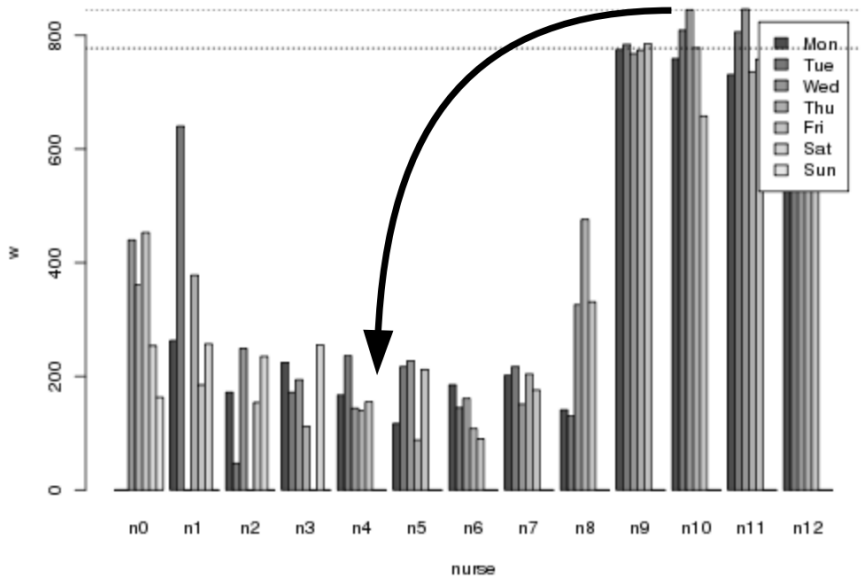


Figura 7.3: Nell'asse delle x i tour sono raggruppati per infermiera, e per ogni infermiera i tour sono ordinati per il giorno della settimana assegnato. Sull'asse delle y è riportata la durata totale di un tour ℓ. Le prime nove infermiere hanno un carico di lavoro u^j, diversamente dalle ultime quattro. Una freccia indica un possibile spostamento di un servizio dal tour assegnato all'infermiera n_{10} al tour assegnato all'infermiera n_4

Selezioniamo un tour T_i assegnato all'infermiera j, e un tour $T_{i'}$ assegnato all'infermiera j' tali che sia massima la differenza di carico di lavoro $u^j - u^{j'}$:

$$\mathcal{A} = \arg\max_{(i,i')}\{u^j - u^{j'} \mid j, j' \in N, T_i \in \mathcal{T}^j, T_{i'} \in \mathcal{T}^{j'}\}\,. \tag{7.3}$$

La complessità di questa mossa è $O(|\mathcal{T}|^2)$.

M5: da tour lungo di un'infermiera sovraccarica a tour corto di un'infermiera sotto carico

Questa mossa equivale alla mossa M2, ove al posto di tutti i tour $\mathcal{T}$ usiamo l'insieme dei tour $\mathcal{A}$ di tutte le coppie di tour assegnati alle infermiere j e j' tali che la differenza di carico di lavoro è massima, eq. 7.3. In altre parole, riduciamo le possibili soluzioni date dalla mossa M2, optando per quella ove il valore di ℓ_i è molto grande. Pertanto, spostare il servizio da T_i aiuta ad abbassare il valore ℓ_i e auspicabilmente anche il valore di u^j; per quanto detto in precedenza

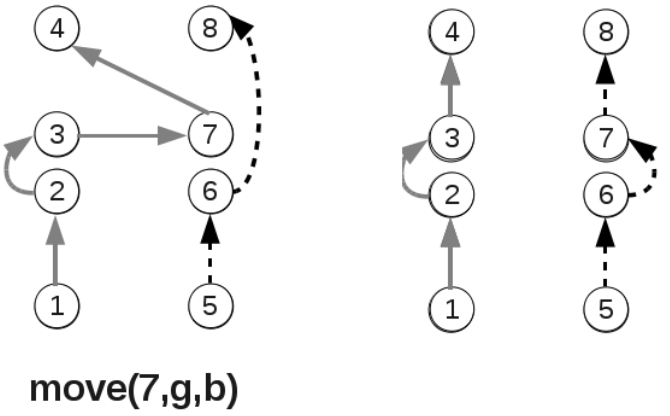


Figura 7.4: Il tour grigio è denotato con g, il tour nero con b e i servizi con i numeri. L'immagini di sinistra mostra la situazione nella quale l'operazione `move`$(7, g, b)$ *produce la soluzione graficata nell'immagine di destra.*

ha come effetto abbassare u. Formalmente:

$$\arg\max\{\ell_i - \ell_{i'} \mid (i, i') \in \mathcal{A},\ T_i, T_{i'} \in \mathcal{T}, \exists h \in H : q_i^h \wedge q_{i'}^h\} \tag{7.4}$$

La complessità computazionale è la somma delle complessità delle rispettive mosse (M2 + M4), ossia $O(|\mathcal{T}|^2)$. Sebbene asintoticamente M5 abbia la stessa complessità dell'operatore M4, ai fini pratici la costante moltiplicativa di M5 porta questa mossa ad essere più costosa. In altre parole, sebbene M4 e M5 abbiano stessa complessità, in un dato lasso di tempo il numero di mosse M4 è maggiore del numero di mosse M5.

M6: spostare s in un tour dove c'è un servizio vicino

Nelle precedenti mosse l'enfasi della scelta dei tour sui quali applicare la `move` era posta esclusivamente sulle caratteristiche dei tour. In questa mossa, poniamo l'attenzione sul trovare servizi s e s' che siano vicini spazialmente ma in due tour differenti, su questi tour verrà applicata la `move`. Per esempio, si veda il caso graficato in figura 7.4, ove servizi 6 e 7 appartengono a tour differenti, sebbene s e s' siano localizzati in luoghi geograficamente vicini. C'è vicinanza tra un servizio s ed un servizio s' se $d_{ss'} \leq \delta$. La ricerca inizia con $\delta \leftarrow 0$, ovvero s e s' sono nello stesso luogo, nel caso il tipo di istanze ammetta la non esistenza di servizi con δ si reitera la ricerca ponendo $\delta \leftarrow \delta + \Delta$ finché $\delta \geq \delta_{max}$. La mossa raccoglie in uno stesso tour servizi dispersi in differenti tour, quando $\delta = 0$ i servizi sono quasi sempre dello stesso paziente, pertanto l'accorpamento di servizi di uno stesso paziente in un unico tour, quindi ad un unica infermiera, rende auspicabile il miglioramento del valore di fidelizzazione paziente-infermiera.

$$\{(i, i') \mid T_i, T_{i'} \in \mathcal{T},\ s \in T_i,\ s' \in T_{i'},\ \bigvee_{h \in H} q_i^h \wedge q_{i'}^h,\ d_{ss'} \leq \delta\} \tag{7.5}$$

La complessità computazionale della mossa è $O(|S|^2)$.

7.2 swap

La mossa `swap` consiste nello scambiare un servizio $s \in T_i$ ed un servizio $s' \in T_{i'}$, ossia nell'ordine di visita dei servizi di T_i comparirà il servizio s' al posto del servizio s, e analogamente in $T_{i'}$ vi sarà il servizio s al posto del servizio s'. Segue una rigenerazione dei valori della distanza totale ℓ dei rispettivi tour (si veda algoritmo 7.2). Analogamente al discorso svolto per l'operazione

algoritmo 7.2 `swap`$(s, s', T_i, T_{i'})$

1: $T_i \leftarrow$ `swap`(s, s', T_i) ..... # $T_i = \{0, s_a, \ldots, s, \ldots, s_b, 0\} \rightarrow \{0, s_a, \ldots, s', \ldots, s_b, 0\}$
2: $T_{i'} \leftarrow$ `swap`$(s', s, T_{i'})$ ..... # $T_{i'} = \{0, s_c, \ldots, s', \ldots, s_d, 0\} \rightarrow \{0, s_c, \ldots, s, \ldots, s_d, 0\}$
3: $\ell_i \leftarrow$ regenerate(T_i)
4: $\ell_{i'} \leftarrow$ regenerate$(T_{i'})$

`move`, lo spostamento dei servizi e rigenerazione tour possiamo considerarli come un costo costante.

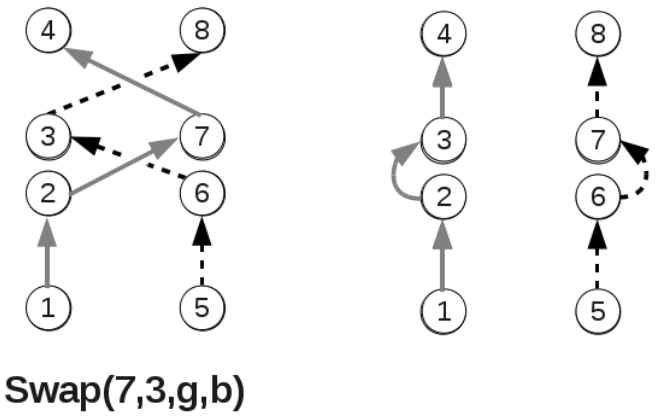


Figura 7.5: Il tour grigio è denotato con g, il tour nero con b e i servizi con i numeri. L'immagini di sinistra mostra la situazione nella quale l'operazione `swap`$(7, 3, g, b)$ *produce la soluzione graficata nell'immagine di destra.*

In seguito presentiamo due principali modi di selezionare i due tour, T_i e $T_{i'}$, ed il servizi s e s': una selezione casuale, ed una selezione per prossimità. Il concetto di prossimità si può intuire osservando il caso graficato in figura 7.5, ove i servizi 2 e 3 sono vicini, ed anche i servizi 6 e 7 sono vicini. I servizi 2 e 3 appartengono a tour differenti, ed i servizi 6 e 7 appartengono a tour differenti, sebbene vicini.

Talvolta, con le stesse caratteristiche indicate per la selezione potrebbero esistere più coppie di tour, in particolare se i costi di viaggio e di servizio tendono ad essere omogenei; in questo caso viene scelta la prima coppia di tour con riscontro positivo. Analogamente alla mossa `move`, nelle seguenti operazioni teniamo conto di non riutilizzare nella stesso tipo di mossa successiva la coppia di tour precedentemente richiesta. Se non diversamente descritto, i servizi sono scelti deterministicamente dall'operatore. Al termine dell'operazione verrà restituita la tupla (s, s', i, i').

S1: casuale

La selezione dei tour e dei servizi è ottenuta mediante un generatore di numeri pseudocasuali. La complessità computazionale è $O(1)$.

S2: servizi vicini ma in tour differenti

Se esistono due coppie di servizi (s, s') e (r, r') tali che $d_{ss'} \leq \delta$ e $d_{rr'} \leq \delta$, con $s, r \in T_i$ e $s', r' \in T_{i'}$, allora possiamo scambiare r con s':

$$\mathcal{B} = \{(i, i') \mid s \in T_i,\ s' \in T_{i'},\ d_{ss'} \leq \delta, r \in T_i,\ r' \in T_{i'},\ d_{rr'} \leq \delta\}\,. \tag{7.6}$$

Analogamente alla mossa M6, quando $\delta = 0$ i servizi sono nello stesso luogo, generalmente per lo stesso paziente. In questo caso è auspicabile la decrescita del valore della fidelizzazione paziente-infermiere. Inoltre accorpando i servizi dei due tour, ci si aspetta la decrescita del tempo totale dei tour ℓ_i e $\ell_{i'}$, ossia una possibile decrescita del valore di carico di lavoro u. Se applicando l'operatore l'insieme $\mathcal{B}$ è vuoto, allora verrà svolta l'operazione S1. Il costo computazionale è di $O(|S|^2)$.

S3: servizi vicini ma in tour differenti nello stesso giorno

In analogia alla mossa M3, computiamo con un generatore pseudo casuale un giorno h, e restringiamo la ricerca dei possibili tour della mossa S2:

$$\mathcal{B}^h = \{(i, i') \in \mathcal{B} \mid T_i, T_{i'} \in \mathcal{T}^h\}\,. \tag{7.7}$$

Con la stessa argomentazione portata per la mossa `move` M3, possiamo affermare che questo operatore evita la verifica della parte di scheduling. Se applicando l'operatore l'insieme $\mathcal{B}^h$ è vuoto, allora verrà svolta l'operazione S1. Il costo computazionale è di $O(|S^h|^2)$.

7.3 k`swap`

Abbiamo appena presentato un operatore che presi un servizio s di un tour T_i e un servizio s' di un altro tour $T_{i'}$ li scambia. Ora estendiamo questo scambio non a un solo servizio ma a k servizi consecutivi, cioè scambiamo una sequenza contigua di k servizi $s_1, \ldots, s_k$ di un tour T_i con una sequenza contigua di k servizi $s'_1, \ldots, s'_k$ di un altro tour $T_{i'}$, vedi algoritmo 7.3. In figura 7.6 è

algoritmo 7.3 k`swap`$(\overline{s}, \overline{s}', T_i, T_{i'})$

1: **for all** $l \in \{1, \ldots, k\}$ **do**
2: $\quad T_i \leftarrow \text{swap}(s_l, s'_l, T_i)$ ..... $\#\ T_i = \{0, a, \ldots, s_l, s_{l+1}, \ldots, b, 0\} \rightarrow \{0, a, \ldots, s'_l, s_{l+1}, \ldots, b, 0\}$
3: $\quad T_i \leftarrow \text{swap}(s'_l, s_l, T_{i'})$ ..... $\#\ T_{i'} = \{0, c, \ldots, s'_l, s'_{l+1}, \ldots, d, 0\} \rightarrow \{0, c, \ldots, s_l, s'_{l+1}, \ldots, d, 0\}$
4: **end for**
5: $\ell_i \leftarrow \text{regenerate}(T_i)$
6: $\ell_{i'} \leftarrow \text{regenerate}(T_{i'})$

riportato un esempio di k`swap` con $k = 2$, ove sono scambiate tra tour le coppie di servizi $(6, 7)$

e $(2, 3)$. La k`swap` permette di evitare il seguente comportamento. Se per un paziente vi sono più servizi, come $\{s_1, s_2, s_3\}$, che hanno un giorno h comune, ossia $g^h_{s_1} = g^h_{s_2} = g^h_{s_3} = 1$, allora è possibile che i parametri di frequenza f_s e intermezzo e_s permettano la presenza di tutti i servizi in uno tour. Se partiamo da una soluzione ove un tour contiene questa sequenza consecutiva di servizi a distanza nulla , l'operazione `swap` presenta due aspetti negativi. Da un lato potrebbe restituire una soluzione con peggiori valori della funzione obiettivo, rendendo inutile la mossa di `swap`. Dall'altro lato l'operazione `swap` potrebbe trovare un ottimo locale, ove i servizi a distanza nulla sono riposti in tour diversi di infermiere diverse; questa seconda situazione, sebbene si sia ottenuto un ottimo locale, può distruggere una buona soluzione (ossia, che raggruppa i servizi di un paziente) ritardando la comparsa di una possibile migliore soluzione ove i servizi sono nuovamente nello stesso tour. A seguire la descrizione formale del comportamento dell'operatore

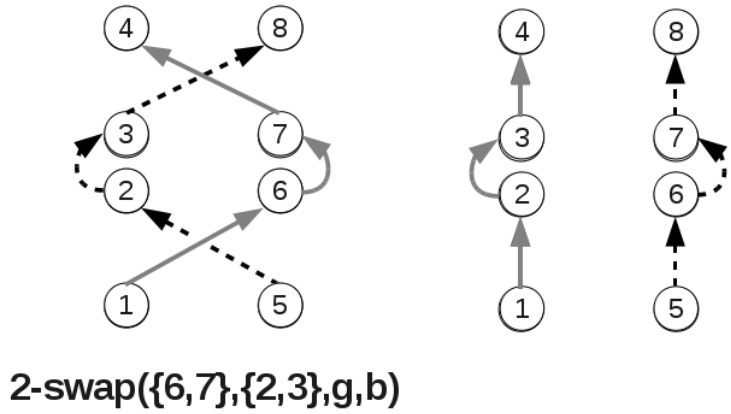


Figura 7.6: Il tour grigio è denotato con g, il tour nero con b e i servizi con i numeri. L'immagini di sinistra mostra la situazione nella quale l'operazione k`swap`$(\{6, 7, \}, \{2, 3\}, g, b)$ *produce la soluzione graficata nell'immagine di destra.*

k`swap`, e una variante ottenuta fissando il giorno.

K1: generica

Impostiamo $\delta \leftarrow 0$. Spostiamo dal tour T_i al tour $T_{i'}$ una sequenza di servizi consecutivi $\overline{s} = \{s_1, \ldots, s_k\}$, tali che la distanza tra ogni coppia di servizi (s_l, s_{l+1}), con $l = 1, \ldots, k-1$, è $d_{s_l, s_{l+1}} \leq \delta$. Analogamente, spostiamo dal tour $T_{i'}$ al tour T_i una sequenza di servizi consecutivi $\overline{s'} = \{s'_1, \ldots, s'_k\}$, tali che la distanza tra ogni coppia di servizi (s'_l, s'_{l+1}), con $l = 1, \ldots, k-1$, è $d_{s'_l, s'_{l+1}} \leq \delta$. Quando una sequenza non è trovata con δ, allora incrementiamo $\delta \leftarrow \delta + \Delta$ reiterando la ricerca fintanto che $\delta \leq \delta_{max}$.

K2: nello stesso giorno

Questo operatore è simile al precedente, dove ogni tour selezionato ha assegnato il giorno h, ossia $T \in \mathcal{T}^h$. Come per le mosse M3 e S3, il giorno h è scelto mediante un generatore di numeri pseudocasuali. Il giorno fissato, permette di risparmiare il tempo di compitazione destinato alla verifica di ammissibilità dei vincoli sulla frequenza f_s e l'intermezzo e_s.

7.4 group

Vediamo in figura 7.1, una delle soluzioni iniziali, è possibile osservare che in uno stesso giorno della settimana esistono tour il cui tempo totale di viaggio è nettamente sotto il valore L richiesto. L'idea di questo operatore è di ridistribuire i servizi in modo da avere un tour in meno, o con distanza totale ℓ nulla.

Formalmente si traduce nel fatto che in un insieme di tour $\mathcal{T}' \subset \mathcal{T}$ la somma dei tempi residui $L-\ell_i$ è maggiore del tempo limite L. L'insieme Ξ di insiemi di tour con le caratteristiche appena descritte è così definito:

$$\Xi \leftarrow \{\mathcal{T}' \subset \mathcal{T} \mid |\mathcal{T}'| \leq \texttt{maxQuantiTour}, \sum_{T_i \in \mathcal{T}'} (L - \ell_i)) \geq L\}\,. \tag{7.8}$$

Restringendo l'insieme $\mathcal{T}^h$ dei tour ai tour ai quali è stato assegnato il giorno h abbiamo:

$$\Xi^h \leftarrow \{\mathcal{T}' \subset \mathcal{T}^h \mid |\mathcal{T}'| \leq \texttt{maxQuantiTour}, \sum_{T_i \in \mathcal{T}'} (L - \ell_i)) \geq L\}\,. \tag{7.9}$$

Ora possiamo descrivere l'operatore `group` dell'algoritmo 7.4: preso un insieme di alcuni tour $\mathcal{T}'$, `group` ne ricava l'insieme dei servizi S' (linea 1). `group` cerca di verificare se è possibile avere una soluzione del DCVRP con un numero minore di tour, $|\mathcal{T}'|-1$ (linea 2). Se il problema del DCVRP ammette soluzione allora sostituisce i tour di $\mathcal{T}'$ con i tour della soluzione ottenuta $\mathcal{T}''$ (linee 3–8).

algoritmo 7.4 `group`

Require: $\mathcal{T}' \in \Xi$, oppure $\mathcal{T}' \in \Xi_h$
1: $S' \leftarrow \{s \in S \mid \exists T_i \in \mathcal{T}' : s \in T_i\}$
2: $\mathcal{T}'' \leftarrow DCVRP(V(S'), |S'| - 1, L)$
3: **if** σ is feasible **then**
4: **for all** $T_i \in \mathcal{T}'$ **do**
5: $\mathcal{T} \leftarrow \mathcal{T} \setminus T_i$
6: **end for**
7: $\mathcal{T} \leftarrow \mathcal{T} \cup \mathcal{T}''$
8: **end if**

Fissato un giorno h, quando i tour sono in $\mathcal{T}^h$ denotiamo l'insieme con Ξ^h. In questo modo possiamo ridurre la ricerca del tour, ed evitare la verifica dell'ammissibilità dell'assegnazione dei servizio, come argomentato per le mosse M3, S3 e K2.

Per selezionare un insieme di tour in Ξ, o in Ξ^h, si può procedere ordinando l'insieme di tour $\mathcal{T}$, o $\mathcal{T}^h$, con la funzione d'ordinamento $T_i \leq T_{i'} \Leftrightarrow \ell_i \leq \ell_{i'}$; infine si prendono i primi `maxQuantiTour` tour. La complessità computazionale è data dall'ordinamento, cioè $O(n \log n)$, con $n = |\mathcal{T}|$, o $n = |\mathcal{T}^h|$. Pertanto la complessità computazionale della mossa è dominata dai tempi di risoluzione del DCVRP.

7.5 merge

L'operatore `merge` prende due tour e se è possibile li unisce insieme, vedi algoritmo 7.5.

algoritmo 7.5 `merge`$(T_i, T_{i'})$

1: $T_i \leftarrow T_i \cup T_{i'}$ ..... # $T_i = \{0, a, \ldots, b, 0\} \rightarrow \{0, a, \ldots, b, \overbrace{c, \ldots, d}^{T_{i'}}, 0\}$
2: $w_i \leftarrow$ regenerate(T_i)

Nel contesto dei problemi di routing, ad esempio il CVRP, una sequenza di operazioni `merge` è applicata della nota euristica 'basata sui risparmi' proposta da Clarke e Wright (1964). Tale euristica sarà presentata ed estesa per ottenere una soluzione dell'HHCP nel capitolo 9.

Quando si ripete sistematicamente una sequenza di merge possiamo memorizzare l'esito dell'operazione `merge`(i, i'). Accedendo allo storico dell'esito delle operazioni `merge`(i, i') è possibile non ripetere quelle che in passato non hanno dato esito positivo, evitando la verifica di ammissibilità che vedremo nel prossimo capitolo.

Tuttavia, se la sequenza di `merge` è interrotta da una mossa, ad esempio una `move`, tutto viene messo in discussione; poiché potrebbe ampliarsi l'insieme ammissibile di giorni g_i per un tour T_i e rendere ammissibile l'unione della coppia di tour (i, i') che non lo era, per esempio il T_i dopo un operazione `swap` passa da un solo giorno ammissibile $q_i^1 = 1$ a tre giorni ammissibili $q_i^1 = q_i^4 = q_i^6 = 1$ ammettendo l'unione con $T_{i'}$ poiché era disponibile nel solo giorno $q_{i'}^4 = 1$.

7.6 Conclusioni

In questo capitolo abbiamo presentato diversi operatori. Abbiamo visto che l'applicazione di operatori, quale la `move`, ha una complessità computazionale $O(1)$. Sebbene l'applicazione di un operatore sia poco costosa, non è possibile svolgere un'esplorazione esaustiva dell'intorno che descrivono. Inoltre, l'applicazione casuale degli operatori può distruggere una buona soluzione: occorre quindi selezionare a priori su quali elementi (tour, servizio) applicare una mossa.

La selezione di alcuni elementi è motivata dall'auspicio di avere un miglioramento di alcuni degli aspetti del problema, ad esempio azzerare i tempi di lavoro straordinario. Per ogni mossa abbiamo descritto gli operatori, gli aspetti del problema che ci si aspetta siano presi in considerazione ed il costo computazionale della ricerca degli elementi sui quali applicare l'operatore.

In tabella 7.1 è riportata una sintesi delle caratteristiche principali della mosse. Le prime tre colonne colgono le tre componenti principali della funzione obiettivo: il carico di lavoro medio giornaliero u, la fidelizzazione paziente-infermiera l ed il tempo di lavoro straordinario w. Nel caso delle mosse M3 e M5, si ricorda che non è colto tutto l'aspetto del carico di lavoro u, ma solo quello relativo al giorno h e all'infermiera j. Il simbolo '✓' nelle colonne u , l e w indica che ci si attende un miglioramento per quella componente. Il simbolo '✓' nelle colonne $\mathcal{T}$ e S sta ad indicare che la mossa utilizza un predeterminato criterio per scegliere i tour per $\mathcal{T}$ o i servizi per S; in caso contrario l'elemento è scelto in modo casuale. La colonna '*costo ricerca*'

riporta la complessità computazionale del costo di cercare quegli elementi (tour, servizio) sui quali la mossa deve operare. Invece, la colonna '*costo mossa*' riporta il costo computazionale dell'operare una mossa.

Come vedremo nel prossimo capitolo, la verifica di ammissibilità è particolarmente costosa ed uno dei punti critici è la procedura di verifica della sola parte di pianificazione dei servizi (frequenza, intermezzo, *etc.*). Se la mossa permette di evitare tale verifica, nell'ultima colonna segnaliamo questo fatto con il simbolo '✓'.

nome	u	l	w	$\mathcal{T}$	S	costo ricerca	costo mossa	evitare check scheduling
M1						$O(1)$	$O(1)$	
M2			✓	✓		$O(\|\mathcal{T}\|^2)$	$O(1)$	
M3	(u^h)		✓	✓		$O(\|\mathcal{T}^h\|^2)$	$O(1)$	✓
M4	✓		✓	✓		$O(\|\mathcal{T}\|^2)$	$O(1)$	
M5	(u^j)		✓	✓		$O(\|\mathcal{T}\|^2)$	$O(1)$	
M6		✓		✓	✓	$O(\|S\|^2)$	$O(1)$	
S1						$O(1)$	$O(1)$	
S2	✓	✓		✓	✓	$O(\|S\|^2)$	$O(1)$	
S3	✓	✓		✓	✓	$O(\|S^h\|^2)$	$O(1)$	✓
K1	✓	✓		✓	✓	—	$O(1)$	
K2	✓	✓		✓	✓	—	$O(1)$	✓
`merge`						$O(1)$	$O(1)$	
`group`				✓		$O(\|\mathcal{T}\|^2)$	NP-Hard	
`group`^{h}				✓		$O(\|\mathcal{T}^h\|^2)$	NP-Hard	✓

Tabella 7.1: Sunto delle mosse.

Capitolo 8

Verifiche di ammissibilità

Una sequenza di nodi viene detta catena (*chain*). L'operazione '*chain relocation*' nota per il TSP (Babin et al., 2007), svolge una ricollocazione di una catena di nodi nello stesso tour. In questo caso l'operazione trasforma una soluzione in un'altra soluzione ammissibile, senza richiedere una verifica. Per le operazioni presentate nel capitolo precedente l'ammissibilità deve essere verificata, ed è molto costosa. Una codifica naïf dell'algoritmo porta a una bassa efficienza computazionale, ovvero non possiamo identificare buone soluzioni per istanze tratte da casi reali in dieci ore di calcolo. Per questo sono stati messi a punto una serie di accorgimenti allo scopo di superare i colli di bottiglia sul calcolo, rendendo possibile l'applicazione di un'euristica ad un problema così ricco. In questo capitolo presentiamo le verifiche preventive e posteriori all'applicazione di un operatore, e gli accorgimenti per mettere in essere le verifiche più costose solo se necessarie. Nella parte dei risultati computazionali sarà possibile osservare l'importanza in termini di tempo di calcolo dell'uso di alcune di queste tecniche.

8.1 Introduzione

Svolgendo una verifica sommaria della nuova soluzione, potrebbero essere scartate mosse migliorative. Come esempio assumiamo di svolgere una `move`, ovvero di accodare un servizio ad un altro tour $T_{i'}$. Se la verifica del tempo di percorrenza $\ell_{i'}$ del tour $T_{i'}$ eccede di un tempo irrisorio il tempo limite L allora dovremmo scartarla, oppure potremmo ottimizzare il tour $T_{i'}$ nella speranza di rendere la nuova soluzione ammissibile. Analogamente occorrerà verificare la parte relativa all'assegnazione dei servizi. La soluzione proposta ibridizza codice *ad hoc* e tecniche di programmazione matematica. Alcune verifiche sono svolte prima di un operatore per evitare di svolgere l'operazione e di svolgere le verifiche di ammissibilità finali.

Si indichi con `op` una generalizzazione degli operatori presentati. `op` lavora su un'insieme di servizi $S_A = \{s_1, \ldots, s_m\}$ presenti nel tour T_i, e un insieme di servizi $S_B = \{s'_1, \ldots, s'_m\}$ presenti nel tour $T_{i'}$. Definiamo $\mathcal{S}_{\texttt{op}} = \{S_A, S_B\}$ e $\mathcal{T}_{\texttt{op}} = \{T_i, T_{i'}\}$. L'insieme di servizi S_B può essere vuoto, *i.e.* nell'operazione `move` $S_A = \{s\}$ e $S_B = \emptyset$. Nell'algoritmo 8.1 forniamo un modello per ogni operatore. In linea 1, verifichiamo se è possibile applicare l'operatore, per esempio verifichiamo se muovere un servizio s in un tour T_i è possibile, ossia se esista h tale

algoritmo 8.1 `op` template

Require: $S_{\texttt{op}}, \mathcal{T}_{\texttt{op}}, \rho \in \{\ell_{max}, u, L\}$, `opt`, `always optimize`, $\ell_{max} = L + w$

```
1: if is possible to operate op then
2:    Backup ← save(State)
3:    ℓ_i ← op
4:    if ¬TSP_part(T_op, opt) then
5:       return false
6:    end if
7:    check ← false
8:    if h_i ≠ h_i' then
9:       for all s ∈ S_op do
10:         check ← check ∨ (e_s > e(s))
11:      end for
12:   end if
13:   return scheduling_part(check, T_op, u_old)
14: else
15:   return false
16: end if
```

$q_i^h = g_s^h = 1$. Prima di ogni operazione salviamo la soluzione (line 2). Dopo ogni operazione `op`, verifichiamo se la nuova soluzione è ammissibile. La verifica è suddivisa in una parte inerente all'instradamento (linee 4–6) e una parte inerente all'assegnazione dei servizi (linee 7–13). In dettaglio a linea 10 ricaviamo se possiamo evitare l'onerosa verifica dell'assegnazione dei servizi, nel caso risulti positiva la violazione del minimo tempo e_s del servizio s con il nuovo minimo tempo d'attesa $e(s)$:

$$\begin{aligned} e(s) \;=\; \min\{ & \overbrace{0}^{e_s^{ii}} \mid \exists\, T_i \in \mathcal{T} : |\{s \in T_i\}| > 1\} \\ & \cup \{\overbrace{|h_{i'} - h_i|}^{e_s^{ii'}} \mid T_i, T_{i'} \in \mathcal{T} : i \neq i' \\ & \wedge s \in T_i \cap T_{i'}\}\,; \end{aligned} \tag{8.1}$$

ottenuto dopo l'applicazione dell'operazione `op`. Il costo computazionale della verifica $e(s)$ è polinomiale. Vedremo che la procedura `scheduling_part` (linea 13) è NP-Hard. Se eliminiamo le righe 7–12 ponendo $check \leftarrow true$, questo significa che `scheduling_part` svolgerà la verifica sempre; la semantica dell'operazione `op` rimane equivalente, poiché `scheduling_part` si occupa anche di verificare quanto verificato da $e(s)$. In tempo costante possiamo evitare di svolgere la verifica `scheduling_part`, infatti se il giorno h_i o $h_{i'}$ precedentemente assegnato era ammissibile continuerà ad esserlo poiché, $h_i = h_{i'}$. Se per ogni servizio s la verifica $e(s)$ attesta che l'intermezzo è ancora ammissibile, allora in tempo polinomiale abbiamo evitato la verifica `scheduling_part`.

In precedenza abbiamo visto che le mosse possono avere un costo computazionale che possiamo considerare costante, ad esempio la `move`. Successivamente abbiamo visto che il costo computazionale può aumentare se desideriamo selezionare tour e servizi in modo da sfruttare alcune caratteristiche di alcune soluzioni, come le mosse M2–M6. Da questo esame sommario dell'algoritmo 8.1 aggiungiamo ai costi computazionali della selezione dei tour e dei servizi, i costi computazionali del controllo che permette di sapere se si può svolgere una mossa (linea 1) e i costi di verifica dell'ammissibilità (linea 7). Inoltre, al fine di aggiornare il valore della funzione obiettivo, ad ogni passo è necessario calcolare i valori di u, l e w, rispettivamente con costo computazionale $O(|N| \cdot |H|)$, $O(|N| \cdot |P|)$ e $O(|N| \cdot |H|)$. La somma totale di tutti questi costi diminuisce il numero di soluzioni che un'euristica può esplorare in un certo tempo. Ai costi si aggiunge un '*trade-off*', argomentato all'inizio della sezione, dovuto al fatto che non si vorrebbero scartare mosse migliorative per non aver ottimizzato ed allo stesso tempo ottimizzare comporta l'esplorazione di un minor numero di soluzioni. Questo secondo aspetto, unito ai costi, rende ardua la messa a punto di un'euristica performante.

Una caratteristica che permette di migliorare le prestazioni della nostra variante di ALNS è data dall'uscire prima della linea 7 dell'algoritmo 8.1 se $f(\sigma) \geq f(\sigma^*)$. In altre parole combiniamo la verifica riguardo alla migliore soluzione (line 10 dell'algoritmo 6.5) con la verifica di ammissibilità (line 6 dell'algoritmo 6.5).

Introduciamo nelle successive sezioni gli aspetti di verifica preliminare di disponibilità, e gli aspetti delle verifiche conclusiva della parte d'instradamento (routing) e della parte di assegnazione di servizi (scheduling).

8.2 Verifica di disponibilità

Poiché questo controllo varia da operatore a operatore, abbiamo descritto genericamente nell'algoritmo 8.1 a linea 1 che l'algoritmo prosegue se è possibile applicare l'opeartore `op`. Infatti, il controllo per l'operatore `merge` differisce dal controllo comune agli operatori `move`, `swap` e k`swap`. Nel caso dell'operatore `merge` si può attuare se ogni servizio $s \in T_{min} \cup T_{max}$ ha un giorno h di disponibilità g_s^h comune, ossia $\bigvee_{h \in H} q_{min}^h \wedge q_{max}^h = 1$; in caso contrario la fusione di T_{min} e T_{max} darebbe un tour T_i tale che $\bigvee_{h \in H} q_{max}^h = 0$, che è inammissibile. Inoltre non si desidera che un servizio che non ammette ripetizioni nello stesso giorno ($e_s = 0$) possa comparire più di una volta nell'unione dei servizi dei due tour, vedi algoritmo 8.2.

algoritmo 8.2 se possible eseguire la `merge`

1: **if** $\bigvee_{h \in H} q_{min}^h \wedge q_{max}^h = 1 \wedge \bigwedge_{s \in T_{max} \cup T_{min}} (e_s = 0 \veebar |\{s \in T_{max} \cup T_{min}\}| = 1)$ **then**
2: **return** `true`
3: **else**
4: **return** `false`
5: **end if**

Per gli operatori `move`, `swap` e k`swap` e successive verifiche di ammissibilità possono essere evitati le due seguenti verifiche:

1. Verifichiamo se è possibile inserire s_l in $T_{i'}$ e s'_l in T_i. Prima di inserire un servizio s'_l in T_i e un altro servizio s_l in $T_{i'}$, verifichiamo che i giorni di disponibilità siano compatibili:

$$\bigwedge_{s'_l \in S_B} \left(\bigvee_{h \in H} g^h_{s'_l} \wedge q^h_i \right) \wedge \bigwedge_{s_l \in S_A} \left(\bigvee_{h \in H} g^h_{s_l} \wedge q^h_{i'} \right) . \tag{8.2}$$

2. È sempre possibile muovere s_l in $T_{i'}$ quando $e_{s_l} = 0$, poiché il servizio s_l può essere ripetuto più volte nello stesso giorno. Quando $e_{s_l} \neq 0$, le ripetizioni nello stesso giorno non sono permesse. Questo significa che dobbiamo verificare se $s_l \notin T_{i'}$. Simmetricamente, è sempre possibile muovere s'_l in T_i quando $e_{s'_l} = 0$; quando $e_{s'_l} \neq 0$, le ripetizioni nello stesso giorno non sono permesse, ovvero dobbiamo verificare se $s'_l \notin T_i$:

$$\bigwedge_{s_l \in S_A} (s_l \notin T_{i'} \veebar e_{s_l} = 0) \wedge \bigwedge_{s'_l \in S_B} (s'_l \notin T_i \veebar e_{s'_l} = 0) . \tag{8.3}$$

Con le verifiche preliminari appena descritte possiamo risparmiare il tempo di calcolo richiesto dall'applicazione della mossa e dei controlli descritti nelle seguenti sezioni.

