UNIVERSITÀ DI PISA

DIPARTIMENTO DI MATEMATICA
CORSO DI LAUREA TRIENNALE IN MATEMATICA

TESI DI LAUREA TRIENNALE

Valutazione sperimentale di solutori per il Master Problem all'interno di approcci di Generazione di Colonne

CANDIDATA:
Sara Gennari

RELATORE:
Prof. Antonio Frangioni

ANNO ACCADEMICO 2015/2016

Indice

Capitolo 1

Introduzione

Nello sviluppo di un algoritmo di ottimizzazione basato su un problema reale ci si trova a dover affrontare diverse problematiche: tra queste vi sono la ricerca di un compromesso tra l'ottimalità della soluzione ed il tempo impiegato per ottenerla e la necessità di manipolare un numero di variabili che, a priori, è talmente elevato da renderne impossibile l'enumerazione.

In questo contesto si inserisce l'algoritmo per il *Vehicle (o Bus Driving) Scheduling* sviluppato dall'azienda italiana M.A.I.O.R S.r.l. (MAIOR).

Qui, per affrontare problemi di grandi dimensioni viene utilizzata la tecnica della *Column Generation*. Essa consiste nel selezionare un numero ridotto di variabili (e quindi di colonne) per generare e risolvere una versione parziale (il *Master Problem*) del problema di partenza. La soluzione trovata viene poi impiegata per determinare altre variabili che, se aggiunte al sottoinsieme, potrebbero portare iterativamente ad un miglioramento della soluzione stessa.

D'altra parte, per motivi di tempo anche il Master Problem non viene approcciato in modo diretto con solutori lineari e la soluzione viene fornita dal suo *duale lagrangiano*, su cui si utilizza un solver di tipo *bundle*: esso risulta infatti un un buon trade-off tra velocità, necessaria nella realtà aziendale, e accuratezza.

Tale componente, nell'implementazione attuale dell'algoritmo, risale al 2005. Si è reso dunque necessario sostituirlo con la sua versione più aggiornata e testare eventuali cambiamenti nella soluzione trovata e nel tempo impiegato.

Nell'ambito di questa sperimentazione è stato inserito anche un confronto del bundle con altri solutori, così da verificare l'effettivo vantaggio che l'utilizzo di quest'ultimo comporta.

Il nostro lavoro di tesi nasce a partire da precedenti studi, in cui è stata analizzata l'azione di diversi solutori su numerose istanze del Master Problem,

ponendo l'attenzione su come variassero i risultati in base alle caratteristiche dell'istanza di partenza.

Un miglioramento nella soluzione di una singolo sottoproblema, però, non sempre implica un miglioramento nell'andamento del problema globale, in quanto piccole differenze nella soluzione di un Master Problem possono indurre cambiamenti nell'insieme delle colonne generate nella fase successiva e far intraprendere all'algoritmo una nuova strada.

Il nostro obbiettivo sarà, dunque, studiare come i vari solver influiscano differentemente sull'esecuzione complessiva dell'algoritmo, cercando di capire le motivazioni di tali comportamenti.

Nella trattazione che segue verrà esposto nello specifico il problema proposto e le soluzioni adottate dall'azienda nello sviluppo dell'algoritmo. Seguiranno poi una breve descrizione del codice e del lavoro implementativo. Per finire, riporteremo le modalità con cui è stata svolta la sperimentazione, i risultati ottenuti e le conclusioni che ne possiamo trarre.

Capitolo 2

Ambito della tesi

La MAIOR è un'azienda lucchese che si occupa di ottimizzazione matematica per la gestione delle risorse nel settore del trasporto pubblico (aereo, ferroviario e su gomma). Nello specifico, vengono sviluppati algoritmi per la formazione di turni del personale (*Crew Scheduling, CS, o Bus Driver, DB*) e di veicoli (*Vehicle Scheduling*, VS), con lo scopo di coprire tutte le attività programmate del servizio con turni minimizzando i costi per l'azienda utente.

Anche se per tutti i settori viene utilizzato lo stesso modello di partenza, ognuno di essi presenta delle peculiarità che ne rendono difficile la generalizzazione.

2.1 Modellizzazione del problema

In letteratura i problemi di VS e CS vengono generalmente approcciati mediante due particolari modelli di Programmazione Lineare Intera (PLI): *flusso di costo minimo (MCF, Min Cost Flow)* e *Set Partitioning (SP)*.

Per loro la descrizione è utile definire il cosiddetto *grafo di compatibilità* $G = (N, A)$. In esso, l'insieme dei nodi $N = T \cup \{O, D\}$ comprende tutte attività da coprire, T, e due nodi deposito, O e D, che rappresentano intuitivamente delle rimesse in cui i turni hanno rispettivamente inizio e fine; A rappresenta invece l'insieme degli archi ed un elemento $(i, j) \in T \times T$ vi appartiene se e solo se le due attività i e j possono essere svolte consecutivamente in un turno. Analogamente $(O, j) \in A$ se e solo se un turno può iniziare con l'attività $j \in T$ e $(i, D) \in A$ se e solo se un turno può terminare con $i \in T$. Così, un turno può essere visto come un cammino da O a D su tale grafo.

A questo punto, per ottenere modello di flusso di costo minimo sarà sufficiente associare ad ogni arco $(i, j) \in A$ un costo $c_{i,j}$ (supponiamo quindi che il costo di un turno sia esprimibile come somma dei costi delle attività che lo compongono) e introdurre delle variabili booleane $x_{i,j}$ che assumano valore 1 se le attività i e j vengono svolte in sequenza, 0 altrimenti. Da questa formalizzazione ricaviamo il seguente modello:

$$\begin{aligned}
&\min \sum_{(i,j)\in A} c_{i,j}x_{i,j} \\
&\sum_{j:(i,j)\in A} x_{i,j} = 1 && i \in T && (2.1)\\
&\sum_{j:(i,j)\in A} x_{i,j} - \sum_{j:(j,i)\in A} x_{j,i} = 0 && i \in T && (2.2)\\
&x_{i,j} \in \{0,1\} && (i,j) \in A
\end{aligned}$$

dove il vincolo (2.1) impone che ogni attività sia coperta da uno e un solo turno e il vincolo (2.2) garantisce la conservazione del flusso per ogni nodo.

Definendo, invece, l'insieme P dei cammini ammissibili sul grafo (cioè dei turni) e associando ad ognuno di essi ($p \in P$) un costo c_p e dei coefficienti $a_{i,p}$ di valore 1 se l'attività i può essere svolta nel turno p e 0 altrimenti, ricaveremo la seguente formulazione del Set Partitioning:

$$\begin{aligned}
&\min \sum_{p\in P} c_p x_p \\
&\sum_{p\in P} a_{i,p}x_p = 1 && \forall i \in N && (SP)\\
&x_p \in \{0,1\} && \forall p \in P
\end{aligned}$$

Per semplicità in entrambi i modelli abbiamo omesso i cosiddetti *vincoli globali*, ovvero la descrizione lineare delle regole imposte da eventuali normative vigenti e che riguardano i turni nel loro complesso.

La principale motivazione per cui il modello MCF viene in alcuni casi preferito all'altro è che per risolverlo si può sfruttare numerosi algoritmi sviluppati nel campo dell'ottimizzazione combinatoria e della teoria dei grafi.

Il problema, però, sorge quando si tratta di dover formalizzare alcuni vincoli imposti dalle normative aziendali. Esistono, infatti, soprattutto nell'ambito del *Crew Scheduling*, restrizioni che impediscono di inserire un'attività

in un turno in relazione a quelle svolte precedentemente (come ad esempio la durata massima di guida continuativa di un autista o l'obbligatorietà di una pausa pranzo all'interno di un turno).

In generale perciò si preferisce utilizzare il modello del *Set Partitioning*, nel quale l'ammissibilità di un turno si riconduce alla sola appartenenza di esso all'insieme P.

Purtroppo nelle applicazioni considerate il numero dei cammini ammissibili è molto grande e ciò non permette neppure di scrivere esplicitamente la matrice dei coefficienti.

Si ricorre perciò alla tecnica di *Column Generation (CG)*.

2.2 Column Generation

Si consideri il rilassamento continuo del problema (SP)

$$\min \left\{ \sum_{p \in P} c_p x_p : \sum_{p \in P} a_{i,p} x_p = 1 \;\; i \in N, x_p \in [0,1] \;\; p \in P \right\} \tag{P}$$

e il suo duale

$$\max \left\{ \sum_{i \in N} \pi_i : c_p - \sum_{i \in N} \pi_i a_{i,p} \geq 0, p \in P \right\}. \tag{D}$$

Scegliamo un sottoinsieme $B \subset P$ di dimensioni ridotte e definiamo il sottoproblema P_B che presenta solo le colonne relative a B (poniamo $x_p = 0$ se $p \notin B$). Stavolta P_B, che prende in nome di *Master Problem*, è più facile da approcciare. Osserviamo inoltre che il suo duale (D_B) è un rilassamento di (D).

A questo punto possiamo risolvere il Master Problem e trovare due soluzioni ottime x^* per (P_B) e π^* per (D_B).

Tuttavia, mentre x^* è sempre una soluzione ammissibile per (P) se completata con zeri, π^* potrebbe violare alcuni dei vincoli duali.

Per questo motivo occorre definire per ogni turno $p \in P$ il suo *costo ridotto*:

$$\bar{c}_p = c_p - \sum_{i \in N} \pi_i^* a_{i,p}.$$

Esso sarà negativo se e solo se il vincolo duale corrispondente non viene rispettato. In quest'ottica, la soluzione π^* ammissibile viene trovata in modo iterativo seguendo la seguente strategia:

- i cammini vengono ordinati per costo ridotto crescente;
- il cammino di costo ridotto minore, se negativo, viene inserito in B;
- si risolve il nuovo Master Problem.

Il procedimento termina quando tutti cammini hanno costo ridotto positivo o nullo. In tal caso, infatti, la soluzione è ammissibile per (D) e quindi ottima.

In realtà trovare il cammino di costo ridotto minimo non è affatto banale. In generale infatti il *problema di pricing*, come viene denominato, è NP-completo, perciò viene risolto tramite un'euristica. Ciò avviene principalmente a causa dei vincoli di normativa che spesso impongono regole molto complesse da soddisfare.

Non è necessario, tuttavia, soffermarci ulteriormente su tale argomento in quanto si allontana dagli obbiettivi di questo lavoro.