8.3 Verifica della parte di routing

Verificare che un ciclo sia hamiltoniano ha un costo computazionale lineare; con pari complessità computazionale si verifica che ℓ_i, il tempo di lavoro del turno i, non superi il massimo tempo di un turno L. Se a seguito di una mossa la sequenza di servizi dei rispettivi tour coinvolti è sempre ottimizzata, allora siamo certi che la verifica di massima durata di un turno è una sentenza definitiva sull'ammissibilità della parte d'instradamento. Tuttavia, ottimizzare la sequenza di servizi richiede la soluzione del problema del commesso viaggiatore (TSP), il quale è NP-Hard . Sebbene nella pratica il problema è risolto in tempi che non superano un secondo, preso un algoritmo che ottimizza sempre il tour dopo una mossa e un altro che mai ottimizza, osserviamo che in un lasso di tempo il primo algoritmo esplora un numero di intorni minore rispetto al secondo algoritmo. Abbiamo anche osservato che la Clarke & Wright è passata da meno di una decina di minuti senza ottimizzazione a più di un'ora ottimizzando sempre dopo ogni `merge`.

Ora descriviamo come la scelta estremistica di ottimizzare sempre oppure di ottimizzare mai, possa essere mitigata con la scelta di ottimizzare all'occorrenza e pseudo-casualmente. A seguito di una mossa `move` il servizio è accodato al nuovo tour i', assumiamo che la verifica informi che il massimo tempo di percorrenza è violato, cioè $\ell_{i'} > L$. In questo caso possiamo scartare la mossa, oppure ottimizzare la sequenza dei servizi risolvendo un TSP. Nella pratica abbiamo osservato che ci sono altre circostanze ove si potrebbe trarre beneficio dall'ottimizzazione del tour. Nella fase iniziale, per molti tour $\ell_i > L$; in attesa del raggiungimento del limite può essere opportuno selezionare un altro indice per scegliere se ottimizzare, come ℓ_{max}, o u. Questa scelta può essere altrettanto positiva nel caso il tour sia ammissibile, come ora mostriamo. Assumiamo che la mossa abbia lasciato i tour coinvolti nella situazione di ammissibilità, cioè $\ell_{i'} \leq L$. In questo caso è possibile che il tour contenga alcuni servizi che hanno distanza nulla tra loro,

ma non sono visitati consecutivamente; ed è molto probabile che ottimizzando la sequenza di visita risolvendo un TSP i servizi vicini possano essere consecutivi. In questo modo è possibile predisporre la soluzione al caso richiesto nella mossa k`swap`; inoltre può abbassare il costo del tour, di conseguenza del carico medio di lavoro della relativa infermiera. Pertanto, può essere utile svolgere un ottimizzazione del tour anche se $\ell_{i'} \leq L$. In un campione di esecuzioni, è sembrato che la convergenza verso l'ammissibilità di un algoritmo sia più rapida con la scelta pseudo casuale tra la violazione dei vincoli $\ell_i > \ell_{max}$, $\ell_i > u$ e $\ell_i > L$, rispetto ad scegliere totalmente a caso se ottimizzare. Pertanto, associamo alla variabile ρ la scelta della violazione che richiede se ottimizzare il tour, e al parametro `always optimize` se ottimizzare sempre o come descritto (si veda linee 2–8 dell'algoritmo 8.3).

algoritmo 8.3 `TSP_part`

Require: $\mathcal{T}_{\texttt{op}}, \rho \in \{\ell_{max}, u, L\}$, `opt`, `always optimize`, $\ell_{max} = L + w$

1: **for all** $T_i \in \mathcal{T}_{\texttt{op}} : opt \wedge (\ell_i > L \vee \ell_i > u)$ **do**
2: **if** `always optimize` **then**
3: $\ell_i \leftarrow$ `TSP`(T_i)
4: **else**
5: **if** `opt` $\wedge\, \ell_i > \rho$ **then**
6: $\ell_i \leftarrow$ `TSP`(T_i)
7: **end if**
8: **end if**
9: **if** $\ell_{max} > L \wedge |\ell_{max} - u| < L \cdot \underline{\lambda}$ **then**
10: **if** $\ell_i > \ell_{max}$ **then**
11: `restore`$(Backup)$
12: **return** $false$
13: **end if**
14: **else**
15: **if** $\ell_i > \rho$ **then**
16: `restore`$(Backup)$
17: **return** $false$
18: **end if**
19: **end if**
20: **return** $true$
21: **end for**

Successivamente alla scelta se ottimizzare il tour, c'è la scelta di convalidare la soluzione come ammissibile. Generalmente se esiste un tour di durata massima ℓ_{max} che supera il tempo massimo L, e la differenza con il carico di lavoro medio u è ampia quanto $L \cdot \underline{\lambda}$, cioè $|\ell_{max} - u| < L \cdot \underline{\lambda}$, allora siamo lontani da una buona soluzione (si veda linea 9 dell'algoritmo 8.3). In questa circostanza se la lunghezza del tour T_i ove ha operato la mossa supera ℓ_{max}, allora la mossa allontana dall'abbattimento dei costi di straordinario delle infermiere e deve essere scartata (si veda linea 9 dell'algoritmo 8.3). Altrimenti se la lunghezza del tour T_i ove ha operato la

mossa non supera ℓ_{max}, non consideriamo ammissible il tour che viola la condizione $\ell_i > \rho$ (si veda linea 15 dell'algoritmo 8.3). Questa scelta è utile per premiare miglioramenti, ovvero per aiutare la convergenza, senza specializzare ulteriormente la scelta di ρ in relazione all'operatore.

Per le mosse `move`, `swap` e k`swap` l'insieme $\mathcal{T}_{\texttt{op}}$ è costituito di soli due tour T_i e $T_{i'}$; abbiamo ritenuto opportuno sostituire l'algoritmo 8.3 con l'algoritmo 8.4.

algoritmo 8.4 `TSP_part` (specializzazione)

1: **for all** $T_i \in \mathcal{T}_{\texttt{op}} : opt \wedge (\ell_i > L \vee \ell_i > u)$ **do**
2: **if** `always optimize` **then**
3: $\ell_i \leftarrow \texttt{TSP}(T_i)$
4: **else**
5: **if** `opt` $\wedge\ \ell_i > \rho$ **then**
6: $\ell_i \leftarrow \texttt{TSP}(T_i)$
7: **end if**
8: **end if**
9: **end for**
10: **if** $\ell_{max} > L \wedge |\ell_{max} - u| < L \cdot \underline{\lambda}$ **then**
11: **if** $\ell_{i'} > \rho \vee \ell_i > \ell_{max}$ **then**
12: `restore`$(Backup)$
13: **return** $false$
14: **end if**
15: **else**
16: **if** $\ell_{i'} > \rho \vee \ell_i > \rho$ **then**
17: `restore`$(Backup)$
18: **return** $false$
19: **end if**
20: **end if**
21: **return** $true$

Il motivo di questa variante è dato dalla presenza di mosse `move` M2, M3 e M5, le quali tendono scegliere come tour T_i con $\ell_i = \ell_{max}$ e $T_{i'}$ con $\ell_{i'} = \ell_{min}$. Un esempio dell'utlità di questo meccanismo è quando si desiderava scegliere una coppia di tour (i, i') tali da avere una massima differenza $|\ell_i - \ell_{i'}|$ in un giorno fissato h; otteniamo che $\ell_i = \ell_{max}$ ma abbiamo $\ell_{i'} > L$ anche dopo la mossa. Pertanto scartare $\ell_{i'} > \rho$ equivale a controllare che la mossa `move` non tenda ad essere peggiorativa; poiché essendo $\ell_{max} > L$ è fortemente richiesto annulare i costi di straordinario w.

8.4 Verifica della parte di scheduling

Nel caso in cui il giorno assegnato ai tour sia lo stesso ($h_i = h_{i'}$, linea 7 nell'algoritmo 8.1) non occorre verificare la frequenza, l'intermezzo e la disponibilità del servizio ad essere svolto

il giorno h; tale verifica la chiameremo *verifica della parte di scheduling*, o *verifica dell'assegnamento*. La verifica dell'assegnamento è evitata anche quando tutti i servizi mantengono il rispetto dell'intermezzo e_s, in tal caso poniamo $check \leftarrow false$; altrimenti $check \leftarrow true$. In questo secondo caso ci potrebbe essere un diverso assegnamento dei tour alle infermiere, tale da giustapporre i giorni affinché la soluzione sia ammissibile (vedi algoritmo 8.5). Dobbiamo

algoritmo 8.5 `scheduling_part`

Require: *check*
1: **if** *check* **then**
2: **if** ¬ `scheduling_is_feasible` **then**
3: `restore`(*Backup*)
4: **return** *false*
5: **end if**
6: **end if**
7: **return** *true*

algoritmo 8.6 `scheduling_is_feasible`:

Require: $Data : \{S, \mathcal{T}, H, N, \{\Delta_s : s \in S\}, \{f_s : s \in S\}\}$
1: $\sigma \leftarrow$ `solve`(`min`$\{\alpha_2 w \mid (8.8), (8.11), (8.12), (8.4), (8.7), (8.10)\}$)
2: **if** σ is feasible **then**
3: **return** true
4: **else**
5: **return** false
6: **end if**

quindi risolvere un problema combinatorio che può essere modellato come un problema di programmazione matematica (Martello e Toth, 1990; Kellerer et al., 2004). Definiamo la variabile booleana χ_i^{jh} che assume valore 1 se il tour i è assegnato all'infermiera j il giorno h.

$$\sum_{i:s\in T_i} \sum_{h\in H} \sum_{j\in N_h} \chi_i^{jh} = f_s \quad \forall s \in S : e_s \neq 0 \tag{8.4}$$

$$\sum_{i:s\in T_i} \sum_{h\in H} \sum_{j\in N_h} \chi_i^{jh} \leq f_s \quad \forall s \in S : e_s = 0 \tag{8.5}$$

$$\sum_{h\in H} \sum_{j\in N_h} \chi_i^{jh} = 1 \quad \forall i \in \mathcal{T} \tag{8.6}$$

$$\sum_{k=h}^{h+e_s-1} \sum_{i:s\in T_i \wedge q_i^k=1} \sum_{j\in N_k} \chi_i^{jk} \leq 1 \quad \forall s \in S_+,\ \forall h \in \{1, \ldots, |H| - e_s\} \tag{8.7}$$

$$\sum_{i\in\mathcal{T}} \ell_i \chi_i^{jh} - L \leq w \quad \forall h \in H,\ \forall j \in N_h \tag{8.8}$$

$$\chi_i^{jh} \leq q_i^h \quad \forall i \in \{1, \ldots, |\mathcal{T}|\},\ \forall j \in N,\ \forall h \in H \tag{8.9}$$

$$\chi_i^{jh} \in \{0, 1\} \quad \forall i \in \{1, \ldots, |\mathcal{T}|\},\ \forall j \in N,\ \forall h \in H \tag{8.10}$$

$$0 \leq w \leq \underline{W}_{max} \tag{8.11}$$

$$w \in \mathbb{R}^+ \tag{8.12}$$

I vincoli (8.4) garantiscono che un servizio s sia servito f_s volte. I vincoli (8.5) gestiscono il caso particolare di servizi nello stesso giorno. Viene garantito che ogni turno sia assegnato ad un'infermiera j in un giorno h nei vincoli (8.6). Il minimo numero di giorni di attesa prima di una ripetizione è gestito dai vincoli (8.7). Potrebbe essere possibile, più frequentemente nelle fasi iniziali, che una soluzione presenti vari tour di pochissimi nodi e con stesso insieme di giorni ammissibili. I vincoli 8.8 consentono di unire questi tour; il che corrisponde al permettere all'infermiera di tornare più volte al deposito, o anche virtualizzare un'operazione `merge`. Desideriamo minimizzare la funzione obiettivo $\alpha_2 w$, algoritmo 8.6; il risolutore di programmazione matematica è impostato per uscire appena trova una soluzione ammissibile.

8.5 Conclusioni

In questo capitolo abbiamo presentato una serie di accorgimenti che hanno premesso di superare i colli di bottiglia nel calcolo della verifica di ammissibilità, rendendo possibile l'applicazione di un'euristica. Il modello (8.4)–(8.12) usato per la verifica della parte di scheduling sarà usato nel prossimo capitolo, nel quale mostreremo come il lavoro di generalizzazione degli operatori permette di estendere euristiche note, come quella proposta da Clarke e Wright (1964).

Capitolo 9

Clarke and Wright

In questo capitolo trattiamo un adattamento della nota euristica costruttiva conosciuta come *algoritmo basato sui risparmi* (*saving algorithm*), proposta da Clarke e Wright (1964). Lo scopo è duplice: da un lato si mostra che è possibile usare gli operatori anche in euristiche non neighbourhood-based; dall'altro lato è una potenziale alternativa per ottenere una soluzione iniziale, a quella presentata in sezione 6.4, dalla quale possono partire metodi MILP (ad esempio Column Generation) ed euristici (per esempio la VNS). Nella versione originale dell'algoritmo non c'è una limitazione sul numero dei veicoli e tutti i veicoli hanno la stessa capacità. In Mendoza (2009) la Clarke & Wright è estesa ed applicata al DCVRP, senza ottimizzare i tour dopo la fusione. Nella prossima sezione introduciamo la versione originale di Clarke e Wright (1964), e successivamente la estendiamo all'HHCP, la quale gestisce la ripetizione di servizi, si considera il caso ove i tour sono ottimizzati dopo la fusione, e si verifica che la fusione dia luogo ad una soluzione ammissibile.

9.1 Versione originale dell'algoritmo

Ad ogni passo dell'algoritmo 9.1 proposto da Clarke e Wright (1964), la soluzione è costituita da una famiglia $\mathcal{T} = \{T_1, \ldots, T_m\}$ di tour che rispettano una serie di proprietà. Per i problemi di base, come il CVRP, la verifica richiede tempo polinomiale, *e.g.* che il tour non superi la capacità. Nel prossimo capitolo, vedremo che per problemi più grandi, come l'HHCP, la verifica può non essere polinomiale.

Ad ogni passo vengono scelti tour da fondere insieme. I tour sono scelti usando un *criterio di risparmio* (*saving*)

$$\varsigma_{ss'} = c_{0s} + c_{s'0} - c_{ss'} > 0 \qquad \forall s, s' \in S\,, \tag{9.1}$$

il quale permette di abbassare il valore della funzione obiettivo, quando essa dipende dai costi $c_{ss'}$. È possibile scegliere la coppia di tour per la quale si ha il massimo risparmio $\varsigma_{ss'}$, questo richiede un ordinamento di tutti i valori di $\varsigma_{ss'}$.

La fusione di due tour T_i e $T_{i'}$, denotata con $\texttt{merge}(T_i, T_{i'})$, restituisce un nuovo tour T_k che sarà così formato: parte dalla centrale operativa 0 svolgendo i servizi del tour T_i e, al posto

di tornare alla centrale operativa, si prosegue con i servizi del tour $T_{i'}$, ed infine si torna alla centrale operativa.

La fusione ha luogo se sono verificate le proprietà di ammissibilità del problema e se è verificato il criterio di risparmio (9.1). Dopo la fusione, la soluzione sarà data da $\mathcal{T} = T_k \cup \mathcal{T} \setminus \{T_i, T_{i'}\}$. Il processo termina quando non è più possibile fondere i tour.

algoritmo 9.1 Clarke and Wright

1: compute saving cost $\varsigma_{uv} = d_{0u} + d_{v0} - d_{uv}$ for every pair (u, v)
2: **repeat**
3: **if** best saving > 0 **then**
4: `merge`$(T_i, T_{i'})$
5: **end if**
6: **until** best saving ≤ 0

9.2 Adattamento per l'HHCP

Rispetto al CVRP nell'HHCP dobbiamo gestire i seguenti aspetti non gestiti dalla versione originale dell'algoritmo: il vincolo sulla massima lunghezza di un tour; la ripetizione di servizi; l'ottimizzazione del tour dopo la fusione; la verifica dell'ammisibilità della soluzione. Nel lavoro di Mendoza (2009) viene svolto il controllo sulla massima lunghezza del tour dopo la fusione, senza ottimizzare il tour dopo la fusione. Inoltre in Mendoza (2009) il numero di veicoli, ossia infermiere in una settimana, non è fissato; mentre nell'HHCP non sono ammesse soluzioni che richiedano più infermiere.

Nel caso dell'HHCP occorre anche ricordare che un servizio s deve essere presente esattamente in f_s tour. Infatti nelle linee 3–8 dell'algoritmo 9.2 sono generati f_s tour per ogni servizio s.

Per selezionare il miglior criterio di risparmio è possibile usare un insieme ordinato Σ di tutti i criteri di risparmio $\varsigma_{ss'}$. Nei lavori citati non è presente la frequenza del servizio, pertanto una volta fuso il tour i che ha s come ultimo servizio e il tour i' che ha s' come primo servizio, ottenendo un tour ammissibile, è possibile scartare da Σ sia $\varsigma_{ss'}$ che tutti i ς_{sk} e $\varsigma_{ks'}$.

Per l'HHCP non possiamo eliminare $\varsigma_{ss'}$ senza controllare che vi siano altri tour con i servizi s e s' in coda e in testa ad un tour. Occorre quindi contare il numero dei tour che hanno s come servizio iniziale, $|r_{s\bullet}|$, o come servizio finale, $|r_{\bullet s}|$. Gli insiemi $r_{s\bullet}$ e $r_{\bullet s}$ sono definiti nell'algoritmo 9.2 rispettivamente nelle line 9 e 10. Sapendo se esistono tour che hanno un servizio s in testa, possiamo inserire $\varsigma_{ss'}$ nell'insieme $\Sigma_{s\bullet}$ (linea 11). Analogamente, sapendo se esistono tour che hanno un servizio s in coda, possiamo inserire $\varsigma_{s's}$ nell'insieme $\Sigma_{\bullet s}$ (linea 12). Infine possiamo ottenere Σ come l'unione di tutti gli insiemi $\Sigma_{\bullet s}$ e $\Sigma_{\bullet s}$, ed ordinarlo (linee 13 e 14).

Abbiamo detto che tra le proprietà che dobbiamo verificare nell'HHCP c'è il rispetto del vincolo sulla massima lunghezza, pertanto dopo una `merge` può essere che il tour T_k sia di

algoritmo 9.2 Clarke and Wright:

Require: *opt* is true if we optimize after the merge

1: $\varsigma_{ss'} \leftarrow c_{0s} + c_{s'0} - c_{ss'}\ \forall s, s' \in S$ ..... # saving costs
2: $feasible_{ss'} \leftarrow 1\ \forall s, s' \in S$
3: **for all** $s \in S$ **do**
4: **for all** $d \in \{1, \ldots, f_s\}$ **do**
5: $T \leftarrow [0, s, 0]$
6: $\mathcal{T} \leftarrow \mathcal{T} \cup T$
7: **end for**
8: **end for**
9: $r_{s\bullet} \stackrel{\text{def}}{=} \{t \mid T_t \in \mathcal{T}\ T_t = [0, s, \ldots, k, 0]\} \qquad \forall s \in S$
10: $r_{\bullet s'} \stackrel{\text{def}}{=} \{t \mid T_t \in \mathcal{T}\ T_t = [0, l, \ldots, s', 0]\} \qquad \forall s \in S$
11: $\Sigma_{s\bullet} \stackrel{\text{def}}{=} \{\varsigma_{sk} \mid r_{s\bullet} \neq \emptyset,\ k \in S\} \qquad \forall s \in S$
12: $\Sigma_{\bullet s'} \stackrel{\text{def}}{=} \{\varsigma_{ks'} \mid r_{\bullet s'} \neq \emptyset,\ k \in S\} \qquad \forall s \in S$
13: $\Sigma \leftarrow \bigcup_{k \in S} (\Sigma_{\bullet k} \cup \Sigma_{k\bullet})$
14: $\Sigma \leftarrow \text{sort}(\Sigma, \geq)$ ..... # $\Sigma = \{\varsigma_{ss'} \mid s, s' \in S\}$
15: **for all** $\varsigma_{s's} \in \Sigma$ **do**
16: **for all** $(\rho_{s'}, \rho_s) \in r_{\bullet s'} \times r_{s\bullet}$ **do**
17: **if** $feasible_{ss'} = 1$ **then**
18: $good \leftarrow$ `merge`$(\rho_{s'}, \rho_s, opt)$ ..... # $good = true$ if merge is feasible
19: **if** *good* **then**
20: **if** *opt* **then**
21: `goto` linea 9
22: **else**
23: update $r_{s\bullet}$ and $r_{\bullet s'}$
24: update $\Sigma_{s\bullet}$, $\Sigma_{\bullet s'}$, and Σ
25: `goto` linea 15
26: **end if**
27: **else**
28: $feasible_{ss'} \leftarrow 0$
29: **end if**
30: **end if**
31: **end for**
32: **end for**

poco oltre a tale valore. Per ovviare è plausibile ottimizzare il tour, tuttavia c'è differenza computazionale tra ottimizzare tour e non ottimizzarli. Ora vediamo come segue il flusso dell'algoritmo.

Per eliminare svariate ripetizioni, si usa una variabile $feasible_{ss'} \in \{0, 1\}$ che tiene traccia dello storico dei successi: se $feasible_{ss'} = 0$ significa che in una precedente operazione `merge` non ha fuso i tour (s, s'). Prima di intraprendere l'operazione di fusione di due tour (linea 18),

verificando se era inammissibile (linea 17) evitiamo di ripeterla.

Quando due tour sono uniti si presentano due casi: l'unione è seguita da un ottimizzazione del nuovo tour, all'unione non segue un ottimizzazione del nuovo tour. Nel primo caso (linee 20–21), occorre rigenerare gli insiemi $r_{s\bullet}, r_{\bullet s'}$ e i valori di saving $\Sigma_{s\bullet}, \Sigma_{\bullet s'}$, andando a linea 9; questo perché l'ottimizzazione di un tour può spostare un servizio posto al centro di un tour alla fine o all'inizio. Nel secondo caso (linee 22–26), si aggiornano $r_{s\bullet}, r_{\bullet s'}$ e i valori di saving $\Sigma_{s\bullet}, \Sigma_{\bullet s'}, \Sigma$, il quale è meno costoso che rigenerare tutti gli insiemi (come in linee 9–14).

Una volta impostato opportunamente l'upper bound $\underline{W}_{max}$ di w del vincolo (8.11) si ha che un tour è fusione di due tour se la `merge` ha restituito *true*, ovvero se il sistema (8.4)–(8.12). Pertanto alla fine dell'algoritmo abbiamo la garanzia che il sistema (8.4)–(8.12) ha soluzione.

Ora mostriamo come espandere all'HHCP la versione per il CVRP, o DCVRP, gestendo l'assegnazione dell'infermiera e del giorno ad ogni tour. Alla fine dell'esecuzione dell'algoritmo 9.2 abbiamo un insieme $\mathcal{T}$ di tour T_i che hanno distanza totale $\ell \leq L$. Il numero dei tour è maggiore del numero di turni in una settimana, ossia $|\mathcal{T}| \gg |N| \cdot |H|$. Per fornire una soluzione occorre risolvere il problema di programmazione lineare intera (8.4)–(8.12), usato per la verifica della parte di scheduling, la cui soluzione ci permette di sapere che il tour i è assegnato all'infermiera j nel giorno h quando $\chi_i^{jh} = 1$. Infine possiamo avere una pianificazione settimanale ponendo il turno assegnato all'infermiera j nel giorno h come la visita consecutiva dei servizi dei tour assegnategli quel giorno:

$$T^{jh} = \bigcup_{i:\chi_i^{jk}=1} T_i \tag{9.2}$$

Ottimizzando i tour la complessità dell'algoritmo aumenta, sia perché è risolto un TSP che per l'aggiornamento delle strutture dati.

9.3 Conclusioni

Abbiamo mostrato che il lavoro di generalizzazione della verifica di ammissibilità ha permesso di espandere un'euristica nota. Allo stesso modo, dall'insieme di operazioni proposto, altre euristiche o meta euristiche possono essere sviluppate.

Riepilogo della parte III

I capitoli precedenti hanno introdotto ed analizzato le meta euristiche *neighbourhood-based*, gli operatori di trasformazione delle soluzioni, la verifica di ammissibilità di tali soluzioni ed infine l'euristica Clarke & Wright.

Si è potuto così vedere che nell'approccio alla soluzione del problema HHC sono di fondamentale importanza sia la riduzione della regione ammissibile nella ricerca delle soluzioni, che la progettazione di opportuni accorgimenti per ridurre i tempi della verifica dell'ammissibilità delle soluzioni ottenute dalle mosse.

Abbiamo quindi ora a disposizione uno schema completo per risolvere il problema HHC nella sua formulazione più generale.

Nei capitoli successivi la sperimentazione numerica, su una serie di istanze ottenute da dati reali, consentirà di valutare oggettivamente il comportamento computazionale degli algoritmi proposti per la soluzione di HHCP.

Parte IV

Risultati sperimentali

Sommario della parte IV

Quest'ultima parte del lavoro è dedicata alla sperimentazione numerica. I capitoli precedenti hanno descritto e analizzato tutte le componenti con le quali si sono potuti assemblare gli algoritmi che consentono di risolvere l'HHCP.

Per verificare l'oggettiva funzionalità di tali algoritmi sarebbe stato sufficiente utilizzare problemi HHC di piccole dimensioni. Tuttavia, uno degli scopi di questa tesi è mostrare che gli approcci proposti sono efficaci nei casi reali, i quali hanno tipicamente grandi dimensioni.

L'AUSL di Ferrara ha messo a disposizione un insieme di dati storici corrispondente alle richieste di servizi lungo un mese: si è dunque deciso di utilizzare tali dati per generare istanze verosimili delle dimensioni volute.

Il capitolo 10 descrive la struttura e le proprietà dei dati reali ricevuti dall'AUSL di Ferrara, mentre il capitolo 11 riporta l'intero processo di generazione che permette di ottenere istanze di qualunque dimensione a partire dai dati reali.

Come è ben evidente dai capitoli precedenti, gli algoritmi proposti dipendono da numerosi parametri: essi hanno un ruolo significativo per adattare gli algoritmi ad istanze di caratteristiche differenti e per esplorare la qualità delle soluzioni ottenute, oltre ad avere un impatto sui tempi di esecuzione.

La scelta dei valori da attribuire a tali parametri è dunque cruciale sia per la riproducibilità dei risultati, che per la velocità di risoluzione, ma un'indagine esaustiva nello spazio dei parametri è al momento non proponibile. Pertanto, nel capitolo 12 riportiamo le motivazioni, basate sull'esperienza diretta, che hanno condotto alla determinazione dei valori assegnati a tutti i parametri nelle sperimentazioni numeriche di questo lavoro. All'interno dello stesso capitolo sono anche menzionati l'ambiente di calcolo e di sviluppo utilizzati.

Infine, il capitolo 13 raccoglie i risultati degli esperimenti effettuati su un ampio insieme di istanze verosimili, generate dal procedimento del capitolo 11.

Capitolo 10

Descrizione dati reali

I dati reali a nostra disposizione sono stati forniti dall'ASL di Ferrara. Essi rappresentano uno storico di richieste R archiviate durante il mese di febbraio 2010. Una *richiesta*, o *prestazione*, r è una tupla $(h, \texttt{tipo}_s, p_s) \in R$ rispettivamente giorno, tipo del servizio e luogo ove svolgere il servizio; quest'ultimo identifica il paziente. Consideriamo un servizio s come la coppia $(\texttt{tipo}_s, p_s)$, e l'insieme dei servizi $S = \{s = (\texttt{tipo}_s, p_s) \mid \exists h \in H : (h, \texttt{tipo}_s, p_s) \in R\}$. In questo capitolo presenteremo un'analisi statistica dello storico delle richieste, relativamente alla tipologia, alla frequenza, all'intermezzo e alle distanze tra servizi. Osserveremo che i dati reali sono eterogenei.

10.1 Tipologia di servizio $\texttt{tipo}_s$

Abbiamo un elenco `Tipi` di tipi di servizio $\texttt{tipo}_s$, con associata una durata a_s. In tabella 10.1 riportiamo la quantità di volte che una durata è ripetuta nell'elenco di tipi di servizio `Tipi` e nelle richieste R. Si ha che esiste una tipologia di servizio che richiede dieci minuti di tempo, ma non è mai stata richiesta; mentre per 624 volte in un mese è stato svolto un servizio che richiede 45 minuti di tempo di lavoro.

Tra queste tipologie di servizio, ce ne sono alcune per le quali occorre ritornare al centro di servizio dopo breve tempo, come il prelievo ematico o il campione biologico. Denotiamo questo insieme di tipologie di servizi *urgenti* con $U \subset \texttt{Tipi}$. Per modellare i servizi urgenti, senza modificare il modello descritto nel capitolo 1, occorre applicare la seguente trasformazione nella tabella dei costi di viaggio relativamente ai servizi urgenti:

$$\forall s \in S,\ \forall \texttt{tipo}_s \in U \qquad d_{ss'} = \begin{cases} L & \forall s' \in S \setminus \{s, 0\} \\ d_{s0} & s' = 0 \end{cases} . \tag{10.1}$$

10.2 Rapporto servizi-paziente giornalieri

Il numero delle richieste totali avute in quattro settimane è di 3344. In media si hanno 836 richieste settimanali ricevute da 229 pazienti. Maggiori dettagli in tabella 10.2, dove la quantità di pazienti $|P|$ e di servizi $|R|$ sono suddivisi per settimane e per giorno. Nella colonna 'totale'

a_s	Tipi	R
5	1	1
10	1	0
15	22	1729
20	1	237
30	12	696
45	2	624
60	3	57
	42	3344

Tabella 10.1: Nella prima colonna la quantità di tempo di lavoro a_s; nella seconda colonna la quantità di tipi di servizio con tempo di lavoro a_s; nella terza colonna la quantità di volte che un tipo di servizio, con tempo di lavoro a_s, è stato richiesti.

		H							
		1	2	3	4	5	6	7	totale
week 1	$\|P\|$	70	62	61	67	55	15	7	205
	$\|R\|$	159	145	116	139	135	34	12	740
week 2	$\|P\|$	69	59	79	70	69	20	4	214
	$\|R\|$	161	135	183	166	159	58	5	867
week 3	$\|P\|$	67	64	68	84	76	15	0	229
	$\|R\|$	165	155	166	191	149	15	0	841
week 4	$\|P\|$	81	79	86	76	69	18	2	270
	$\|R\|$	183	171	185	167	140	48	2	896

Tabella 10.2: Una colonna si riferisce ad un giorno, la riga al numero di pazienti serviti $|P|$ o numero di richieste svolte $|R|$ nelle 4 settimane

le richieste riportate equivalgono alla somma delle richieste giornaliere, questo poiché la tupla $(h, \texttt{tipo}_s, p_s)$ è unica nel giorno h. Invece, pazienti possono essere serviti più volte in una settimana, ossia la somma del numero di pazienti al giorno è maggiore del numero di pazienti serviti nella settimana riportato nella colonna 'totale'.

Prima di proseguire dobbiamo tenere presente che esistono servizi con stesso tipo $\texttt{tipo}_s$, i quali sono compiuti più volte in uno stesso giorno per lo stesso paziente. Il numero di tali servizi è basso, tale che parlare di servizi giornalieri equivale a parlare di richieste giornaliere. Nel seguito di questa sezione parleremo di servizi giornalieri o richieste giornaliere intendendo la stessa cosa.

Osservando il rapporto tra il numero di richieste $|R|$ e il numero di pazienti $|P|$, sia nel

k	1	2	3	4	5	6	7	8	9
$\Phi(k)$	685	215	219	181	85	23	28	10	1

Tabella 10.3: Valori di $\Phi(k)$

campione settimanale che giornaliero, abbiamo che in media vi sono tra due e tre tipi di servizio per paziente al giorno, ossia in media tra due e tre richieste al giorno per paziente. Entrando nel dettaglio vediamo ora che tale rapporto servizi-paziente è maggiore. Ricaviamo $\varphi(h, p_s)$ la quantità di servizi svolti nel giorno h al paziente p_s, ossia $\varphi(h, p_s) = |\{(h, \texttt{tipo}_s, p_s)\}|$. Ora possiamo quantificare il numero di pazienti ai quali sono compiuti k servizi in uno stesso giorno definendo $\Phi(k) = |\{(h, p_s) | \varphi(h, p_s) = k\}|$.

Otteniamo che in uno stesso giorno un paziente può avere da 1 a 9 servizi con la distribuzione riportata in tabella 10.3, cioè nell'arco di tempo di quattro settimane 1 paziente ha ricevuto 9 servizi nello stesso giorno, 10 pazienti hanno ricevuto 8 servizi nello stesso giorno, 28 pazienti hanno ricevuto 7 servizi nello stesso giorno, *etc.*. Da quest'osservazione emerge che il numero di pazienti che ha ricevuto k servizi nello stesso giorno con $k \in 2, 3, 4$ è circa lo stesso, e che $\Phi(1) \approx \sum_{k=2}^{9} \Phi(k)$.

10.3 Frequenza f_s e intermezzo e_s

Un servizio s può essere ripetuto f_s volte, e richiede un intermezzo di tempo $\tilde{e}_s$ tra due ripetizioni consecutive. Per ricavare questi dati procediamo nel seguente modo. Dall'insieme delle richieste, e dato un orizzonte temporale H, possiamo ricavare la *frequenza del servizio ricostruita* come la quantità di richieste del servizio s, ossia la quantità di richieste della tipologia di servizio $\texttt{tipo}_s$ per il paziente p_s:

$$\tilde{f}_s^H = \tilde{f}_{(\texttt{tipo}_s, p_s)}^H = |\{h \in H \mid (h, \texttt{tipo}_s, p_s) \in R\}| . \tag{10.2}$$

Con H molto grande ci si aspetta che la frequenza ricostruita $\tilde{f}_s^H$ tenda ad essere f_s.

Osservando la figura 10.1 si vede che il 60.74% dei servizi ha frequenza uno. Si presti attenzione al fatto che un servizio ripetuto k volte equivale a k richieste, quindi i 107 servizi con frequenza 3, equivalgono a 321 richieste. Sebbene sia vero che il 60.74% dei servizi ha frequenza uno, è anche vero che le 933 su 3344 richieste equivale a 27.90% delle richieste con frequenza uno. Da quest'osservazione possiamo dedurre che $|R| \approx 3 \cdot |S|$. È quindi opportuno tenere ben distinte le richieste dai servizi quando $|H| > 1$. Quando ci si riferisce alla singola giornata $|H| = 1$ le richieste sono circa lo stesso numero dei servizi, ossia alla coppia tipo-paziente; esclusi 18 casi su 2421 (0.74%) in cui una coppia tipo-paziente è stata ripetuta più volte nello stesso giorno, ad esempio un servizio s ripetuto due volte in uno stesso giorno poiché svolto singolarmente a due coniugi. Queste 18 richieste le poniamo in $\underline{R}$ e definiamo $\overline{R} = R \setminus \underline{R}$.

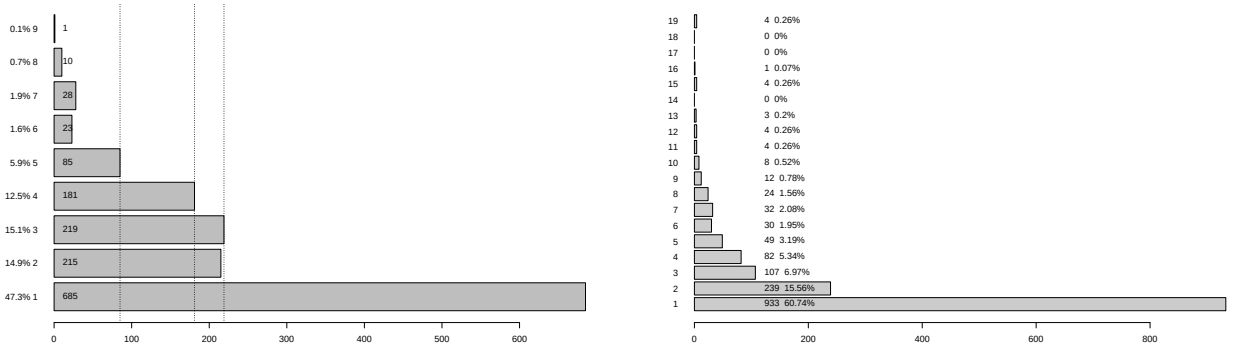


Figura 10.1: L'immagine di sinistra, sull'asse y è il valore k mentre sull'asse x il valore $\Phi(k)$. Nel grafico di destra, sull'asse y è riportata la frequenza $\tilde{f}_s^H$ mentre sull'asse x enumeriamo quanti sono i servizi che in un mese sono ripetuti $\tilde{f}_s^H$ volte.