2.3 Solver

Il Master Problem è un problema di programmazione lineare e di conseguenza potrebbe essere risolto in modo diretto tramite uno dei numerosi solutori disponibili. Questo approccio, però, è fortemente sconsigliabile per via delle dimensioni rilevanti che il problema può assumere nei casi presi in esame dall'azienda, soprattutto al crescere di B.

La linea seguita è quindi quella di non risolvere esplicitamente P_B ma di considerare al suo posto il *rilassamento lagrangiano* dei vincoli di copertura, compresi eventuali vincoli globali:

$$\phi(\lambda) = \left\{ \sum_{p \in P} c_p x_p - \sum_{i \in N} \lambda_i \left(1 - \sum_{p \in B} a_{i,p} x_p \right) : x_p \in [0,1] \;\; p \in B \right\}$$

e il corrispondente problema duale:

$$max \left\{ \phi(\lambda) : \lambda \in \mathbb{R}^n \right\}. \tag{DL}$$

Come è noto, il duale lagrangiano (DL) di un problema lineare è equivalente al duale (D), così risolvendo il primo avremmo la soluzione π^* dell'altro.

Purtroppo la funzione ϕ in generale è convessa ma non differenziabile, di conseguenza non possono essere utilizzati la maggior parte degli algoritmi di ottimizzazione nonlineare che richiedono una certa regolarità delle derivate prime o seconde.

Per questo la MAIOR ha scelto di utilizzare come solver un algoritmo di tipo Bundle, che verrà trattato più approfonditamente più avanti.

2.4 Ricerca della miglior soluzione discreta

Una volta risolto il Master Problem e trovata la coppia di soluzioni ottime (x^*, π^*) per (P) e (D), si procede a partire da essa per cercare euristicamente delle soluzioni intere.

Poiché, appunto, l'approccio seguito non è "esatto", calcolando un'unica soluzione intera si rischia che essa sia molto lontana dall'ottimo. Per questo motivo, il procedimento viene ripetuto più volte e tra tutte le soluzioni calcolate viene scelta la migliore. In particolare l'algoritmo alterna cicli di fissaggio delle colonne (*fixing*), soluzione di un nuovo problema "master" e sfissaggio (*sfixing*).

Vediamo più nel dettaglio tale procedimento.

2.4.1 Fixing

Nella fase di *fixing* vengono scelti alcuni turni $p \in P$ da far entrare in soluzione (ponendo $x_p = 1$), quindi iterativamente viene risolto il Master Problem relativo alle variabili introdotte dal fissaggio fino alla totale copertura delle attività.

È evidente che la scelta di quali turni fissare sia cruciale per l'efficacia del metodo.

Inizialmente venivano sfruttate solo le informazioni fornite dalla soluzione duale. In particolare, dopo aver ordinato al termine della CG i turni appartenenti a B per costo ridotto crescente, venivano fissati i primi della lista, fino ad aver coperto una determinata percentuale di attività.

Successivamente, invece, è stato introdotto un criterio più restrittivo, basato anche sulla soluzione continua primale: come prima si seleziona un sottoinsieme $\hat{B} \subset B$ in base ai costi ridotti di π^*, però al suo interno si privilegiano i turni $p \in \hat{B}$ con x_p vicino ad 1. Si spera, infatti, che essi abbiano più probabilità di far parte di una buona soluzione intera.

2.4.2 Sfixing

Al termine del calcolo ha inizio la fase di *sfixing*, in cui una parte di turni precedentemente fissati vengono rimossi dalla soluzione intera. La scelta viene fatta in base a delle variabili "accumulate" durante l'interno processo. Ancora una volta, però, è necessario scendere più nel dettaglio.

Terminato lo sfixing l'algoritmo costruisce una nuova soluzione intera a partire quella parziale appena trovata.

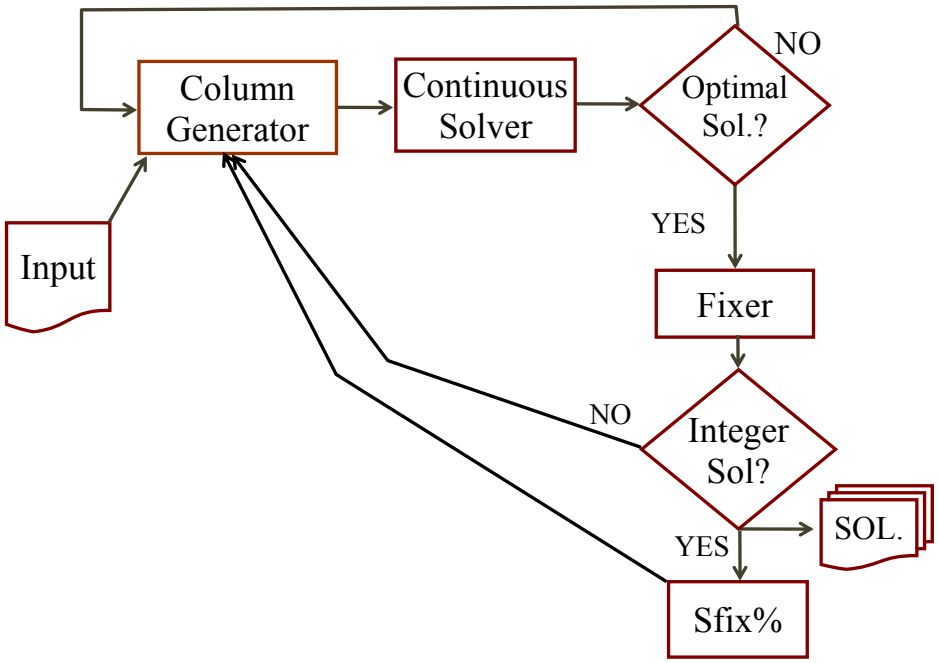

Figura 2.1: Struttura sommaria dell'algoritmo

2.5 Bundle

Per completezza concludiamo questa sezione illustrando brevemente la struttura degli algoritmi di tipo *bundle*.

Con questo termine ci si riferisce ad un ampia classe di metodi considerati tra i più efficienti per l'ottimizzazione convessa non differenziabile.

Siano $f : \mathbb{R}^n \to \mathbb{R}$ una funzione a valori finiti, non differenziabile e convessa ed $X \subseteq \mathbb{R}^n$ un sottoinsieme convesso e chiuso.

Vogliamo risolvere un generico problema della forma

$$\inf \{f(x) : x \in X\} \tag{2.3}$$

dove assumiamo, però, che la funzione sia nota solo tramite un *oracolo* (o *Black Box*): dato un x nello spazio delle soluzioni ammissibili viene fornito il valore $f(x)$ ed un elemento $z \in \partial f(x)$ del subdifferenziale [1] di f in x.

L'idea dei metodi di tipo bundle è quella generare una sequenza finita di punti provvisori $\{x_i\}_{i \leq n}$ di cui vengono memorizzate le informazioni fornite dall'oracolo. Fissato uno specifico vettore $\bar{x}$, detto *punto corrente*, tale *bundle* (insieme, gruppo) di informazioni $\mathcal{B} = \{(x_i, f(x_i), z_i)\}_{i \leq n}$ viene utilizzato per stabilire una direzione di decrescita provvisoria e calcolare un nuovo punto x_{n+1}. Se tale punto soddisfa determinate condizioni (che in questo contesto non è utile specificare), viene aggiornato a nuovo punto corrente, effetuando un *Serious Step*, altrimenti viene aggiunto a $\mathcal{B}$, sperando che le nuove informazioni aiutino a scegliere una direzione migliore all'iterazione successiva (*Null Step*).

Nello specifico, il bundle viene utilizzato per costruire un modello $f_{\mathcal{B}}$ di f, generalmente una sua approssimazione inferiore, in modo che il risultante problema

$$\inf_d \{f_{\mathcal{B}}(\bar{x} + d)\} \tag{2.4}$$

fornisca un lower bound per 2.3.

Il modello più utilizzato è quello dei *piani di taglio*:

$$f_{\mathcal{B}}(x) = \max\{f(x_i) + z_i(x - x_i) : (x_i, f(x_i), z_i) \in \mathcal{B}\}$$

, nonostante spesso risulti computazionalmente instabile, in quanto la sequenza di punti trovati tende ad oscillare incontrollabilmente in prossimità del minimo.

Per ovviare al problema, viene introdotto un *termine di stabilizzazione* D_t, che impedisca al punto successivo di allontanarsi troppo dal precedente. D_t è una funzione chiusa e convessa, dove $t > 0$ è un *parametro di prossimità*

[1] Ricordiamo che il *subdifferenziale* di una funzione g in un punto x_0 è definito come $\partial g(x_0) = \{y \in \mathbb{R}^n : g(x) \geq g(x_0) + < y, x - x_0 >\}$. Un tale $y_0 \in \partial g(x_0)$ è detto *subgradiente*.

che regola la forza con cui lo stabilizzatore agisce. In tal modo, la soluzione di decrescita viene calcolata risolvendo il seguente problema stabilizzato:

$$\inf_d \{f_{\mathcal{B}}(\bar{x} + d) + D_t(d)\}. \tag{2.5}$$

È noto, infine, che aggiornare di volta in volta il parametro t sia fondamentale per l'efficacia del modello.

Per maggiori dettagli si rimanda alla letteratura citata.

Capitolo 3

Implementazione

Adesso che abbiamo introdotto le nozioni necessarie, possiamo procedere a trattare più nello specifico il lavoro svolto.

Come già accennato, la versione del Bundle attualmente utilizzata dall'azienda risale al 2005. Poiché nel 2015 ne è stata rilasciata una versione aggiornata, è naturale chiederci se il nuovo solver produca dei miglioramenti, se non in termini di soluzione finale, almeno in termini di tempo, all'algoritmo della MAIOR. D'altra parte, il Bundle non fornisce una soluzione ottima al Master Problem: è quindi dovroso confrontarlo anche con solutori esatti, sia commerciali che open source, per vedere se effettivamente sia la scelta migliore.

La fase implementativa, per la quale ci siamo avvalsi dell'IDE Visual Studio 2015, è stata, perciò, suddivisa in due parti:

1. l'inserimento all'interno del codice del nuovo bundle e delle classi ad esso legate;

2. l'implementazione dell'interfaccia *OsiSolverInterface*, che permette di utilizzare due software di ottimizzazione (*Cplex* e *Clp*) per risolvere il problema master tramite tramite metodi quali il simplesso ed il barrier.