In modo analogo alla frequenza ricostruita, possiamo ricavare l'*intermezzo ricostruito* come minimo numero di giorni che intercorrono tra due ripetizioni di servizio:

$$\tilde{e}_s^H = \tilde{e}^H_{(\texttt{tipo}_s, p_s)} = \begin{cases} \min\{h-k \mid h \neq k \wedge & f_s > 1 \wedge (h, \texttt{tipo}_s, p_s) \in \overline{R} \\ \quad \wedge (h, \texttt{tipo}_s, p_s), (k, \texttt{tipo}_s, p_s) \in \overline{R}\} & \\ 0 & f_s > 1 \wedge (h, \texttt{tipo}_s, p_s) \in \underline{R} \\ -1 & f_s = 1 \end{cases} . \tag{10.3}$$

È più difficile ricostruire il valore reale dell'intermezzo rispetto alla ricostruzione del valore reale della frequenza. Infatti, consideriamo una richiesta di un servizio s appartenente allo storico, ossia già erogata, ricaviamo l'informazione che il servizio s è stato svolto due volte in un mese ($|H| = 28$) con un $\tilde{e}_s = 3$, tuttavia le esigenze reali del tipo di servizio o del paziente potevano essere valide anche per $e_s \leq \tilde{e}_s$; pertanto $\tilde{e}_s$ è un upper bound all'intermezzo "reale" e_s.

Una volta ricavato $\tilde{f}_s^H$, ed assunto equivalente a f_s, possiamo ricavare $\mathbb{P}(e_s = X \mid f_s)$ usando la distribuzione di probabilità subordinata al conoscere il valore di f_s. In tabella 10.4 mostriamo i valori $\mathbb{P}(e_s = X \mid f_s) > 0$, le celle vuote sono i casi $\mathbb{P}(e_s = X \mid f_s) = 0$.

Da un'analisi visuale di un campione dei dati, non sono emersi motivi sufficienti per considerare $\tilde{e}_s \neq e_s$. Difatti se un servizio è ripetuto più volte nel breve tempo, è sensato pensare che al tendere di $f_s \to H$ il $\tilde{e}_s = e_s$. Per esempio prendiamo un campione di richieste ristretto ad una settimana con frequenza alta che tendente al valore di $|H|$, ossia $f_s > 3$, si ha $\tilde{e}_s^H = 1$, il quale equivale con alta probabilità a e_s (vedi tabella 10.4). Diversamente un servizio ripetuto con frequenza bassa, per esempio $f_s = 2$ volte in un mese, il valore dell'intermezzo ricostruito ha un margine d'incertezza maggiore, si pensi a $\tilde{e}_s^H = 11$. È possibile misurare l'incertezza relativa al valore dell'intermezzo ricostruito rispetto all'intermezzo reale per mezzo del concetto di entropia $\mathcal{H}(X)$, introdotto in (Shannon, 1948), ma l'indagine esula dagli scopi di questo lavoro.

		X						
		-1	0	1	2	3	4	5
f_s	1	$\frac{1827}{1827}$						
	2		$\frac{13}{382}$	$\frac{58}{382}$	$\frac{106}{382}$	$\frac{134}{382}$	$\frac{69}{382}$	$\frac{2}{382}$
	3		$\frac{2}{131}$	$\frac{49}{131}$	$\frac{80}{131}$			
	4		$\frac{1}{49}$	$\frac{48}{49}$				
	5		$\frac{2}{23}$	$\frac{21}{23}$				
	6			$\frac{8}{8}$				
	7			$\frac{1}{1}$				

Tabella 10.4: La tabella riporta i valori della probabilità di dover aspettare e_s giorni sapendo che il servizio s subordinata al valore della frequenza f_s, ossia $\mathbb{P}(e_s = X \mid f_s)$

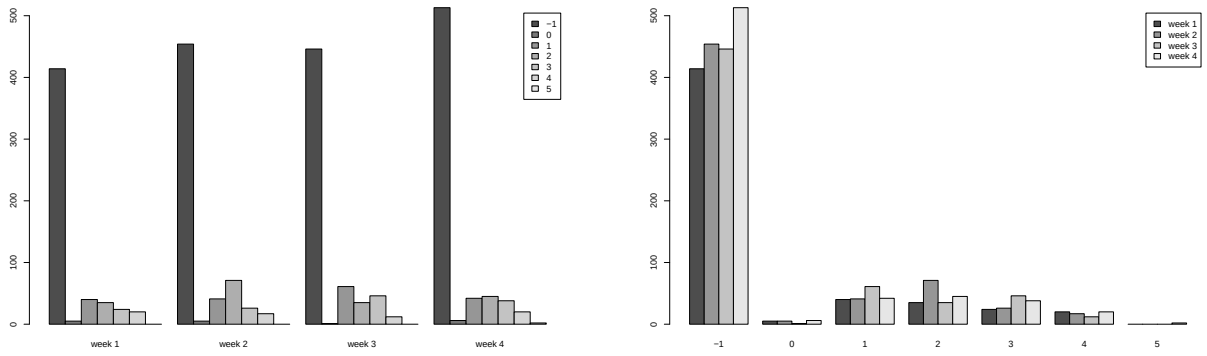


Figura 10.2: I due grafici presentano la distribuzione di valori di $\Delta_s \in \{-1, 0, 1, 2, 3, 4, 5\}$; nel grafico di sinistra ad ogni settimana corrispondono sette barre relative alla distribuzione di valori di Δ_s; nel grafico di destra ad ogni possibile valore di Δ_s corrispondono quattro barre delle relative settimane.

10.4 Tempo di viaggio $d_{ss'}$

Abbiamo detto che un luogo dove viene svolto un servizio identifica il paziente p_s. Per leggibilità identifichiamo con $d_{ss'}$ il tempo di viaggio tra l'indirizzo del paziente p_s e l'indirizzo del paziente $p_{s'}$, o semplicemente tempo di viaggio tra il servizio s e il servizio s'. Il valore del tempo di viaggio è ottenuto con l'ausilio di un *Geographical Information System* (GIS), Google Maps™, che tiene conto della presenza di sensi unici. Tenuti in considerazione nel calcolo dei tempi di viaggio, i sensi unici possono portare due luoghi spazialmente vicini ad essere temporalmente lontani (vedi figura 10.3), e danno luogo a una matrice dei tempi di viaggio asimmetrica.

Ottenuti i tempi di viaggio $d_{ss'}$ per ogni coppia di servizi (s, s'), possiamo ricavare il costo di viaggio $c_{ss'} = d_{ss'} + a_{s'}$. Abbiamo raggruppato in $T^h = \{0, s, s', \dots\}$ i servizi svolti nel giorno

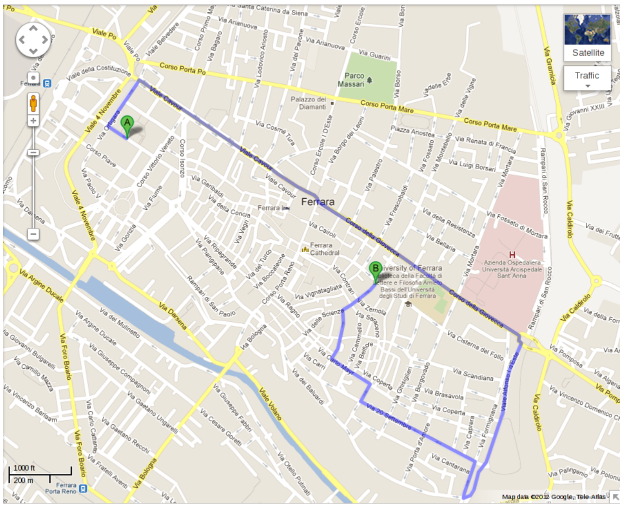


Figura 10.3: Alcune strade sono raggiunte 12 minuti, sebbene vicine, per via dalla presenza di sensi unici

h; ne abbiamo calcolato il TSP; il valore ottenuto è stato diviso per il tempo limite L, ossia $\lceil \mathrm{TSP}(T^h)/L \rceil$, ottenendo un approssimazione per difetto del numero di infermiere richieste.

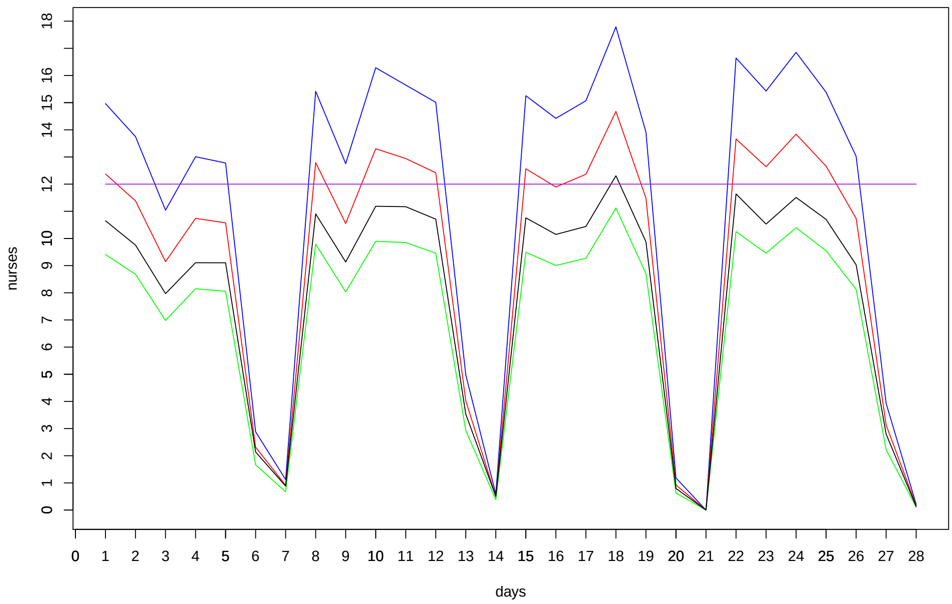


Figura 10.4: Sull'asse delle x il giorno h, sull'asse delle y il numero di infermiere richiesto; i valori riportano $\lceil \mathrm{TSP}(T_h)/L \rceil$, dove T_h è l'insieme dei servizi del giorno h; con il colore nero $d_{ss'}$ è il valore dato dal GIS, mentre costante per gli altri colori: blue per $d_{ss'} = 15$, rosso per $d_{ss'} = 8$ e verde per $d_{ss'} = 0$. La linea viola identifica il massimo numero di infermiere disponibili.

Dalle immagini in figura 10.4 la linea nera, tra la linea rossa e la linea verde, rappresenta il numero di infermiere stimato, usando come costo di viaggio la somma del tempo di lavoro e della distanza tra due servizi ottenuta dal GIS. Si può osservare che l'istanza del giorno 18

il numero di infermiere $\lceil \text{TSP}(T^{18})/L \rceil$ che è maggiore del numero di infermiere a disposizione (linea viola). Ovvero, se un'infermiera lavorasse senza sosta e senza tornare al centro operativo, si avrebbe un tempo di lavoro maggiore a $12 \cdot L$; pertanto, sotto l'ipotesi che i tempi di lavoro e di viaggio siano stati rispettati, è molto probabile che in quel giorno sia stato necessario un tempo di lavoro straordinario w per qualche infermiera.

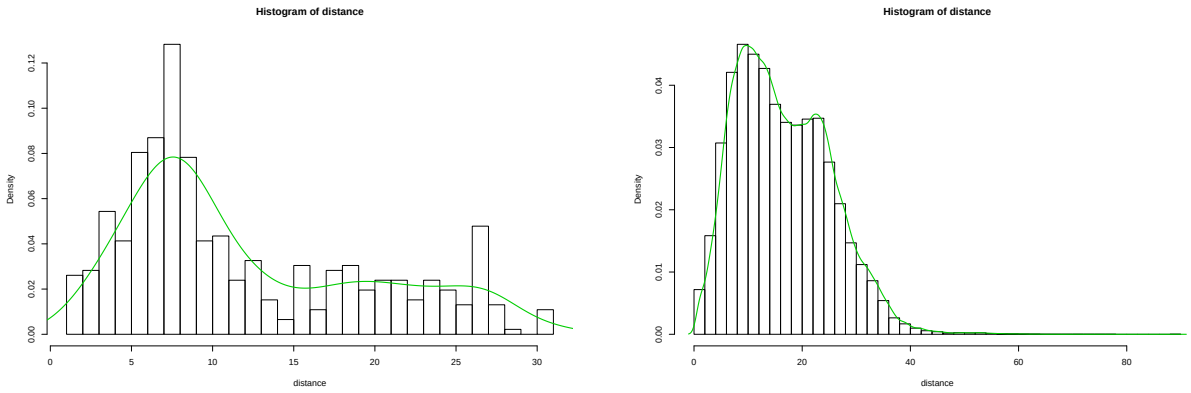


Figura 10.5: Il grafici rappresenta la distribuzione di probabilità del valore della distanza d_{rs}: quello di sinistra riguardo le distanze dalla centrale operativa (deposito), quello di destra tutte le distanze

Abbiamo ricavato altre informazioni quali: 13.27 il tempo medio di viaggio tra due servizi; la distribuzione dei tempi di viaggio dalla centrale operativa 0 (figura 10.5 di sinistra), ove la maggior densità vicino al valore 8; la distribuzione di tutti i tempi di viaggio ricavati mediante il GIS (figura 10.5 di destra), ove la maggior densità è vicino al valore 15.

Abbiamo ripetuto la procedura precedentemente descritta la quale per ogni giorno del mese abbiamo risolto un TSP per ottenere T^h, ma al posto di usare il tempo di viaggio fornito dal GIS abbiamo usato come valori dei tempi di viaggio le costanti 0, 8, e 15. Si può osservare in figura 10.4 che le soluzioni del TSP, prendendo come tempi di viaggio un valore costante nell'intorno del tempo medio di viaggio (13 minuti), si allontanano molto dal valore ottenuto con i tempi di viaggio del GIS.

10.5 Conclusioni

Questo capitolo, dedicato all'analisi dello storico di richieste reali, ha permesso di identificare alcune caratteristiche delle istanze. Circa metà dei servizi sono richiesti una volta sola, l'altra metà dei servizi è ripetuta almeno una volta. Il numero di richieste è quindi due o tre volte il numero di servizi.

Dagli elementi a nostra disposizione, ossia il giorno, il tipo del servizio ed il luogo ove svolgere il servizio, abbiamo ricavato i valori $\texttt{tipo}_s$, f_s, e_s e $d_{ss'}$, osservando la non uniformità delle rispettive distribuzioni. Inoltre, ottimizzando con un TSP l'insieme delle richieste giornaliere, abbiamo visto che il tempo medio di viaggio fornisce soluzioni molto lontane da quelle ottime. Abbiamo quindi dedotto che i turni svolti nel periodo analizzato, probabilmente, sono stati ottemperati con l'ausilio di tempo di lavoro straordinario.

Capitolo 11

Generazione istanze

L'insieme delle richieste analizzato nel capitolo 10 ci permette di poter generare delle istanze che rispettino le caratteristiche di eterogeneità dei tempi di viaggio e di servizio. Oltre ai tempi di viaggio tra i servizi tratti dalla realtà, è possibile generare istanze per le quali i servizi sono collocati su un piano euclideo. In questo capitolo descriviamo la generazione delle istanze realistiche tratte dai casi reali, le cui dimensioni variano in un intorno del valore di richieste considerato elevato dal fornitore di servizi.

11.1 Generatore

Dando uno sguardo d'insieme, il generatore delle istanze è composto di due parti: la parte di generazione e quella di verifica. La generazione consiste nell'assegnazione dei valori delle caratteristiche (frequenza, intermezzo, *etc.*) ad ogni servizio di un prescelto insieme di servizi S. La generazione dei valori è svolta usando un generatore di numeri pseudo-casuali, tenendo conto della distribuzione di probabilità ottenuta dall'analisi dei dati reali presentata nel capitolo 10. Ottenuti i valori pseudo-casualmente, occorre verificare che non vi siano ripetizioni, ovvero che non vi siano servizi che hanno le stesse caratteristiche. Inoltre occorre verificare che la coppia di frequenza e intermezzo ammetta una soluzione. Una volta generate tutte le caratteristiche dei servizi di S è opportuno verificare che un algoritmo atto alla generazione di una soluzione iniziale, come l'algoritmo proposto in sezione 6.4, ammetta una soluzione.

Ora guardiamo alcuni dettagli della generazione dell'istanza descritta in pseudo codice nell'algoritmo 11.1. Nelle prime cinque righe sono inizializzati i valori di e_s, $\texttt{tipo}_s$ e p_s ad uno stesso valore. L'algoritmo rimane nel ciclo principale finché esistono servizi s al quale non è stato assegnato un valore ammissibile ($p_s = -1$) o servizi doppi, ossia che hanno le stesse caratteristiche di altri (vedi line 6). In linee 7–10 vengono assegnati i valori pseudo casualmente; in particolare a linea 10 il valore di e_s è ottenuto dalla probabilità subordinata alla conoscenza del valore della frequenza. Il flusso si divide in due casi:

- nel primo caso (linee 11–15), il servizio è richiesto ogni giorno ($f_s = 7$); oppure il parametro $allSet = true$, quindi si desidera che per ogni servizio possa essere sempre compiuto in ogni giorno della settimana. Pertanto, per ogni giorno h abbiamo $g_s^h = 1$.

algoritmo 11.1 `instance_generator`$(S, allSet)$:

1: **for all** $s \in S$ **do**
2: $p_s \leftarrow -1$
3: $e_s \leftarrow -2$
4: $\texttt{tipo}_s \leftarrow -3$
5: **end for**
6: **for all** $s \in S : p_s = -1 \vee \exists s' \in S \setminus \{s\} : p_{s'} = p_s \wedge \texttt{tipo}_{s'} = \texttt{tipo}_s \wedge (e_s \neq 0 \vee e_{s'} \neq 0)$ **do**
7: $p_s \leftarrow \text{random}(\{\text{set of addresses}\})$ ..... # distribuzione di probabilità per posizione
8: $\texttt{tipo}_s \leftarrow \text{random}(X = \{1, \ldots, 31\}, \mathbb{P}(\texttt{tipo}_s = X))$ ... # distribuzione di probabilità per tipo
9: $f_s \leftarrow \text{random}(\{1,2,3,4,5,6,7\}, \{\frac{1827}{2421}, \frac{382}{2421}, \frac{131}{2421}, \frac{49}{2421}, \frac{23}{2421}, \frac{8}{2421}, \frac{1}{2421}\})$
10: $e_s \leftarrow \text{random}(X = \{-1,0,1,2,3,4,5\}, \mathbb{P}(e_s = X \mid f_s))$ ..... # tabella 10.4
11: **if** $f_s = 7 \vee allSet$ **then**
12: **for all** $h \in H$ **do**
13: $g_s^h \leftarrow 1$
14: $G_s \leftarrow G_s \cup \{(g_s^h, h)\}$
15: **end for**
16: **else**
17: **repeat**
18: **repeat**
19: **for all** $h \in H$ **do**
20: $g_s^h \leftarrow \text{random}(\{0,1\})$ ... # sabato e domenica devono avere meno probabilità
21: $G_s \leftarrow G_s \cup \{(g_s^h, h)\}$
22: **end for**
23: **until** $|G_s| < f_s$
24: **until** $\neg\,\text{check}(e_s, f_s, G_s) \wedge e_s \neq 0$
25: **end if**
26: **end for**

- Nel secondo caso (linee 16–26), occorre generare un insieme di giorni disponibili. Per prima cosa si sceglie pseudo-casualmente i giorni in cui il servizio può essere svolto (linee 19–22). Poi si verifica che vi siano tanti giorni possibili almeno quanta la disponibilità (linea 23), e se non ammette ripetizioni nello stesso giorno ($e_s = 0$) allora si verifica che i valori dell'intermezzo, della frequenza e di G_s siano ammissibili. Se la verifica fallisce si continua a generare valori pseudo-casualmente, altrimenti un nuovo servizio è stato correttamente generato.

algoritmo 11.2 check(e_s, f_s, G_s):

1: $g_s^h \leftarrow \{\tilde{x} \mid (\tilde{x}, h) \in G_s\}$
2: $\sigma \leftarrow$ solve$(\{(11.1), (11.2), (11.3), (11.4)\})$
3: **return** true se σ ammissibile, false altrimenti

Rimane da spiegare la verifica dei valori dell'intermezzo, della frequenza e di G_s svolta dall'algoritmo 11.2. Dopo aver ricavato i valori di g_s^h, possiamo risolvere il seguente problema di ammissibilità:

$$\sum_{h \in H} \gamma_s^h \geq f_s \tag{11.1}$$

$$\sum_{k=h}^{h+e_s-1} \gamma_s^k \leq 1 \qquad \forall h \in H \tag{11.2}$$

$$g_s^h \geq \gamma_s^h \qquad \forall h \in H \tag{11.3}$$

$$\gamma_s^h \in \{0, 1\} \qquad \forall h \in H\,. \tag{11.4}$$

La variabile booleana γ_s^h è 1 se il servizio è assegnato nel giorno h. Il vincolo (11.1) verifica che il servizio sia ripetuto almeno f_s volte. I vincoli (11.2) garantiscono che l'intermezzo sia rispettato. Nei vincoli (11.3) si richiede che il giorno di servizio sia compatibile con la possibilità di svolgere il servizio. Se il problema ammette soluzione, allora i parametri e_s, f_s, G_s sono accettabili.

11.2 Distanze euclidee

È usuale negl'articoli relativi ai problemi di instradamento non entrare nel dettaglio della matrice delle distanze, avendo informazioni generiche come la simmetria della matrice stessa. Si è a conoscenza che il software GIS scelto computa il tempo di percorso tenendo traccia della velocità urbana o extraurbana, oltre a ricavare percorsi che rispettano i sensi unici. L'esempio portato in figura 10.3 solleva un'esigenza di comprendere quanto una distanza euclidea deformi la realtà dei costi.

Nell'algoritmo 11.4 richiediamo che una quantità q di servizi sia inserita nel "quanto di tempo" $[min, max]$ (vedi algoritmo 11.3). I "quanti di tempo" sono tratti dalla distribuzione delle distanze dal deposito, vedi figura 10.5.

In figura 11.1 possiamo osservare la differenza tra il grafico ottenuto dalle latitudini e longitudini dei servizi tratti dal GIS e dalle posizioni euclidee.

algoritmo 11.3 addVal(q, min, max)

1: **for** $i = 0; i < q; i++$ **do**
2: **repeat**
3: $x \leftarrow \text{random}(0, 60)$
4: $y \leftarrow \text{random}(0, 60)$
5: $d_{xy} \leftarrow \sqrt{(x-30)^2 + (y-30)^2}$
6: **until** $min \geq d_{xy} \vee d_{xy} \geq max$
7: **end for**

algoritmo 11.4 euc2d

1: addVal$(69, 0.0, 5.0)$
2: addVal$(191, 5.0, 10.0)$
3: addVal$(56, 10.0, 15.0)$
4: addVal$(55, 15.0, 20.0)$
5: addVal$(49, 20.0, 25.0)$
6: addVal$(35, 25.0, 30.0)$
7: addVal$(5, 30.0, 35.0)$

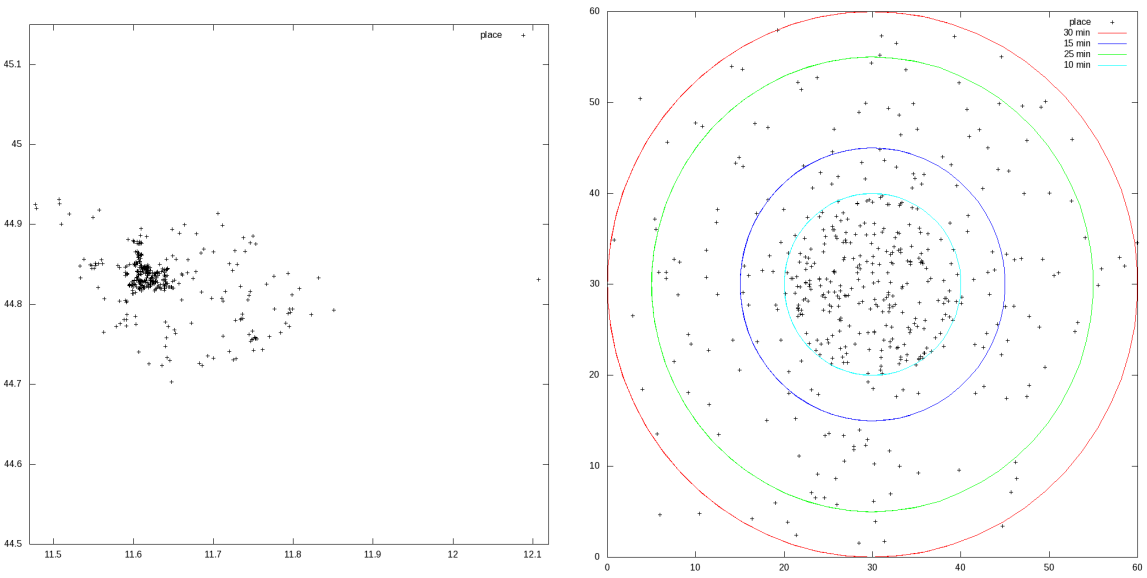


Figura 11.1: I grafici rappresentano la posizione dei luoghi dove viene svolto il servizio. Nell'immagine di sinistra sono rappresentati i luoghi dei servizi tratti dalle coordinate GIS. Nell'immagine di destra i luoghi dei servizi γ_s sono posizionati su un piano euclideo, la centrale operativa 0 è al centro dell'immagine $(30, 30)$, ed ogni cerchio rappresenta il tempo $d_{0\gamma_s}$; due cerchi che non contengono un altro cerchio identificano il quanto di tempo $[min, max]$ nel quale vi saranno q servizi.

11.3 Conclusione

In questo capitolo abbiamo descritto la generazione delle istanze realistiche tratte dai casi reali. Le dimensioni di tali istanze variano in un intorno del valore di richieste considerato elevato dal fornitore di servizi.

Capitolo 12

Calibrazione parametri

Lo scopo di questo capitolo è duplice: si desidera mettere il lettore nelle condizioni di replicare il lavoro svolto e si desidera riportare l'esperienza avuta durante la fase di sviluppo e messa a punto di questi parametri (detta anche calibrazione).

Per il secondo scopo è ardua una verifica esaustiva: infatti, se per esplorare l'effetto di k parametri booleani su un campione di istanze il numero di impostazioni che occorre provare è 2^k, poiché nel caso in esame alcuni parametri hanno valori reali o interi, la quantità di verifiche necessarie aumenta considerevolmente.

Abbiamo, pertanto, scelto di riportare l'esperienza e congetturare possibili correlazioni tra l'impostazione di un valore e le caratteristiche di un istanza, al fine di aiutare sia una celere applicazione dei metodi in differenti contesti, che l'impostazione di una successiva accurata indagine.

12.1 Ambiente di calcolo

Il codice è stato implementato in C++ e compilato usando il compilatore g++ 4.6.0 con parametro di ottimizzazione `-O2`. Il generatore di numeri pseudo casuali scelto è la funzione `std::rand`, al quale è stato posto il seme di inizializzazione 99. Per la soluzione di problemi MILP viene usata la libreria gurobi. Per la soluzione del TSP è stato valutato l'utilizzo di Concorde (`http://www.tsp.gatech.edu/concorde/`); tuttavia nel contesto di istanze reali Concorde termina restituendo la soluzione iniziale data dall'euristica e un codice d'errore. Questo comportamento è imputabile ad un'instabilità numerica, conseguenza del dover porre un costo molto alto per gli archi che non sono presenti nel grafo.

Le esecuzioni sono state svolte su due differenti macchine, entrambe con sistema operativo linux:

name: PC1
CPU: 2 × Intel Xeon X5620 2.40GHz
cores: 2 × 4 = 8
threads: 2 × 8 = 16

RAM: 18 GB DDR3
MILP solver: gurobi 5.0.0

e

name: PC2
CPU: 2 × Intel Xeon X5690 3.46GHz
cores: 2 × 6 = 12
threads: 2 × 12 = 24
RAM: 188 GB DDR3
MILP solver: gurobi 5.0.1

Occorre infine ricordare che, essendo gli elaboratori condivisi con altri utenti, è possibile che talvolta le computazioni siano svolte in contemporanea con altri processi, il che può rallentare l'esecuzione relativa ad un certa istanza.

12.2 Lower Bound

Nel capitolo 5 abbiamo introdotto un lower bound per il DCVRP, sottoproblema di routing dell'HHCP, ed un lower bound che rilassa le componenti d'instradamento annullando i tempi di viaggio ottenendo un sottoproblema di scheduling.

Routing. Con il rilassamento lagrangiano introdotto nella sezione 5.2 otteniamo un lower bound del DCVRP. Per ottenere i valori dei moltiplicatori di Lagrange abbiamo presentato il metodo del sottogradiente (algoritmo 5.1). Durante la fase di calibrazione del passo μ (linee 4 abbiamo riscontrato una difficoltà a scegliere una regola opportuna alla convergenza, tale difficoltà è riscontrata anche da altri autori nel contesto del m-TSP (Desrosiers et al., 1988). Nell'algoritmo 5.1 dimezziamo ϵ_t (linee 6–10), abbiamo impostato che il dimezzamento avvenga dopo 10 volte consecutive che nel ciclo si ha $\mathbf{x}_t$ con lo stesso valore. Oltre alla regola per definire il passo μ descritta nell'algoritmo, è stata applicata la regola banale $\mu = \epsilon_t$ con $\epsilon_0 = 0.625$. Per la pesantezza della soluzione del sottoproblema non è stato possibile svolgere sufficienti confronti tra le due regole.

Scheduling. Abbiamo visto in sezione 5.2 che annullando i tempi di viaggio otteniamo un rilassamento dell'HHCP. Dalla formulazione presentata nelle equazioni (5.23)–(5.31) otteniamo:

- il lower bound $\mathcal{LB}$ se i parametri sono $\alpha_0 = 1$ e $\alpha_1 = 1$;
- il lower bound $\mathcal{LB}_u$ se i parametri sono $\alpha_0 = 1$ e $\alpha_1 = 0$;
- il lower bound $\mathcal{LB}_l$ se i parametri $\alpha_0 = 0$ e $\alpha_1 = 1$.

Si ha che $\mathcal{LB} \geq \mathcal{LB}_u + \mathcal{LB}_l$.

12.3 VNS e ALNS

12.3.1 Soluzione iniziale

Nella sezione 6.4 abbiamo presentato l'algoritmo 6.6 per ottenere rapidamente una soluzione iniziale dalla quale le meta euristiche VNS e ALNS potessero partire. L'algoritmo presentato ha alcuni aspetti critici, correlati alle caratteristiche delle istanze ed al fatto che deve ottenere una soluzione ammissibile per un problema NP-Hard. Questi aspetti vengono superati con un opportuna impostazione dei parametri. Elencheremo alcune caratteristiche osservate durante la messa a punto dei parametri.

Le seguenti impostazioni in generale hanno permesso di ottenere soluzioni in pochi secondi, senza eccedere il tempo limite $\texttt{timelimit}_{max} = 125$ secondi: $\alpha_0 = 1$; $\alpha_1 = 1$; $\alpha_2 = 0$; $\alpha_3 = M$, $W^{max} = 0$, $\texttt{numSolutions} = 1$; $c_s = \hat{d}_s + a_s$; $\Lambda = 0$; $\Upsilon = \lceil L/a_{min} \rceil = 26$, con $a_{min} = \min\{a_s \mid s \in S\}$; $\zeta = 0$.

Per l'insieme d'istanze prese in esame, è stata sufficiente la soluzione del modello ottenuta dalla linea 4 dell'algoritmo 6.6. Tuttavia non è da escludere che per istanze più grandi, o con tempi di transito da un servizio ad un altro diversamente distribuiti, possa essere richiamato il risolutore del modello (6.2)–(6.15) con i parametri delle linee successive.

Nel seguito discutiamo alcune congetture riguardo le possibili correlazioni fra parametri e soluzioni, oltre alle impostazioni apparentemente più idonee per altri tipi di istanze.

Scelta del costo c_s. Il parametro c_s è l'approssimazione del tempo speso per il servizio s, dato dalla somma del tempo di servizio a_s e dell'approssimazione del tempo di viaggio d_s, quindi $c_s = d_s + a_s$. Abbiamo scelto tre approssimazioni di d_s: la distanza nulla 0, la distanza media $\tilde{d}_s$, la distanza ottenuta dal primo quartile $\hat{d}_s$.

Nel seguito consideriamo impostato il parametro $W^{max} = 0$, per richiede di avere turni che non eccedano L, evitando il tempo straordinario. Con questa scelta abbiamo osservato una difficoltà del solutore a fornire soluzioni ammissibili con $\tilde{d}_s$: per tale motivo si è poi deciso di usare $\hat{d}_s$. Quest'ultimo, sebbene abbia sempre permesso al solutore di giungere in tempi celeri ad una soluzione ammissibile, non dà però garanzie di esistenza di una soluzione ammissibile, a differenza di quanto avviene per la distanza nulla.

Il fatto che non si giunga ad una soluzione ammissibile con $\tilde{d}_s$ può essere in correlazione con le osservazioni svolte in sezione 10.4: abbiamo visto che il valore della distanza media $\tilde{d}_s$ è circa 13 ed abbiamo anche osservato che l'intorno dei valori vicini al valore medio tende ad essere una stima eccessiva del numero di infermiere richieste (figura 10.4).

Eccedere il tempo limite ($W^{max} = 0$). Ammettendo il tempo di lavoro straordinario, cioè $W^{max} = \infty$, si permette al solutore di restituire subito una soluzione ammissibile per qualsiasi scelta del costo di servizio c_s. Quando impostiamo $W^{max} = \infty$, e scegliendo di fermare il solutore alla prima soluzione ammissibile, abbiamo un eccessivo sbilanciamento dei tour, ovvero un tour con una quantità di servizi troppo grande e vari tour senza servizi. Quando impostiamo $W^{max} = 0$, ossia si deve rispettare il massimo tempo L di un turno giornaliero,

ed abbiamo il costo di servizio $c_s = \tilde{c}_s$ (dato dalla media dei tempi di viaggio $\tilde{d}_s$), allora il solutore non giunge ad una soluzione ammissibile in tempi rapidi. Consideriamo la *qualità* di una configurazione come il rapporto tra massimo numero di servizi in un turno giornaliero e tempo di calcolo per avere la prima soluzione ammisibile. Nessuna delle configurazioni del modello esplorate ha permesso di eliminare il fenomeno descritto per $W^{max} = \infty$ con la stessa qualità dalla configurazione $W^{max} = 0$ e $\hat{c}_s$, ossia non ammettendo tempo di lavoro straordinario e considerando il costo dato dal 'primo quartile' $\hat{d}_s$ (vedi sezione 6.4).

Numero di servizi per turno. Dallo storico delle richieste fornito dall'AUSL è emerso che, sovente, nei giorni feriali il numero giornaliero dei nodi, ossia dei servizi, è compreso tra due estremi $[\Lambda, \Upsilon]$. I vincoli (6.6)–(6.8), sul minimo (Λ) e massimo (Υ) numero di servizi da svolgere in una giornata, possono permettere di ottenere soluzioni ammissibili più vicine alle soluzioni bilanciate.

È possibile soddisfare sempre il vincolo sul massimo numero giornaliero di nodi Υ con il seguente procedimento. Dato $a_{min} = \min\{a_s \mid s \in S\}$ il costo minimo di servizio e dato il tempo L di massima durata di un turno, allora possiamo imporre $\Upsilon = \lceil L/a_{min} \rceil$.