Prima di esporre le modifiche apportate al progetto, però, è utile dare un'idea di come la MAIOR abbia implementato l'algoritmo descritto nel capitolo 2.

3.1 Classi di interesse

Riportiamo una rapida descrizione, della struttura del codice, ponendo particolare attenzione alle principali classi che lo costituiscono e a quelle che

saranno più rilevanti in seguito.

La seguente relazione è tratta dalla direttamente dalla tesi di G.Vallese che abbiamo precedentemente citato.

ConstrainedSetPartitioning È la classe di più alto livello e coordina il lavoro di tutte le altre per ottenere le soluzioni del problema. In breve:

- istanzia le altre classi coinvolte nel calcolo;
- calcola il *lower bound* tramite una chiamata al *MainModule*;
- calcola un numero prestabilito di soluzioni intere e per ognuna di esse:
  - se la soluzione attuale non è la prima, chiama il *RigeneraPezzi* per effettuare lo sfissaggio di alcune colonne della precedente;
  - fino al competa copertura delle attività, alterna chiamate al MainModule, per il ciclo di generazione-solver, ed alla classe *IncrTurniInSol* per il fixing di alcune colonne;
  - notifica la foluzione intera trovata.

MainModule Si occupa di gestire il procedimento di Column Generation mediante chiamate alternate al *GeneratoreColonne* ed al *MasterProblemSolver*. In particolare:

- istanzia le due classi sopracitate;
- nel caso sia necessario calcolare il lower bound, genera un primo insieme di colonne con costo ridotto qualsiasi;
- fino a quando non è soddisfatta una delle condizioni di arresto, come ad esempio il raggiungimento della stabilità del ciclo, il raggiungimento di un numero limite di passi stabilito o l'interruzione da parte dell'utente, esegue le seguenti operazioni:
  - se il numero di colonne disponibili è maggiore di una certa soglia, elimina alcuni turni tra i peggiori;
  - se l'ultimo fissaggio delle colonne ha comportato un incremento eccessivo del costo della soluzione ottima del sottoproblema, viene effettuato il *backtrack*, ovvero vengono rimosse alcune delle colonne con costo ridotto eccessivo;
  - chiama il GeneraColonne per generare colonne a costo ridotto negativo;

- chiama il MasterProblemSolver per calcolare la soluzione del problema corrente;
- se il ciclo è stabile per più di due iterazioni, termina.

MasterProblemSolver Permette di utilizzare un solver per risolvere il sottoproblema attuale. Oltre ad interfacciarsi con il Bundle tramite la sottoclasse *BundleMaior2005*, può essere impiegata per calcolare la soluzione tramite due particolari software di ottimizzazione: *Clp* e *Cplex*.

GeneratoreColonne È incaricata principalmente di generare colonne a costo ridotto negativo, in base alla soluzione duale corrente, ed aggiungerle a quelle già presenti nel *GestoreColonne*.

BundleMaior2005 La classe costituisce l'interfaccia tra il codice MAIOR ed il Bundle. Tale solver, infatti, insieme ad altre classi ad esso legate, come la *FiOracle*, è stato sviluppato separatamente dal professor Antonio Frangioni del gruppo di Ricerca Operativa del Dipartimento di Informatica dell'Università di Pisa e successivamente inserito nell'algoritmo. Fondamentalmente, la classe si occupa di impostare tutti i parametri necessari al funzionamento del solver ed a lanciarlo tramite il metodo *Calcola()*.

Per quanto riguarda i parametri, molti di essi sono fissati in fase di inizializzazione in base a valori di default; altri, invece, dipendono di volta in volta dalle caratteristiche del Master Problem. Tra questi vi è il numero massimo di iterazioni eseguite dal bundle prima di fermarsi, che può assumere tre valori in base alla precisione necessaria andando ad influire sull'accuratezza del solver.

FiMaior Implementa la classe concreta dell'interfaccia astratta *FiOracle*.

3.2 Il nuovo bundle

Per facilitare la comprensione, da ora in avanti chiameremo le due versioni del solver *bundle05* e *bundle15*.

Inizialmente sono state create due copie separate del progetto che utilizzassero ognuna una versione del solver.

Poiché, come è già stato accennato, il bundle è una componente esterna al codice, di fatto è indipendente da esso, fatta eccezione per le classi BundleMaior2005 e FiMaior con cui si interfaccia.

Il primo obbiettivo, dunque, è stato inserire la classe Bundle15 al posto della precedente Bundle05 come classe derivata di BundleMaior2005.

Una volta verificato che, in tal modo, il nuovo algoritmo risultava effettivamente "migliore" del primo (in termini che saranno chiariti meglio nella sezione 4.2), si è deciso di unire le due versioni in un unico progetto. Questa decisione mira principalmente a lasciare all'azienda utente la libertà di poter scegliere quale pacchetto utilizzare.

Per raggiungere l'intento è stato necessario considerare attentamente quali fossero le migliori modalità inserimento per non alterare l'omogeneità strutturale del progetto. Alla fine è stato deciso di inserire entrambe le versioni come sottoclassi di BundleMaior2005: risolti i problemi di compatibilità che si erano venuti a creare (come ad esempio conflitti nell'utilizzo dei *namespace*), attualmente le due componenti corrispondono a due diverse configurazioni del progetto e convivono alternativamente nel codice, generando due diversi eseguibili.

3.3 OsiSolverInterface

L'OsiSolverInterface (OSI) è una classe astratta rilasciata dalla COIN-OR (COmputational INfrastructure for Operations Research) che permette di interfacciarsi, tramite sue sottoclassi, con numerosi software di ottimizzazione. Tra questi, come già accennato, si è scelto di utilizzare *Clp* e *Cplex*.

Il primo è un solutore open source per problemi di programmazione lineare, sviluppato dalla stessa COIN-OR.

L'altro, invece, è un solver commercializzato dalla IBM per programmazione lineare, intera e quadratica.

Entrambi i software permettono al loro interno di scegliere quale metodo impiegare per la soluzione del problema: nell'algoritmo della MAIOR si è deciso di restringere le opzioni a *Simplesso Primale*, *Simplesso Duale* e *Barrier*.

La classe concreta che è stata creata per implementare l'OSI nel codice è la *MPSOsi*. Di seguito ne forniamo una breve descrizione.

MPSOsi classe derivata di MasterProblemSolver, implementa le sottoclassi astratte specifiche per Cplex e Clp, *OsiCplexSolverInterface* e *OsiClpSolverInterface*.

Si occupa principalmente di:

- inizializzare i parametri per la scelta del solver e del metodo utilizzati (tramite la procedura *Inizializza*);

- scrivere il modello come un *CoinModel*, così che possa essere passato in input al solutore (*BuildCoinModel*);
- lanciare il solutore e salvare le soluzioni primali e duali trovate (*Calcola*).

È utile sottolineare che, in alcune configurazioni dell'algoritmo (VSA, vedi 4.1), in caso di utilizzo del Bundle, le soluzioni primali vengono calcolate tramite Clp, in quanto forniscono dei risultati più accurati. Se il solutore scelto, invece, è esatto, come quelli utilizzati in questo caso da Cplex o Clp, si preferisce ricavare da un solo lancio entrambe le soluzioni.

Capitolo 4

Sperimentazione

Per semplificare la fase di sperimentazione e garantire una miglior precisione dei risultati, tutti i test sono stati lanciati da remoto su una macchina situata a La Spezia.

Abbiamo così potuto disporre di 8 host dotati di 2 processori intel xeon da 6 core l'uno, grazie ai quali è stato possibile lanciare più test in parallelo.

4.1 Casi Test

La MAIOR ci ha fornito per questo studio circa trenta casi di test da analizzare che, per dimensione e complessità dei vincoli globali, rappresentano un ventaglio abbastanza vario e completo di problemi sottoposti all'azienda.

I casi sono tutti relativi al settore del trasporto su gomma e sono suddivisi in tre categorie:

- BDSA (Bus Driver Scheduling Algorithm)
- VBDSA (Vehicle Bus Driving Scheduling Algorithm)
- VSA (Vehicle Scheduling Algorithm).

Per analizzare al meglio i risultati ottenuti, all'interno delle tre categorie i casi di test sono stati ordinati per numero di elementi da partizionare (*numero di tasks*) decrescente. Tale valore, nonostante sia chiaramente indicativo, può dare un'idea della "difficoltà" del problema.

Per facilitare il lettore, sono stati colorati in nero i risultati "positivi" e in rosso quelli "negativi".

Nei test effettuati sono state calcolate 10 soluzioni intere, a differenza delle 40 utilizzate di default dalla MAIOR.

Nel presentare i risultati ci focalizzeremo prima sull'analisi delle prestazioni dell'algoritmo con il bundle15 rispetto a quelle con il bundle05, poi sul cambiamento indotto dalla modifica di alcuni parametri del bundle ed infine sul confronto tra vari solver per il Master Problem.

4.2 Confronto tra le due versioni del bundle

I dati che sono stati raccolti sono:

- tempo totale di esecuzione del bundle all'interno dell'algoritmo (*tempo*);
- numero di chiamate del bundle all'interno dell'algoritmo ($n.chiam$);
- numero totale di iterazioni interne del bundle su tutte le sue chiamate ($n.it$);
- tempo totale di esecuzione dell'algoritmo ($t.TOT$);
- migliore soluzione intera trovata (*best sol*).

A partire da essi, poi, sono stati calcolti i seguenti valori:

- tempo medio di una chiamata del bundle ($t/n.ch$);
- numero medio di iterazioni al secondo effettuate dal bundle ($n.it/t$);
- numero medio di iterazioni effettuate in una chiamata dal bundle ($n.it/n.ch$).

Nella tabella 4.1 riportiamo i risultati ottenuti. In particolare, per ogni campo è esibita la differenza relativa percentuale tra i due bundle, espressa nel seguente modo:

$$\frac{bundle05 - bundle15}{bundle15}.$$

Nella tabella 4.2 mostriamo invece media e deviazione standard dei dati sopracitati, così che siano più chiari, a colpo d'occhio, i risultati ottenuti.

Per motivi di privacy i nomi delle città interessate sono stati rimossi e sostituiti con nomi fittizi.