Se l'insieme di istanze ha caratteristiche quali l'omogeneità della distribuzione dei servizi, allora in ogni giorno ci sarà un numero di servizi che tende ad essere equivalente al numero dei servizi di un altro giorno: pertanto, il minimo numero di nodi tenderà ad essere un valore positivo, quindi un valore di Λ abbastanza alto è d'ausilio. In alcuni casi è necessario modellare periodi con distribuzione di servizi eterogenea, pertanto il numero di servizi giornalieri nei giorni festivi e prefestivi potrebbe scendere anche sotto la metà, come nel nostro insieme d'istanze. In questo caso, l'algoritmo 6.6 potrebbe non terminare in tempi accettabili, oppure terminare affermando che non esistono soluzioni ammissibili. Alla non esistenza di soluzioni ammissibili, è possibile ovviare limitando i vincoli (6.6) al sottoinsieme di giorni feriali H_f, invece che alla settimana lavorativa H. Nell'esperienza tratta durante la messa a punto dei parametri, il valore di Λ che tende a zero aiuta la celerità del solutore nel trovare una prima soluzione ammissibile.

Massima durata del tour. Il parametro ζ definisce un lower bound sulla durata totale approssimata di un singolo turno giornaliero (vincolo (6.9)). Nel modello (6.2)–(6.15), con l'attivazione del vincolo (6.9), ovvero $\zeta > 0$, il metodo potrebbe non terminare in tempi accettabili, oppure terminare affermando che non esistono soluzioni ammissibili. Per esperienza, si ipotizza vi sia una correlazione tra l'inammissibilità e istanze che presentano elevata eterogenità della quantità di servizi giornalieri, del tempo di servizio o del tempo di viaggio. Per quel campione d'istanze che hanno mostrato di ammettere soluzione con $\zeta > 0$, l'esperienza suggerisce una correlazione tra l'aumento del tempo per ottenere una prima soluzione ammissibile e la grandezza di ζ.

Analogamente a quanto appena affermato nel contesto del limite del numero giornaliero di servizi in un tour, nei giorni festivi e prefestivi non si riesce a garantire un lower bound sulla durata minima del turno giornaliero.

Vincolo frequenza. Il vincolo (6.2) richiede che un servizio s sia svolto almeno f_s volte:

$$\sum_{h \in H} \sum_{j \in N_h} y_s^{jh} \geq f_s \qquad \forall s \in S$$

Quando il vincolo è posto come disuguaglianza, aiuta a far convergere celermente il risolutore, particolarmente indicata per un lower bound, del quale interessa il valore della funzione obiettivo. Tuttavia, non si hanno garanzie che il solutore di programmazione matematica fornisca soluzioni con servizi svolti esattamente f_s volte. Se invece si pone il vincolo ad uguaglianza, allora certamente la soluzione non ha ripetizioni indesiderate del servizio. Volendo utilizzare la prima soluzione ammissibile che il solutore restituisce, abbiamo optato per la formulazione con l'uguaglianza.

12.3.2 Operazioni

Sono state scelte per le euristiche VNS e ALNS le seguenti operazioni: M1–M6; S1–S3; K1–K2 con $k = 2, 3$. Per ogni operazione che lo richiede, è stato impostato il parametro di massima distanza $\delta_{max} = 25$ e l'incremento $\Delta = 5$. È stato implementato un controllo per evitare di ripetere due volte l'applicazione di uno stesso operatore con gli stessi argomenti, ossia: è possibile avere $\texttt{op}_1(1,2)$–$\texttt{op}_2(1,2)$, poiché sono due operatori differenti, ma non è possibile avere $\texttt{op}_1(1,2)$–$\texttt{op}_3(2,3)$–$\texttt{op}_1(1,2)$, perché l'operatore 1 è applicato con gli stessi argomenti della precedente applicazione. Per la VNS è stato impostato il parametro $\overline{\kappa} = 10$.

Per l'ALNS non occorre definire un ordine di sequenza. Per la VNS, invece, la quale passa sistematicamente all'operazione successiva, impostiamo l'ordine come riportato in precedenza. L'ordine scelto è motivato dal fatto che le soluzioni iniziali sono sbilanciate, pertanto si preferisce inizialmente l'uso di mosse `move`, per abbattere i costi del lavoro straordinario. Le mosse `swap` sono precedute dalle mosse k`swap`, per via del costo computazionale.

12.3.3 Verifica di ammisibilità

Nella verifica della parte di routing, il valore di $\underline{\lambda}$ è stato impostato empiricamente a 0.10. Denotiamo con VNS–A e ALNS–A le euristiche che hanno il parametro `always optimize` posto a $true$, mentre è $false$ per le euristiche VNS e ALNS.

Abbiamo impostato il valore $\underline{W}_{max} = L$. Generalmente la verifica della parte di scheduling richiede meno di un secondo, sporadicamente richiede circa tre secondi, raramente richiede tempi non accettabili. Per questo noi imponiamo un tempo limite di tre secondi per il solutore. Nel caso si scelga di avere un tetto massimo alto per w , è possibile diminuire il tempo di risoluzione nella fase iniziale. Diversamente, un tetto massimo troppo stringente scarta tutte le mosse della fase iniziale ove i tour superano il tempo limite L. Alla fine della verifica è possibile riassegnare i tour alle infermiere in diversi giorni: in tal caso è consigliabile mantenere il vincolo (8.4) come uguaglianza. Analogamente al modello della soluzione iniziale, nel modello della verifica della parte di assegnamento (sezione 8.4) possiamo usare una disuguaglianza, la quale permette al risolutore di essere più veloce, in tutti i casi nei quali non occorra una soluzione ammissibile,

ossia non occorra che dopo la verifica della parte di scheduling sia noto l'assegnamento esatto del tour all'infermiera. Un esempio è l'euristica Clarke & Wright (algoritmi 9.2), nella quale la procedura di fusione non necessita di conoscere una soluzione ammissibile per proseguire, e solo alla fine sarà assegnata l'infermiera ed il giorno ad ogni tour.

12.4 Conclusioni

In questo capitolo abbiamo riportato le informazioni sperimentali che è stato possibile raccogliere durante la fase di calibrazione dei parametri. Abbiamo inoltre congetturato possibili correlazioni tra l'impostazione di un valore e le caratteristiche di un istanza, al fine di aiutare sia una celere applicazione dei metodi in differenti contesti, che l'impostazione di una successiva accurata indagine.

I risultati presentati nel prossimo capitolo sono ottenuti con le impostazioni dei parametri qui descritte per i rispettivi algoritmi.

Capitolo 13

Risultati computazionali

Nel presente capitolo saranno riportati i risultati delle computazioni del bound e delle euristiche presentate in questa tesi. Nella sezione dedicata all'euristica Clarke & Wright sarà possibile osservare la pesantezza computazionale della verifica della parte di scheduling di una soluzione. Successivamente sarà presentato un confronto tra VNS e ALNS sia per le istanze realistiche che per le istanze euclidee.

13.1 Lower bound

In questa sezione analizziamo i risultati computazionali ottenuti imponendo i lower bound proposti nel capitolo 5.

13.1.1 Instradamento

In letteratura non si è trovato risultato teorico che attesti una dominanza tra il rilassamento lineare del problema (5.11)–(5.17) con il vincolo (5.18) e lo stesso rilassamento senza tale vincolo. Computazionalmente, l'aggiunta del vincolo ha sempre permesso di avere un miglior bound rispetto al modello che ne è privo.

La messa a punto del metodo del sottogradiente coniugato ha mostrato la convergenza del metodo; tuttavia uno dei punti critici è che il sottoproblema è NP-Hard, cioè per alcune istanze, o valori di λ, la soluzione ottima del sottoproblema viene raggiunta in tempi accettabili, mentre per altre istanze l'algoritmo non raggiunge la soluzione ottima nemmeno dopo tempi inaccettabili, e non ci sono segni di convergenza. Uno degli aspetti determinanti che ha permesso di diminuire apprezzabilmente i tempi di calcolo è l'uso della soluzione $\mathcal{L}(\lambda^{t-1}, \mathbf{x}_{t-1})$ del passo precedente come soluzione iniziale, che chiamiamo '*soluzione innestata*'.

L'aver ottenuto soluzioni in tempi accettabili ha motivato la messa a punto di un'euristica basata sul rilassamento lagrangiano: il metodo parte con una soluzione del m-TSP innestata, ad ogni passo è limitato il tempo del solutore di programmazione matematica per risolvere il sottoproblema, infine, la soluzione ottenuta è un ottimo locale che viene innestata come soluzione iniziale del passo successivo.

Quando non esistono soluzioni ottime, ossia soluzioni che permettano a tutti i tour di essere entro la lunghezza massima L, l'euristica basata sul rilassamento lagrangiano giunge a soluzione ove il tour di massima lunghezza supera di poco la lunghezza massima consentita. Quest'aspetto è apprezzabile per via del fatto che uno scostamento minimale dalla lunghezza massima equivale a superare di pochi minuti il tempo massimo del turno di lavoro, ossia nella pratica è una soluzione accettabile. In particolare, è apprezzabile quando il decisore è a conoscenza che il periodo di lavoro è così intenso da richiedere, con alta probabilità, del tempo di lavoro straordinario. La procedura risulta troppo lenta per un insieme grande di nodi, sebbene rimane d'interesse per l'ottimizzazione di sottoproblemi di pochi tour al fine di accorpare i servizi (per esempio, la mossa `group` in sezione 7.4).

Questo lavoro ci ha permesso di indagare il fenomeno nel quale, a fronte di istanze per le quali l'ottimizzazione del sottoproblema converge celermente ad una soluzione ottima, esistano altre istanze sulle quali il metodo non compie progressi anche per tempi molto lunghi (giorni). In due casi si è constatato che il branch-and-bound non restituisce un miglioramento del lower bound nemmeno dopo diversi giorni di calcolo, non ottenendo così una soluzione ammissibile rispetto al valore impostato negli m veicoli. Per le stesse istanze il metodo descritto raramente fornisce una soluzione ammissibile diversa da quella innestata, sia variando il lasso di tempo concesso per la soluzione del sottoproblema, sia per i valori di λ^t e la scelta del passo. Queste due istanze erano abbastanza grandi da non essere banali e abbastanza piccole per permettere ad un algoritmo di generare tutti i tour di grandezza L. È stato quindi possibile osservare che in tali due istanze la quantità di tour di lunghezza L, e quindi di soluzioni ammissibili, è bassa. Nelle altre istanze di pari dimensione, nelle quali non appare il comportamento di stallo descritto, il numero di possibili tour di grandezza minore di L è ampio; per queste istanze, il metodo branch-and-bound del risolutore non entra in stallo, cioè incrementa sporadicamente il valore del lower bound. Inoltre l'euristica basata sul rilassamento lagrangiano trova nuove soluzioni nel tempo limite d'esecuzione impostato. Da questo nasce la congettura di una correlazione tra la bontà del lower bound e la quantità di soluzioni ammissibili del problema.

In sintesi, lo strumento presentato non si è mostrato idoneo per ottenere un lower bound per il nostro problema in tempi accettabili, pertanto non abbiamo approfondito l'indagine. Dal lavoro di messa a punto del software è emersa una possibile correlazione tra la quantità di soluzioni ammissibili e la bontà del lower bound. Abbiamo individuato un possibile riuso dell'euristica basata sul rilassamento lagrangiano in un contesto euristico per un ottimizzazione locale di problemi come l'HHCP che includono il DCVRP.

13.1.2 Pianificazione

Si ricorda che abbiamo ottenuto il lower bound presentato in sezione 5.2 attraverso un rilassamento dei vincoli d'instradamento ottenuto ponendo ogni distanza a zero. Pertanto, le istanze reali e quelle sintetiche (euclidee), che differiscono solo per il tipo di distanza, avranno stesso lower bound in quanto in esse tutte distanze sono nulle. Per le 60 istanze reali e le 60 euclidee la media dei lower bound $\mathcal{LB}_l$, che corrisponde al valore di u, è 296.3 con deviazione standard

parametri			m540_s13 ($\|R\| = 719$)		m590_s17 ($\|R\| = 848$)	
skip	opt	hist	tempo	scheduling	tempo	scheduling
			$632'50''$	257682	$1556'04''$	519259
✓			$3'55''$	0	$7'35''$	0
		✓	$215'56''$	65179	$425'15''$	110846
✓		✓	$3'04''$	0	$5'05''$	0
	✓		$689'53''$	260167	$1649'42''$	524099
✓	✓		$34'42''$	0	$76'51''$	0
	✓	✓	$236'27''$	65575	$365'30''$	111546
✓	✓	✓	$27'02''$	0	$42'48''$	0

Tabella 13.1: Tempi delle esecuzioni della Clarke & Wright su PC1 impostando o meno i parametri 'skip' per non svolgere il controllo con `scheduling_part`*, 'opt' per l'ottimizzazione del TSP e 'hist' per l'accesso allo storico delle* `merge`*.*

13.9 e massimo valore 325. Il calcolo dei lower bound $\mathcal{LB}_u$ e $\mathcal{LB}$ richiede più di 10 minuti usando 24 thread su PC2; nelle istanze campione eseguite sino al raggiungimento della soluzione ottima osserviamo che $\mathcal{LB}$ mantiene lo stesso valore di u di $\mathcal{LB}_l$, ed il valore della fidelizzazione paziente-infermiera tende a 0: pertanto $\mathcal{LB}$ non aumenta sensibilmente il $\mathcal{LB}_l$.

13.2 Clarke & Wright

Ora consideriamo alcune esecuzioni dell'euristica Clarke & Wright presentata nel capitolo 9. Le istanze campione scelte sono la m540_s13 e m590_s17, le quali sono rispettivamente l'istanza con il più basso e il più alto numero di richieste $|R|$.

La Clarke & Wright è stata eseguita attivando o meno le seguenti opzioni:

- 'skip', quando è disattiva la procedura `scheduling_part` viene sostituita da una procedura che restituisce sempre il valore booleano $false$;
- 'opt': equivale a ottimizzare o meno il tempo totale del turno, se attivata l'algoritmo risolve sempre un TSP dopo ogni mossa, altrimenti non risolve mai il TSP;
- 'hist': l'attivazione richiede all'euristica di usare l'informazione $feasible_{ss'}$ per evitare di ripetere verifiche inutili, ricordiamo che $feasible_{ss'}$ è una variabile booleana che ricorda se un'operazione `merge` non ha avuto esito positivo, ovvero tiene memoria della sentenza negativa fornita da una precedente verifica di ammissibilità;

Nella tabella 13.1 riportiamo nella colonna 'tempo' il tempo reale delle computazioni. Il numero di esecuzioni della procedura `scheduling_part` è riportato nella colonna 'scheduling'. Quando le opzioni 'skip', 'opt' e 'hist' precedentemente descritte sono attive nella tabella comparirà il simbolo ✓ nelle rispettive colonne.

Heuristic	u	l	w	f	Moves	Time limit
ALNS	397.87	25.53	8.48	508.24	54510.25	1500 s
ALNS–A	396.66	31.20	14.05	568.36	19984.02	1500 s
ALNS	393.73	21.78	2.88	444.35	123149.08	3200 s
ALNS–A	390.95	26.33	8.68	504.12	44194.55	3200 s
ALNS	389.97	18.35	0.75	415.82	291201.42	7200 s
ALNS–A	386.91	22.07	4.58	454.81	102116.03	7200 s
VNS(10)	472.00	27.35	92.02	1419.52	—	7200 s
VNS–A(10)	447.11	22.98	59.68	1066.93	—	7200 s

Tabella 13.2: Media dei migliori risultati per VNS, VNS–A, ALNS e ALNS–A su sessanta istanze reali (non euclidee) su PC1 con tempo limite di calcolo pari a 7200 secondi.

Nel caso dell'euristica Clarke & Wright la procedura `scheduling_part` è necessaria per assegnare nuovamente i giorni, tuttavia l'abbiamo disattivata permettendo un raffronto in termini di tempo. Possiamo quindi osservare l'incremento del tempo quando l'opzione 'skip' è disattiva.

Attivando o meno l'opzione 'opt' il tempo di computazione è incrementato di dieci volte, passando da 8 a 80 o da 4 a 40, quando 'skip' è attivo; tale incremento è difficilmente osservabile quando l'opzione 'skip' è attiva.

In tabella 13.1, è possibile osservare come il parametro $feasible_{ss'}$ permetta di rendere considerevolmente inferiori i tempi di computazione quando desidera ottimizzare sempre il tour. Si può osservare la correlazione tra i valori bassi riportati nella colonna 'tempo' e bassi nella colonna 'scheduling'.

13.3 Confronto tra VNS e ALNS

Abbiamo 60 istanze reali e 60 istanze euclidee, per la seguenti otto euristiche: ALNS, ALNS–A, VNS($\overline{\kappa}$) e VNS–A($\overline{\kappa}$) con $\overline{\kappa} = 1, 10, 50$. Se svolte sequenzialmente con un tempo limite di 2 ore si hanno 80 giorni di calcolo. Avendo impostato 8 thread per il solver di problemi MILP, possiamo avviare contemporaneamente più processi sino alla quantità massima di thread del elaboratore. In PC1 è stato impostato a due ore (7200 secondi) il massimo tempo di calcolo. Su PC2 abbiamo a disposizione meno tempo di calcolo che su PC1, pertanto è stata avviata una sequenza di un'ora (3600 secondi).

Al fine di confrontare due euristiche per un elemento della funzione obiettivo, ad esempio u, prendiamo un insieme d'istanze è contiamo per quante istanze un'euristica ha ricavato un migliore valore dell'altra euristica. In questo tipo di analisi, che chiameremo *quantitativa*, consideriamo anche il caso in cui le due euristiche raggiungano lo stesso valore. Un altro modo per osservare la bontà del metodo è quello di ricavare la media dei valori di tutte le sessanta

Heuristic	u	l	w	f	Moves	Time limit
ALNS	365.90	20.03	48.30	868.93	52927.73	1500 s
ALNS–A	374.58	25.78	57.03	970.70	17198.73	1500 s
ALNS	359.60	17.90	26.63	643.83	118024.55	3200 s
ALNS–A	353.44	21.77	38.25	757.70	39823.48	3200 s
ALNS	354.91	16.02	7.20	442.93	266992.95	7200 s
ALNS–A	346.95	18.70	16.68	532.48	96591.63	7200 s
VNS(10)	530.28	17.18	150.05	2047.96	—	7200 s
VNS–A(10)	514.31	14.93	138.22	1911.41	—	7200 s

Tabella 13.3: Media dei migliori risultati per VNS, VNS–A, ALNS e ALNS–A su sesanta istanze euclidee su PC1 con tempo limite di calcolo pari a 7200 secondi.

Heuristic	u	l	w	f	Moves	Time limit
ALNS	396.76	24.63	6.33	484.73	67774.82	1500 s
ALNS–A	395.03	30.12	13.03	555.48	24886.20	1500 s
ALNS	392.73	20.93	1.32	426.83	152068.43	3200 s
ALNS–A	390.31	25.58	7.25	488.40	53811.97	3200 s
ALNS	392.04	20.47	1.23	424.84	172703.97	3600 s
ALNS–A	389.59	24.85	6.67	481.11	60815.32	3600 s
VNS(1)	540.61	40.03	188.25	2463.14	—	3600 s
VNS–A(1)	506.78	32.03	137.35	1912.31	—	3600 s
VNS(10)	491.82	30.48	113.15	1653.81	—	3600 s
VNS–A(10)	460.65	24.97	73.33	1218.95	—	3600 s
VNS(50)	490.63	28.80	108.95	1608.93	—	3600 s
VNS–A(50)	462.52	24.65	78.57	1272.84	—	3600 s

Tabella 13.4: Media dei migliori risultati per VNS, VNS–A, ALNS e ALNS–A su sessanta istanze reali (non euclidee) su PC2 con tempo limite di calcolo pari a 3600 secondi.

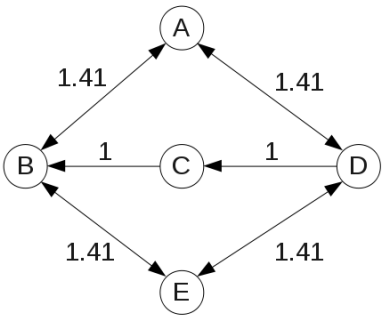
Heuristic	u	l	w	f	Moves	Time limit
ALNS	361.44	18.82	39.38	774.09	77094.67	1500 s
ALNS–A	360.98	23.92	49.10	875.90	25275.52	1500 s
ALNS	357.71	17.05	16.53	540.09	167472.02	3200 s
ALNS–A	349.27	20.60	29.15	661.37	57370.57	3200 s
ALNS	357.12	16.78	13.72	511.07	187833.70	3600 s
ALNS–A	348.60	20.10	24.70	615.70	65203.20	3600 s
VNS(1)	593.01	27.68	239.53	3016.03	—	3600 s
VNS–A(1)	572.60	21.80	213.15	2725.90	—	3600 s
VNS(10)	538.19	18.60	156.15	2118.29	—	3600 s
VNS–A(10)	519.75	15.98	142.08	1956.56	—	3600 s
VNS(50)	535.83	17.62	163.28	2186.28	—	3600 s
VNS–A(50)	532.42	15.68	158.88	2136.94	—	3600 s

Tabella 13.5: Media dei migliori risultati per VNS, VNS–A, ALNS e ALNS–A su sessanta istanze euclidee su PC2 con tempo limite di calcolo pari a 3600 secondi.

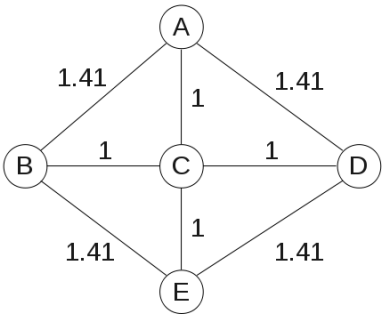
istanze. Questo tipo di analisi sarà chiamata del *valor medio*. In seguito analizziamo i dati raccolti in termini quantitativi nella tabella 13.6 ed in termini di valor medio nelle tabelle 13.2, 13.3, 13.4 e 13.5.

Nel dettaglio, la tabella 13.6 è divisa nella parte di quantificazione per le sessanta istanze reali e per le sessanta istanze euclidee. Per ogni parte le prime tre colonne si riferiscono alle componenti: carico medio di lavoro giornaliero u, fidelizzazione paziente-infermiera l e il tempo di lavoro straordinario w; il valore della funzione obiettivo f è nell'ultima colonna. Per ogni riga il nome nella prima colonna identifica l'euristica che ha ottenuto un migliore valore rispetto ad un altra euristica, cioè 'ALNS' o 'ALNS–A', oppure il nome identifica che entrambe le euristiche hanno lo stesso valore, per esempio con 'ALNS=ALNS–A'. In ogni cella si ha un numero di istanze in cui l'euristica ha dato un migliore valore, o stessi valori, di un'altra.

Nelle tabelle 13.2, 13.3, 13.4 e 13.5 in ogni riga ci sono i valori medi per l'euristica, il cui nome è nella colonna. Le successive tre colonne si riferiscono alle componenti u, l e w, seguite dal valore della funzione obiettivo f. Infine le ultime due colonne sono rispettivamente il numero delle mosse svolte ed il tempo di computazione. Nel caso delle varianti ALNS non viene ripetuta l'esecuzione per ogni tempo riportato, ovvero il valori del tempo di computazione sono quelli della ultima miglior soluzione ottenuta nei secondi precedenti al tempo indicato in colonna.



	A	B	C	D	E
A	∞	1.41	2.41	1.41	2.82
B	1.41	∞	3.82	2.82	1.41
C	2.41	1	∞	3.82	2.41
D	1.41	2	1	∞	1.41
E	2.82	1.41	2.41	1.41	∞



	A	B	C	D	E
A	∞	1.41	1	1.41	2
B	1.41	∞	1	2	1.41
C	1	1	∞	1	1
D	1.41	2	1	∞	1.41
E	2	1.41	1	1.41	∞

Figura 13.1: Il grafo in alto ha distanze non euclidee. Il grafo in basso ha distanze non euclidee.

13.3.1 ALNS e ALNS–A.

Consideriamo ora i valori medi per le istanze reali riportati nelle tabelle 13.2 e 13.4 delle istanze reali. Il valore l di fidelizzazione paziente-infermiera per ALNS è sempre migliore di quello dato da ALNS–A. C'è una correlazione tra basso valore di l e alto numero di mosse. Questa correlazione può essere spiegata dal fatto che un maggior numero di mosse permette di svolgere più `swap` e k`swap`, le quali ci aspettiamo migliorino la fidelizzazione paziente-infermiera.

Il valore del carico di lavoro medio giornaliero u è sempre migliore per ALNS–A rispetto ad ALNS. Questo può essere dovuto al fatto che in ALNS–A si ottimizza sempre il tour: ne segue che la media dei tempi di lavoro dei turni giornalieri è più bassa. Il valore medio del tempo straordinario e il valore medio per la funzione obiettivo sono migliori per ALNS.

Nelle istanze euclidee, vedi tabelle 13.3 e 13.5, sono più bassi i valori di carico di lavoro medio giornaliero u rispetto alle istanze reali. Una possibile spiegazione può essere data dalle caratteristiche delle matrici dei costi. Nelle istanze reali le distanze risentono dei sensi unici, come abbiamo visto in figura 10.3. In figura 13.1 riproponiamo schematicamente l'obbligo di passare per sensi unici. Si osserva che il percorso da B a C nella matrice delle distanze reali è $d_{BC} = 3.82$, mentre nel caso di distanze euclidee è $d_{BC} = 1$. Pertanto è plausibile che i valori di u relativi all'instradamento siano più bassi.

L'andamento osservato dal valor medio non è sempre confermato nell'analisi quantitativa riportata in tabella 13.6. Sono confermate le tendenze ad avere migliori valori di l per ALNS e ad avere migliori valori di u per ALNS–A. Sebbene il valore medio della funzione obiettivo f sia migliore per ALNS, quantitativamente ALNS–A nelle istanze reali e euclidee ottiene circa lo stesso numero di migliori valori di f rispetto a ALNS. ALNS e ALNS–A tendono entrambe

	reali				euclidee			
euristica	u	l	w	f	u	l	w	f
ALNS	9	**49**	10	*34*	0	**46**	13	23
ALNS = ALNS–A	2	6	*50*	0	0	3	*44*	2
ALNS–A	**49**	5	0	*26*	**60**	11	3	**35**

Tabella 13.6: In questa tabella vengono contate le istanze nelle quali l'euristica, eseguita su PC1 con 7200 secondi.

ad azzerare il tempo di lavoro straordinario.

13.3.2 VNS e VNS–A.

Gli algoritmi VNS sono stati eseguiti con parametro $Q = 0.5$ ed in tre diverse modalità, ottenute sostituendo l'algoritmo 6.4 `FirstImprovement` con l'algoritmo 6.3 $\overline{\kappa}$`Improvement` usando per $\overline{\kappa}$ i valori $1, 10, 50$.

L'attuale implementazione della VNS non restituisce soluzioni adeguate in tempi accettabili al variare di $\overline{\kappa}$ fra i tre valori dati, ossia il tempo di lavoro straordinario è troppo elevato.

Da un punto di vista di analisi del valor medio, rispetto a VNS, si osserva la tendenza per VNS–A ad avere migliori valori u, l e w. Mentre un analisi quantitativa tra tutte le possibili combinazioni delle varianti di VNS i valori tendono ad un bilanciamento di quanità non mostrando rilevanti dominanze, per esempio per un elemento talvolta abbiamo 24 valori migliori per VNS e 36 per VNS–A, viceversa per un altro elemento.

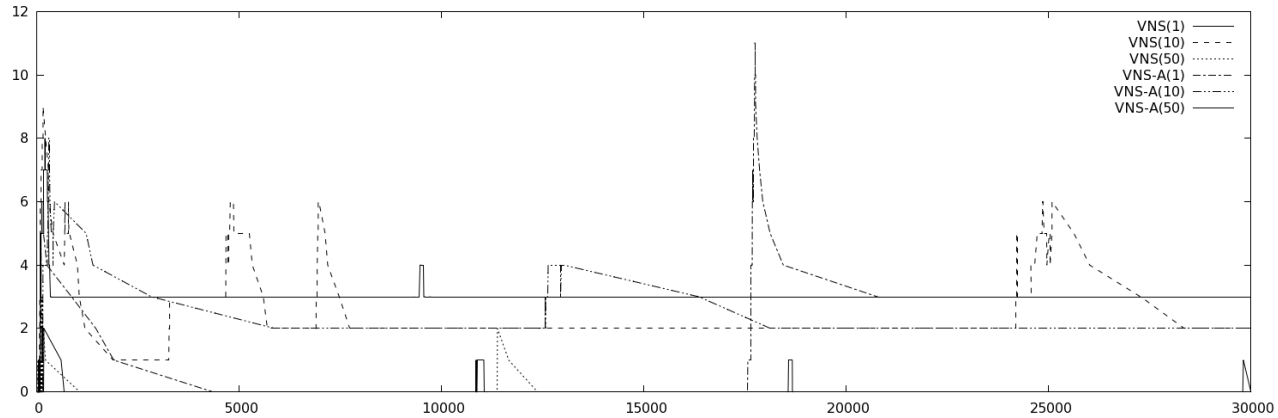


Figura 13.2: Il grafico presenta l'evoluzione del valore di fidelizzazione paziente-infermiera nelle esecuzioni di VNS e VNS-A con $\overline{\kappa} \in \{1, 10, 50\}$ per l'istanza m540_s13 euclidea con tempo limite 30000 secondi.

Prestando attenzione ai valori riportati per le singole istanze nelle tabelle in appendice (B.8, C.8, C.10 e C.12), è possibile osservare un fenomeno di azzeramento della fidelizzazione per le sole istanze euclidee con numero di richieste basso. Per queste istanze osserviamo lo storico delle migliori soluzioni calcolate dall'esecuzione della VNS e VNS–A: nei primi miglioramenti della

Heuristic	m540_s13	m540_s13 (euc)	m590_s17	m590_s17 (euc)
ALNS	365.0	336.8	**421**.4	390.8
ALNS–A	**359**.0	**328**.6	428.4	**385**.8
VNS(1)	404.0	3441.6	836.6	421.8
VNS–A(1)	387.0	2896.8	761.6	1591.4
VNS(10)	414.2	3186.4	707.0	421.2
VNS–A(10)	390.8	3436.8	644.4	419.4
VNS(50)	397.4	3464.4	1443.0	424.6
VNS–A(50)	390.4	3243.4	755.0	414.4

Tabella 13.7: Riportiamo i valori della funzione obiettivo ottenuti da un esecuzioni su PC2 con tempo limite di calcolo pari a 30000 secondi.

funzione obiettivo viene azzerato il valore di fidelizzazione, tale valore poi cresce lentamente (vedi figura 13.2) mentre diminuiscono il valori del tempo di lavoro straordinario ed del carico medio di lavoro giornaliero.

13.3.3 VNS e ALNS.

L'euristica ALNS fornisce sempre soluzioni adeguate in tempi accettabili, mentre l'attuale implementazione di VNS giunge sporadicamente a soluzioni adeguate. Una possibile spiegazione di questo può essere dovuta al fatto che, diversamente da quanto ci aspettavamo, la mossa `move` da sola non abbatte i tempi di lavoro straordinario; occorrono le mosse S3 e K2 per accorpare servizi vicini.

Il fenomeno osservato per VNS e VNS–A con $\overline{\kappa} = 50$, sembra mettere in luce la necessità di una fase di *shake* 'forte', ovvero ripartire da una nuova soluzione iniziale. La scelta di avere una fase di shake 'debole', ovvero di ripartire dalla miglior soluzione conosciuta, è dovuta al basso numero di mosse per secondo il quale non permette una veloce convergenza.

In un problema complesso come l'HHCP, la capacità dell'ALNS di tener memoria delle mosse di successo permette l'adattamento automatico della scelta degli intorni alle peculiarità delle singole istanza.

Ora confrontiamo ALNS e VNS con un ampio tempo di calcolo. In tabella 13.7 sono riportati i valori della funzione obiettivo dopo un esecuzione di 30000 secondi su PC2. Abbiamo scelto l'istanza m540_s13 e m590_s17, la prima ha il valore più basso di richieste $|R| = 719$ mentre la seconda il valore più alto $|R| = 848$. Per entrambe consideriamo le istanze con matrice dei costi sia reali che euclidea, quest'ultima denotata con '(euc)'.

Possiamo osservare che le euristiche ALNS giungono a migliori valori della funzione obiettivo rispetto alle varianti VNS. Si può anche osservare che l'euristica VNS in un tempo ampio tende a giungere a soluzione accettabili.

Nel caso delle istanze euclidee con un basso numero di richieste le euristiche VNS tendono a far crescere lentamente il valore l (vedi figura 13.2) mantenendo la soluzione lontana da una soluzione accettabile.

Una possibile motivazione del miglior andamento delle varianti ALNS può essere spiegato dal comportamento adattivo della scelta degli operatori. Il comportamento varia istanza per istanza. Prendiamo ad esempio il comportamento graficato nella figura 13.3, nella quale riportiamo i valori di $\varsigma_{\texttt{op}}$ dell'esecuzione di ALNS–A per l'istanza m590_s17. Abbiamo riportato tre grafici relativi ai gruppi di operazioni `move`, `swap` e k`swap`. Per le mosse K1 con $k = 2$, e $k = 3$, la denoteremo '*2-K1*', e '*3-K1*'; analogamente per K2.

Le mosse `move` per l'ALNS dopo un primo tempo smette di essere di successo e di essere usata. Rispetto all'andamento appena osservato, nella VNS le operazioni `move` sono sempre rieseguite dopo ogni successo di un'altra operazione, inoltre questo alto numero di `move` non abbassa il valore del tempo di lavoro straordinario w. Si osservi che la mossa M1, la quale sceglie casualmente i turni e i servizi, ha meno successo delle mosse specializzate M2 e M3.

Nell'immagine le mosse `swap` hanno un comportamento analogo alla `move`, ossia l'operazione S1, che sceglie casualmente i turni e i servizi, non ha successo. Invece, hanno periodi di successo le mosse S2 e S3, che scelgono in modo non casuale i tour o i servizi.

Le mosse k`swap` hanno sporadicamente Interessante il fatto che la mossa S3 ritorna di moda, intorno al secondo 25000, e negli istanti successivi si ha successo per la mossa 2-K1.

13.3.4 Influenza del generatore di numeri pseudo casuale

Abbiamo preso l'istanza m590_s17 ed abbiamo eseguito le euristiche con un seme di inizializzazione che varia tra 90 e 99. Le esecuzioni sono state svolte su PC2 con tempo limite di calcolo pari a 600 secondi. Riportiamo nella tabella 13.8 la sintesi dei dati mostrando il nome dell'euristica nella prima colonna, il minimo valore della funzione obiettivo nella seconda colonna 'min', la media dei dieci valori della funzione obiettivo nella terza colonna 'media', il massimo valore della funzione obiettivo nella quarta colonna 'max' e nell'ultima colonna la deviazione standard.

Possiamo osservare che al variare del seme d'inizializzazione si ripresenta il comportamento generale descritto in precedenza, ovvero che ALNS ottiene migliori soluzioni di VNS. Possiamo osservare che la deviazione standard è molto alta per il campione di esecuzioni delle varianti VNS. Questo, può essere spiegato dal fatto che la mossa M1, dal quale l'algoritmo parte, richiede una scelta casuale dei tour e del servizio. Si consideri anche che nel momento in cui le mosse successive restituiscono una miglir soluzione, l'algoritmo riparte usando il primo operatore, cioé M1. Osserviamo che il 20% delle mosse M1 ha portato ad un miglioramento sia per ALNS che per VNS, però il numero di mosse M1 richieste per gli algoritmi ALNS è un decimo delle mosse M1 degl'algoritmi VNS.