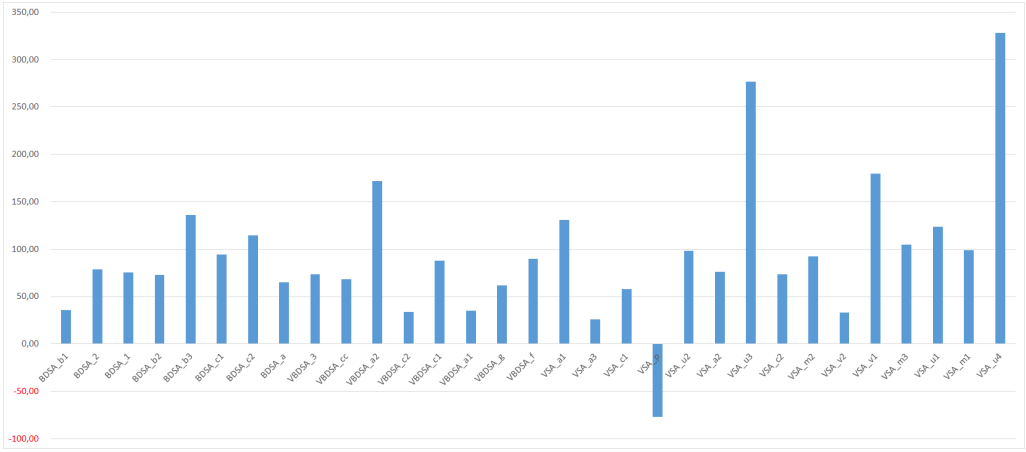

Figura 4.1: Istogramma delle differenze relative percentuali tra i tempi del bundle15 rispetto al bundle05

Tabella 4.1: Differenze relative percentuali tra bundle15 e bundle05

CASO	tempo	n.chiam	t/n.ch	n.it	n.it/t	n.it/n.ch	t.TOT	best sol
BDSA_b1	35,73	-11,58	53,51	-21,35	-42,05	-11,05	-18,78	0,02
BDSA_2	78,52	-13,97	107,50	-16,77	-53,38	-3,26	-11,11	-0,01
BDSA_1	75,61	6,40	65,05	7,01	-39,06	0,57	-0,43	1,11
BDSA_b2	72,78	11,82	54,52	-5,02	-45,03	-15,06	16,30	-0,04
BDSA_b3	136,05	0,84	134,08	44,77	-38,67	43,57	3,52	-0,06
BDSA_c1	94,48	0,00	94,48	4,56	-46,24	4,56	11,43	0,10
BDSA_c2	114,26	6,53	101,12	4,31	-51,32	-2,09	14,58	0,00
BDSA_a	64,85	-11,26	85,77	-13,58	-47,58	-2,61	5,77	-0,02
VBDSA_3	73,17	0,93	71,57	-13,27	-49,92	-14,07	2,04	-2,30
VBDSA_cc	68,00	3,74	61,94	-5,78	-43,92	-9,18	29,42	-0,37
VBDSA_a2	171,71	51,35	79,52	55,98	-42,59	3,06	34,44	0,38
VBDSA_c2	33,55	-10,93	49,95	-8,28	-31,32	2,98	6,15	0,37
VBDSA_c1	88,03	-10,96	111,18	-2,94	-48,38	9,01	2,50	-0,08
VBDSA_a1	34,77	1,94	32,21	-13,76	-36,01	-15,40	-9,76	1,58
VBDSA_g	61,48	-12,92	85,44	-20,65	-50,86	-8,87	4,35	-0,14
VBDSA_f	89,64	0,00	89,64	-0,00	-47,27	-0,00	7,50	0,00
VSA_a1	130,48	4,03	121,54	6,87	-53,63	2,72	72,51	0,06
VSA_a3	25,68	-16,90	51,23	-23,26	-38,94	-7,66	-12,35	-0,21
VSA_c1	57,67	-6,77	69,11	-4,15	-39,20	2,81	47,59	0,01
VSA_p	-76,91	-72,50	-16,02	-74,72	9,49	-8,05	-73,73	-0,99
VSA_u2	98,37	0,00	98,37	0,00	-49,59	0,00	54,55	0,00
VSA_a2	76,21	-3,59	82,76	-6,78	-47,10	-3,31	37,05	-0,03
VSA_u3	276,80	18,36	218,34	15,30	-69,40	-2,59	174,16	0,01
VSA_c2	73,52	-21,61	121,36	-32,19	-60,92	-13,50	17,25	-0,21
VSA_m2	92,49	10,73	73,84	6,52	-44,66	-3,81	26,94	0,04
VSA_v2	33,28	-6,31	42,25	-15,38	-36,51	-9,68	15,13	-0,00
VSA_v1	179,94	30,36	114,75	31,09	-53,17	0,56	94,77	-0,04
VSA_m3	104,89	6,40	92,56	5,14	-48,68	-1,19	95,00	0,00
VSA_u1	123,41	5,84	111,09	4,20	-53,36	-1,55	27,27	-0,00
VSA_m1	98,96	-6,01	111,68	-6,20	-52,85	-0,20	38,20	-0,00
VSA_u4	327,96	-4,91	350,05	1,62	-76,26	6,86	130,64	-0,01

Tabella 4.2: Riassunto differenze relative tra bundle 15 e bundle05

	tempo	n.chiam	t/n.ch	n.it	n.it/t	n.it/n.ch	t.TOT	best sol
media	94,04	-1,64	94,21	-3,12	-46,08	-1,82	27,19	-0,03
dev.std	71,93	19,09	60,96	22,44	13,64	10,45	46,42	0,58

Come possiamo osservare, il tempo di esecuzione del nuovo bundle è nettamente migliore del precedente, con un miglioramento medio di circa il 95% sul tempo del solver e del 27% su quello totale.

Tale comportamento è attribuibile ad una minor durata delle singole iterazioni del bundle, che in quasi tutti i casi scende più del 30%.

Il costo della miglior soluzione trovata, invece, resta praticamente invariato (lo scarto massimo è del 2,3%).

Gli altri dati non ci forniscono informazioni rilevanti.
È doveroso notare che il test VSA_p presenta un comportamento decisamente controcorrente. Purtroppo ancora non sono stati fatti ulteriori test per approfondire la questione.

Gli esiti diretti dei suddetti test sono riportati nelle tabelle 4.3 e 4.4.

4.3 Doppie iterazioni limite del bundle

Come abbiamo già accennato, il bundle è caratterizzato da molti parametri. Fino ad adesso, nello studio che è stato fatto, sono stati mantenuti i valori di default stabiliti dalla MAIOR.

D'altra parte, abbiamo appena visto che il nuovo solver permette di ottenere soluzioni finali con costo comparabile al precedente, ma con tempi minori. Per questo non è da escludere la possibilità che variando alcuni parametri si possa ottenere soluzioni migliori con tempi comunque accettabili.

L'analisi che segue è stata fatta, solo sui test di tipo VSA, raddoppiando il numero limite di iterazioni interne del bundle rispetto ai valori standard (vedi tabella 4.5).

Riportiamo nella tabella 4.6 le differenze relative percentuali tra l'algoritmo che utilizza il bundle15 con il numero massimo standard di iterazioni e quello con i limiti nuovi. Le differenze relative sono calcolate mediante la seguente formula:

$$\frac{standardLimIt - doppioLimIt}{doppioLimIt}.$$

Tabella 4.3: Risultati dei test con bundle15 e bundle05, prima parte

	bundle15				bundle05			
CASO	tempo	n.chiam	t/n.ch	n.it	tempo	n.chiam	t/n.ch	n.it
BDSA_b1	2951,84	950	3,11	218362	4006,61	840	4,77	171743
BDSA_2	17501,00	2993	5,85	793318	31242,40	2575	12,13	660279
BDSA_1	841,24	1578	0,53	323016	1477,29	1679	0,88	345654
BDSA_b2	48,59	440	0,11	61312	83,95	492	0,17	58232
BDSA_b3	5,58	119	0,05	11728	13,16	120	0,11	16979
BDSA_c1	11,28	151	0,07	15102	21,93	151	0,15	15791
BDSA_c2	84,37	199	0,42	38672	180,78	212	0,85	40337
BDSA_a	12,93	293	0,04	26401	21,32	260	0,08	22816
VBDSA_3	209,66	967	0,22	149560	363,07	976	0,37	129717
VBDSA_cc	13526,70	3687	3,67	1348705	22724,50	3825	5,94	1270706
VBDSA_a2	264,80	592	0,45	107663	719,48	896	0,80	167933
VBDSA_c2	85,95	567	0,15	82899	114,79	505	0,23	76036
VBDSA_c1	71,72	447	0,16	63739	134,85	398	0,34	61863
VBDSA_a1	11,64	309	0,04	29828	15,68	315	0,05	25723
VBDSA_g	90,63	356	0,25	100569	146,35	310	0,47	79805
VBDSA_f	16,53	130	0,13	20087	31,35	130	0,24	20086
VSA_a1	3923,31	496	7,91	83506	9042,27	516	17,52	89240
VSA_a3	5448,86	870	6,26	209600	6847,97	723	9,47	160837
VSA_c1	7098,30	1271	5,58	302236	11191,60	1185	9,44	289705
VSA_p	13038,00	3073	4,24	593721	3010,71	845	3,56	150118
VSA_u2	10,14	42	0,24	2339	20,12	42	0,48	2339
VSA_a2	1645,97	502	3,28	89226	2900,36	484	5,99	83177
VSA_u3	2967,39	1307	2,27	307080	11181,10	1547	7,23	354063
VSA_c2	582,24	310	1,88	54432	1010,31	243	4,16	36908
VSA_m2	107,62	559	0,19	82332	207,16	619	0,33	87699
VSA_v2	1591,23	1015	1,57	226904	2120,74	951	2,23	192009
VSA_v1	7474,06	1871	3,99	336296	20922,90	2439	8,58	440847
VSA_m3	3672,01	734	5,00	140492	7523,41	781	9,63	147716
VSA_u1	14,24	137	0,10	22246	31,82	145	0,22	23180
VSA_m1	240,58	283	0,85	45339	478,66	266	1,80	42529
VSA_u4	39,25	387	0,10	42345	167,99	368	0,46	43030