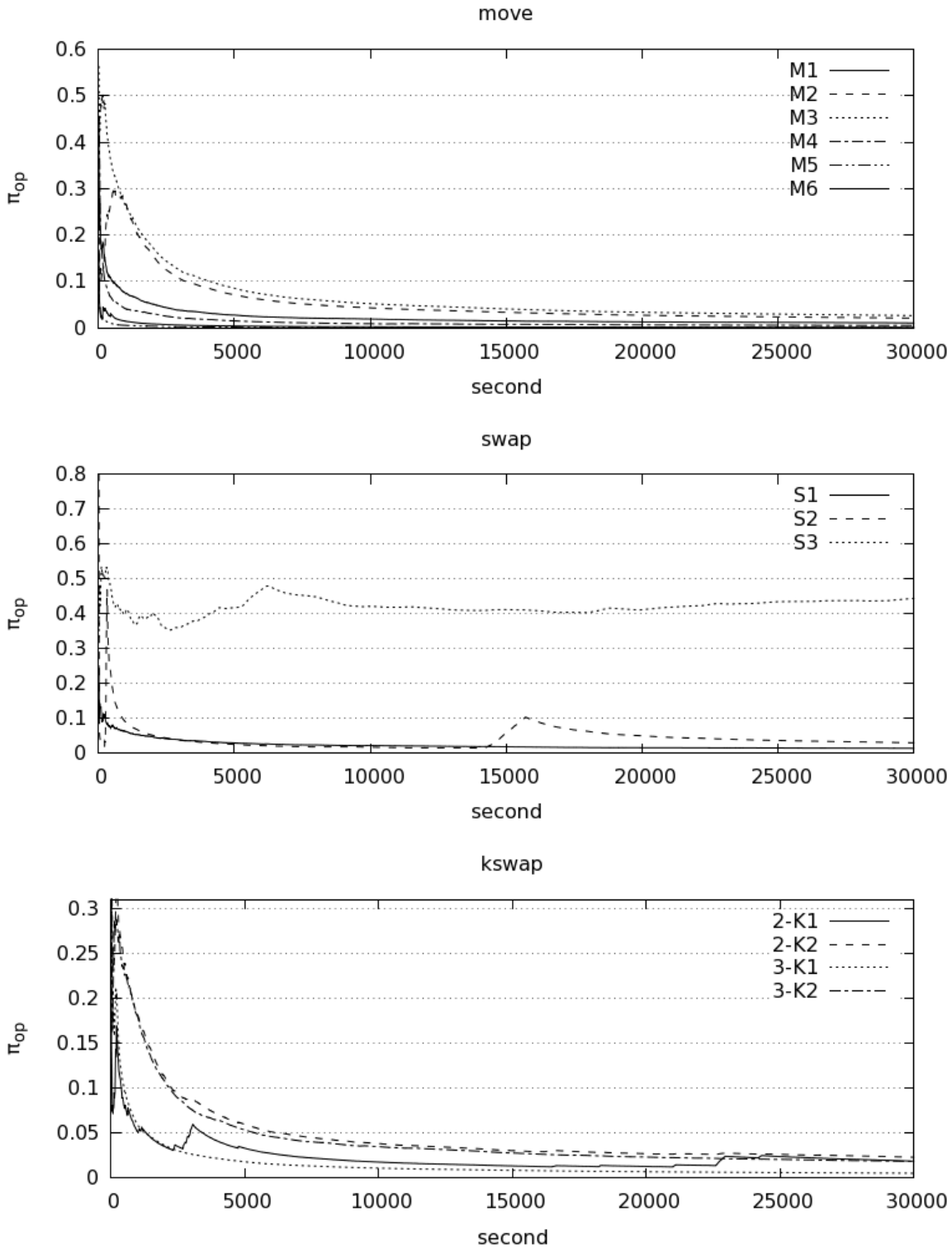


Figura 13.3: I tre grafici rappresentano il valore di ς_{op} raggruppando le operazioni `move`, `swap` *e* `kswap`. *I dati sono relativi all'esecuzione dell'euristica ALNS–A per l'istanza m590_s17 con tempo limite 30000 secondi.*

Heuristic	min	media	max	std. dev.
ALNS	589.80	657.20	804.00	66.96
ALNS–A	724.20	812.70	925.20	63.91
VNS(1)	1358.00	3438.00	5139.00	1131.53
VNS–A(1)	2021.00	3999.00	5037.00	871.87
VNS(10)	1074.00	2200.00	4049.00	1090.64
VNS–A(10)	754.40	3005.00	5115.00	1556.63
VNS(50)	1478.00	2866.00	4338.00	783.00
VNS–A(50)	2544.00	3232.00	4231.00	569.58

Tabella 13.8: Minimo, media, massimo e deviazione standard dei valori della funzione obiettivo in un campione di esecuzioni ripetute con diverso seme di inizializzazione.

13.4 Conclusioni

Abbiamo osservato gli elevati tempi di calcolo dell'euristica Clarke & Wright rispetto alla stessa euristica quando si omette la verifica delle parte di scheduling. Si è anche osservato come l'accorgimento di avere uno storico delle mosse riduca considerevolmente il tempo di calcolo.

L'analisi del valor medio ha mostrato che l'euristica VNS richiede un'ulteriore fase di messa a punto per poter giungere a soluzioni accettabili nei tempi celeri dell'euristica ALNS. Nel lungo periodo abbiamo osservato che l'euristica VNS giunge a soluzioni accettabili per le istanze realistiche. Tuttavia, l'euristica ALNS è risultata in ogni circostanza lo strumento ideale per poter affrontare il problema HHC.

Riepilogo della parte IV

In quest'ultima parte del lavoro si sono descritti gli aspetti computazionali relativi all'implementazione degli algoritmi proposti e i risultati della sperimentazione numerica su numerose istanze verosimili di HHCP.

Nel capitolo 10 l'analisi di un insieme di dati storici di richieste di servizi, tratti dalla realtà, ha permesso di realizzare un generatore di istanze, descritto nel capitolo 11. Utilizzando tale generatore sono state create 60 istanze realistiche e 60 istanze euclidee, ciascuna delle quali con numerose centinaia di servizi.

Su questo ampio insieme di problemi test si sono provati gli algoritmi proposti ed analizzati nella parte III, configurati con i valori dei parametri scelti in accordo con le considerazioni del capitolo 12.

I risultati di quest'indagine sperimentale dimostrano che l'algoritmo ALNS è l'unico strumento attualmente disponibile in grado di fornire soluzioni ammissibili di HHCP in tempi accettabili, anche su istanze verosimili di dimensioni grandi quanto quelle dei problemi HHC reali.

Conclusioni generali e sviluppi futuri

In questo lavoro è stato affrontato il problema di pianificazione di prestazioni sanitarie a domicilio in essere presso l'AUSL di Ferrara. Si hanno due obiettivi: si desidera bilanciare il carico di lavoro e gestire la fidelizzazione paziente-infermiera. Abbiamo visto che l'eterogeneità dei dati mette in crisi molte formulazioni che modellano il bilanciamento di carico. La nostra analisi dello storico delle richieste reali ha mostrato l'eterogeneità del numero delle richieste giornaliere, dei tempi di servizio e dei tempi di viaggio. Pertanto, è risultato idoneo, per cogliere le esigenze reali, modellare il bilanciamento di carico con una funzione min-max, sebbene sia difficile da affrontare dal punto di vista dell'ottimizzazione.

Rispetto ai sottoproblemi per i quali le euristiche propongono operatori che danno soluzioni ammissibili, o con celere verifica di ammissibilità, l'HHCP risente della pesantezza della verifica di tutti i vincoli che descrivono le esigenze reali. La sinergia tra la programmazione matematica e l'approccio euristico, oltre ad aver reso possibile la verifica di ammissibilità in tempi accettabili, ha motivato un'analisi a largo spettro sui metodi e sui problemi.

È stato esplorata la possibilità di usare la decomposizione di Benders e la generazione di colonne. In analogia ad altri problemi complessi, l'arricchimento dei vincoli e la presenza di più obiettivi rendono difficile l'individuazione di sottoproblemi, o la loro soluzione in tempi accettabili. È stata implementata la trasformazione da PCTSP a TSP: essa ha messo in luce nel software Concorde fenomeni d'instabilità numerica dovuti alla matrice dei costi. La soluzione con solver MILP del PCTSP, e del TSP derivato dal PCTSP, si è mostrata troppo lenta per poter usare il PCTSP come sottoproblema di pricing nella CG.

La messa in essere di approcci euristici si è scontrata con la difficoltà di avere un numero di operazioni al secondo sufficiente a permettere ad un'euristica di giungere a soluzioni ammissibili in tempi accettabili. Avendo un problema con più obiettivi, è stato necessario progettare intorni che indirizzassero la ricerca tenendo conto dei diversi obiettivi del problema, sia singolarmente, che congiuntamente.

La definizioni di intorni *ad hoc* per l'HHCP, la verifica di applicabilità di un operatore e di verifica di ammissibilità della soluzione ottenuta dall'applicazione di un operatore hanno consentito di mettere a punto euristiche quali VNS e ALNS. L'astrazione del concetto di operatore ha permesso di definire un operatore di fusione, con il quale siamo stati in grado di estendere l'approccio di Clarke & Wright all'HHCP.

Sono state generate istanze simulate dopo aver analizzato i dati reali: complessivamente si sono utilizzate sessanta istanze con matrice dei tempi reali di viaggio e sessanta istanze con tempi di viaggio verosimili con distanza euclidea. Si è osservato che l'abbattimento dei tempi

di lavoro straordinario è più semplice nelle istanze euclidee che nelle istanze reali.

Per le istanze prese in esame, l'ALNS si è mostrata un'euristica adeguata, riuscendo a calcolare una soluzione ammissibile in tempi accettabili. La variante ALNS–A ha dato migliori risultati per i singoli obiettivi relativi all'instradamento, mentre l'aspetto della fidelizzazione paziente-infermiera è ben gestito dall'ALNS.

Con questo lavoro si è colta la sfida di mantenere nel modello astratto i diversi aspetti del problema reale, ottenendo risultati accettabili mediante l'uso congiunto di metodi euristici e di programmazione matematica. La ricerca può proseguire focalizzandosi su più aspetti:

- una diversa scelta della sequenza di intorni, per la VNS;
- il comportamento di euristiche note, usando gli operatori progettati;
- la progettazione degli operatori come frutto di una maggiore indagine;
- la ricerca di una migliore soluzione di partenza;
- la messa a punto di migliori formulazioni dei sottoproblemi usati nella verifica di ammissibilità, per esempio diverse formulazioni del TSP;
- l'utilizzo di diverse funzioni obiettivo (non lineari) per il bilanciamento di carico nell'ambito della VNS e dell'ALNS;
- l'impiego di un'analisi multi-criterio;
- l'uso parallelo e cooperativo di differenti euristiche.

Appendice A

Istanze

Tabella A.1: Istanze.

$\mathcal{I}$			f_s							e_s							nome
R	P	S	1	2	3	4	5	6	7	−1	0	1	2	3	4	5	
719	228	540	310	67	18	5	3	1	0	309	0	32	24	23	16	0	m540_s13
731	219	540	280	54	24	6	2	0	0	279	1	34	20	18	14	0	m540_s19
732	220	540	325	63	19	6	4	3	0	324	0	38	24	22	12	0	m540_s11
733	229	550	313	67	20	3	3	1	0	312	0	31	26	22	16	0	m550_s13
737	219	540	305	52	21	7	3	2	2	304	2	35	21	19	11	0	m540_s18
737	223	540	316	60	21	6	3	1	0	315	0	32	30	19	11	0	m540_s14
740	218	540	300	63	23	7	3	2	1	299	0	43	29	18	10	0	m540_s15
742	215	540	299	62	19	5	6	3	0	298	0	36	33	19	8	0	m540_s16
742	219	550	283	55	24	6	2	0	0	282	1	34	21	18	14	0	m550_s19
743	238	540	311	55	27	7	8	1	0	310	2	44	31	14	8	0	m540_s10
747	228	560	313	71	19	4	3	1	0	312	0	31	28	24	16	0	m560_s13
747	237	540	299	71	23	10	6	1	0	298	3	40	35	21	11	2	m540_s17
750	225	550	321	61	20	6	3	1	0	320	0	31	31	19	11	0	m550_s14
751	222	550	330	62	19	6	4	3	0	329	0	38	24	22	11	0	m550_s11
752	218	550	304	61	23	6	3	2	1	303	0	41	27	20	9	0	m550_s15
753	222	560	290	55	24	6	2	0	0	289	1	34	20	18	15	0	m560_s19
754	215	550	303	64	19	5	6	3	0	302	0	37	34	19	8	0	m550_s16
757	241	550	307	70	23	8	5	1	0	306	2	38	35	21	10	2	m550_s17
762	218	560	297	69	15	4	8	2	0	296	1	30	28	23	15	2	m560_s15
763	228	560	325	62	21	6	3	1	0	324	0	32	31	19	12	0	m560_s14
763	231	570	317	71	21	6	3	1	0	316	1	34	30	23	15	0	m570_s13
765	216	560	309	64	19	5	6	3	0	308	0	37	34	19	8	0	m560_s16
765	223	560	336	64	19	6	4	3	0	335	0	38	25	22	12	0	m560_s11
771	221	570	304	71	16	4	7	2	0	303	1	30	28	25	15	2	m570_s15
771	224	540	303	65	27	8	8	0	0	302	0	45	26	21	17	0	m540_s12
773	217	560	306	58	20	2	6	3	1	305	1	33	26	17	14	0	m560_s17
775	222	570	295	56	26	6	2	0	0	294	1	36	20	19	15	0	m570_s19
776	221	560	303	52	19	7	4	3	2	302	1	36	21	19	11	0	m560_s18
777	216	570	311	64	19	5	6	2	0	310	0	36	34	19	8	0	m570_s16
778	224	570	337	64	19	5	4	3	0	336	0	37	25	22	12	0	m570_s11
781	229	570	326	65	21	6	3	1	0	325	0	32	34	19	12	0	m570_s14
783	218	550	312	57	28	8	4	5	0	311	1	51	25	15	9	2	m550_s18
785	226	550	305	65	27	8	8	0	0	304	0	45	26	21	17	0	m550_s12
$\mathcal{I}$			f_s							e_s							nome

Tabella A.1: Istanze.

$\mathcal{I}$			f_s							e_s							nome
R	P	S	1	2	3	4	5	6	7	−1	0	1	2	3	4	5	
787	225	580	305	71	17	4	7	2	0	304	1	31	28	25	15	2	m580_s15
788	220	570	311	59	20	2	7	3	1	310	1	34	27	17	14	0	m570_s17
788	220	580	317	64	19	5	6	2	0	316	0	36	34	19	8	0	m580_s16
789	222	580	294	56	26	7	2	0	0	293	1	37	20	18	16	0	m580_s19
789	227	580	343	65	19	4	4	3	0	342	0	36	25	23	12	0	m580_s11
789	240	560	303	53	31	9	10	1	0	302	2	50	30	13	10	0	m560_s10
790	225	570	308	54	19	7	4	3	2	307	1	36	21	20	12	0	m570_s18
790	236	580	324	69	25	7	3	1	1	323	1	39	29	25	13	0	m580_s13
794	232	580	332	67	21	6	3	1	0	331	0	32	35	20	12	0	m580_s14
795	220	550	310	69	24	12	4	0	0	309	1	42	29	24	14	0	m550_s10
797	229	590	314	71	18	4	6	2	0	313	1	31	28	25	15	2	m590_s15
800	228	560	308	64	28	8	8	0	0	307	0	46	26	20	17	0	m560_s12
800	228	590	350	66	19	4	4	3	0	349	0	36	26	23	12	0	m590_s11
801	241	570	308	52	31	9	10	1	0	307	2	51	29	12	10	0	m570_s10
803	224	590	300	60	26	7	2	0	0	299	1	37	24	18	16	0	m590_s19
804	223	590	319	65	19	5	6	2	0	318	0	36	34	19	9	0	m590_s16
806	223	580	317	60	21	2	7	4	1	316	1	36	27	18	14	0	m580_s17
807	234	590	340	68	22	6	3	1	0	339	0	33	36	20	12	0	m590_s14
808	236	590	326	70	27	7	3	1	1	325	1	40	30	25	14	0	m590_s13
811	227	580	311	55	20	8	5	3	2	310	1	39	21	21	12	0	m580_s18
812	231	570	308	64	29	8	8	0	0	307	0	47	26	20	17	0	m570_s12
814	225	580	328	59	21	15	1	0	0	327	0	39	25	21	12	0	m580_s10
822	229	590	316	56	20	8	5	3	2	315	1	39	22	21	12	0	m590_s18
825	232	580	314	64	30	8	8	0	0	313	0	48	26	20	17	0	m580_s12
834	223	590	329	58	20	15	1	0	0	328	0	38	23	22	12	0	m590_s10
841	233	590	316	64	30	8	8	0	0	315	0	48	25	21	17	0	m590_s12
848	228	590	290	77	29	11	7	2	0	289	1	52	37	20	16	1	m590_s17
$\mathcal{I}$			f_s							e_s							nome

Appendice B

Computazioni su PC1

		ALNS					ALNS–A				
R	P	u	l	w	f	step	u	l	w	f	step
719	228	371.4	20	0	391.4	69008	364.8	28	0	392.8	24343
731	219	383.6	19	0	402.6	85540	380.6	31	0	411.6	25892
732	220	364.2	21	0	385.2	90770	363.6	30	0	393.6	21340
733	229	384	21	0	405	58751	377.2	27	0	404.2	21819
737	219	378.6	25	0	403.6	66703	371	25	0	396	23253
737	223	378.8	25	0	403.8	76817	377.8	28	0	405.8	22201
740	218	370.8	17	0	387.8	78677	367.4	28	0	395.4	21200
742	215	372.4	26	0	398.4	76712	368.2	28	0	396.2	22524
742	219	391.8	24	0	415.8	59329	385.8	28	0	413.8	27852
743	238	372.8	24	0	396.8	62853	375.6	30	0	405.6	18357
747	228	381.8	22	0	403.8	60600	380.6	30	0	410.6	21640
747	237	381.4	26	0	407.4	46274	380.2	29	0	409.2	19079
750	225	386.8	19	0	405.8	69828	385	30	0	415	22988
751	222	375	23	0	398	88944	373.8	31	0	404.8	20486
752	218	378.2	20	0	398.2	75498	378.6	28	0	406.6	19643
753	222	390.2	21	0	411.2	62449	389.2	26	0	415.2	21375
754	215	375	26	0	401	68374	375.8	26	0	401.8	20678
757	241	377	28	0	405	64117	378.6	36	0	414.6	22062
762	218	393.4	23	0	416.4	51462	396.6	28	16	584.6	19062
763	228	392	25	0	417	51891	386.4	33	0	419.4	21006
763	231	407.2	19	0	426.2	56625	410	41	20	651	23647
765	216	385.2	20	0	405.2	66174	383.8	28	0	411.8	19848
765	223	383.8	19	0	402.8	55772	381.6	34	0	415.6	16937
771	221	395.6	24	0	419.6	50740	397.2	36	3	463.2	20077
771	224	407.6	26	62	1053.6	49437	402.4	34	13	566.4	18361
773	217	388.8	26	0	414.8	58762	385.4	24	0	409.4	23993
775	222	402.4	29	20	631.4	47619	399.2	27	0	426.2	19611
776	221	403.4	25	0	428.4	45260	405.2	30	23	665.2	18611
777	216	390.6	35	0	425.6	54019	388.8	33	7	491.8	21553
778	224	391.6	22	0	413.6	64985	385.6	34	0	419.6	16818
781	229	401.2	29	13	560.2	44243	396.8	31	0	427.8	16520
783	218	393.4	23	0	416.4	50122	390.4	31	0	421.4	17007
785	226	409.6	29	40	838.6	55578	409.8	31	63	1070.8	19118
787	225	406.8	23	0	429.8	45712	407.2	32	5	489.2	20520
788	220	392.4	25	0	417.4	49945	392.4	31	0	423.4	19816
788	220	395	20	0	415	55012	395.4	24	0	419.4	22182
789	222	410.8	28	31	748.8	47955	405.2	32	58	1017.2	19913
789	227	391	19	0	410	66886	387.8	29	0	416.8	17845
789	240	398	32	0	430	50560	402	30	3	462	19723
790	225	421.2	34	28	735.2	36117	416.4	41	32	777.4	18840
790	236	412.4	26	12	558.4	44039	413	39	31	762	18398
794	232	402.8	24	0	426.8	44776	400.4	36	0	436.4	17031
795	220	406.8	27	0	433.8	52300	405.4	28	0	433.4	21813
797	229	402.8	19	0	421.8	48115	401.8	30	0	431.8	21945
800	228	394.6	28	0	422.6	52813	396.2	34	0	430.2	20515
800	228	409.6	26	0	435.6	45110	412.4	33	16	605.4	19652
801	241	403.4	28	0	431.4	54413	405	33	17	608	18935
803	224	418.2	33	46	911.2	38828	416.8	35	63	1081.8	16760
804	223	416.2	28	19	634.2	33870	409.4	30	34	779.4	16878
806	223	402.4	30	0	432.4	44968	404	31	0	435	20403
807	234	404.6	30	0	434.6	43032	405	35	0	440	16113
808	236	423	29	12	572	38532	430.6	33	47	933.6	17545
811	227	435.2	29	60	1064.2	32852	421.8	37	50	958.8	19268
812	231	417.2	25	43	872.2	52529	415.6	33	54	988.6	22909
814	225	406.2	32	0	438.2	46812	410	24	0	434	18713
822	229	435.8	27	30	762.8	31722	432	34	60	1066	18049
825	232	423	36	32	779	36694	428.6	35	68	1143.6	15038
834	223	426.4	25	10	551.4	36466	430.6	33	47	933.6	16947
841	233	429.8	38	40	867.8	40489	429.4	28	86	1317.4	17568
848	228	427	30	11	567	36135	432.4	38	27	740.4	16821
mean		397.87	25.53	8.48	508.24	54510.25	396.66	31.20	14.05	568.36	19984.02

Tabella B.1: ALNS e ALNS–A dopo 1500 secondi in un grafo non euclideo su PC1.

		ALNS					ALNS–A				
R	P	u	l	w	f	step	u	l	w	f	step
719	228	358.4	15	136	1733.4	32949	348.4	19	124	1607.4	11619
731	219	351.8	12	115	1513.8	33498	342.6	17	58	939.6	16685
732	220	346.8	16	59	952.8	39347	326	19	111	1455	18775
733	229	356	17	97	1343	41318	416.4	28	98	1424.4	11306
737	219	364.4	20	106	1444.4	43330	374.4	17	210	2491.4	13282
737	223	359.6	12	130	1671.6	41016	366.2	28	142	1814.2	12129
740	218	356	14	78	1150	34742	528.8	6	247	3004.8	8185
742	215	354	14	60	968	38733	342.4	17	72	1079.4	13069
742	219	353.2	16	131	1679.2	42855	350.4	22	140	1772.4	15109
743	238	371	17	159	1978	26200	429.8	17	120	1646.8	13205
747	228	344.2	21	0	365.2	80689	339.6	26	0	365.6	19576
747	237	344.2	21	63	995.2	53036	346.2	22	159	1958.2	16599
750	225	388	18	187	2276	21512	445.8	22	179	2257.8	9176
751	222	352.4	15	109	1457.4	42817	452.6	21	150	1973.6	10056
752	218	384.2	16	218	2580.2	33801	418.6	16	215	2584.6	11778
753	222	367.2	17	148	1864.2	34734	349.6	23	114	1512.6	14084
754	215	349.8	16	77	1135.8	42148	355.8	26	47	851.8	11883
757	241	357.2	15	130	1672.2	42685	347.6	19	126	1626.6	16454
762	218	350.2	16	0	366.2	62979	348.2	27	0	375.2	22066
763	228	384.2	17	189	2291.2	25775	465	16	193	2411	8917
763	231	373.6	24	0	397.6	85615	368.2	22	0	390.2	26858
765	216	366.6	18	90	1284.6	37694	363	25	93	1318	12649
765	223	367.4	16	120	1583.4	35499	354	19	139	1763	15027
771	221	357.6	25	0	382.6	70141	349.8	25	0	374.8	23120
771	224	365.8	23	0	388.8	62228	360.4	34	0	394.4	19644
773	217	361	14	110	1475	49685	348	14	55	912	21410
775	222	357	21	0	378	70407	355.4	25	0	380.4	17892
776	221	386	20	205	2456	30855	452	17	223	2699	11774
777	216	349.8	21	0	370.8	63037	348.2	32	0	380.2	19493
778	224	341.2	22	0	363.2	69104	343	35	0	378	16951
781	229	358.6	26	0	384.6	61965	355.4	34	0	389.4	19682
783	218	365.8	18	89	1273.8	36942	492.2	13	196	2465.2	8817
785	226	366.2	20	0	386.2	57725	362.2	27	0	389.2	22466
787	225	363.2	20	0	383.2	63389	364.4	35	0	399.4	17443
788	220	359.2	16	0	375.2	69573	350.2	23	0	373.2	18360
788	220	365.8	15	92	1300.8	50257	351.8	21	137	1742.8	17472
789	222	365.4	19	0	384.4	84757	358.8	31	0	389.8	21529
789	227	355.6	26	0	381.6	49942	352.2	31	0	383.2	16871
789	240	365	23	0	388	67952	362.8	31	0	393.8	19778
790	225	376	31	0	407	44532	369	29	0	398	16783
790	236	361.4	27	0	388.4	59821	359.6	31	0	390.6	20608
794	232	359.8	16	0	375.8	58356	359	39	0	398	17076
795	220	360.8	16	0	376.8	91170	360.8	23	0	383.8	23608
797	229	366	24	0	390	62798	363.4	29	0	392.4	21089
800	228	362.6	20	0	382.6	63095	355.8	28	0	383.8	19386
800	228	375	26	0	401	51587	367.4	30	0	397.4	23480
801	241	374.8	26	0	400.8	60716	367	32	0	399	21997
803	224	369.8	24	0	393.8	57289	363.2	29	0	392.2	18898
804	223	363.2	25	0	388.2	55402	355	28	0	383	19322
806	223	357.6	21	0	378.6	101311	358	24	0	382	27927
807	234	365.6	21	0	386.6	53728	359.8	35	0	394.8	16350
808	236	375.4	17	0	392.4	55749	373.2	39	0	412.2	20057
811	227	389.2	26	0	415.2	39290	375.4	35	17	580.4	15145
812	231	405	22	0	427	77850	402.8	29	2	451.8	27064
814	225	367.6	22	0	389.6	62716	368.4	27	0	395.4	19733
822	229	390.6	24	0	414.6	48362	384.4	34	28	698.4	14503
825	232	384	25	0	409	49353	383.2	33	27	686.2	15487
834	223	387.6	23	0	410.6	62108	389.2	29	0	418.2	17499
841	233	389.2	28	0	417.2	49206	383.2	31	0	414.2	17848
848	228	389	26	0	415	40294	390.8	31	0	421.8	16875
mean		365.90	20.03	48.30	868.93	52927.73	374.58	25.78	57.03	970.70	17198.73

Tabella B.2: ALNS e ALNS–A dopo 1500 secondi in un grafo euclideo su PC1.

		ALNS					ALNS–A				
R	P	u	l	w	f	step	u	l	w	f	step
719	228	367.8	19	0	386.8	144723	362	24	0	386	53969
731	219	381.6	16	0	397.6	175136	375.8	23	0	398.8	57402
732	220	363.2	18	0	381.2	216536	357.8	24	0	381.8	50114
733	229	379.6	19	0	398.6	156374	371.8	23	0	394.8	48501
737	219	372.8	25	0	397.8	127996	366.4	23	0	389.4	54088
737	223	378.4	24	0	402.4	156429	373.2	23	0	396.2	49637
740	218	366.6	14	0	380.6	173263	364.8	22	0	386.8	42943
742	215	365.8	21	0	386.8	163143	363.2	21	0	384.2	47893
742	219	385.4	22	0	407.4	135159	382.6	24	0	406.6	58443
743	238	370.8	24	0	394.8	128045	366.2	19	0	385.2	40843
747	228	380	20	0	400	129247	374	28	0	402	48715
747	237	376.4	26	0	402.4	131453	374.4	25	0	399.4	46499
750	225	382.6	18	0	400.6	145479	381	26	0	407	48397
751	222	372	19	0	391	219928	368.6	28	0	396.6	46541
752	218	374.6	19	0	393.6	186016	373.8	24	0	397.8	44793
753	222	386	18	0	404	153618	384.8	23	0	407.8	44888
754	215	371.2	21	0	392.2	153997	371.8	20	0	391.8	44957
757	241	376.2	26	0	402.2	138570	374.2	29	0	403.2	49210
762	218	390.8	21	0	411.8	99981	385.4	17	0	402.4	45023
763	228	389.2	20	0	409.2	113718	382.2	27	0	409.2	44950
763	231	403	15	0	418	126217	402.6	31	0	433.6	55568
765	216	380	18	0	398	166768	378	23	0	401	44588
765	223	380.4	17	0	397.4	131697	375.2	27	0	402.2	39095
771	221	392.2	17	0	409.2	116782	392.6	31	0	423.6	44235
771	224	410.8	21	13	561.8	111258	395	28	0	423	41074
773	217	385.8	23	0	408.8	133036	378.6	21	0	399.6	54861
775	222	395	21	0	416	109704	392.4	26	0	418.4	42250
776	221	399	21	0	420	112404	395	31	8	506	41415
777	216	385.6	29	0	414.6	129715	382.2	29	0	411.2	47065
778	224	385.4	19	0	404.4	166308	379.4	30	0	409.4	41503
781	229	394.2	21	0	415.2	99482	388.4	25	0	413.4	38565
783	218	390.6	23	0	413.6	102742	386.6	26	0	412.6	39028
785	226	408	23	28	711	122590	403.8	25	56	988.8	43297
787	225	400.2	20	0	420.2	112934	402	26	0	428	46062
788	220	390.6	21	0	411.6	99118	389	27	0	416	41676
788	220	390.8	18	0	408.8	120964	388.8	23	0	411.8	49589
789	222	407.6	26	0	433.6	105091	401.4	29	43	860.4	42740
789	227	387.4	16	0	403.4	175884	383.4	27	0	410.4	40943
789	240	396.8	24	0	420.8	115077	397.4	26	0	423.4	41277
790	225	413.6	27	4	480.6	80262	410.2	32	23	672.2	44566
790	236	406.4	21	0	427.4	95413	407	36	9	533	41232
794	232	397	20	0	417	99010	393.6	27	0	420.6	37749
795	220	401.6	20	0	421.6	103461	400	24	0	424	46336
797	229	399.8	18	0	417.8	105470	395.6	28	0	423.6	49035
800	228	389.8	22	0	411.8	128771	389.8	25	0	414.8	44560
800	228	405.4	22	0	427.4	100254	407.6	29	1	446.6	41398
801	241	400.8	25	0	425.8	126586	400	32	17	602	39991
803	224	414	28	21	652	92456	411.8	31	49	932.8	38862
804	223	408.8	26	0	434.8	75333	407.4	27	11	544.4	36344
806	223	398	29	0	427	118934	397.4	26	0	423.4	45778
807	234	402	25	0	427	99929	400.6	29	0	429.6	36820
808	236	416.2	25	0	441.2	83823	420.2	27	20	647.2	38514
811	227	425.4	25	44	890.4	83314	420.6	34	42	874.6	33330
812	231	414	25	7	509	108579	408	30	44	878	51163
814	225	402.8	19	0	421.8	100655	403.8	22	0	425.8	42182
822	229	428.4	23	22	671.4	67720	427.2	29	49	946.2	35843
825	232	418.4	31	14	589.4	76552	416	29	52	965	32691
834	223	418	20	0	438	77857	425.4	27	26	712.4	36197
841	233	430.4	27	20	657.4	81917	424.4	22	60	1046.4	39279
848	228	418.6	26	0	444.6	76067	424.6	30	11	564.6	37166
mean		393.73	21.78	2.88	444.35	123149.08	390.95	26.33	8.68	504.12	44194.55

Tabella B.3: ALNS e ALNS–A dopo 3200 secondi in un grafo non euclideo su PC1.

		ALNS					ALNS–A				
R	P	u	l	w	f	step	u	l	w	f	step
719	228	342.2	16	72	1078.2	86438	329.2	13	92	1262.2	32036
731	219	344.4	12	63	986.4	83946	334	19	24	593	41189
732	220	339	14	28	633	100107	325	17	75	1092	42928
733	229	350.6	16	17	536.6	102492	331.6	17	64	988.6	38347
737	219	358.6	12	58	950.6	92771	344.2	17	163	1991.2	34049
737	223	343.4	12	108	1435.4	89746	331.4	18	134	1689.4	34975
740	218	336.8	15	68	1031.8	92844	439.8	9	216	2608.8	21407
742	215	346.4	13	12	479.4	97123	331.2	15	27	616.2	34083
742	219	355.4	13	82	1188.4	87511	334.6	18	118	1532.6	35155
743	238	360.2	19	92	1299.2	68182	332.2	14	95	1296.2	35217
747	228	339.8	17	0	356.8	191016	334	23	0	357	46102
747	237	336.8	17	38	733.8	113894	335.6	19	110	1454.6	38769
750	225	358.6	15	119	1563.6	59294	347.6	20	91	1277.6	29157
751	222	355.2	11	51	876.2	104314	339	16	71	1065	33251
752	218	348	16	83	1194	87854	334.6	15	185	2199.6	30256
753	222	356.2	14	99	1360.2	87678	342.6	17	105	1409.6	36048
754	215	350	16	6	426	102683	335	19	0	354	35170
757	241	345.8	13	123	1588.8	111647	337.6	17	87	1224.6	40327
762	218	346.4	15	0	361.4	146224	342.4	23	0	365.4	45646
763	228	364.8	15	114	1519.8	69357	364.6	14	133	1708.6	24887
763	231	369.4	19	0	388.4	201355	364.6	21	0	385.6	62778
765	216	358	17	31	685	78566	340.4	18	48	838.4	31489
765	223	360.6	15	34	715.6	80567	339	14	105	1403	34970
771	221	351.2	21	0	372.2	154072	347.4	23	0	370.4	49858
771	224	361.8	19	0	380.8	137648	356	28	0	384	44324
773	217	363.2	18	76	1141.2	113673	343	12	27	625	52340
775	222	353	18	0	371	149292	347.4	23	0	370.4	43046
776	221	381.8	18	162	2019.8	72940	370.2	18	164	2028.2	29975
777	216	345.8	20	0	365.8	125809	342.2	28	0	370.2	41462
778	224	339.4	20	0	359.4	134581	338.2	25	0	363.2	37765
781	229	353.8	20	0	373.8	145515	351.8	28	0	379.8	44950
783	218	367.4	21	0	388.4	77342	353.6	21	67	1044.6	25250
785	226	362.2	18	0	380.2	130239	354.8	25	0	379.8	48918
787	225	357.8	16	0	373.8	125196	358.2	29	0	387.2	38631
788	220	351.2	15	62	986.2	108374	349.2	16	85	1215.2	43132
788	220	352.2	15	0	367.2	151944	347.6	19	0	366.6	39947
789	222	364.4	16	0	380.4	179060	355.8	24	0	379.8	48181
789	227	352.6	24	0	376.6	125225	345.6	27	0	372.6	35964
789	240	359.6	20	0	379.6	149084	357.8	25	0	382.8	42535
790	225	368.8	26	0	394.8	104459	362.2	26	0	388.2	40085
790	236	359.8	23	0	382.8	122820	354.4	27	0	381.4	43204
794	232	358.6	15	0	373.6	127819	352.6	29	0	381.6	36611
795	220	359	15	0	374	207805	358	20	0	378	49526
797	229	362.4	23	0	385.4	144378	359	25	0	384	47483
800	228	357.8	20	0	377.8	158145	351.8	26	0	377.8	43968
800	228	371.2	24	0	395.2	102264	363	26	0	389	46403
801	241	371.8	21	0	392.8	129770	363.2	23	0	386.2	47771
803	224	363.8	22	0	385.8	117360	358.8	24	0	382.8	43292
804	223	358	24	0	382	106294	350.8	28	0	378.8	41226
806	223	355.4	20	0	375.4	215468	353.4	19	0	372.4	53543
807	234	361	19	0	380	117026	356.4	32	0	388.4	34499
808	236	372.2	16	0	388.2	121795	368.8	30	0	398.8	42216
811	227	384	22	0	406	81802	381	28	2	429	33212
812	231	403.2	19	0	422.2	157709	399.8	26	0	425.8	57989
814	225	362	21	0	383	133595	363	18	0	381	41442
822	229	387.2	21	0	408.2	91621	381.6	29	7	480.6	35864
825	232	379.6	18	0	397.6	105922	378	26	0	404	33761
834	223	382.8	16	0	398.8	124516	381.6	26	0	407.6	36464
841	233	386.8	25	0	411.8	99629	379.4	27	0	406.4	38779
848	228	386.4	23	0	409.4	95673	380.4	27	0	407.4	37557
mean		359.60	17.90	26.63	643.83	118024.55	353.44	21.77	38.25	757.70	39823.48

Tabella B.4: ALNS e ALNS–A dopo 3200 secondi in un grafo euclideo su PC1.