Tabella 4.4: Risultati dei test con bundle15 e bundle05, prima parte

	bundle15				bundle05			
CASO	n.it/t	n.it/n.ch	t.TOT	best sol.	n.it/t	n.it/n.ch	t.TOT	best sol.
BDSA_b1	73,97	229,85	321,52	605,42	42,86	204,46	261,15	605,54
BDSA_2	45,33	265,06	2981,22	231,18	21,13	256,42	2649,95	231,16
BDSA_1	383,98	204,70	112,65	109,49	233,98	205,87	112,17	110,71
BDSA_b2	1261,90	139,35	3,07	174,13	693,65	118,36	3,57	174,07
BDSA_b3	2103,30	98,55	3,32	196,78	1290,00	141,49	3,43	196,65
BDSA_c1	1339,07	100,01	2,33	51,24	719,93	104,58	2,60	51,29
BDSA_c2	458,35	194,33	34,53	33,35	223,13	190,27	39,57	33,35
BDSA_a	2041,84	90,11	1,73	62,85	1070,42	87,75	1,83	62,84
VBDSA_3	713,35	154,66	112,65	389,55	357,28	132,91	114,95	380,61
VBDSA_cc	99,71	365,80	563,47	1821,85	55,92	332,21	729,22	1815,04
VBDSA_a2	406,59	181,86	74,33	130,20	233,41	187,43	99,93	130,69
VBDSA_c2	964,53	146,21	37,40	107,38	662,41	150,57	39,70	107,78
VBDSA_c1	888,77	142,59	27,28	169,52	458,76	155,43	27,97	169,38
VBDSA_a1	2563,64	96,53	24,60	92,27	1640,39	81,66	22,20	93,72
VBDSA_g	1109,70	282,50	8,43	51,73	545,32	257,44	8,80	51,66
VBDSA_f	1214,96	154,52	3,33	65,65	640,64	154,51	3,58	65,65
VSA_a1	21,28	168,36	125,32	76,10	9,87	172,95	216,18	76,15
VSA_a3	38,47	240,92	388,53	378,63	23,49	222,46	340,57	377,83
VSA_c1	42,58	237,79	138,97	197,64	25,89	244,48	205,10	197,66
VSA_p	45,54	193,21	593,58	111,78	49,86	177,65	155,93	110,68
VSA_u2	230,58	55,69	0,37	74,49	116,24	55,69	0,57	74,49
VSA_a2	54,21	177,74	52,45	623,79	28,68	171,85	71,88	623,58
VSA_u3	103,48	234,95	94,05	108,77	31,67	228,87	257,85	108,78
VSA_c2	93,49	175,59	31,30	223,75	36,53	151,88	36,70	223,28
VSA_m2	765,03	147,28	6,87	246,78	423,34	141,68	8,72	246,87
VSA_v2	142,60	223,55	44,73	149,66	90,54	201,90	51,50	149,66
VSA_v1	45,00	179,74	288,62	55,75	21,07	180,75	562,15	55,73
VSA_m3	38,26	191,41	68,27	130,49	19,63	189,14	133,12	130,49
VSA_u1	1561,78	162,38	1,28	97,97	728,40	159,86	1,63	97,97
VSA_m1	188,46	160,21	11,65	48,16	88,85	159,88	16,10	48,16
VSA_u4	1078,74	109,42	2,07	31,62	256,15	116,93	4,77	31,61

Tabella 4.5: Valori limite per le iterazioni interne del bundle

	maxItBundle	medioItBundle	minItBundle
default MAIOR	2000	250	150
doppie	4000	500	300

Come potevamo aspettarci, l'aumento del numero massimo di iterazioni ha causato un innalzamento medio del tempo di esecuzione del bundle, che solo in pochi casi risulta migliore (si noti i casi *VSA_a3* e *VSA_v2*). D'altra parte, si riscontra anche una riduzione media del numero di chiamate del solver di circa il 12% ed un abbassamento del tempo medio di esecuzione in una singola iterazione, anche se con una varianza abbastanza alta.

Il miglioramento nel costo della soluzione finale, invece, è dell'ordine dello 0,1%.

Tabella 4.6: Differenze relative percentuali tra limite di iterazioni normali e doppie con bundle15

CASO	tempo	n.chiam	t/n.ch	n.it	n.it/t	n.it/n.ch	t.TOT	best sol
VSA_a3	45,27	63,23	-11,00	17,72	-18,97	-27,88	98,82	0,15
VSA_c1	-26,97	14,61	-36,28	-20,09	9,42	-30,27	-23,63	-0,02
VSA_p	4,96	50,05	-30,05	-14,61	-18,64	-43,09	34,02	1,04
VSA_u2	-32,23	0,00	-32,23	-28,82	5,03	-28,82	-21,43	0,00
VSA_a2	-16,71	6,58	-21,85	-16,99	-0,33	-22,11	-5,52	-0,07
VSA_u3	-47,21	13,06	-53,31	-26,68	38,88	-35,15	-38,21	0,11
VSA_c2	-37,73	6,16	-41,35	-21,72	25,72	-26,27	-15,90	0,31
VSA_m2	-41,39	-12,52	-33,00	-35,93	9,32	-26,76	-15,23	0,13
VSA_v2	56,30	12,65	38,74	-4,25	-38,74	-15,00	46,83	-0,00
VSA_v1	-34,65	-3,85	-32,03	-44,99	-15,83	-42,79	-19,01	0,06
VSA_m3	-62,88	0,96	-63,23	-10,17	141,97	-11,03	-4,52	0,00
VSA_u1	-53,13	-14,91	-44,91	-33,50	41,86	-21,85	-18,09	0,00
VSA_a1	-27,07	23,08	-40,75	-20,01	9,68	-35,01	-5,99	0,06
VSA_m1	5,93	27,48	-16,90	10,85	4,65	-13,04	20,31	0,03
VSA_u4	-49,22	-3,49	-47,39	-28,45	40,91	-25,86	-36,73	0,01

Tabella 4.7: Riassunto differenze relative tra limite di iterazioni normali e doppie con bundle15

	tempo	n.chiam	t/n.ch	n.it	n.it/t	n.it/n.ch	t.TOT	best sol.
media	-21,12	12,21	-31,04	-18,51	15,66	-27,00	-0,29	0,12
dev.std	33,80	20,96	22,78	16,29	40,70	9,31	35,23	0,26

Analizziamo, ora, come si comportano le due versioni del bundle con limite di iterazioni doppio. I risultati sono riportati nelle tabelle 4.8 e 4.9.

In generale viene mantenuto il comportamento riscontrato già dai primi studi: il vecchio solver supera decisamente in termini di tempo impiegato il nuovo, a causa delle iterazioni interne che sono più lente. Tuttavia, se confrontato con gli esiti del primo test, la riduzione media del tempo di esecuzione del bundle15 rispetto al bundle05 risulta essere un po' più alta e la varianza più bassa.

Notiamo, infine, che il caso critico VSA_p è rientrato nei valori "standard".

Tabella 4.8: Differenze relative percentuali tra bundle15 e bundle05 con doppie iterazioni limite

CASO	tempo	n.chiam	t/n.ch	n.it	n.it/t	n.it/n.ch	t.TOT	best sol
VSA_a3	145,02	14,63	113,74	31,04	-46,52	14,31	49,87	0,24
VSA_c1	40,94	3,88	35,68	-3,44	-31,49	-7,05	34,60	-0,03
VSA_p	259,16	45,17	147,42	49,32	-58,42	2,86	154,42	0,10
VSA_u2	95,84	0,00	95,84	0,00	-48,94	0,00	57,14	0,00
VSA_a2	84,44	-0,85	86,02	-0,08	-45,83	0,78	48,93	0,00
VSA_u3	135,97	13,93	107,12	10,90	-53,00	-2,66	92,38	0,00
VSA_c2	74,67	8,22	61,40	-2,20	-44,01	-9,63	25,21	-0,20
VSA_m2	95,04	9,39	78,30	12,62	-42,26	2,96	48,56	0,09
VSA_v2	103,55	-1,89	107,47	8,13	-46,88	10,21	60,07	0,00
VSA_v1	269,65	13,62	225,34	18,68	-67,89	4,46	174,27	0,00
VSA_m3	76,17	-3,85	83,22	-8,82	-48,24	-5,16	68,60	0,00
VSA_u1	78,91	-60,87	357,21	7,63	-39,84	175,05	19,15	-0,00
VSA_a1	86,37	6,70	74,67	4,18	-44,10	-2,36	54,56	-0,09
VSA_m1	81,63	-3,60	88,42	-3,57	-46,91	0,03	29,95	-0,01
VSA_u4	182,88	-11,47	219,54	-3,44	-65,86	9,08	73,47	0,00

Tabella 4.9: Riassunto differenze relative tra bundle15 e bundle05 con doppie iterazioni limite

	tempo	n.chiam	t/n.ch	n.it	n.it/t	n.it/n.ch	t.TOT	best sol
media	120,68	2,20	125,43	8,06	-48,68	12,86	66,08	0,01
dev.std	65,24	21,05	80,16	14,85	9,15	43,80	42,85	0,09

Nelle tabelle 4.10 e 4.11 sono esposti gli esiti diretti dei test, con entrambe le versioni del bundle.

4.4 Nuovi parametri del bundle

I parametri del bundle, come l'algoritmo stesso, sono rimasti invariati da più di dieci anni. Potrebbe accadere, quindi, che con la nuova versione del solver si possano ottenere risultati migliori modificandone alcuni.

In tale ottica, nel 2015, è stato effettuato da Giuliano Vallese (si rimanda a [8]) un *tuning* dei parametri, i cui risultati giustificano l'analisi che segue.