		ALNS					ALNS–A				
R	P	u	l	w	f	step	u	l	w	f	step
719	228	363.2	14	0	377.2	342554	354.6	17	0	371.6	129664
731	219	380	13	0	393	379796	372.8	23	0	395.8	128150
732	220	357.4	17	0	374.4	587888	351	20	0	371	121476
733	229	377.4	12	0	389.4	365588	369.2	21	0	390.2	110591
737	219	370.8	20	0	390.8	296920	362.2	23	0	385.2	129402
737	223	375	17	0	392	320865	370.8	20	0	390.8	108931
740	218	364.2	14	0	378.2	319627	360	19	0	379	100059
742	215	364.4	19	0	383.4	309855	359.2	20	0	379.2	105050
742	219	383.2	21	0	404.2	275867	378.8	22	0	400.8	129183
743	238	364.4	21	0	385.4	317997	362.2	18	0	380.2	98615
747	228	374.8	18	0	392.8	339153	370.2	22	0	392.2	117030
747	237	372.2	20	0	392.2	287216	368.6	22	0	390.6	107438
750	225	378.8	15	0	393.8	331988	376.2	18	0	394.2	111446
751	222	366.6	18	0	384.6	566907	365.6	21	0	386.6	102517
752	218	370.6	17	0	387.6	458586	369.8	19	0	388.8	103397
753	222	384.4	16	0	400.4	336626	380	19	0	399	107972
754	215	369.2	17	0	386.2	370285	366.2	17	0	383.2	105321
757	241	375	22	0	397	293768	368.2	24	0	392.2	113040
762	218	388.2	17	0	405.2	292638	381.6	14	0	395.6	106971
763	228	385.6	18	0	403.6	244167	379.2	24	0	403.2	106427
763	231	399.2	13	0	412.2	292264	396.8	28	0	424.8	109878
765	216	376.8	15	0	391.8	383542	373	18	0	391	102549
765	223	378.4	15	0	393.4	302220	371.8	21	0	392.8	94510
771	221	388.4	17	0	405.4	261956	388.4	21	0	409.4	105314
771	224	397.6	15	0	412.6	252180	389.8	21	0	410.8	98874
773	217	381.4	22	0	403.4	325606	375.2	20	0	395.2	139075
775	222	389.4	16	0	405.4	282568	387.4	25	0	412.4	97675
776	221	393.6	15	0	408.6	257578	392.4	25	0	417.4	93988
777	216	385.2	22	0	407.2	300491	378.2	23	0	401.2	106481
778	224	382.2	16	0	398.2	414852	377	26	0	403	90360
781	229	391.4	18	0	409.4	237159	384.6	20	0	404.6	93043
783	218	389	21	0	410	236059	383.4	18	0	401.4	89526
785	226	400.2	18	0	418.2	315023	402.4	23	23	655.4	107195
787	225	397.2	18	0	415.2	254298	396.8	21	0	417.8	107328
788	220	386.4	16	0	402.4	296284	383.6	23	0	406.6	127876
788	220	388.6	19	0	407.6	247692	385.4	20	0	405.4	99571
789	222	397	22	0	419	246106	396.4	24	27	690.4	97868
789	227	384	16	0	400	483186	383	23	0	406	91408
789	240	392.6	22	0	414.6	248988	390	22	0	412	100499
790	225	407.2	23	0	430.2	219763	408	27	4	475	104784
790	236	401.2	18	0	419.2	267067	400	24	0	424	99302
794	232	393.4	19	0	412.4	212916	392.2	23	0	415.2	78615
795	220	397.8	19	0	416.8	229526	397.4	20	0	417.4	109482
797	229	396.2	14	0	410.2	260107	393.8	23	0	416.8	109295
800	228	386.8	19	0	405.8	324480	387.8	19	0	406.8	102543
800	228	402	17	0	419	272783	402	25	0	427	99494
801	241	398.4	21	0	419.4	314286	400.2	25	0	425.2	86818
803	224	407.2	21	0	428.2	208530	404	24	16	588	91762
804	223	405.2	20	0	425.2	189739	401.4	28	0	429.4	85926
806	223	395.4	26	0	421.4	303454	396.2	24	0	420.2	109480
807	234	398.8	19	0	417.8	249806	396	25	0	421	83489
808	236	411	22	0	433	204316	410.2	22	0	432.2	95254
811	227	426.2	21	18	627.2	177773	419.2	29	29	738.2	69439
812	231	407.4	20	0	427.4	258603	409.4	27	30	736.4	104043
814	225	400.4	18	0	418.4	236629	398.6	19	0	417.6	97765
822	229	429.2	19	18	628.2	168094	425.4	25	39	840.4	74712
825	232	413.6	24	0	437.6	167668	410.2	25	41	845.2	80279
834	223	417.4	18	0	435.4	172930	418.8	18	13	566.8	83493
841	233	424	21	9	535	172812	424.6	21	53	975.6	86488
848	228	415.2	20	0	435.2	184460	417.2	26	0	443.2	78801
mean		389.97	18.35	0.75	415.82	291201.42	386.91	22.07	4.58	454.81	102116.03

Tabella B.5: ALNS e ALNS–A dopo 7200 secondi in un grafo non euclideo su PC1.

		ALNS					ALNS–A				
R	P	u	l	w	f	step	u	l	w	f	step
719	228	339	11	0	350	227306	328.4	11	0	339.4	91404
731	219	345.6	13	0	358.6	214922	327.2	14	0	341.2	102585
732	220	330.8	13	0	343.8	294679	324.6	13	48	817.6	100030
733	229	335.8	13	0	348.8	270427	328.6	14	0	342.6	110565
737	219	357	12	58	949	94966	339.8	11	66	1010.8	92546
737	223	340	11	26	611	246635	330.2	13	102	1363.2	94378
740	218	336.4	12	0	348.4	232574	325.8	14	126	1599.8	71987
742	215	331.4	13	0	344.4	249819	323	14	0	337	89335
742	219	348.8	10	11	468.8	213275	333.6	15	51	858.6	86337
743	238	346.8	15	37	731.8	184656	325.4	13	89	1228.4	92211
747	228	337.8	16	0	353.8	385779	330.8	21	0	351.8	127650
747	237	335.2	16	0	351.2	264626	324.8	15	42	759.8	102606
750	225	343.6	13	100	1356.6	214573	336	15	25	601	83744
751	222	340.4	10	0	350.4	295120	335.2	14	0	349.2	95279
752	218	352.4	18	5	420.4	229113	330.4	16	130	1646.4	75919
753	222	355.8	13	73	1098.8	222528	345.4	12	35	707.4	96485
754	215	337.2	14	0	351.2	266079	325.6	15	0	340.6	90709
757	241	348.4	16	39	754.4	269118	333.6	15	7	418.6	99089
762	218	342.6	14	0	356.6	325030	338.6	19	0	357.6	103280
763	228	365	15	14	520	172924	346.4	13	68	1039.4	66448
763	231	366.8	16	0	382.8	428422	360.4	19	0	379.4	141092
765	216	343.2	14	0	357.2	222455	335.8	15	0	350.8	81839
765	223	341.2	10	0	351.2	246862	332.6	15	46	807.6	90630
771	221	346.4	20	0	366.4	333750	342.4	19	0	361.4	119798
771	224	359.8	18	0	377.8	286469	353.4	22	0	375.4	100521
773	217	351	10	0	361	279477	341.4	13	0	354.4	129548
775	222	348.2	17	0	365.2	320955	342.2	21	0	363.2	101682
776	221	373.6	16	63	1019.6	183484	358.2	15	122	1593.2	74275
777	216	343	20	0	363	240280	338.4	25	0	363.4	92377
778	224	338	20	0	358	245113	333.2	23	0	356.2	92302
781	229	350.2	20	0	370.2	319678	347.4	22	0	369.4	106573
783	218	350.6	18	0	368.6	217205	343.8	18	24	601.8	70855
785	226	357.4	18	0	375.4	259266	352	22	0	374	113252
787	225	356	13	0	369	254841	352.2	24	0	376.2	88873
788	220	348.8	14	0	362.8	304291	342.6	15	0	357.6	93427
788	220	360.2	10	6	430.2	246047	343	12	20	555	107506
789	222	360.4	16	0	376.4	317043	353.4	19	0	372.4	108738
789	227	349.2	23	0	372.2	265055	343.4	24	0	367.4	88587
789	240	356.8	18	0	374.8	340872	353.4	20	0	373.4	105697
790	225	366.8	24	0	390.8	209676	360.4	21	0	381.4	89845
790	236	356	19	0	375	239782	351	21	0	372	94756
794	232	354.8	12	0	366.8	309195	348	26	0	374	85926
795	220	356.2	14	0	370.2	444452	352.2	18	0	370.2	119883
797	229	361	21	0	382	249558	355.2	23	0	378.2	107111
800	228	355.2	19	0	374.2	388180	350.2	24	0	374.2	95642
800	228	370.2	18	0	388.2	223867	360.6	21	0	381.6	104012
801	241	369.6	18	0	387.6	226009	359.2	19	0	378.2	105871
803	224	361.8	19	0	380.8	263044	356	22	0	378	96044
804	223	353.6	22	0	375.6	235293	347.2	26	0	373.2	93251
806	223	353	18	0	371	575074	349.6	19	0	368.6	117503
807	234	358.2	17	0	375.2	245474	353.6	26	0	379.6	81878
808	236	369	16	0	385	271834	365.8	23	0	388.8	101312
811	227	377.6	18	0	395.6	182345	369.4	24	0	393.4	75787
812	231	401.4	16	0	417.4	318044	396.2	22	0	418.2	131360
814	225	358	20	0	378	271439	356.6	18	0	374.6	91646
822	229	383	20	0	403	194315	376.8	24	0	400.8	86244
825	232	376	14	0	390	245558	373.2	21	0	394.2	80000
834	223	378.6	15	0	393.6	285727	376.8	23	0	399.8	82740
841	233	384.2	22	0	406.2	230257	378.6	25	0	403.6	87404
848	228	379.6	20	0	399.6	224740	377.6	26	0	403.6	87124
mean		354.91	16.02	7.20	442.93	266992.95	346.95	18.70	16.68	532.48	96591.63

Tabella B.6: ALNS e ALNS–A dopo 7200 secondi in un grafo euclideo su PC1.

		VNS(10)				VNS–A(10)			
R	P	u	l	w	f	u	l	w	f
719	228	372.8	23	0	395.8	392.2	22	0	414.2
731	219	609	32	230	2941	443.2	20	46	923.2
732	220	378.2	20	0	398.2	384.6	19	12	523.6
733	229	387.2	30	0	417.2	400	22	27	692
737	219	384.2	23	0	407.2	421.6	22	35	793.6
737	223	390.6	22	0	412.6	391.6	20	0	411.6
740	218	468.4	25	99	1483.4	394.8	23	0	417.8
742	215	429	22	33	781	395.6	20	0	415.6
742	219	609.2	32	192	2561.2	406	15	0	421
743	238	381.2	24	0	405.2	396.8	21	0	417.8
747	228	384.2	26	6	470.2	405	25	0	430
747	237	383.8	22	0	405.8	385.6	22	0	407.6
750	225	523.4	30	123	1783.4	403.2	23	0	426.2
751	222	547.4	36	246	3043.4	421.4	22	30	743.4
752	218	535	31	136	1926	410.4	26	0	436.4
753	222	401.8	23	0	424.8	552.6	27	186	2439.6
754	215	389	23	0	412	408	17	8	505
757	241	391	19	4	450	396.6	23	13	549.6
762	218	495	21	100	1516	415	21	15	586
763	228	613	33	325	3896	729.4	36	371	4475.4
763	231	414.8	22	50	936.8	427.8	23	60	1050.8
765	216	528.4	25	173	2283.4	642	24	262	3286
765	223	522.4	33	115	1705.4	404.2	23	0	427.2
771	221	406.2	22	20	628.2	419.8	19	6	498.8
771	224	650.8	33	282	3503.8	415.4	25	14	580.4
773	217	398.8	19	0	417.8	408.4	23	4	471.4
775	222	455.4	26	69	1171.4	410.2	20	18	610.2
776	221	468.2	27	98	1475.2	435.4	25	97	1430.4
777	216	413.8	27	43	870.8	420.4	19	31	749.4
778	224	398.4	24	26	682.4	409.4	26	31	745.4
781	229	402.6	25	0	427.6	423.4	18	21	651.4
783	218	403.2	26	36	789.2	685.6	32	291	3627.6
785	226	479	48	78	1307	433	21	44	894
787	225	421.8	22	22	663.8	425.4	21	41	856.4
788	220	471.6	25	121	1706.6	676.4	30	356	4266.4
788	220	682	33	273	3445	414.6	18	12	552.6
789	222	680.6	42	331	4032.6	421.4	22	64	1083.4
789	227	396.4	24	18	600.4	405.6	25	28	710.6
789	240	410.4	31	30	741.4	458	27	56	1045
790	225	622.2	31	298	3633.2	456.8	24	71	1190.8
790	236	657	34	286	3551	420.2	22	37	812.2
794	232	494	30	119	1714	414.2	21	34	775.2
795	220	424.6	23	23	677.6	427.2	27	20	654.2
797	229	419.2	20	12	559.2	442.8	20	34	802.8
800	228	407.8	21	26	688.8	408	22	14	570
800	228	692	36	300	3728	442.2	18	75	1210.2
801	241	412.8	30	5	492.8	474.8	24	70	1198.8
803	224	474.4	27	89	1391.4	441.6	23	51	974.6
804	223	409.8	24	121	1643.8	425.8	23	81	1258.8
806	223	417.4	25	16	602.4	419.2	21	28	720.2
807	234	414	28	0	442	418.8	25	29	733.8
808	236	670	44	291	3624	447.8	24	51	981.8
811	227	488	27	126	1775	771.4	39	405	4860.4
812	231	437.2	29	53	996.2	443.6	23	66	1126.6
814	225	419.8	22	24	681.8	432.4	23	56	1015.4
822	229	432.2	26	49	948.2	443.8	24	56	1027.8
825	232	433.2	28	44	901.2	450.6	25	66	1135.6
834	223	642.6	37	280	3479.6	452.6	20	52	992.6
841	233	436.2	26	60	1062.2	451.2	22	63	1103.2
848	228	437.6	22	20	659.6	451.6	22	43	903.6
media		472.00	27.35	92.02	1419.52	447.11	22.98	59.68	1066.93

Tabella B.7: VNS e VNS–A con $\overline{\kappa} = 10$ dopo 7200 secondi in un grafo non euclideo su PC1.

		VNS(10)				VNS–A(10)			
R	P	u	l	w	f	u	l	w	f
719	228	650.8	3	280	3453.8	620.2	5	281	3435.2
731	219	684	0	338	4064	672	1	340	4073
732	220	672.4	0	276	3432.4	648.8	0	245	3098.8
733	229	651	1	275	3402	664	0	291	3574
737	219	700.4	0	290	3600.4	663.6	2	272	3385.6
737	223	656	1	316	3817	691.2	0	388	4571.2
740	218	680.6	0	292	3600.6	634.2	2	228	2916.2
742	215	628.8	2	272	3350.8	590.4	3	286	3453.4
742	219	702.8	0	332	4022.8	653	1	292	3574
743	238	614.6	0	235	2964.6	632.4	2	278	3414.4
747	228	346.6	20	0	366.6	376.2	25	0	401.2
747	237	653.8	0	256	3213.8	684.2	0	300	3684.2
750	225	633.6	4	229	2927.6	680.8	0	318	3860.8
751	222	687.6	0	316	3847.6	655.8	3	291	3568.8
752	218	665.6	0	342	4085.6	615.4	2	267	3287.4
753	222	594.4	6	218	2780.4	692.2	0	348	4172.2
754	215	622.6	1	238	3003.6	597.8	6	340	4003.8
757	241	640.8	1	255	3191.8	687.8	0	325	3937.8
762	218	485.8	22	77	1277.8	365.4	17	0	382.4
763	228	678.6	1	299	3669.6	651.8	5	281	3466.8
763	231	375.6	22	26	657.6	388.2	18	0	406.2
765	216	656.4	0	282	3476.4	580.6	3	184	2423.6
765	223	603.8	4	220	2807.8	639.2	4	266	3303.2
771	221	380.2	24	0	404.2	377.6	20	0	397.6
771	224	630.8	37	241	3077.8	384.4	20	0	404.4
773	217	710.2	0	322	3930.2	668.6	1	250	3169.6
775	222	501.2	26	91	1437.2	504	29	132	1853
776	221	685.4	1	294	3626.4	709.8	2	340	4111.8
777	216	530	28	173	2288	375.8	19	0	394.8
778	224	350.2	23	0	373.2	389.4	18	0	407.4
781	229	364.6	23	0	387.6	392.6	21	0	413.6
783	218	652.2	3	298	3635.2	640.8	1	269	3331.8
785	226	393.6	25	12	538.6	385	24	0	409
787	225	681.8	47	302	3748.8	385.2	21	2	426.2
788	220	528.8	33	161	2171.8	570.2	32	227	2872.2
788	220	684.4	0	275	3434.4	623.8	2	230	2925.8
789	222	381.2	25	0	406.2	394.4	22	14	556.4
789	227	447	27	36	834	389.4	21	0	410.4
789	240	371.2	24	0	395.2	626	34	224	2900
790	225	454	33	81	1297	391	24	0	415
790	236	574.2	44	231	2928.2	397.8	22	0	419.8
794	232	377.2	25	0	402.2	380	26	0	406
795	220	376.6	23	0	399.6	405	24	0	429
797	229	381.4	19	0	400.4	514.6	26	107	1610.6
800	228	380.6	29	0	409.6	392.6	21	0	413.6
800	228	424.2	28	48	932.2	391.2	22	0	413.2
801	241	383.2	22	0	405.2	393	23	0	416
803	224	374.6	23	0	397.6	397.8	22	9	509.8
804	223	363.8	26	0	389.8	377	20	0	397
806	223	369.6	24	0	393.6	393.4	20	0	413.4
807	234	547.8	34	138	1961.8	386.6	19	0	405.6
808	236	568	33	170	2301	583	34	225	2867
811	227	555.8	40	145	2045.8	628.2	43	224	2911.2
812	231	458.8	27	73	1215.8	419.8	20	47	909.8
814	225	448.4	32	46	940.4	394.2	19	0	413.2
822	229	385.6	25	13	540.6	448.8	24	86	1332.8
825	232	416.4	28	9	534.4	436.4	28	34	804.4
834	223	539.8	30	126	1829.8	406.4	19	0	425.4
841	233	462.8	25	54	1027.8	417.4	29	52	966.4
848	228	394.4	27	0	421.4	402.2	25	0	427.2
media		530.28	17.18	150.05	2047.96	514.31	14.93	138.22	1911.41

Tabella B.8: VNS e VNS–A con $\overline{\kappa} = 10$ dopo 7200 secondi in un grafo euclideo su PC1.

Appendice C

Computazioni su PC2

		ALNS					ALNS–A				
R	P	u	l	w	f	step	u	l	w	f	step
719	228	369.4	20	0	389.4	100416	363.6	24	0	387.6	37892
731	219	383.6	18	0	401.6	112036	378.4	28	0	406.4	36047
732	220	363.6	19	0	382.6	137984	361.4	28	0	389.4	30846
733	229	380.8	19	0	399.8	88662	374.6	25	0	399.6	30172
737	219	375.8	25	0	400.8	95720	368.8	25	0	393.8	35833
737	223	378.8	25	0	403.8	79480	377.8	28	0	405.8	22274
740	218	372	20	0	392	72132	367.8	28	0	395.8	19653
742	215	373	28	0	401	65710	368.6	30	0	398.6	18879
742	219	388	22	0	410	96624	384.6	24	0	408.6	44290
743	238	371.8	24	0	395.8	86697	369.8	25	0	394.8	25687
747	228	381.2	22	0	403.2	90150	379.2	28	0	407.2	30752
747	237	382.8	27	0	409.8	38994	383	30	0	413	15341
750	225	387.4	20	0	407.4	66272	385	30	0	415	23403
751	222	374	21	0	395	122889	371	30	0	401	29451
752	218	378.2	20	0	398.2	70076	379.2	28	0	407.2	19081
753	222	387.8	21	0	408.8	92628	388.6	25	0	413.6	28274
754	215	375	26	0	401	57809	379	28	0	407	17474
757	241	377.4	28	0	405.4	58217	381.2	37	0	418.2	18505
762	218	393.6	23	0	416.6	49558	397	28	16	585	18513
763	228	392.6	25	0	417.6	51218	386.4	33	0	419.4	20528
763	231	406	18	0	424	72124	407.2	36	15	593.2	30491
765	216	386	20	0	406	55591	384.6	29	0	413.6	17062
765	223	381.4	17	0	398.4	80434	378	27	0	405	25253
771	221	395.2	24	0	419.2	52138	397.2	36	3	463.2	20356
771	224	410	23	47	903	68884	399.6	32	8	511.6	25520
773	217	391	29	0	420	47153	387.4	27	0	414.4	19620
775	222	401	24	1	435	71124	394.6	26	0	420.6	27894
776	221	400.6	23	0	423.6	66364	399.2	30	20	629.2	25432
777	216	390.6	35	0	425.6	45630	387.8	32	17	589.8	18377
778	224	387.8	20	0	407.8	97459	382	33	0	415	25283
781	229	400.4	27	13	557.4	45854	393	26	0	419	21012
783	218	391.2	23	0	414.2	78965	387.8	29	0	416.8	26566
785	226	406.2	27	40	833.2	78517	404.4	30	63	1064.4	27583
787	225	406.8	23	0	429.8	45306	407.2	32	5	489.2	20236
788	220	392.4	25	0	417.4	43469	393	33	0	426	17302
788	220	396.8	22	0	418.8	43717	394.8	26	0	420.8	17768
789	222	405.6	26	29	721.6	69211	402.4	30	49	922.4	28571
789	227	389.6	18	0	407.6	99362	385.2	29	0	414.2	25738
789	240	397.2	29	0	426.2	75105	399.6	27	0	426.6	27485
790	225	417.8	31	23	678.8	50466	413.6	37	27	720.6	26331
790	236	413	28	0	441	54360	408.6	40	30	748.6	22323
794	232	402.6	24	0	426.6	45371	400.4	36	0	436.4	17324
795	220	404	21	0	425	71019	404.2	27	0	431.2	29561
797	229	402.6	19	0	421.6	51975	404.2	37	0	441.2	25484
800	228	392	24	0	416	82413	392.2	32	0	424.2	30838
800	228	409	24	0	433	63497	409.2	33	10	542.2	27162
801	241	403.6	26	0	429.6	77602	402.6	33	17	605.6	27022
803	224	417.4	33	34	790.4	64997	409.8	32	62	1061.8	26243
804	223	416	28	22	664	32286	410.2	30	34	780.2	15414
806	223	402.2	30	0	432.2	50709	402.2	30	0	432.2	22808
807	234	405	30	0	435	42288	405	35	0	440	15999
808	236	421.6	30	5	501.6	48875	427.4	32	47	929.4	22099
811	227	428.4	27	59	1045.4	52339	420.4	35	48	935.4	26397
812	231	417.8	25	27	712.8	71361	413.8	31	52	964.8	32433
814	225	405.4	26	0	431.4	66652	406	23	0	429	27212
822	229	432	26	22	678	49670	429.4	29	53	988.4	27811
825	232	423.2	36	19	649.2	52518	427.6	35	67	1132.6	22083
834	223	419.2	23	0	442.2	58494	426.6	28	42	874.6	27014
841	233	428.8	33	33	791.8	55801	429.8	27	77	1226.8	25689
848	228	421.6	28	6	509.6	54117	428.4	33	20	661.4	25481
media		396.76	24.63	6.33	484.73	67774.82	395.03	30.12	13.03	555.48	24886.20

Tabella C.1: ALNS e ALNS–A dopo 1500 secondi in un grafo non euclideo su PC2.

		ALNS					ALNS–A				
R	P	u	l	w	f	step	u	l	w	f	step
719	228	344.4	14	116	1518.4	53511	334.4	15	124	1589.4	18931
731	219	352.4	13	83	1195.4	51260	334.4	18	35	702.4	25368
732	220	341.6	14	55	905.6	50637	332	19	85	1201	25957
733	229	348	18	76	1126	61087	338.8	21	78	1139.8	20642
737	219	357.6	16	100	1373.6	56340	353.2	16	197	2339.2	20669
737	223	346.4	13	127	1629.4	60493	337	24	134	1701	20493
740	218	338.8	15	68	1033.8	61184	485.4	5	238	2870.4	13431
742	215	345.4	13	44	798.4	70240	331.2	17	63	978.2	23050
742	219	350.6	12	120	1562.6	60753	338.6	20	124	1598.6	22389
743	238	353	16	151	1879	43616	345.6	19	100	1364.6	21412
747	228	341.4	19	0	360.4	121224	337	23	0	360	27726
747	237	347.4	18	43	795.4	78210	332.2	21	137	1723.2	25015
750	225	378.2	14	172	2112.2	29559	356.2	23	154	1919.2	17669
751	222	358.6	11	78	1149.6	64646	353	23	111	1486	17375
752	218	359.6	18	149	1867.6	52184	359.8	18	209	2467.8	18304
753	222	359.8	15	115	1524.8	55849	343.8	19	109	1452.8	22277
754	215	346.2	16	36	722.2	70011	340.8	23	19	553.8	21772
757	241	348.2	13	125	1611.2	63682	414.2	20	153	1964.2	18344
762	218	347.8	15	0	362.8	98766	345.8	25	0	370.8	31091
763	228	362.6	19	155	1931.6	41539	416.8	16	163	2062.8	14249
763	231	373.2	21	0	394.2	119670	367.2	22	0	389.2	35701
765	216	357	17	62	994	57069	343.8	22	65	1015.8	20601
765	223	359	17	70	1076	51262	340	19	125	1609	22157
771	221	353.4	23	0	376.4	104659	349.4	24	0	373.4	32892
771	224	364.2	22	0	386.2	84772	353.8	26	0	379.8	30838
773	217	356.8	18	90	1274.8	70792	344.4	12	40	756.4	31855
775	222	355	20	0	375	105186	351.2	24	0	375.2	28542
776	221	384.8	19	192	2323.8	43984	402.8	17	177	2189.8	17710
777	216	347.2	20	0	367.2	90547	345.2	30	0	375.2	28471
778	224	340.2	20	0	360.2	97237	340.4	30	0	370.4	24049
781	229	356.8	24	0	380.8	94938	355.8	25	0	380.8	25592
783	218	363.4	19	61	992.4	53918	410.6	17	149	1917.6	16211
785	226	364	19	0	383	87319	358.4	26	0	384.4	32069
787	225	361.2	19	0	380.2	90338	363	31	0	394	26995
788	220	355.8	15	0	370.8	107654	349.2	23	0	372.2	27119
788	220	359.6	14	75	1123.6	68037	347.4	19	105	1416.4	25400
789	222	366	16	0	382	121752	356.8	29	0	385.8	32366
789	227	353.8	25	0	378.8	78047	350	29	0	379	23605
789	240	362.8	22	0	384.8	95063	361.4	30	0	391.4	27224
790	225	370.8	27	0	397.8	67607	366.8	27	0	393.8	25305
790	236	360.4	24	0	384.4	85000	355.6	30	0	385.6	29952
794	232	358.4	16	0	374.4	92249	357.2	34	0	391.2	25205
795	220	359.6	15	0	374.6	126364	358.8	22	0	380.8	32274
797	229	365.2	23	0	388.2	78029	362	28	0	390	27332
800	228	359.8	20	0	379.8	90713	353	28	0	381	27176
800	228	372.4	26	0	398.4	69818	365.2	28	0	393.2	30820
801	241	372	24	0	396	87550	365.8	26	0	391.8	31597
803	224	365.2	22	0	387.2	89151	360.4	25	0	385.4	31136
804	223	359.8	24	0	383.8	75209	353.2	28	0	381.2	26619
806	223	356.4	21	0	377.4	140651	356.6	22	0	378.6	36501
807	234	363.4	19	0	382.4	75474	358.4	34	0	392.4	22643
808	236	372.6	17	0	389.6	87057	369.6	33	0	402.6	30162
811	227	385	23	0	408	58151	373.4	31	15	554.4	23129
812	231	404.2	20	0	424.2	101616	401.2	28	0	429.2	35449
814	225	364.6	21	0	385.6	87917	365.4	23	0	388.4	27143
822	229	389	23	0	412	60854	382.4	31	25	663.4	20793
825	232	381.6	22	0	403.6	76808	382.6	31	12	533.6	23246
834	223	387	20	0	407	79273	386	28	0	414	24505
841	233	388.6	26	0	414.6	68951	381.2	28	0	409.2	25235
848	228	388.2	24	0	412.2	60203	383	30	0	413	24748
media		361.44	18.82	39.38	774.09	77094.67	360.98	23.92	49.10	875.90	25275.52

Tabella C.2: ALNS e ALNS–A dopo 1500 secondi in un grafo euclideo su PC2.

		ALNS					ALNS–A				
R	P	u	l	w	f	step	u	l	w	f	step
719	228	366.2	19	0	385.2	225085	358.6	20	0	378.6	81320
731	219	380.4	15	0	395.4	235941	375.2	23	0	398.2	79479
732	220	361.2	17	0	378.2	321846	353	23	0	376	69318
733	229	378	15	0	393	220730	370.6	22	0	392.6	67117
737	219	372.4	23	0	395.4	181454	364.2	23	0	387.2	83411
737	223	379	22	0	401	166375	373.2	23	0	396.2	52321
740	218	366.6	14	0	380.6	169533	365.6	22	0	387.6	40723
742	215	366.4	21	0	387.4	144369	364	25	0	389	39438
742	219	385	21	0	406	205351	380.8	22	0	402.8	93821
743	238	370.6	24	0	394.6	183619	366.2	17	0	383.2	55874
747	228	378.2	20	0	398.2	178722	374	25	0	399	66371
747	237	376.4	26	0	402.4	105491	374.8	26	0	400.8	36638
750	225	382.8	18	0	400.8	139236	381.6	26	0	407.6	45309
751	222	370.6	19	0	389.6	310698	367.2	24	0	391.2	65699
752	218	375	19	0	394	168108	373.8	24	0	397.8	41670
753	222	385.2	18	0	403.2	215513	383.4	22	0	405.4	61843
754	215	372.6	23	0	395.6	125461	372.2	20	0	392.2	38996
757	241	376.2	27	0	403.2	115451	375.8	30	0	405.8	39575
762	218	390	21	0	411	112275	385	15	0	400	50891
763	228	389.6	20	0	409.6	111500	382.2	27	0	409.2	43566
763	231	401.8	14	0	415.8	161292	400	29	0	429	66132
765	216	380	18	0	398	134968	378.8	23	0	401.8	37009
765	223	380	15	0	395	194231	374.6	25	0	399.6	55521
771	221	392.2	17	0	409.2	118363	392.6	31	0	423.6	44224
771	224	400.6	17	0	417.6	154346	392.8	23	0	415.8	59315
773	217	387.2	23	0	410.2	106009	381.4	22	0	403.4	44086
775	222	392	18	0	410	162973	390.2	26	0	416.2	59445
776	221	396.4	21	0	417.4	151866	393.4	28	0	421.4	56525
777	216	386	30	0	416	107259	383.6	32	0	415.6	39814
778	224	383.6	17	0	400.6	248009	377.8	30	0	407.8	54598
781	229	393.2	20	0	413.2	107017	388.2	23	0	411.2	42853
783	218	388.8	23	0	411.8	158167	386	21	0	407	59531
785	226	409.6	22	3	461.6	186233	402.6	24	51	936.6	60916
787	225	400.4	20	0	420.4	110629	402	26	0	428	44980
788	220	391	19	0	410	96714	389.2	23	0	412.2	39161
788	220	392.4	22	0	414.4	85386	389.8	27	0	416.8	35485
789	222	400.4	24	0	424.4	146563	401.4	29	37	800.4	59489
789	227	386.4	16	0	402.4	278138	382.4	26	0	408.4	57805
789	240	395.6	22	0	417.6	160835	394.8	25	0	419.8	60394
790	225	410.8	27	0	437.8	113104	410.2	31	17	611.2	60300
790	236	405.4	18	0	423.4	123477	405.6	36	4	481.6	50184
794	232	397.2	20	0	417.2	97651	393.8	27	0	420.8	37495
795	220	401	19	0	420	135430	398.6	22	0	420.6	64398
797	229	399	18	0	417	122535	400	33	0	433	54314
800	228	389.4	21	0	410.4	198359	388	23	0	411	64133
800	228	405.6	21	0	426.6	141416	405.4	28	0	433.4	56874
801	241	400.6	21	0	421.6	182121	402.6	30	5	482.6	56381
803	224	410.6	25	0	435.6	133721	407	30	30	737	57421
804	223	409.8	26	0	435.8	68005	411.2	26	16	597.2	32253
806	223	397	29	0	426	137767	397.2	26	0	423.2	51121
807	234	403	25	0	428	94652	400.8	29	0	429.8	35184
808	236	415.6	24	0	439.6	98521	419.8	27	20	646.8	45801
811	227	423.2	22	39	835.2	117645	420	35	36	815	42224
812	231	411.2	24	0	435.2	148264	409.2	29	42	858.2	67636
814	225	402	19	0	421	138343	401.2	20	0	421.2	58299
822	229	426.6	21	22	667.6	113443	426.8	27	44	893.8	50052
825	232	417.2	29	0	446.2	103581	412	29	48	921	49493
834	223	417	19	0	436	120300	420.6	25	23	675.6	57311
841	233	423.8	25	15	598.8	111015	422.8	22	57	1014.8	53248
848	228	417.6	23	0	440.6	119000	423	28	5	501	53933
media		392.73	20.93	1.32	426.83	152068.43	390.31	25.58	7.25	488.40	53811.97

Tabella C.3: ALNS e ALNS–A dopo 3200 secondi in un grafo non euclideo su PC2.

		ALNS					ALNS–A				
R	P	u	l	w	f	step	u	l	w	f	step
719	228	343.8	15	46	818.8	127771	328.8	12	66	1000.8	47904
731	219	349.6	13	29	652.6	120104	334.8	16	1	360.8	60423
732	220	344.8	13	0	357.8	145308	322	15	71	1047	58001
733	229	342.4	15	0	357.4	153699	333.2	15	0	348.2	57257
737	219	352.8	13	76	1125.8	147302	344.8	16	145	1810.8	51900
737	223	355.2	13	67	1038.2	138542	330.8	17	124	1587.8	54077
740	218	336	15	52	871	139760	339.8	13	179	2142.8	38138
742	215	334.8	14	0	348.8	163923	325	14	0	339	58645
742	219	356.2	12	58	948.2	121061	335.2	16	96	1311.2	51930
743	238	351	17	43	798	111244	328.8	14	95	1292.8	53809
747	228	339.2	16	0	355.2	255008	333	23	0	356	64936
747	237	347.2	17	0	364.2	159040	326.4	18	90	1244.4	56474
750	225	348.6	13	101	1371.6	117381	342.2	21	57	933.2	48979
751	222	349.2	13	10	462.2	159020	334.4	17	28	631.4	51593
752	218	340.4	15	62	975.4	123253	337.4	15	165	2002.4	44667
753	222	355	14	83	1199	134066	342.4	16	99	1348.4	54831
754	215	340.6	15	0	355.6	165744	329.6	17	0	346.6	56635
757	241	341.2	12	103	1383.2	156262	333	19	66	1012	51160
762	218	344.2	15	0	359.2	220197	339.6	20	0	359.6	68806
763	228	360	15	56	935	108833	346.4	15	100	1361.4	40344
763	231	366.8	18	0	384.8	271017	362.6	21	0	383.6	81423
765	216	349	14	3	393	128925	342	17	10	459	51210
765	223	351.4	13	0	364.4	125700	337	16	90	1253	52581
771	221	348.4	21	0	369.4	208915	344	23	0	367	72491
771	224	361.4	19	0	380.4	173422	350.6	23	0	373.6	64100
773	217	370.6	13	14	523.6	158825	343	12	17	525	72862
775	222	350.2	18	0	368.2	204038	344.6	22	0	366.6	61456
776	221	387.2	14	136	1761.2	99943	365.6	17	156	1942.6	42481
777	216	344.6	20	0	364.6	169475	339.8	27	0	366.8	58986
778	224	339	20	0	359	177164	334.8	24	0	358.8	54219
781	229	353.2	20	0	373.2	200831	349.4	20	0	369.4	56997
783	218	355.2	19	0	374.2	121410	346	20	37	736	40262
785	226	359.4	18	0	377.4	181036	353.2	24	0	377.2	66652
787	225	356.8	13	0	369.8	172230	355.6	27	0	382.6	56423
788	220	350	15	0	365	201197	344.8	18	0	362.8	59634
788	220	354	14	53	898	147549	347.4	15	55	912.4	61578
789	222	364	16	0	380	240372	354.8	21	0	375.8	66174
789	227	351.4	24	0	375.4	161258	344.4	26	0	370.4	51464
789	240	359.6	18	0	377.6	206836	356	23	0	379	58768
790	225	368.4	24	0	392.4	144148	362	24	0	386	55549
790	236	357.4	19	0	376.4	182492	352.4	24	0	376.4	61299
794	232	356	13	0	369	209544	351.4	27	0	378.4	51257
795	220	358	15	0	373	282773	356	19	0	375	69092
797	229	361.8	22	0	383.8	180025	358	25	0	383	61652
800	228	357	20	0	377	226153	350.8	26	0	376.8	59362
800	228	369.4	22	0	391.4	134422	361.6	25	0	386.6	61707
801	241	369.8	20	0	389.8	159648	361	22	0	383	67830
803	224	363	21	0	384	177592	357	23	0	380	64337
804	223	355.6	23	0	378.6	142607	349.2	27	0	376.2	57910
806	223	354.2	20	0	374.2	320847	351	19	0	370	75929
807	234	359.4	18	0	377.4	157514	354.4	30	0	384.4	52704
808	236	370.8	16	0	386.8	184725	367	23	0	390	63993
811	227	378.8	19	0	397.8	122906	373.2	27	0	400.2	50188
812	231	403	19	0	422	195641	398.8	26	0	424.8	74732
814	225	360.4	21	0	381.4	177935	359	18	0	377	57574
822	229	386.6	20	0	406.6	110688	379	25	2	424	49980
825	232	377.6	18	0	395.6	164250	374.8	23	0	397.8	48809
834	223	381.2	16	0	397.2	172586	378.6	24	0	402.6	50133
841	233	386	23	0	409	148033	379	27	0	406	54797
848	228	383.6	22	0	405.6	136131	379	27	0	406	53130
media		357.71	17.05	16.53	540.09	167472.02	349.27	20.60	29.15	661.37	57370.57

Tabella C.4: ALNS e ALNS–A dopo 3200 secondi in un grafo euclideo su PC2.