Prima di riportare i risultati, però, è necessario fornire una breve descrizione di suddetti parametri.

tStar: rappresenta una stima del massimo passo che può essere effettuato nella ricerca del *Serious Step*.

tInit: è il primo valore assegnato al parametro di prossimità t, definito nella sezione 2.5. Il suo ordine di grandezza può essere calcolato tramite delle

Tabella 4.10: Risultati dei test Risultati dei test con bundle15 e bundle05 con iterazioni limite doppie, prima parte

	bundle15				bundle05			
CASO	tempo	n.chiam	t/n.ch	n.it	tempo	n.chiam	t/n.ch	n.it
VSA_a3	3750,94	533	7,04	178057	9190,70	611	15,04	233332
VSA_c1	9719,33	1109	8,76	378198	13698,50	1152	11,89	365171
VSA_p	12422,10	2048	6,07	695284	44615,80	2973	15,01	1038225
VSA_u2	14,97	42	0,36	3286	29,31	42	0,70	3286
VSA_a2	1976,19	471	4,20	107484	3644,93	467	7,80	107398
VSA_u3	5620,75	1156	4,86	418819	13263,30	1317	10,07	464461
VSA_c2	935,08	292	3,20	69535	1633,29	316	5,17	68004
VSA_m2	183,63	639	0,29	128499	358,14	699	0,51	144721
VSA_v2	1018,07	901	1,13	236977	2072,30	884	2,34	256252
VSA_v1	11437,00	1946	5,88	611380	42276,60	2211	19,12	725582
VSA_m3	3891,52	727	5,35	156402	6855,59	699	9,81	142613
VSA_u1	30,39	161	0,19	33454	54,37	63	0,86	36006
VSA_a1	5379,78	403	13,35	104401	10026,40	430	23,32	108770
VSA_m1	227,11	222	1,02	40900	412,49	214	1,93	39439
VSA_u4	77,31	401	0,19	59182	218,69	355	0,62	57149

Tabella 4.11: Risultati dei test Risultati dei test con bundle15 e bundle05 con iterazioni limite doppie, seconda parte

	bundle15				bundle05			
CASO	n.it/t	n.it/n.ch	t.TOT	best sol.	n.it/t	n.it/n.ch	t.TOT	best sol.
VSA_a3	47,47	334,07	195,42	378,06	25,39	381,89	292,87	378,96
VSA_c1	38,91	341,03	181,97	197,69	26,66	316,99	244,93	197,63
VSA_p	55,97	339,49	442,92	110,63	23,27	349,22	1126,85	110,74
VSA_u2	219,54	78,24	0,47	74,49	112,10	78,24	0,73	74,49
VSA_a2	54,39	228,20	55,52	624,22	29,47	229,97	82,68	624,22
VSA_u3	74,51	362,30	152,20	108,65	35,02	352,67	292,80	108,65
VSA_c2	74,36	238,13	37,22	223,07	41,64	215,20	46,60	222,61
VSA_m2	699,79	201,09	8,10	246,47	404,09	207,04	12,03	246,69
VSA_v2	232,77	263,02	30,47	149,67	123,66	289,88	48,77	149,67
VSA_v1	53,46	314,17	356,35	55,71	17,16	328,17	977,37	55,72
VSA_m3	40,19	215,13	71,50	130,49	20,80	204,02	120,55	130,49
VSA_u1	1100,90	207,79	1,57	97,97	662,28	571,52	1,87	97,97
VSA_a1	19,41	259,06	133,30	76,05	10,85	252,95	206,03	75,98
VSA_m1	180,09	184,23	9,68	48,15	95,61	184,29	12,58	48,14
VSA_u4	765,54	147,59	3,27	31,62	261,32	160,98	5,67	31,62

euristiche e spesso risulta essere uno o due volte minore di quello di *tStar*.

m1: una volta trovata una direzione di discesa d^* rispetto al punto corrente $\bar{x}$ viene imposta una condizione per stabilire se $\bar{x} + d^*$ migliori abbastanza la soluzione da poter effettuare un *Serious Step*. Questa condizione è gestita dal parametro $m1$.

In particolare, il passo viene effettuato se

$$f(\bar{x} + d^*) - f(\bar{x}) \geq |m1|\Delta_f,$$

dove

$$\Delta_f = \begin{cases} f(\bar{x} + d^*) - f(\bar{x}) & \text{se } m1 > 0 \\ f(\bar{x} + d^*) - f(\bar{x}) - D_t(d^*) & \text{se } m1 < 0 \end{cases}.$$

Poiché la seconda condizione è più debole, potrebbe portare ad un maggior numero di Serious Step, quindi ad una convergenza più rapida.

Nella tabella 4.12 sono riportati i valori attualmente utilizzati dalla MAIOR e quelli che, a seguito dello studio effettuato, si sono rivelati i migliori.

Tabella 4.12: Valori parametri bundle

Parametri	default MAIOR	nuovi
tStar	2000	250
tInit	4000	500
m1	4000	500

Nella tabella 4.15 sono mostrati i risultati ottenuti. Come possiamo osservare, i nuovi parametri non apportano un miglioramento effettivo né in termini di tempo né nella qualità della soluzione, fatta eccezione per i casi VBDSA in cui il costo della miglior soluzione è sempre più basso.

Questa tendenza negativa si è amplificata notevolmente in alcuni casi critici, BDSA_2 e VBDSA_cc, in cui si è superato gli 8 giorni di esecuzione (addirittura il VBDSA_cc in tale intervallo di tempo ha calcolato una sola soluzione intera). Per questo motivo i due casi di test non presentano risultati.

Tale comportamento non è del tutto inaspettato: nella sperimentazione tenuta per il tuning, infatti, non veniva imposto un limite alle iterazioni del bundle, mentre, come spiegato nella sezione 4.3, esso è attivo di default nel codice dell'azienda. La scelta di limitare il numero massimo di iterazioni, è stata dettata principalmente da motivi di tempo. Dunque scoprire se vi siano dei limiti più alti con cui i nuovi parametri possano risultare vantaggiosi è un ottimo spunto per indagini future.

Per completezza, nelle tabelle 4.13 e 4.14 riportiamo i risultati diretti dei test effettuati rispettivamente con bundle15 e con bundle05.

Tabella 4.13: Risultati dei test con bundle15 e bundle05 con i nuovi parametri, prima parte

	bundle15				bundle05			
CASO	tempo	n.chiam	t/n.ch	n.it	tempo	n.chiam	t/n.ch	n.it
BDSA_b1	18248,80	5199	3,51	1270748	37661,50	5202	7,24	1270739
BDSA_2	#	#	#	#	#	#	#	#
BDSA_1	19160,70	8086	2,37	3341470	35959,70	8344	4,31	3501157
BDSA_b2	137,93	597	0,23	164686	259,10	625	0,41	158714
BDSA_b3	18,87	196	0,10	41801	26,34	187	0,14	33972
BDSA_c1	258,38	754	0,34	215143	460,34	741	0,62	208967
BDSA_c2	1395,41	1128	1,24	424067	2518,25	1106	2,28	401582
BDSA_a	33,04	366	0,09	67750	65,72	327	0,20	70545
VBDSA_3	2177,59	2368	0,92	758768	3142,87	2269	1,39	682625
VBDSA_cc	#	#	#	#	#	#	#	#
VBDSA_a2	4205,38	2627	1,60	855233	7772,30	2618	2,97	753225
VBDSA_c2	470,61	826	0,57	263172	902,49	919	0,98	271993
VBDSA_c1	512,70	857	0,60	208206	1037,33	821	1,26	189374
VBDSA_a1	43,00	415	0,10	99618	74,58	361	0,21	93702
VBDSA_g	391,35	760	0,51	304738	726,17	930	0,78	375519
VBDSA_f	107,66	357	0,30	96722	221,71	365	0,61	100442
VSA_a1	20896,3	1161	18,00	271179	38952,20	1083	35,97	258179
VSA_a3	6555,80	1103	5,94	238201	16842,50	1551	10,86	358481
VSA_c1	8294,50	1057	7,85	257249	8049,49	328	24,54	109783
VSA_p	11095,20	1943	5,71	427497	21368,60	2017	10,59	452286
VSA_u2	86,66	73	1,19	20195	203,46	73	2,79	20195
VSA_a2	3606,37	619	5,83	198240	5931,65	525	11,30	148772
VSA_u3	1282,03	475	2,70	112143	1762,43	626	2,82	139884
VSA_c2	8125,92	1231	6,60	293114	10856,80	972	11,17	235807
VSA_m2	1095,25	1089	1,01	264662	1971,28	1088	1,81	262919
VSA_v2	2134,37	633	3,37	140480	3182,87	648	4,91	134536
VSA_v1	2323,58	780	2,98	177436	3808,01	735	5,18	165112
VSA_m3	8764,57	902	9,72	310695	16270,50	1006	16,17	314399
VSA_u1	110,07	535	0,21	104867	238,35	529	0,45	102931
VSA_m1	920,04	459	2,00	111152	1950,65	532	3,67	120223
VSA_u4	231,83	509	0,46	121776	409,23	471	0,87	113284

4.5 Confronto con Cplex e Clp

In questa sezione vengono confrontati i risultati ottenuti tramite il bundle con quelli di altri solutori. Come abbiamo visto, l'utilizzo del bundle15 al posto del bundle05 apporta notevoli miglioramenti all'algoritmo, quindi, similmente a ciò che abbiamo fatto nei paragrafi precedenti, ci limiteremo ad analizzare il comportamento di questa versione.

Nella sezione 3.3 abbiamo visto che tramite l'MPSOsi è possibile risolvere il Master Problem in modo diretto, utilizzando Cplex o Clp.

Tabella 4.14: Risultati dei test con bundle15 e bundle05 con i nuovi parametri, seconda parte

	bundle15				bundle05			
CASO	n.it/t	n.it/n.ch	t.TOT	best sol.	n.it/t	n.it/n.ch	t.TOT	best sol.
BDSA_b1	69,63	244,42	3430,87	626,54	33,74	244,28	3793,77	626,74
BDSA_2	#	#	#	#	#	#	#	#
BDSA_1	174,39	413,24	963,48	109,58	97,36	419,60	1281,25	111,21
BDSA_b2	1193,98	275,86	6,47	174,08	612,56	253,94	8,73	175,97
BDSA_b3	2215,21	213,27	5,80	196,84	1289,90	181,67	5,83	196,72
BDSA_c1	832,66	285,34	15,33	51,59	453,94	282,01	19,05	51,59
BDSA_c2	303,90	375,95	207,00	33,52	159,47	363,09	200,20	33,43
BDSA_a	2050,36	185,11	2,63	62,95	1073,35	215,73	3,23	62,89
VBDSA_3	348,44	320,43	446,93	381,61	217,20	300,85	440,28	391,80
VBDSA_cc	#	#	#	#	#	#	#	#
VBDSA_a2	203,37	325,56	390,60	129,39	96,91	287,71	446,48	131,78
VBDSA_c2	559,22	318,61	89,85	107,21	301,38	295,97	96,30	107,07
VBDSA_c1	406,10	242,95	67,55	169,44	182,56	230,66	78,53	169,44
VBDSA_a1	2316,59	240,04	40,72	91,93	1256,48	259,56	42,00	91,98
VBDSA_g	778,68	400,97	21,17	51,60	517,12	403,78	25,68	51,60
VBDSA_f	898,40	270,93	9,63	65,25	453,04	275,18	11,47	65,25
VSA_a1	12,98	233,57	517,58	76,20	6,63	238,39	814,42	76,21
VSA_a3	36,33	215,96	348,68	379,71	21,28	231,13	548,87	378,36
VSA_c1	31,01	243,38	158,08	197,85	13,64	334,70	146,57	197,89
VSA_p	38,53	220,02	484,62	111,93	21,17	224,24	656,02	111,91
VSA_u2	233,05	276,64	1,83	74,49	99,26	276,64	3,88	74,49
VSA_a2	54,97	320,26	101,37	621,65	25,08	283,38	141,52	621,77
VSA_u3	87,47	236,09	40,75	108,94	79,37	223,46	46,70	108,96
VSA_c2	36,07	238,11	234,68	222,52	21,72	242,60	258,95	222,62
VSA_m2	241,65	243,03	40,35	246,06	133,37	241,65	54,97	246,05
VSA_v2	65,82	221,93	59,72	149,73	42,27	207,62	76,97	149,79
VSA_v1	76,36	227,48	64,63	56,37	43,36	224,64	87,02	56,39
VSA_m3	35,45	344,45	157,28	130,49	19,32	312,52	282,28	130,49
VSA_u1	952,75	196,01	6,20	97,97	431,84	194,58	9,07	97,97
VSA_m1	120,81	242,16	29,35	48,21	61,63	225,98	47,42	48,19
VSA_u4	525,29	239,25	7,07	31,63	276,82	240,52	10,30	31,63

Tabella 4.15: Differenze relative percentuali tra parametri nuovi e di default con bundle15

CASO	tempo	n.chiam	t/n.ch	n.it	n.it/t	n.it/n.ch	t.TOT	best sol
BDSA_b1	-83,82	-81,73	-11,48	-82,82	6,23	-5,96	-90,63	-3,37
BDSA_2	#	#	#	#	#	#	#	#
BDSA_1	-95,61	-80,48	-77,50	-90,33	120,18	-50,46	-88,31	-0,08
BDSA_b2	-64,77	-26,30	-52,20	-62,77	5,69	-49,49	-52,58	0,03
BDSA_b3	-70,45	-39,29	-51,33	-71,94	-5,05	-53,79	-42,82	-0,03
BDSA_c1	-95,64	-79,97	-78,20	-92,98	60,82	-64,95	-84,78	-0,68
BDSA_c2	-93,95	-82,36	-65,73	-90,88	50,82	-48,31	-83,32	-0,53
BDSA_a	-60,87	-19,95	-51,12	-61,03	-0,42	-51,32	-34,18	-0,15
VBDSA_3	-90,37	-59,16	-76,42	-80,29	104,73	-51,73	-74,79	2,08
VBDSA_cc	#	#	#	#	#	#	#	#
VBDSA_a2	-93,70	-77,46	-72,06	-87,41	99,93	-44,14	-80,97	0,62
VBDSA_c2	-81,74	-31,36	-73,39	-68,50	72,48	-54,11	-58,38	0,17
VBDSA_c1	-86,01	-47,84	-73,18	-69,39	118,86	-41,31	-59,61	0,05
VBDSA_a1	-72,94	-25,54	-63,66	-70,06	10,66	-59,79	-39,58	0,37
VBDSA_g	-76,84	-53,16	-50,56	-67,00	42,51	-29,55	-60,16	0,24
VBDSA_f	-84,64	-63,59	-57,83	-79,23	35,24	-42,97	-65,40	0,62
VSA_a1	-81,22	-57,28	-56,05	-69,21	64,01	-27,92	-75,79	-0,13
VSA_a3	-16,88	-21,12	5,37	-12,01	5,87	11,56	11,43	-0,28
VSA_c1	-14,42	20,25	-28,83	17,49	37,29	-2,29	-12,09	-0,11
VSA_p	17,51	58,16	-25,70	38,88	18,19	-12,19	22,48	-0,13
VSA_u2	-88,29	-42,47	-79,65	-88,42	-1,06	-79,87	-80,00	-0,00
VSA_a2	-54,36	-18,90	-43,72	-54,99	-1,38	-44,50	-48,26	0,34
VSA_u3	131,46	175,16	-15,88	173,83	18,30	-0,48	130,80	-0,16
VSA_c2	-92,83	-74,82	-71,55	-81,43	159,17	-26,26	-86,66	0,55
VSA_m2	-90,17	-48,67	-80,86	-68,89	216,59	-39,40	-82,98	0,29
VSA_v2	-25,45	60,35	-53,51	61,52	116,65	0,73	-25,09	-0,04
VSA_v1	221,66	139,87	34,10	89,53	-41,08	-20,99	346,55	-1,11
VSA_m3	-58,10	-18,63	-48,51	-54,78	7,93	-44,43	-56,60	0,00
VSA_u1	-87,06	-74,39	-49,46	-78,79	63,92	-17,16	-79,30	-0,00
VSA_m1	-73,85	-38,34	-57,59	-59,21	55,99	-33,84	-60,31	-0,09
VSA_u4	-83,07	-23,97	-77,73	-65,23	105,36	-54,27	-70,75	-0,04

Tra i vari metodi disponibili si è scelto di effettuare i test con il simplesso duale. Tale decisione è motivata principalmente da sperimentazioni svolte in passato che hanno mostrato la maggior efficacia di questo approccio rispetto all'uso del primale e del barrier.

Nella tabella 4.16 abbiamo riportato i valori ottenuti con i tre solutori. Nella tabella 4.17, invece, vi sono le differenze relative percentuali tra tali valori. I colori sono stati scelti in modo da far vedere quale sia il solver con la miglior prestazione in ogni campo: arancione per Cplex, azzurro per Clp e fucsia per Bundle.

Osserviamo subito che Cplex, ad eccezione di alcuni sporadici casi, ha prestazioni decisamente migliori di Cpl, soprattutto in termini di tempo ma a volte anche per costo della soluzione finale (si veda ad esempio BSDA_b3 e VSA_a3). Per un corretto confronto bisogna però ricordare che il secondo, a differenza del primo, dispone di una licenza *open source*.

Analizziamo ora nello specifico i risultati dei test.

- Le soluzioni finali ottenute da Cplex sono paragonabili a quelle del bundle, fatta eccezione per il caso VSA_p. Con Cpl, invece, vi è un numero maggiore di casi in cui il costo della soluzione è nettamente più alto rispetto a quello ottenuto con gli altri due solutori.

- Cplex ha un tempo medio per chiamata molto ridotto: solo in due casi, che rientrano nella "fascia di difficoltà" più alta, si supera il tempo di esecuzione di $1s$. Gli altri due solver, invece, presentano tempi più alti: $2s$ di media il bundle e quasi $5s$ Clp, che con VSA_a3 supera perfino i $65s$.

- Il numero di esecuzioni del solver (quindi di generazioni di colonne) di Cplex è di media maggiore di quello del bundle e ciò potrebbe causare anche un aumento del tempo totale di esecuzione del programma; in effetti, la maggior parte dei casi in cui il tempo totale del Cplex è minore, vi anche una riduzione nel numero di chiamate del solver.

- Il bundle offre prestazioni decisamente migliori sul tempo totale di esecuzione rispetto a Cplex e Clp: il primo, infatti, ha tempi più alti del 330%, l'altro del 1070%.

A prescindere dai vari trends, si può notare che la varianza dei valori è molto alta.

Tabella 4.16: Risultati dei test con Cplex, Clp e Bundle

	Cplex					Clp					Bundle15				
CASO	tempo	n.ch.	t/n.ch.	t.TOT	best sol	tempo	n.ch.	t/n.ch.	t.TOT	best sol	tempo	n.ch.	t/n.ch.	t.TOT	best sol
BDSA_b2	35,72	1209	0,03	19,98	174,04	170,41	795	0,21	16,45	239,35	48,59	440	0,11	3,07	174,13
BDSA_b3	22,67	547	0,04	38,42	197,90	389,78	906	0,43	73,33	563,08	5,58	119	0,05	3,32	196,78
BDSA_c1	166,32	955	0,17	17,37	52,09	2234,92	1233	1,81	59,38	52,04	11,28	151	0,07	2,33	51,24
BDSA_a	2,67	178	0,01	4,32	62,91	6,28	240	0,03	5,87	62,88	12,93	293	0,04	1,73	62,85
VBDSA_a1	14,94	421	0,04	212,40	92,13	90,46	1618	0,06	846,55	93,43	11,64	309	0,04	24,60	92,27
VBDSA_g	600,82	839	0,72	69,97	51,70	1732,05	2558	0,68	213,18	51,98	90,63	356	0,25	8,43	51,73
VBDSA_f	21,58	236	0,09	15,17	65,57	159,80	729	0,22	48,13	65,29	16,53	130	0,13	3,33	65,65
VSA_a1	679,89	1252	0,54	164,27	75,91	2733,52	995	2,75	199,75	75,94	3923,31	496	7,91	125,32	76,10
VSA_a3	2150,94	1271	1,69	509,90	378,63	116893,00	1812	64,51	2897,40	871,47	5448,86	870	6,26	388,53	378,63
VSA_c1	948,58	2004	0,47	136,33	197,65	3355,31	1793	1,87	181,62	197,69	7098,30	1271	5,58	138,97	197,64
VSA_p	571,88	186	3,07	82,65	965,96	9335,60	801	11,65	649,85	1000,24	13038,00	3073	4,24	593,58	111,78
VSA_u2	0,32	41	0,01	0,27	74,49	0,50	45	0,01	0,30	74,49	10,14	42	0,24	0,37	74,49
VSA_a2	500,16	1533	0,33	307,75	621,52	2534,99	1661	1,53	394,68	625,30	1645,97	502	3,28	52,45	623,79
VSA_u3	750,96	1009	0,74	70,23	108,66	4566,28	1269	3,60	174,82	108,87	2967,39	1307	2,27	94,05	108,77
VSA_c2	1013,30	1166	0,87	174,72	224,09	4269,01	1490	2,87	313,43	223,44	582,24	310	1,88	31,30	223,75
VSA_m2	1181,20	2455	0,48	50,62	246,24	3058,29	1192	2,57	88,93	246,51	107,62	559	0,19	6,87	246,78
VSA_v2	857,44	2909	0,29	98,90	149,64	9725,18	2206	4,41	283,77	159,01	1591,23	1015	1,57	44,73	149,66
VSA_v1	1418,92	1568	0,90	193,50	55,75	4620,34	1516	3,05	260,77	56,07	7474,06	1871	3,99	288,62	55,75
VSA_m3	19,39	741	0,03	21,88	130,49	72,53	535	0,14	16,10	130,49	3672,01	734	5,00	68,27	130,49
VSA_u1	199,64	671	0,30	11,20	97,96	1625,32	1210	1,34	44,45	97,96	14,24	137	0,10	1,28	97,97
VSA_m1	49,60	249	0,20	18,72	48,15	2501,49	1087	2,30	144,25	48,15	240,58	283	0,85	11,65	48,16
VSA_u4	217,86	860	0,25	14,43	31,61	2180,67	996	2,19	58,58	31,61	39,25	387	0,10	2,07	31,62