		ALNS					ALNS–A				
R	P	u	l	w	f	step	u	l	w	f	step
719	228	365.2	18	0	383.2	261327	356.6	19	0	375.6	91470
731	219	380.4	15	0	395.4	263880	374.4	23	0	397.4	89740
732	220	360	17	0	377	378822	353	20	0	373	78505
733	229	377.8	15	0	392.8	245392	370.2	22	0	392.2	75224
737	219	371	22	0	393	210022	363.4	23	0	386.4	95221
737	223	377.6	21	0	398.6	193705	374	22	0	396	60875
740	218	366.6	14	0	380.6	182143	363.8	22	0	385.8	46132
742	215	366	21	0	387	157495	363.2	23	0	386.2	44663
742	219	384.6	21	0	405.6	221425	380.4	22	0	402.4	104720
743	238	370	24	0	394	210261	364.8	18	0	382.8	63181
747	228	377.2	19	0	396.2	205091	372	24	0	396	75358
747	237	376.4	26	0	402.4	121720	374.6	25	0	399.6	42098
750	225	382.6	18	0	400.6	156132	380.8	26	0	406.8	50960
751	222	369	18	0	387	373009	367.6	22	0	389.6	73111
752	218	374.6	19	0	393.6	186854	373.6	24	0	397.6	47627
753	222	384.8	18	0	402.8	241131	383.2	22	0	405.2	69871
754	215	371.2	21	0	392.2	147080	371.8	20	0	391.8	43290
757	241	376.4	26	0	402.4	132297	374.6	29	0	403.6	45373
762	218	390.6	20	0	410.6	128559	384.2	15	0	399.2	59599
763	228	388.6	20	0	408.6	125980	382	27	0	409	50075
763	231	401.2	14	0	415.2	178603	399.8	29	0	428.8	72619
765	216	380	18	0	398	157434	378	23	0	401	42581
765	223	379.6	15	0	394.6	222954	373.8	25	0	398.8	63048
771	221	391.2	17	0	408.2	131323	391.8	29	0	420.8	49892
771	224	400	16	0	416	173622	391.2	23	0	414.2	67201
773	217	386.4	23	0	409.4	122255	379.6	21	0	400.6	50131
775	222	391.4	18	0	409.4	188751	389.2	25	0	414.2	67693
776	221	396	19	0	415	172853	393.8	27	0	420.8	64281
777	216	385.6	30	0	415.6	124303	383	29	0	412	45183
778	224	383.4	17	0	400.4	273962	377.4	29	0	406.4	59912
781	229	392.6	19	0	411.6	124225	387.4	23	0	410.4	48094
783	218	388.2	23	0	411.2	175481	385.2	20	0	405.2	66925
785	226	405.6	20	0	425.6	212439	401	21	50	922	69117
787	225	400	20	0	420	124298	401.2	26	0	427.2	49687
788	220	390.6	21	0	411.6	95477	389.2	27	0	416.2	40432
788	220	391	18	0	409	113689	389	23	0	412	45863
789	222	399.4	23	0	422.4	161383	398.8	28	37	796.8	67784
789	227	385.2	16	0	401.2	323694	382.4	26	0	408.4	66692
789	240	394.6	22	0	416.6	181637	393.6	24	0	417.6	68151
790	225	409	24	0	433	134931	410.4	31	14	581.4	67017
790	236	405.2	18	0	423.2	139645	406.6	33	0	439.6	57092
794	232	396	20	0	416	115094	393.4	26	0	419.4	43320
795	220	400.4	19	0	419.4	152620	398.6	22	0	420.6	72296
797	229	399	18	0	417	134915	399.2	28	0	427.2	62067
800	228	387.8	21	0	408.8	223729	387.2	22	0	409.2	72627
800	228	405	19	0	424	161865	405	28	0	433	63829
801	241	400.2	21	0	421.2	203653	403.4	29	4	472.4	62981
803	224	410	25	0	435	150444	407.6	27	22	654.6	65191
804	223	408.8	26	0	434.8	78358	407.4	27	11	544.4	36454
806	223	396.8	29	0	425.8	157849	396.8	24	0	420.8	58077
807	234	401.8	25	0	426.8	108919	400.6	28	0	428.6	39561
808	236	415.2	24	0	439.2	108856	418.8	27	14	585.8	51398
811	227	423.8	22	38	825.8	131729	419.4	36	32	775.4	47357
812	231	410.2	23	0	433.2	167129	408	29	42	857	73741
814	225	401.6	19	0	420.6	154504	400	20	0	420	66356
822	229	426.6	21	21	657.6	128242	426.8	27	44	893.8	55783
825	232	416.8	27	0	443.8	114042	411.8	28	48	919.8	56713
834	223	416.4	19	0	435.4	133595	420.8	23	22	663.8	64292
841	233	422.8	24	15	596.8	127076	421.4	22	57	1013.4	60764
848	228	416.4	22	0	438.4	134335	418.8	28	3	476.8	59624
media		392.04	20.47	1.23	424.84	172703.97	389.59	24.85	6.67	481.11	60815.32

Tabella C.5: ALNS e ALNS–A dopo 3600 secondi in un grafo non euclideo su PC2.

		ALNS					ALNS–A				
R	P	u	l	w	f	step	u	l	w	f	step
719	228	344.2	13	26	617.2	143697	328.4	12	53	870.4	54969
731	219	347.2	13	21	570.2	139247	334.2	15	0	349.2	68620
732	220	340.4	13	0	353.4	168175	323.6	15	68	1018.6	65947
733	229	341	14	0	355	176524	331.6	15	0	346.6	66484
737	219	356	12	66	1028	170117	343.2	14	91	1267.2	60013
737	223	352.2	13	62	985.2	159368	334	14	104	1388	63454
740	218	334.8	15	44	789.8	156777	331.6	15	172	2066.6	46132
742	215	333.4	14	0	347.4	190419	324.6	14	0	338.6	67298
742	219	350.6	12	39	752.6	139614	334.2	15	95	1299.2	59697
743	238	350	16	40	766	127903	329	14	89	1233	61535
747	228	338.6	16	0	354.6	278547	332.4	23	0	355.4	74060
747	237	343.8	17	0	360.8	178255	326.8	17	82	1163.8	65026
750	225	346.8	13	100	1359.8	136373	341.4	19	51	870.4	56646
751	222	346.4	12	0	358.4	184382	333.2	17	28	630.2	60837
752	218	340.6	15	58	935.6	137807	332.6	16	146	1808.6	50834
753	222	354.6	14	83	1198.6	152574	344.8	16	78	1140.8	63104
754	215	338.4	15	0	353.4	194373	327.6	16	0	343.6	63705
757	241	355.6	14	69	1059.6	179942	335.4	19	45	804.4	58930
762	218	344.2	15	0	359.2	236939	339	20	0	359	76167
763	228	358.8	15	46	833.8	126476	344.8	15	87	1229.8	46283
763	231	367.4	16	0	383.4	306061	361.6	20	0	381.6	92785
765	216	346.6	14	0	360.6	147750	341.2	15	4	396.2	58041
765	223	345.6	12	0	357.6	146838	335	16	63	981	59141
771	221	347.6	21	0	368.6	233967	344	22	0	366	83119
771	224	361.4	19	0	380.4	188773	349.8	21	0	370.8	72625
773	217	374.8	10	0	384.8	179194	341.6	12	17	523.6	83129
775	222	350.2	17	0	367.2	225658	343.8	22	0	365.8	69960
776	221	384	15	122	1619	113435	364.4	17	132	1701.4	48376
777	216	344	20	0	364	182445	339.4	26	0	365.4	66034
778	224	338.8	20	0	358.8	192013	334.2	24	0	358.2	61615
781	229	352.8	20	0	372.8	221604	349.4	20	0	369.4	63992
783	218	354.6	19	0	373.6	140428	344.2	20	35	714.2	46422
785	226	359	18	0	377	198139	352.6	23	0	375.6	74961
787	225	356.6	13	0	369.6	188995	354.6	26	0	380.6	63437
788	220	349.8	15	0	364.8	226070	343.6	17	0	360.6	66891
788	220	352.8	14	47	836.8	164927	345	15	41	770	70435
789	222	362	16	0	378	260425	354.4	20	0	374.4	75350
789	227	350.6	23	0	373.6	184785	344.4	25	0	369.4	59402
789	240	358	18	0	376	234273	355.2	22	0	377.2	67441
790	225	368.4	24	0	392.4	158853	362	24	0	386	61631
790	236	357	19	0	376	199924	352.4	23	0	375.4	68503
794	232	356	12	0	368	242777	349.8	27	0	376.8	57408
795	220	358	15	0	373	313667	355	19	0	374	77300
797	229	361	22	0	383	196884	358	24	0	382	69119
800	228	356.8	20	0	376.8	254603	350.4	26	0	376.4	66881
800	228	370	21	0	391	149630	360.8	24	0	384.8	68219
801	241	370.2	19	0	389.2	169633	361	22	0	383	75899
803	224	362.8	21	0	383.8	197511	356.6	23	0	379.6	72512
804	223	355.4	23	0	378.4	161009	348.8	27	0	375.8	65029
806	223	354	20	0	374	359622	350.6	19	0	369.6	84766
807	234	358.6	18	0	376.6	176059	354.4	28	0	382.4	59838
808	236	370.6	16	0	386.6	206221	366.4	23	0	389.4	72987
811	227	378.6	18	0	396.6	138061	372	26	0	398	56563
812	231	402.6	19	0	421.6	218127	399.4	24	0	423.4	85107
814	225	359.8	20	0	379.8	195409	358.2	18	0	376.2	64483
822	229	386.4	20	0	406.4	116787	379.2	25	1	414.2	57432
825	232	377.6	18	0	395.6	182687	375	22	0	397	56001
834	223	380.6	16	0	396.6	190348	378	24	0	402	56387
841	233	386	23	0	409	169555	378.2	27	0	405.2	62078
848	228	382.4	22	0	404.4	159366	379	27	0	406	61152
media		357.12	16.78	13.72	511.07	187833.70	348.60	20.10	24.70	615.70	65203.20

Tabella C.6: ALNS e ALNS–A dopo 3600 secondi in un grafo euclideo su PC2.

		VNS(1)				VNS–A(1)			
R	P	u	l	w	f	u	l	w	f
719	228	391.4	30	119	1611.4	400.8	20	4	460.8
731	219	429.8	23	20	652.8	399.2	18	0	417.2
732	220	434	38	93	1402	715.6	47	350	4262.6
733	229	395.4	22	16	577.4	410.8	27	1	447.8
737	219	381.4	23	45	854.4	389.4	26	0	415.4
737	223	425.6	28	81	1263.6	393.6	22	0	415.6
740	218	508.8	52	141	1970.8	403.8	29	22	652.8
742	215	466.2	30	89	1386.2	401.6	20	0	421.6
742	219	651	36	269	3377	417.8	24	3	471.8
743	238	385.2	30	0	415.2	397.6	22	0	419.6
747	228	594.4	50	277	3414.4	515.2	40	193	2485.2
747	237	581.6	46	272	3347.6	550.6	42	190	2492.6
750	225	399	26	0	425	402.4	24	13	556.4
751	222	652.4	50	341	4112.4	632.8	34	327	3936.8
752	218	783.2	71	425	5104.2	490.4	31	116	1681.4
753	222	523.4	45	145	2018.4	496.6	28	95	1474.6
754	215	600.2	30	179	2420.2	483.8	30	71	1223.8
757	241	657	65	319	3912	410.2	27	50	937.2
762	218	527.4	24	118	1731.4	422.2	27	16	609.2
763	228	632.4	51	288	3563.4	427.6	26	43	883.6
763	231	633.8	63	369	4386.8	443.6	23	61	1076.6
765	216	418.6	34	53	982.6	552.2	37	230	2889.2
765	223	660.2	43	320	3903.2	561	34	221	2805
771	221	552.2	41	238	2973.2	537.8	43	213	2710.8
771	224	516.6	44	201	2570.6	649.6	35	306	3744.6
773	217	405.8	28	0	433.8	450	26	48	956
775	222	693.6	46	353	4269.6	457.8	31	74	1228.8
776	221	527.4	45	196	2532.4	490.4	43	171	2243.4
777	216	418.2	31	50	949.2	477.4	21	99	1488.4
778	224	411.4	37	50	948.4	558.8	39	234	2937.8
781	229	658	51	310	3809	433.8	26	48	939.8
783	218	677.2	46	340	4123.2	452.4	21	59	1063.4
785	226	616.4	44	314	3800.4	509.2	34	171	2253.2
787	225	678.6	43	308	3801.6	467.8	27	81	1304.8
788	220	549.4	35	153	2114.4	478.2	33	85	1361.2
788	220	563.6	33	208	2676.6	446.4	24	44	910.4
789	222	428.6	19	30	747.6	532.6	29	150	2061.6
789	227	631	45	310	3776	553.8	33	154	2126.8
789	240	407.8	30	26	697.8	766.6	46	446	5272.6
790	225	578.2	45	282	3443.2	451.6	29	67	1150.6
790	236	462.2	36	101	1508.2	665.6	41	318	3886.6
794	232	410.8	32	48	922.8	512.6	46	163	2188.6
795	220	434.8	27	58	1041.8	453.8	26	74	1219.8
797	229	545.2	34	145	2029.2	676	43	278	3499
800	228	540.8	44	191	2494.8	529.2	39	186	2428.2
800	228	619.4	47	296	3626.4	456.8	30	52	1006.8
801	241	563.8	47	191	2520.8	640.8	35	250	3175.8
803	224	554.4	39	207	2663.4	566.6	35	214	2741.6
804	223	410	37	86	1307	428.8	31	119	1649.8
806	223	619	34	231	2963	456.4	23	83	1309.4
807	234	657.8	45	296	3662.8	433.2	32	51	975.2
808	236	590.4	51	251	3151.4	447.4	22	55	1019.4
811	227	522.4	45	206	2627.4	589.4	42	279	3421.4
812	231	496.2	35	133	1861.2	579.6	36	261	3225.6
814	225	551.4	38	197	2559.4	718.6	45	387	4633.6
822	229	522.8	35	223	2787.8	525	37	183	2392
825	232	643.4	48	307	3761.4	533.2	39	175	2322.2
834	223	490.6	33	143	1953.6	456.2	20	54	1016.2
841	233	798	76	461	5484	862	80	563	6572
848	228	556.6	46	176	2362.6	442.6	22	40	864.6
media		540.61	40.03	188.25	2463.14	506.78	32.03	137.35	1912.31

Tabella C.7: VNS e VNS–A con $\overline{\kappa} = 1$ dopo 3600 secondi in un grafo non euclideo su PC2.

		VNS(1)				VNS–A(1)			
R	P	u	l	w	f	u	l	w	f
719	228	685.6	1	281	3496.6	656.4	3	280	3459.4
731	219	659	5	312	3784	675	1	310	3776
732	220	646.4	2	293	3578.4	693.6	0	368	4373.6
733	229	644.8	3	284	3487.8	695.4	0	330	3995.4
737	219	716.8	0	325	3966.8	723.2	0	352	4243.2
737	223	658.8	2	354	4200.8	707	0	357	4277
740	218	672.6	0	308	3752.6	655.4	1	263	3286.4
742	215	633.8	4	291	3547.8	601.4	3	241	3014.4
742	219	652.2	3	305	3705.2	649.6	4	320	3853.6
743	238	715.6	0	329	4005.6	629	5	235	2984
747	228	546.2	56	246	3062.2	376.4	34	24	650.4
747	237	626.4	2	258	3208.4	655.6	1	262	3276.6
750	225	704.4	0	333	4034.4	688.6	1	323	3919.6
751	222	693.6	0	351	4203.6	680.4	0	299	3670.4
752	218	724	0	343	4154	576.6	5	256	3141.6
753	222	677	0	311	3787	691.6	0	352	4211.6
754	215	685.6	1	319	3876.6	677.2	1	324	3918.2
757	241	626	4	269	3320	670.8	0	280	3470.8
762	218	575.4	44	197	2589.4	406	22	63	1058
763	228	663.8	0	295	3613.8	663.8	0	309	3753.8
763	231	581.2	60	310	3741.2	808.4	75	460	5483.4
765	216	580	5	264	3225	678.6	1	345	4129.6
765	223	631.4	2	285	3483.4	625.8	3	235	2978.8
771	221	483.8	43	140	1926.8	639.2	38	357	4247.2
771	224	569	61	292	3550	492.4	42	190	2434.4
773	217	694.2	0	294	3634.2	724.4	0	360	4324.4
775	222	588.6	39	254	3167.6	620.6	33	309	3743.6
776	221	675	1	293	3606	689.4	1	360	4290.4
777	216	524.2	35	159	2149.2	398.8	33	0	431.8
778	224	606.2	52	244	3098.2	549.2	45	227	2864.2
781	229	573.2	53	265	3276.2	608.6	43	307	3721.6
783	218	672.4	1	311	3783.4	632.6	3	303	3665.6
785	226	435.4	42	76	1237.4	499.4	27	168	2206.4
787	225	556.2	40	196	2556.2	527.8	32	155	2109.8
788	220	537.4	41	193	2508.4	476.6	31	122	1727.6
788	220	666	2	314	3808	654.6	3	253	3187.6
789	222	420	42	40	862	411	27	16	598
789	227	549.2	39	207	2658.2	524.8	35	173	2289.8
789	240	424.8	43	41	877.8	392.2	24	0	416.2
790	225	525.6	42	194	2507.6	487.2	35	119	1712.2
790	236	439.4	41	79	1270.4	438.2	28	102	1486.2
794	232	389.6	41	61	1040.6	481.2	29	133	1840.2
795	220	517.4	36	157	2123.4	604.4	35	222	2859.4
797	229	638.6	38	246	3136.6	425.8	23	74	1188.8
800	228	642	51	272	3413	594	43	228	2917
800	228	643	54	284	3537	526.6	49	171	2285.6
801	241	581.6	52	216	2793.6	449.8	30	46	939.8
803	224	497.8	32	146	1989.8	612.6	38	227	2920.6
804	223	359.4	30	0	389.4	374.4	23	0	397.4
806	223	425	24	35	799	525.6	26	119	1741.6
807	234	516.6	37	130	1853.6	377.8	27	0	404.8
808	236	629.6	57	292	3606.6	663.4	43	332	4026.4
811	227	607.4	48	270	3355.4	418.8	32	71	1160.8
812	231	599.8	42	253	3171.8	464.4	35	102	1519.4
814	225	505.8	47	166	2212.8	406.2	25	3	461.2
822	229	480.4	39	168	2199.4	543.4	36	175	2329.4
825	232	642	56	286	3558	651.6	47	288	3578.6
834	223	620.6	46	281	3476.6	404	23	34	767
841	233	766.4	81	451	5357.4	767.6	81	455	5398.6
848	228	576.6	39	203	2645.6	411.6	23	0	434.6
media		593.01	27.68	239.53	3016.03	572.60	21.80	213.15	2725.90

Tabella C.8: VNS e VNS–A con $\overline{\kappa} = 1$ dopo 3600 secondi in un grafo euclideo su PC2.

		VNS(10)				VNS–A(10)			
R	P	u	l	w	f	u	l	w	f
719	228	372.8	23	0	395.8	392.2	23	0	415.2
731	219	676.4	33	263	3339.4	443.2	20	46	923.2
732	220	378.8	22	0	400.8	384.6	21	12	525.6
733	229	389	31	0	420	400	22	28	702
737	219	397.4	26	0	423.4	421.6	23	35	794.6
737	223	390.6	23	0	413.6	392.4	21	0	413.4
740	218	468.4	30	99	1488.4	394.8	25	0	419.8
742	215	429	30	33	789	395.6	21	0	416.6
742	219	609.2	34	192	2563.2	406	16	0	422
743	238	381.2	24	0	405.2	396.8	21	3	447.8
747	228	413	26	37	809	416.4	23	20	639.4
747	237	385.2	24	0	409.2	394	28	0	422
750	225	675	38	272	3433	403.2	24	0	427.2
751	222	550.6	37	246	3047.6	421.4	22	30	743.4
752	218	577.6	37	218	2794.6	412	26	6	498
753	222	420.6	22	29	732.6	552.6	28	186	2440.6
754	215	499.8	32	108	1611.8	408	18	8	506
757	241	418.2	34	25	702.2	414.8	26	38	820.8
762	218	495	22	100	1517	522.6	21	119	1733.6
763	228	622.2	34	325	3906.2	755.6	44	371	4509.6
763	231	416	22	50	938	427.8	23	60	1050.8
765	216	528.4	28	173	2286.4	647	30	262	3297
765	223	522.4	36	115	1708.4	404.2	23	0	427.2
771	221	447.4	24	73	1201.4	663	33	295	3646
771	224	650.8	34	282	3504.8	420.8	26	14	586.8
773	217	409.4	24	1	443.4	507.2	21	100	1528.2
775	222	504.8	38	139	1932.8	410.8	22	18	612.8
776	221	468.2	30	98	1478.2	435.4	27	97	1432.4
777	216	442.2	26	111	1578.2	447.6	21	54	1008.6
778	224	400.2	26	26	686.2	410.2	27	31	747.2
781	229	693	42	287	3605	423.4	19	21	652.4
783	218	403.2	28	36	791.2	685.6	35	291	3630.6
785	226	600.2	46	236	3006.2	479.4	23	85	1352.4
787	225	426.2	25	22	671.2	427.8	28	41	865.8
788	220	481.8	30	121	1721.8	681.2	33	356	4274.2
788	220	682	35	273	3447	414.6	18	12	552.6
789	222	680.6	47	331	4037.6	421.4	22	64	1083.4
789	227	396.4	27	18	603.4	407.2	25	28	712.2
789	240	410.4	34	30	744.4	458	27	56	1045
790	225	622.2	35	298	3637.2	456.8	24	71	1190.8
790	236	657	37	286	3554	420.2	22	37	812.2
794	232	494	34	119	1718	423	27	40	850
795	220	440.2	24	41	874.2	432.4	24	28	736.4
797	229	425.6	21	12	566.6	485.2	20	64	1145.2
800	228	407.8	23	26	690.8	415.8	24	16	599.8
800	228	692	37	300	3729	442.2	20	75	1212.2
801	241	413.8	36	15	599.8	474.8	26	70	1200.8
803	224	474.4	28	89	1392.4	444.6	25	57	1039.6
804	223	416.4	31	121	1657.4	426.6	27	81	1263.6
806	223	506	31	126	1797	424	21	28	725
807	234	418	30	4	488	423.2	27	29	740.2
808	236	670	45	291	3625	550.8	37	185	2437.8
811	227	488	28	126	1776	781.8	48	420	5029.8
812	231	447.2	32	87	1349.2	443.6	26	66	1129.6
814	225	423.6	22	24	685.6	432.4	26	56	1018.4
822	229	435.2	28	51	973.2	443.8	26	56	1029.8
825	232	443.4	30	44	913.4	451.4	26	66	1137.4
834	223	642.6	44	280	3486.6	454.4	20	52	994.4
841	233	440.8	26	60	1066.8	454.6	23	63	1107.6
848	228	437.6	23	20	660.6	456.8	23	53	1009.8
media		491.82	30.48	113.15	1653.81	460.65	24.97	73.33	1218.95

Tabella C.9: VNS e VNS–A con $\overline{\kappa} = 10$ dopo 3600 secondi in un grafo non euclideo su PC2.

		VNS(10)				VNS–A(10)			
R	P	u	l	w	f	u	l	w	f
719	228	650.8	3	280	3453.8	637.8	3	281	3450.8
731	219	684	0	338	4064	672	1	340	4073
732	220	672.4	0	276	3432.4	648.8	0	245	3098.8
733	229	651	2	275	3403	664	0	291	3574
737	219	700.4	0	290	3600.4	663.6	2	272	3385.6
737	223	656	1	316	3817	691.2	0	388	4571.2
740	218	680.6	0	292	3600.6	634.2	2	228	2916.2
742	215	642.4	2	272	3364.4	590.4	3	286	3453.4
742	219	702.8	0	332	4022.8	653	1	292	3574
743	238	614.6	1	235	2965.6	632.4	4	278	3416.4
747	228	351.4	20	0	371.4	376.2	25	0	401.2
747	237	653.8	0	256	3213.8	684.2	0	300	3684.2
750	225	633.6	5	229	2928.6	680.8	0	318	3860.8
751	222	687.6	0	316	3847.6	655.8	3	291	3568.8
752	218	665.6	0	342	4085.6	615.4	4	267	3289.4
753	222	608.6	4	233	2942.6	692.2	0	348	4172.2
754	215	622.6	1	238	3003.6	597.8	9	340	4006.8
757	241	641.4	1	255	3192.4	695.6	0	325	3945.6
762	218	485.8	22	77	1277.8	367	18	0	385
763	228	678.6	2	299	3670.6	651.8	5	281	3466.8
763	231	377.6	24	26	661.6	388.2	18	0	406.2
765	216	656.4	0	282	3476.4	580.6	3	184	2423.6
765	223	603.8	5	220	2808.8	639.2	4	266	3303.2
771	221	446	31	46	937	377.6	21	0	398.6
771	224	630.8	40	241	3080.8	384.4	21	0	405.4
773	217	710.2	0	322	3930.2	668.6	3	250	3171.6
775	222	501.2	27	91	1438.2	504	30	132	1854
776	221	685.4	1	294	3626.4	709.8	2	340	4111.8
777	216	530	31	173	2291	375.8	19	0	394.8
778	224	372.2	26	0	398.2	389.4	18	0	407.4
781	229	364.6	23	0	387.6	395.4	21	0	416.4
783	218	652.2	3	298	3635.2	645.2	1	269	3336.2
785	226	403	30	18	613	387.8	27	3	444.8
787	225	681.8	49	302	3750.8	386	21	2	427
788	220	528.8	36	161	2174.8	596.6	34	283	3460.6
788	220	684.4	0	275	3434.4	623.8	2	230	2925.8
789	222	405.2	29	0	434.2	481.6	38	61	1129.6
789	227	447	30	36	837	389.4	22	0	411.4
789	240	373.4	25	0	398.4	626	34	224	2900
790	225	513.2	35	155	2098.2	394	29	0	423
790	236	574.2	46	231	2930.2	397.8	22	0	419.8
794	232	387.6	29	0	416.6	393.4	26	0	419.4
795	220	377.2	25	0	402.2	405	24	0	429
797	229	381.4	19	0	400.4	514.6	27	107	1611.6
800	228	424.4	30	48	934.4	391.2	22	0	413.2
800	228	431.4	28	30	759.4	393.4	23	0	416.4
801	241	392.8	23	0	415.8	397.4	23	0	420.4
803	224	374.6	24	0	398.6	401	23	9	514
804	223	363.8	29	0	392.8	381	20	0	401
806	223	374.2	25	0	399.2	393.4	20	0	413.4
807	234	547.8	35	138	1962.8	397	21	0	418
808	236	574.6	36	170	2310.6	623.4	35	262	3278.4
811	227	608.2	50	196	2618.2	628.2	46	224	2914.2
812	231	458.8	29	73	1217.8	419.8	22	47	911.8
814	225	548.4	38	168	2266.4	394.2	19	0	413.2
822	229	407.6	29	35	786.6	448.8	27	86	1335.8
825	232	416.4	28	9	534.4	517.8	32	123	1779.8
834	223	539.8	31	126	1830.8	410.8	20	0	430.8
841	233	462.8	26	54	1028.8	425.2	34	52	979.2
848	228	394.4	27	0	421.4	403.8	25	0	428.8
media		538.19	18.60	156.15	2118.29	519.75	15.98	142.08	1956.56

Tabella C.10: VNS e VNS–A con $\overline{\kappa} = 10$ dopo 3600 secondi in un grafo euclideo su PC2.

		VNS(50)				VNS–A(50)			
R	P	u	l	w	f	u	l	w	f
719	228	376	26	0	402	382.4	17	0	399.4
731	219	533.6	28	143	1991.6	538.8	27	136	1925.8
732	220	379.6	18	16	557.6	380	19	0	399
733	229	403.2	25	20	628.2	396.6	21	18	597.6
737	219	396	25	34	761	404.2	22	44	866.2
737	223	397.2	22	0	419.2	405.6	21	16	586.6
740	218	378.6	23	0	401.6	388.4	29	0	417.4
742	215	385.2	25	0	410.2	403.8	21	10	524.8
742	219	598	35	195	2583	726.2	36	339	4152.2
743	238	394.8	29	17	593.8	395.2	26	24	661.2
747	228	384.8	25	0	409.8	403	21	0	424
747	237	469.6	29	94	1438.6	411.4	25	25	686.4
750	225	409.2	23	8	512.2	486.6	24	106	1570.6
751	222	527.8	28	175	2305.8	429.6	24	86	1313.6
752	218	524.8	27	170	2251.8	400.6	19	0	419.6
753	222	409.4	20	16	589.4	403.4	17	0	420.4
754	215	487.2	22	102	1529.2	393.8	19	14	552.8
757	241	402.6	29	36	791.6	574.2	37	193	2541.2
762	218	667.6	37	323	3934.6	405	17	28	702
763	228	665.2	40	259	3295.2	429.4	22	49	941.4
763	231	424.4	21	53	975.4	504.6	25	124	1769.6
765	216	397.6	22	10	519.6	445.2	18	70	1163.2
765	223	494	28	105	1572	702.2	45	346	4207.2
771	221	428.2	25	47	923.2	416.8	19	29	725.8
771	224	542.4	26	215	2718.4	599	30	276	3389
773	217	492	29	104	1561	449.4	20	88	1349.4
775	222	489.4	31	123	1750.4	700.2	36	358	4316.2
776	221	473.8	29	134	1842.8	667.2	38	302	3725.2
777	216	398.2	24	1	432.2	403.8	22	1	435.8
778	224	471	29	77	1270	412	22	29	724
781	229	723	44	342	4187	409.6	21	3	460.6
783	218	397.4	22	0	419.4	418.6	24	54	982.6
785	226	524.8	31	155	2105.8	434.8	23	25	707.8
787	225	605.6	35	203	2670.6	646.2	33	257	3249.2
788	220	549.2	33	182	2402.2	479.2	23	93	1432.2
788	220	630.4	33	249	3153.4	411.6	20	0	431.6
789	222	416.2	29	33	775.2	424.2	23	36	807.2
789	227	407.4	25	20	632.4	427.2	23	35	800.2
789	240	544.2	36	145	2030.2	440.8	28	46	928.8
790	225	427.2	28	48	935.2	490.8	25	91	1425.8
790	236	577.2	28	172	2325.2	462.6	24	104	1526.6
794	232	632.6	34	237	3036.6	437.8	26	40	863.8
795	220	431.2	28	37	829.2	420.8	25	35	795.8
797	229	707.4	32	317	3909.4	461.8	24	66	1145.8
800	228	407	22	1	439	439.8	29	55	1018.8
800	228	658.4	37	302	3715.4	436.8	21	38	837.8
801	241	431.8	32	37	833.8	543.6	33	143	2006.6
803	224	429.6	29	45	908.6	441.8	23	80	1264.8
804	223	411.4	29	98	1420.4	432.4	29	98	1441.4
806	223	506.4	22	128	1808.4	422.2	20	35	792.2
807	234	432.8	29	53	991.8	411.2	21	10	532.2
808	236	424.8	22	29	736.8	445.4	23	53	998.4
811	227	660.8	45	274	3445.8	443	22	37	835
812	231	436.2	31	46	927.2	426.2	25	21	661.2
814	225	427.2	20	15	597.2	427	19	29	736
822	229	571.8	29	187	2470.8	559.8	25	163	2214.8
825	232	722.2	51	343	4203.2	445.8	30	65	1125.8
834	223	648.2	36	259	3274.2	456.8	23	55	1029.8
841	233	457.4	31	72	1208.4	448.4	26	53	1004.4
848	228	436.8	25	31	771.8	546.6	29	183	2405.6
media		490.63	28.80	108.95	1608.93	462.52	24.65	78.57	1272.84

Tabella C.11: VNS e VNS–A con $\overline{\kappa} = 50$ dopo 3600 secondi in un grafo non euclideo su PC2.

		VNS(50)				VNS–A(50)			
R	P	u	l	w	f	u	l	w	f
719	228	673	0	271	3383	669.6	0	280	3469.6
731	219	682.2	1	345	4133.2	660.2	0	335	4010.2
732	220	642.6	0	294	3582.6	642.4	1	263	3273.4
733	229	650.2	1	274	3391.2	648	1	291	3559
737	219	694	1	302	3715	711.4	0	327	3981.4
737	223	601.6	3	286	3464.6	684.6	0	355	4234.6
740	218	680.2	0	322	3900.2	690.2	0	327	3960.2
742	215	662	0	300	3662	638.8	1	292	3559.8
742	219	688.6	0	326	3948.6	659.2	0	317	3829.2
743	238	673.6	0	361	4283.6	733.4	0	346	4193.4
747	228	355.6	25	0	380.6	445.2	28	97	1443.2
747	237	639.6	0	318	3819.6	653.2	0	273	3383.2
750	225	626.2	1	286	3487.2	660.6	0	267	3330.6
751	222	610.2	2	235	2962.2	604.8	1	256	3165.8
752	218	644.6	1	279	3435.6	690	0	342	4110
753	222	657.6	0	320	3857.6	734.8	0	449	5224.8
754	215	633.4	0	293	3563.4	618.4	0	300	3618.4
757	241	606	1	258	3187	630.4	2	246	3092.4
762	218	631.2	38	246	3129.2	510.2	28	99	1528.2
763	228	696	0	326	3956	678.2	1	317	3849.2
763	231	379.2	21	0	400.2	386.4	21	9	497.4
765	216	609.6	2	235	2961.6	640.8	0	286	3500.8
765	223	653.6	1	321	3864.6	630.2	0	257	3200.2
771	221	669	35	280	3504	497.8	26	114	1663.8
771	224	380.4	23	40	803.4	513.4	26	131	1849.4
773	217	644.6	0	254	3184.6	675	2	271	3387
775	222	484.4	31	80	1315.4	381.8	20	0	401.8
776	221	696.8	2	317	3868.8	673.8	1	286	3534.8
777	216	565.2	33	244	3038.2	375.2	20	0	395.2
778	224	419.2	24	31	753.2	375.2	18	0	393.2
781	229	509.6	32	125	1791.6	432	29	55	1011
783	218	666.6	0	290	3566.6	671	0	317	3841
785	226	578.4	33	184	2451.4	538	28	156	2126
787	225	368.2	25	0	393.2	370.8	24	0	394.8
788	220	458.4	28	76	1246.4	565	32	201	2607
788	220	568.8	0	171	2278.8	623.6	0	228	2903.6
789	222	371	27	0	398	379.6	23	0	402.6
789	227	463.4	32	84	1335.4	658.4	41	287	3569.4
789	240	379.2	28	0	407.2	402.8	23	31	735.8
790	225	388	29	0	417	388.6	25	18	593.6
790	236	481	25	73	1236	394.4	22	0	416.4
794	232	370	25	0	395	388.2	21	0	409.2
795	220	434	33	42	887	414.8	26	30	740.8
797	229	501.4	27	102	1548.4	397	20	0	417
800	228	377.2	25	0	402.2	436.4	25	63	1091.4
800	228	382.6	23	0	405.6	394.6	22	0	416.6
801	241	480.4	30	98	1490.4	451	32	61	1093
803	224	458.8	32	79	1280.8	501.2	28	92	1449.2
804	223	367.2	20	35	737.2	388.2	24	54	952.2
806	223	506	29	113	1665	571.2	30	207	2671.2
807	234	487.2	28	87	1385.2	439.2	30	70	1169.2
808	236	436.4	28	55	1014.4	464.8	30	68	1174.8
811	227	557.2	37	172	2314.2	596.2	33	214	2769.2
812	231	407.6	24	16	591.6	418.2	26	31	754.2
814	225	388	22	0	410	418.8	20	41	848.8
822	229	467.2	28	74	1235.2	405	22	4	467
825	232	629.2	46	232	2995.2	436.6	25	38	841.6
834	223	395.2	26	0	421.2	407.4	24	18	611.4
841	233	520	36	136	1916	417.8	29	40	846.8
848	228	501.4	33	109	1624.4	461.2	30	76	1251.2
media		535.83	17.62	163.28	2186.28	532.42	15.68	158.88	2136.94

Tabella C.12: VNS e VNS–A con $\overline{\kappa} = 50$ dopo 3600 secondi in un grafo euclideo su PC2.

Lista dei simboli

0	centrale operativa, luogo dove le infermiere partono e ritornano
a_s	tempo di lavoro associato al servizio s
α	pesi delle componenti della funzione obiettivo
A	archi del grafo
b	numero di infermiere in turno
c_s	costo associato ad un servizio s, come stima del tempo lavorativo a_s e stima del tempo di spostamento ad un altro luogo d_s, cioè $c_s = a_s + d_s$
$\tilde{c}_s$	$c_s = a_s + \tilde{d}_s$
$\hat{c}_s$	$c_s = a_s + \hat{d}_s$
$d_{ss'}$	tempo speso nel spostarsi dal luogo del servizio s al luogo del servizio s'
d_s	stima del tempo di spostamento da un altro luogo s ad un qualsiasi altro luogo di servizio, per esempio la media
$\tilde{d}_s$	$\tilde{d}_s = \sum_{s' \in Services Depot} d_{ss'} / \mid S_0 \mid$
$\hat{d}_s$	primo quartile dell'insieme ordinato delle distanze $d_{ss'}$
δ	variabile che indentifica una distanza compresa tra $[0, \delta_{max}]$
δ_{max}	massimo valore di δ
Δ	incremento di δ
e_s	intermezzo, tempo che deve passare prima di ripetere il servizio s
f_s	frequenza, ossia numero di ripetizioni del servizio s
γ_s	luogo di svolgimento del servizio s
$\overline{G}_i$	insieme delle coppie $\cup_{h \in H}\{q_i^h\} \times \{h\}$
G_s	insieme delle coppie $\cup_{h \in H}\{g_s^h\} \times \{h\}$
g_s^h	*true* se il servizio s può essere assegnato ad un infermiera il giorno h
$\mathcal{G}$	grafo
H	orizzonte temporale, per esempio una settimana
h	sequenza temporale, per esempio un giorno
$\mathcal{I}$	insieme di istanze
i	tour o anche turno, ossia una sequenza di servizi
j	infermiera
$\underline{\kappa}$	quantità di miglioramenti
$\overline{\kappa}$	massimo numero di miglioramenti
l	fidelizzazione infermiera-paziente
ℓ	tempo di lavoro del turno, totale dei tempi di lavoro dei servizi del turno e dei tempi di spostamento
ℓ^{jh}	durata del turno assegnato all'infermiera j il giorno h
ℓ_{min}	durata del turno min con minimo tempo di lavoro
ℓ_{max}	durata del turno max con massimo tempo di lavoro
L	durata massima di un turno senza tempo di lavoro straordinario
$\mathcal{L}$	lagrangiano
$\underline{\lambda}$	scalare
$\boldsymbol{\lambda}$	moltiplicatori di Lagrange
Λ	lower bound sul numero di nodi

M costante di valore molto grande

$\mathcal{N}$ intorno, l'insieme delle soluzioni ottenibili dall'applicazione di un operatore `op`

N insieme di infermiere

`op` operatore, per esempio `move`

P insieme di pazienti

p paziente

p_s paziente, al quale è associato il servizio s

π_k probabilità che un operatore $\texttt{op}_k$ sia applicato

q_i^h *true* se il turno i può essere assegnato ad un infermiera il giorno h

Q probabilità con la quale accettare una soluzione con stesso valore della funzione obiettivo

r richiesta di servizio, ossia la tupla $(h, \texttt{tipo}_s, p_s)$

R insieme di richieste

$\underline{R}$ insieme di richieste ripetute più volte nello stesso giorno

$\overline{R}$ $\overline{R} = R \setminus \underline{R}$

s servizio, ossia la coppia tipo di servizio-paziente $(\texttt{tipo}_s, p_s)$

S insieme di servizi

S_0 insieme, comprensivo del centro operativo 0, dei luoghi dei relativi servizi

S_0^h insieme, comprensivo del centro operativo 0, dei luoghi dei relativi servizi svolti il giorno h

S_0^{jh} insieme, comprensivo del centro operativo 0, dei luoghi dei relativi servizi svolti il giorno h ed assegnati ad un infermiera j

S^h insieme dei luoghi dei relativi servizi svolti il giorno h

S^{jh} insieme dei luoghi dei relativi servizi svolti il giorno h ed assegnati ad un infermiera j

S_+ insieme dei luoghi dei relativi servizi che richiedono un intermezzo

S_+^h insieme dei luoghi dei relativi servizi svolti il giorno h e che richiedono un intermezzo

ς_k punteggio associato ad un operatore $\texttt{op}_k$, quantificazione dei successi

σ soluzione, ad ogni turno in $\mathcal{T}$ è assegnata una coppia infermiera-giorno

$\varsigma_{ss'}$ criterio di risparmio (saving)

`step` contatore dell'iterazione dell'algoritmo

$\texttt{step}_{max}$ massimo numero di iterazioni in un algoritmo

T turno o tour, ossia una sequenza di servizi

$\mathcal{T}$ insieme di turni

`Tipi` insieme dei tipi di servizio

$\texttt{tipo}_s$ tipo del servizio s, ad esempio prelievo ematico

U insieme di richieste urgenti

u carico di lavoro medio giornaliero

u^j carico di lavoro medio giornaliero dell'infermiera j

u^h carico di lavoro medio giornaliero nella giornata h

Υ upper bound sul numero di nodi

V vertici del grafo

w tempo di lavoro straordinario

W^{max} massimo tempo di lavoro straordinario

$\underline{W}_{max}$ massimo tempo di lavoro straordinario

Bibliografia

Alves, C., P. Brás, J. Valério de Carvalho e T. Pinto (2012). "A variable neighborhood search algorithm for the leather nesting problem". In: *Mathematical Problems in Engineering* 2012, pp. 1–28 (cit. a p. 56).

Amiri, M., M. Zandieh, M. Yazdani e A. Bagheri (2010). "A variable neighbourhood search algorithm for the flexible job-shop scheduling problem". In: *International Journal of Production Research* 48.19, pp. 5671–5689 (cit. a p. 56).

Applegate, D. L., R. E. Bixby, V. Chvàtal e W. J. Cook (2006). *The Traveling Salesman Problem: A Computational Study*. Princeton University Press, pp. 126–127 (cit. alle pp. 5, 16, 29, 33).

Arbib, C., M. Lucertini e F. Nicolò (1991). "Workload balance and part-transfer minimization in flexible manufacturing systems". In: *International Journal of Flexible Manufacturing Systems* 3 (1), pp. 5–25 (cit. a p. 23).

Azar, Y. (1998). "On-line load balancing". In: *Online Algorithms*. A cura di A. Fiat e G. Woeginger. Vol. 1442. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, pp. 178–195 (cit. a p. 19).

Babin, G., S. Deneault e G. Laporte (2007). "Improvements to the Or-Opt Heuristic for the Symmetric Travelling Salesman Problem". In: *The Journal of the Operational Research Society* 58.3, pp. 402–407 (cit. a p. 75).

Balas, E. (1999). "New classes of efficiently solvable generalized Traveling Salesman Problems". In: *Annals of Operations Research* 86 (0), pp. 529–558 (cit. a p. 38).

Balas, E. e N. Simonetti (2001). "Linear Time Dynamic-Programming Algorithms for New Classes of Restricted TSPs: A Computational Study." In: *INFORMS Journal on Computing* 13.1, p. 56 (cit. a p. 38).

Baldacci, R., A. Mingozzi e R. Roberti (2012). "Recent exact algorithms for solving the vehicle routing problem under capacity and time window constraints". In: *European Journal of Operational Research* 218.1, pp. 1–6 (cit. a p. 3).

Begur, S. V., D. M. Miller e J. R. Weaver (1997a). "An Integrated Spatial DSS for Scheduling and Routing Home-Health-Care Nurses". In: *Interfaces* 27.4, pp. 35–48 (cit. a p. 3).

— (1997b). "An Integrated Spatial DSS for Scheduling and Routing Home-Health-Care Nurses". In: *Interfaces* 27.4, pp. 35–48 (cit. a p. 20).

Bektas, T. (2006). "The multiple traveling salesman problem: an overview of formulations and solution procedures". In: *Omega* 34.3, pp. 209–219 (cit. a p. 16).

Bellmore, M. e S. Hong (1974). "Transformation of Multisalesman Problem to the Standard Traveling Salesman Problem". In: *J. ACM* 21.3, pp. 500–504 (cit. a p. 16).

Benders, J. F. (1962). "Partitioning procedures for solving mixed-variables programming problems". In: *Numerische Mathematik* 4 (1), pp. 238–252 (cit. a p. 31).

Bertels, S. e T. Fahle (2006). "A hybrid setup for a hybrid scenario: combining heuristics for the home health care problem". In: *Computers & Operations Research* 33.10, pp. 2866–2890 (cit. alle pp. 2, 3, 38).

Blais, M., S. D. Lapierre e G. Laporte (2003). "Solving a Home–Care Districting Problem in an Urban Setting". In: *Journal of the Operational Research Society* 54.11, pp. 1141–1147 (cit. alle pp. 19, 20).

Blakeley, F., B. Argüello, B. Cao, W. Hall e J. Knolmajer (2003). "Optimizing Periodic Maintenance Operations for Schindler Elevator Corporation". In: *Interfaces* 33.1, pp. 67–79 (cit. a p. 22).

Boccafoli, M., F. Malucelli e M. Nonato (2011). "A Study on the Bi-Objectives Distance-Constrained Vehicle Routing Problem Relaxation". In: *AIRO 2011 Conference: Operational Research in Transportation and Logistics*. Vol. 1. AIRO, p. 179 (cit. a p. 17).

Borsani, V., A. Matta, G. Beschi e F. Sommaruga (2006). "A Home Care Scheduling Model For Human Resources". In: *2006 International Conference on Service Systems and Service Management*. Vol. 1, pp. 449 –454 (cit. alle pp. 3, 20, 22).

Boschetti, M., V. Maniezzo, M. Roffilli e A. Bolufé Röhler (2009). "Matheuristics: Optimization, Simulation and Control". In: *Hybrid Metaheuristics*. A cura di M. J. Blesa, C. Blum, L. Di Gaspero, A. Roli, M. Sampels e A. Schaerf. Vol. 5818. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, pp. 171–177 (cit. a p. 5).

Bozkaya, B., E. Erkut e G. Laporte (2003). "A tabu search heuristic and adaptive memory procedure for political districting". In: *European Journal of Operational Research* 144.1, pp. 12–26 (cit. a p. 19).

Bräysy, O. (2003). "A Reactive Variable Neighborhood Search for the Vehicle-Routing Problem with Time Windows". In: *INFORMS Journal on Computing* 15.4, pp. 347–368 (cit. a p. 56).

Bredström, D. e M. Rönnqvist (2008). "Combined vehicle routing and scheduling with temporal precedence and synchronization constraints". In: *European Journal of Operational Research* 191.1, pp. 19–31 (cit. a p. 21).

Burke, E. K., P. De Causmaecker, G. Vanden Berghe e H. Van Landeghem (2004). "The State of the Art of Nurse Rostering". In: *Journal of Scheduling* 7 (6), pp. 441–499 (cit. alle pp. 2, 20).

Burke, E. K., T. Curtois, G. Post, R. Qu e B. Veltman (2008). "A hybrid heuristic ordering and variable neighbourhood search for the nurse rostering problem". In: *European Journal of Operational Research* 188.2, pp. 330–341 (cit. a p. 56).

Cattafi, M., R. Herrero, M. Gavanelli, M. Nonato, F. Malucelli e J.-J. Ramos (2012). "Improving Quality and Efficiency in Home Health Care: an application of Constraint Logic Programming for the Ferrara NHS unit". In: *Technical Communications of the 28th International Conference on Logic Programming (ICLP'12)*. A cura di A. Dovier e V. S. Costa.

Vol. 17. Leibniz International Proceedings in Informatics (LIPICS). Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, pp. 415–424 (cit. a p. 4).

Cerri, L. e M. Tognoli (2010). "Valutazione delle prestazioni di tecniche di inoltro collaborativo basata su modelli di ottimizzazione". Tesi di laurea mag. Politecnico di Milano.

Cheng, E. e J. L. Rich (1998). *A Home Health Care Routing and Scheduling Problem*. Rapp. tecn. TR98_04. Computational e applied mathematics (CAAM) Rice University (cit. alle pp. 2, 3).

Chiarandini, M., A. Schaerf e F. Tiozzo (2000). "Solving Employee Timetabling Problems with Flexible Workload using Tabu Search". In: *Proceedings of the 3rd International Conference on the Practice and Theory of Automated Timetabling (PATAT-2000)*. Konstanz, Germany, pp. 298–302 (cit. a p. 22).

Christofides, N. e J. E. Beasley (1984). "The period routing problem". In: *Networks* 14.2, pp. 237–256 (cit. a p. 18).

Clarke, G. e J. W. Wright (1964). "Scheduling of Vehicles from a Central Depot to a Number of Delivery Points". In: *Operations Research* 12.4, pp. 568–581 (cit. alle pp. 72, 82, 83).

Contreras, I., J.-F. Cordeau e G. Laporte (nov. 2011). "Benders Decomposition for Large-Scale Uncapacitated Hub Location". In: *Oper. Res.* 59.6, pp. 1477–1490 (cit. a p. 5).

Cordeau, J.-F., G. Laporte e A. Mercier (2001). "A Unified Tabu Search Heuristic for Vehicle Routing Problems with Time Windows". In: *Journal of the Operational Research Society* 52.8, pp. 928–936 (cit. alle pp. 2, 18).

Davidon, W. C. (1991). "Variable Metric Method for Minimization". In: *SIAM Journal on Optimization* 1.1, pp. 1–17 (cit. a p. 56).

Deckard, G., M. Meterko e D. Field (1994). "Physician Burnout: An Examination of Personal, Professional, and Organizational Relationships". In: *Medical Care* 32.7, pp. 745–754 (cit. a p. 19).

Dell'Amico, M., F. Maffioli e P. Värbrand (1995). "On Prize-collecting Tours and the Asymmetric Travelling Salesman Problem". In: *International Transactions in Operational Research* 2.3, pp. 297–308 (cit. a p. 39).

Desrosiers, J. e M. E. Lübbecke (2005). "A Primer in Column Generation". In: *Column Generation*. A cura di G. Desaulniers, J. Desrosiers e M. M. Solomon. Springer US, pp. 1–32 (cit. a p. 38).

Desrosiers, J., M. Sauve e F. Soumis (1988). "Lagrangian Relaxation Methods for Solving the Minimum Fleet Size Multiple Traveling Salesman Problem with Time Windows". In: *Management Science* 34.8, pp. 1005–1022 (cit. a p. 106).

El-Sherbeny, N. (2001). "Resolution of a vehicle routing problem with multi-objective simulated annealing method". Tesi di dott. Mons, Belgique: Faculté Polytechnique de Mons (cit. a p. 21).

Eveborn, P., M. Rönnqvist, H. Einarsdóttir, M. Eklund, K. Lidén e M. Almroth (2009). "Operations research improves quality and efficiency in home care". In: *Interfaces* 39.1, pp. 18–34 (cit. a p. 2).

Eveborn, P., P. Flisberg e M. Rönnqvist (2006). "Laps Care–an operational system for staff planning of home care". In: *European Journal of Operational Research* 171.3, pp. 962–976 (cit. alle pp. 2, 3).

Feillet, D., P. Dejax e M. Gendreau (2005). "Traveling Salesman Problems with Profits". In: *Transportation Science* 39.2, pp. 188–205 (cit. a p. 39).

Fisher, M. L. (2004). "The Lagrangian Relaxation Method for Solving Integer Programming Problems". In: *Management Science* 50.12, pp. 1861–1871 (cit. a p. 44).

Fleszar, K. e K. S. Hindi (2004). "Solving the resource-constrained project scheduling problem by a variable neighbourhood search". In: *European Journal of Operational Research* 155.2, pp. 402–413.

Fletcher, R. e M. J. D. Powell (1963). "A Rapidly Convergent Descent Method for Minimization". In: *The Computer Journal* 6.2, pp. 163–168 (cit. a p. 56).

Francis, P., K. Smilowitz e M. Tzur (nov. 2006). "The Period Vehicle Routing Problem with Service Choice". In: *Transportation Science* 40 (4), pp. 439–454 (cit. alle pp. 2, 38).

Francis, P. M., K. R. Smilowitz e M. Tzur (2008). "The Period Vehicle Routing Problem and its Extensions". In: *The Vehicle Routing Problem: Latest Advances and New Challenges*. A cura di B. Golden, S. Raghavan, E. Wasil, R. Sharda e S. Voß. Vol. 43. Operations Research/Computer Science Interfaces Series. Springer US, pp. 73–102 (cit. alle pp. 17, 38).

Geoffrion, A. M. (1974). "Lagrangean relaxation for integer programming". In: *Approaches to Integer Programming*. Vol. 2. Mathematical Programming Studies. Springer Berlin Heidelberg, pp. 82–114 (cit. a p. 43).

Glaab, H. (2002). "A new variant of a vehicle routing problem: Lower and upper bounds". In: *European Journal of Operational Research* 139.3, pp. 557–577 (cit. a p. 23).

Glover, F. (1968). "Surrogate Constraints". In: *Operations Research* 16.4, pp. 741–749 (cit. a p. 46).

— (1975). "Surrogate Constraint Duality in Mathematical Pragramming". In: *Operations Research* 23.3, pp. 434–451.

Greenberg, H. J. e W. P. Pierskalla (1970). "Surrogate Mathematical Programming". In: *Operations Research* 18.5, pp. 924–939 (cit. a p. 46).

Hansen, P., N. Mladenović e J. A. Moreno-Pérez (2010). "Variable neighbourhood search: methods and applications". In: *Annals of Operations Research* 175 (1), pp. 367–407 (cit. a p. 56).

Held, M. e R. M. Karp (1970). "The Traveling-Salesman Problem and Minimum Spanning Trees". In: *Operations Research* 18.6, pp. 1138–1162 (cit. a p. 45).

Hemmelmayr, V. C., K. F. Doerner e R. F. Hartl (2009). "A variable neighborhood search heuristic for periodic routing problems". In: *European Journal of Operational Research* 195.3, pp. 791 –802 (cit. a p. 18).

Hewitt, M., G. L. Nemhauser e M. W. P. Savelsbergh (2010). "Combining Exact and Heuristic Approaches for the Capacitated Fixed-Charge Network Flow Problem". In: *INFORMS Journal on Computing* 22.2, pp. 314–325 (cit. a p. 56).

Hong, S. e M. W. Padberg (1977). "A Note on the Symmetric Multiple Traveling Salesman Problem with Fixed Charges". In: *Operations Research* 25.5, pp. 871–874 (cit. a p. 16).

Hooker, J. (2005). "A Hybrid Method for the Planning and Scheduling". In: *Constraints* 10 (4), pp. 385–401 (cit. a p. 5).

Hughes, M. (1999). "Nursing workload: an unquantifiable entity". In: *Journal of Nursing Management* 7.6, pp. 317–322 (cit. a p. 19).

Ibrahim, M., S. Chuprat, R. Ahmad, H. Haron e N. Ibrahim (giu. 2010). "A review on the workload in the nurse rostering problem". In: *Information Technology (ITSim), 2010 International Symposium on*. Vol. 3. IEEE. Kuala Lumpur Malaysia, pp. 1591–1595 (cit. a p. 20).

John, C. G. e G. A. Kochenberger (1988). "Using Surrogate Constraints in a Lagrangian Relaxation Approach to Set-Covering Problems". In: *Journal of the Operational Research Society* 39.7, pp. 681–685.

Jonker, R. e T. Volgenant (1988). "An Improved Transformation of the Symmetric Multiple Traveling Salesman Problem". In: *Operations Research* 36.1, pp. 163–167 (cit. a p. 16).

Jozefowiez, N., F. Semet e E.-G. Talbi (2002). "Parallel and Hybrid Models for Multi-objective Optimization: Application to the Vehicle Routing Problem". In: *Parallel Problem Solving from Nature — PPSN VII*. A cura di J. Guervós, P. Adamidis, H.-G. Beyer, H.-P. Schwefel e J.-L. Fernández-Villacañas. Vol. 2439. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, pp. 271–280 (cit. a p. 21).

— (2007). "Target aiming Pareto search and its application to the vehicle routing problem with route balancing". In: *Journal of Heuristics* 13 (5), pp. 455–469 (cit. a p. 21).

Kallehauge, B. (2006). "On the vehicle routing problem with time windows". Tesi di dott. Centre for Traffic and Transport Technical University of Denmark (cit. a p. 38).

— (2008). "Formulations and exact algorithms for the vehicle routing problem with time windows". In: *Computers & Operations Research* 35.7, pp. 2307–2330 (cit. a p. 38).

Kara, I., G. Laporte e T. Bektas (2004). "A note on the lifted Miller–Tucker–Zemlin subtour elimination constraints for the capacitated vehicle routing problem". In: *European Journal of Operational Research* 158.3, pp. 793–795 (cit. a p. 46).

Karwan, M. H. e R. L. Rardin (1984). "Surrogate Dual Multiplier Search Procedures in Integer Programming". In: *Operations Research* 32.1, pp. 52–69.

Kellerer, H., U. Pferschy e D. Pisinger (2004). *Knapsack problems*. Springer Verlag (cit. a p. 81).

Koeleman, P., S. Bhulai e M. van Meersbergen (2012). "Optimal patient and personnel scheduling policies for care-at-home service facilities". In: *European Journal of Operational Research* 219.3, pp. 557–563 (cit. a p. 4).

Kolen, A. W. J., A. H. G. R. Kan e H. W. J. M. Trienekens (1987). "Vehicle Routing with Time Windows". In: *Operations Research* 35.2, pp. 266–273 (cit. a p. 17).

Kovacs, A. A., S. N. Parragh, K. F. Doerner e R. F. Hartl (2012). "Adaptive large neighborhood search for service technician routing and scheduling problems". In: *Journal of Scheduling* 15 (5), pp. 579–600 (cit. a p. 56).

Lahrichi, N. (2008). "Organisation et planification de la main-d'oeuvre: Applications en santé et en industrie". Tesi di dott. Université de Montréal (cit. alle pp. 2, 20, 22).

Lanzarone, E. e A. Matta (2011). "A cost assignment policy for home care patients". In: *Flexible Services and Manufacturing Journal*, pp. 1–31 (cit. a p. 20).

Lanzarone, E., A. Matta e G. Scaccabarozzi (2010). "A patient stochastic model to support human resource planning in home care". In: *Production Planning & Control* 21.1, pp. 3–25 (cit. a p. 4).

Laporte, G., Y. Nobert e S. Taillefer (1987). "A branch-and-bound algorithm for the asymmetrical distance-constrained vehicle routing problem". In: *Mathematical Modelling* 9.12, pp. 857–868 (cit. a p. 16).

Laporte, G. (2009). "Fifty Years of Vehicle Routing". In: *Transportation Science* 43.4, pp. 408–416 (cit. a p. 33).

Laporte, G. e Y. Nobert (1980). "A Cutting Planes Algorithm for the m-Salesmen Problem". In: *Journal of the Operational Research Society* 31.11, pp. 1017–1023 (cit. a p. 16).

Laporte, G., M. Desrochers e Y. Nobert (1984). "Two exact algorithms for the distance-constrained vehicle routing problem". In: *Networks* 14.1, pp. 161–172 (cit. a p. 4).

Laporte, G., Y. Nobert e M. Desrochers (1985). "Optimal Routing under Capacity and Distance Restrictions". In: *Operations Research* 33.5, pp. 1050–1073 (cit. a p. 16).

Laporte, G., R. Musmanno e F. Vocaturo (2010). "An Adaptive Large Neighbourhood Search Heuristic for the Capacitated Arc-Routing Problem with Stochastic Demands". In: *Transportation Science* 44.1, pp. 125–135 (cit. a p. 56).

Lee, T.-R. e J.-H. Ueng (1999). "A study of vehicle routing problems with load-balancing". In: *International Journal of Physical Distribution & Logistics Management* 29.10, pp. 646–657 (cit. a p. 21).

Lenstra, J. K. e A. H. G. Rinnooy Kan (1975). "Some Simple Applications of the Travelling Salesman Problem". In: *Operational Research Quarterly (1970-1977)* 26.4, pp. 717–733 (cit. a p. 16).

Maffioli, F. (1991). *Elementi di programmazione matematica*. Milano: Masson.

Martello, S. e P. Toth (1990). *Knapsack problems: algorithms and computer implementations*. New York, NY, USA: John Wiley & Sons, Inc. (cit. alle pp. 19, 81).

Mathur, K. (1998). "An integer-programming-based heuristic for the balanced loading problem". In: *Operations Research Letters* 22.1, pp. 19–25 (cit. a p. 19).

Mendoza, J.-E. (2009). "Solving real-world vehicle routing problems in uncertain". Tesi di dott. Université de Nantes (cit. alle pp. 17, 83, 84).

Miller, C. E., A. W. Tucker e R. A. Zemlin (ott. 1960). "Integer Programming Formulation of Traveling Salesman Problems". In: *Journal of the ACM* 7.4, pp. 326–329 (cit. alle pp. 15, 46).

Mladenović, N. e P. Hansen (1997). "Variable neighborhood search". In: *Computers & Operations Research* 24.11, pp. 1097–1100 (cit. a p. 56).

Mladenović, N., M. Dražić, V. Kovačevic-Vujčić e M. Čangalović (2008). "General variable neighborhood search for the continuous optimization". In: *European Journal of Operational Research* 191.3, pp. 753 –770 (cit. a p. 56).

Mladenović, N., D. Urošević, S. Hanafi e A. Ilić (2012). "A general variable neighborhood search for the one-commodity pickup-and-delivery travelling salesman problem". In: *European Journal of Operational Research* 220.1, pp. 270–285 (cit. a p. 56).

Monette, J. N., P. Schaus, S. Zampelli, Y. Deville e P. Dupont (2007). "A CP Approach to the Balanced Academic Curriculum Problem". In: *Symcon'07 The Seventh International Workshop on Symmetry and Constraint Satisfaction Problems* (cit. a p. 20).

Mourgaya, M. e F. Vanderbeck (2006). "The periodic Vehicle routing problem: classification and heuristic". In: *RAIRO - Operations Research* 40.02, pp. 169–194 (cit. a p. 17).

— (2007). "Column generation based heuristic for tactical planning in multi-period vehicle routing". In: *European Journal of Operational Research* 183.3, pp. 1028–1041 (cit. alle pp. 17, 38).

Nesti, G., S. Campostrini, S. Garbin, P. Piva, P. Di Santo e F. Tunzi (2004). "Providing integrated health and social care for older persons in Italy". In: *Providing Integrated Health and Social Care for Older Persons: A European Overview of Issues at Stake*. A cura di K. Leichsenring e A. M. Alaszewski. Vol. 28. Ashgate, pp. 371–414 (cit. a p. 1).

Nickel, S., M. Schröder e J. Steeg (2012). "Mid-term and short-term planning support for home health care services". In: *European Journal of Operational Research* 219.3, pp. 574–587 (cit. alle pp. 2, 3).

Okonjo-Adigwe, C. (1989). "The Adult Training Centre Problem: A Case Study". In: *Journal of the Operational Research Society* 40.7, pp. 637–642 (cit. a p. 20).

Pepin, A.-S., G. Desaulniers, A. Hertz e D. Huisman (2009). "A comparison of five heuristics for the multiple depot vehicle scheduling problem". In: *Journal of Scheduling* 12 (1), pp. 17–30 (cit. a p. 56).

Pesant, G. e J.-C. Régin (2005). "SPREAD: A Balancing Constraint Based on Statistics". In: *Principles and Practice of Constraint Programming - CP 2005*. A cura di P. van Beek. Vol. 3709. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, pp. 460–474 (cit. a p. 20).

Petersen, H. e S. Ropke (2011). "The Pickup and Delivery Problem with Cross-Docking Opportunity". In: *Computational Logistics*. A cura di J. Böse, H. Hu, C. Jahn, X. Shi, R. Stahlbock e S. VoSS. Vol. 6971. Lecture Notes in Computer Science. Springer, Berlin / Heidelberg, pp. 101–113 (cit. a p. 56).

Pisinger, D. e S. Ropke (2007). "A general heuristic for vehicle routing problems". In: *Computers & Operations Research* 34.8, pp. 2403–2435 (cit. a p. 56).

Procter, S. (1992). "Subjectivity and objectivity in the measurement of nursing workload". In: *Journal of Clinical Nursing* 1.3, pp. 123–129 (cit. a p. 19).

Rajakumar, S., V. P. Arunachalam e V. Selladurai (2004). "Workflow balancing strategies in parallel machine scheduling". In: *The International Journal of Advanced Manufacturing Technology* 23 (5), pp. 366–374 (cit. a p. 19).

Rao, M. R. (1980). "A Note on the Multiple Traveling Salesmen Problem". In: *Operations Research* 28.3, pp. 628–632 (cit. a p. 16).

Rasmussen, M. S., T. Justesen, A. Dohn e J. Larsen (2012). "The Home Care Crew Scheduling Problem: Preference-based visit clustering and temporal dependencies". In: *European Journal of Operational Research* 219.3, pp. 598–610 (cit. a p. 3).

Ravi, R. e M. Goemans (1996). "The constrained minimum spanning tree problem". In: *Algorithm Theory – SWAT'96*. A cura di R. Karlsson e A. Lingas. Vol. 1097. Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 66–75 (cit. a p. 45).

Rendl, A., M. Prandtstetter, G. Hiermann, J. Puchinger e G. Raidl (2012). "Hybrid Heuristics for Multimodal Homecare Scheduling". In: *Integration of AI and OR Techniques in Contraint Programming for Combinatorial Optimzation Problems*. A cura di N. Beldiceanu, N. Jussien e É. Pinson. Vol. 7298. Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 339–355 (cit. a p. 4).

Ribeiro, G. M. e G. Laporte (2012). "An adaptive large neighborhood search heuristic for the cumulative capacitated vehicle routing problem". In: *Computers & Operations Research* 39.3, pp. 728–735 (cit. a p. 56).

Ribeiro, R. e H. Ramalhinho Dias Lourenço (feb. 2001). "A Multi-Objective Model For A Multi-Period Distribution Management Problem". In: working paper (cit. alle pp. 18, 22, 43).

Roberti, R. e P. Toth (2012). "Models and algorithms for the Asymmetric Traveling Salesman Problem: an experimental comparison". In: *EURO Journal on Transportation and Logistics* 1 (1-2), pp. 113–133 (cit. alle pp. 5, 29).

Russell, R. A. e D. Gribbin (1991). "A multiphase approach to the period routing problem". In: *Networks* 21.7, pp. 747–765 (cit. a p. 18).

Schaus, P., Y. Deville, P. Dupont e J.-C. Régin (2007). "The Deviation Constraint". In: *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems*. A cura di P. Van Hentenryck e L. Wolsey. Vol. 4510. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, pp. 260–274 (cit. a p. 20).

Sevkli, M. e M. Aydin (2006). "A Variable Neighbourhood Search Algorithm for Job Shop Scheduling Problems". In: *Evolutionary Computation in Combinatorial Optimization*. A cura di J. Gottlieb e G. Raidl. Vol. 3906. Lecture Notes in Computer Science. Springer, Berlin / Heidelberg, pp. 261–271 (cit. a p. 56).

Shannon, C. E. (1948). "A mathematical theory of communications". In: *Bell system technical journal* 27, pp. 379–423 (cit. a p. 96).

Shaw, P. (1997). *A new local search algorithm providing high quality solutions to vehicle routing problems*. Rapp. tecn. Department of Computer Sciences, University of Strathclyde, Glasgow, Scottland (cit. a p. 56).

— (1998). "Using Constraint Programming and Local Search Methods to Solve Vehicle Routing Problems". In: *Principles and Practice of Constraint Programming – CP98*. A cura di M. Maher e J.-F. Puget. Vol. 1520. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, pp. 417–431 (cit. a p. 56).

Shinano, Y., T. Achterberg, T. Berthold, S. Heinz e T. Koch (2012). "ParaSCIP: A Parallel Extension of SCIP". In: *Competence in High Performance Computing 2010*. A cura di C. Bischof, H.-G. Hegering, W. E. Nagel e G. Wittum. Springer Berlin Heidelberg, pp. 135–148 (cit. a p. 5).

Stecke, K. E. (1983). "Formulation and Solution of Nonlinear Integer Production Planning Problems for Flexible Manufacturing Systems". In: *Management Science* 29.3, pp. 273–288 (cit. alle pp. 19, 23).

Steeg, J. M. (ott. 2008). "Mathematical Models and Algorithms for Home Health Care Services". Tesi di dott. Germany: Department of Mathematics Technische Universität Kaiserslautern (cit. alle pp. 17, 18).

Stevanato, E., G. Zanghirati e C. Filippi (2012). *Biobjective Combinatorial Optimization and the Tsp with Profits*. LAP Lambert Academic Publishing (cit. a p. 38).

Sutcliffe, C. e J. Boardman (1990). "Optimal Solution of a Vehicle-Routeing Problem: Transporting Mentally Handicapped Adults to an Adult Training Centre". In: *Journal of the Operational Research Society* 41.1, pp. 61–67 (cit. a p. 20).

Tan, C. C. R. e J. E. Beasley (1984). "A heuristic algorithm for the period vehicle routing problem". In: *Omega* 12.5, pp. 497–504 (cit. alle pp. 17, 18).

Toth, P. e D. Vigo, cur. (2001). *The vehicle routing problem*. Philadelphia, PA, USA: Society for Industrial e Applied Mathematics (cit. a p. 16).

Valle, C. A., A. S. da Cunha, G. R. Mateus e L. C. Martinez (2009). "Exact algorithms for a selective Vehicle Routing Problem where the longest route is minimized". In: *Electronic Notes in Discrete Mathematics* 35.0, pp. 133–138 (cit. alle pp. 23, 43).

Vitacca, M., E. Clini, R. Porta e N. Ambrosino (2000). "Preliminary results on nursing workload in a dedicated weaning center". In: *Intensive Care Medicine* 26 (6), pp. 796–799 (cit. a p. 19).

Wilson, J. M. (1992). "Approaches to Machine Load Balancing in Flexible Manufacturing Systems". In: *Journal of the Operational Research Society* 43.5, pp. 415–423 (cit. a p. 23).

Wren, A. (1996). "Scheduling, timetabling and rostering: A special relationship?" In: *Practice and Theory of Automated Timetabling*. A cura di E. Burke e P. Ross. Vol. 1153. Lecture Notes in Computer Science. Springer, Berlin / Heidelberg, pp. 46–75 (cit. a p. 2).