Tabella 4.17: Differenze relative percentuali tra Cplex, Clp e Bundle

	$\frac{Cplex-Clp}{Clp}$%					$\frac{Clp-Bundle}{Bundle}$%					$\frac{Cplex-Bundle}{Bundle}$%				
CASO	tempo	n.ch.	t/n.ch.	t.TOT	best sol	tempo	n.ch.	t/n.ch.	t.TOT	best sol	tempo	n.ch.	t/n.ch.	t.TOT	best sol
BDSA_b2	-79,04	52,08	-86,22	21,48	-27,28	250,73	80,68	94,11	436,41	37,46	-26,49	174,77	-73,25	551,63	-0,05
BDSA_b3	-94,19	-39,62	-90,37	-47,61	-64,85	6890,33	661,34	818,16	2111,05	186,14	306,47	359,66	-11,57	1058,29	0,57
BDSA_c1	-92,56	-22,55	-90,39	-70,75	0,10	19716,63	716,56	2326,85	2445,03	1,56	1374,77	532,45	133,18	644,30	1,66
BDSA_a	-57,53	-25,83	-42,73	-26,42	0,05	-51,44	-18,09	-40,71	238,46	0,05	-79,37	-39,25	-66,05	149,04	0,09
VBDSA_a1	-83,49	-73,98	-36,53	-74,91	-1,40	677,45	423,62	48,47	3341,26	1,27	28,39	36,25	-5,77	763,41	-0,15
VBDSA_g	-65,31	-67,20	5,76	-67,18	-0,55	1811,19	618,54	165,98	2427,86	0,50	562,96	135,67	181,30	729,64	-0,05
VBDSA_f	-86,50	-67,63	-58,29	-68,49	0,42	866,55	460,77	72,36	1344,00	-0,56	30,53	81,54	-28,10	355,00	-0,13
VSA_a1	-75,13	25,83	-80,23	-17,76	-0,05	-30,33	100,60	-65,27	59,40	-0,21	-82,67	152,42	-93,13	31,08	-0,25
VSA_a3	-98,16	-29,86	-97,38	-82,40	-56,55	2045,27	108,28	930,02	645,73	130,16	-60,52	46,09	-72,98	31,24	-0,00
VSA_c1	-71,73	11,77	-74,71	-24,93	-0,02	-52,73	41,07	-66,49	30,69	0,03	-86,64	57,67	-91,52	-1,90	0,01
VSA_p	-93,87	-76,78	-73,62	-87,28	-3,43	-28,40	-73,93	174,70	9,48	794,81	-95,61	-93,95	-27,53	-86,08	764,15
VSA_u2	-35,89	-8,89	-29,63	-11,11	0,00	-95,11	7,14	-95,44	-18,18	0,00	-96,87	-2,38	-96,79	-27,27	0,00
VSA_a2	-80,27	-7,71	-78,62	-22,03	-0,60	54,01	230,88	-53,45	652,49	0,24	-69,61	205,38	-90,05	486,75	-0,36
VSA_u3	-83,55	-20,49	-79,32	-59,82	-0,19	53,88	-2,91	58,49	85,88	0,09	-74,69	-22,80	-67,22	-25,32	-0,10
VSA_c2	-76,26	-21,74	-69,67	-44,26	0,29	633,20	380,65	52,55	901,38	-0,14	74,03	276,13	-53,73	458,20	0,15
VSA_m2	-61,38	105,96	-81,25	-43,08	-0,11	2741,78	113,24	1232,68	1195,14	-0,11	997,58	339,18	149,92	637,14	-0,22
VSA_v2	-91,18	31,87	-93,31	-65,15	-5,89	511,17	117,34	181,21	534,35	6,24	-46,11	186,60	-81,20	121,09	-0,02
VSA_v1	-69,29	3,43	-70,31	-25,80	-0,56	-38,18	-18,97	-23,71	-9,65	0,57	-81,02	-16,19	-77,35	-32,96	0,01
VSA_m3	-73,27	38,50	-80,70	35,92	0,00	-98,02	-27,11	-97,29	-76,42	0,00	-99,47	0,95	-99,48	-67,94	0,00
VSA_u1	-87,72	-44,55	-77,85	-74,80	0,00	11310,56	783,21	1191,94	3363,65	-0,01	1301,59	389,78	186,17	772,73	-0,01
VSA_m1	-98,02	-77,09	-91,34	-87,02	0,00	939,78	284,10	170,71	1138,20	-0,03	-79,38	-12,01	-76,57	60,66	-0,02
VSA_u4	-90,01	-13,65	-88,43	-75,36	0,00	5455,28	157,36	2058,53	2734,67	-0,01	455,00	122,22	149,75	598,38	-0,01

Tabella 4.18: Riassunto dei risultati di Cplex, Clp e Bundle e delle differenze relative

	Cplex				Clp				Bundle15			
	tempo	n.ch.	t/n.ch.	t.TOT	best sol	tempo	n.ch.	t/n.ch.	t.TOT	best sol	tempo	n.ch.	t/n.ch.	t.TOT	best sol
media	519,31	1013,64	0,51	101,50	186,50	7829,81	1213,05	4,92	316,89	230,70	2184,11	666,14	2,01	86,13	147,73
dev.std	557,94	730,79	0,69	120,47	215,06	23944,80	587,04	13,23	600,08	270,36	3332,51	692,29	2,39	147,49	132,22

	Cplex				Clp				Bundle15			
	tempo	n.ch.	t/n.ch.	t.TOT	best sol	tempo	n.ch.	t/n.ch.	t.TOT	best sol	tempo	n.ch.	t/n.ch.	t.TOT	best sol
media	-79,29	-14,92	-71,14	-46,31	-7,30	2434,71	233,83	415,20	1072,31	52,64	188,77	132,28	-14,18	327,60	34,78
dev.std	14,81	45,53	24,43	33,35	17,87	4677,47	261,37	691,14	1122,61	168,39	452,23	159,54	98,34	342,75	159,16

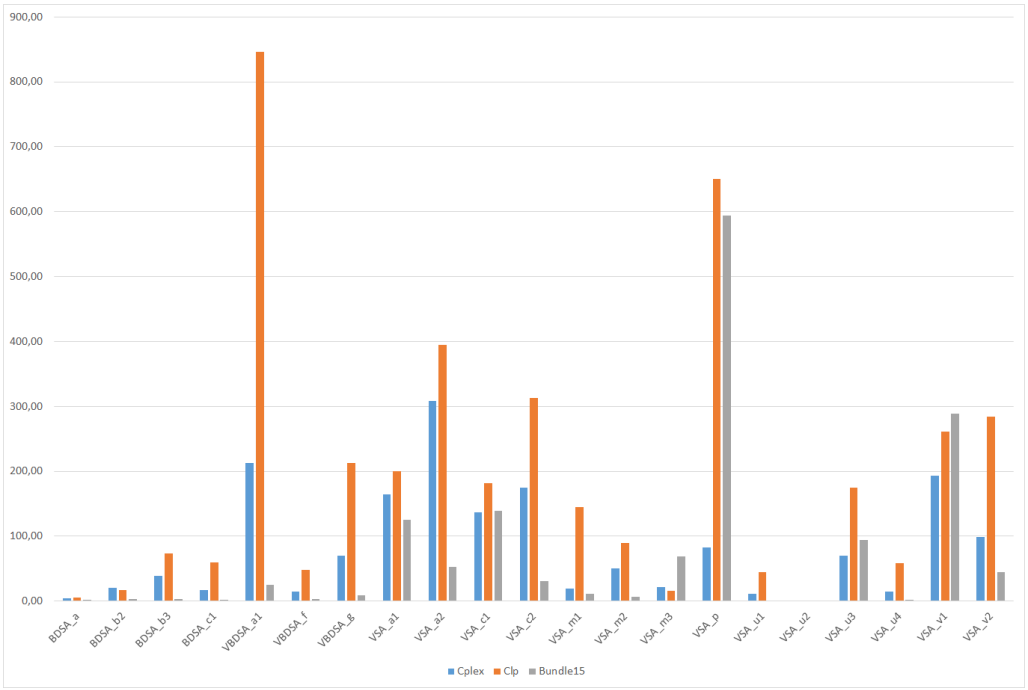

Figura 4.2: Tempi totali di Cplex, Clp e Bundle

Bibliografia

[1] A. Ben Amor, J. Desrosiers, and A. Frangioni, *On the choice of explicit stabilizing terms in column generation*, Discrete Applied Mathematics, 157 (2009), pp. 1170–1172.

[2] F. Bernazzani, S. Carosi, A. Frangioni, A. Gaiffi, and L. Girardi, *Miglioramenti algoritmici nella soluzione di problemi reali di schedulazione di veicoli e personale*, Scienza delle decisioni in Italia: applicazioni della ricerca operativa a problemi aziendali, G. Felici e A. Sciomachen eds., EGIC Genova, p. 429–442.

[3] P. Carraresi, A. Frangioni, and M. Nonato, *Applied bundle methods to the optimization of polyhedral functions: an applications-oriented development*, Technical Report del Dipartimento di Informatica, Università di Pisa, TR-96-17, (1997).

[4] A. Ferruggia, *Redesign di una componente di ottimizzazione per un ambiente parallelo e distribuito*, Tesi di Laurea in Informatica, Facoltà di Scienze Matematiche Fisiche e Naturali Corso di Laurea in Informatica, Università di Pisa, pp. 16–22.

[5] A. Frangioni, *Solving semidefnite quadratic problems within nonsmooth optimization algorithms*, Technical Report del Dipartimento di Informatica, Università di Pisa, TR-95-10, (1995), p. 1.

[6] ——, *Generealized bundle methods*, SIAM Journal on Optimization, 13 (2002), pp. 117–156.

[7] ——, *About lagrangian methods in integer optimization*, Annals of Operations Research, 139 (2005), pp. 163–166.

[8] G. Vallese, *Valutazione sperimentale di metodi risolutivi per problemi master nella generazione di colonne*, Tesi di Laurea Magistrale in Matematica, Università di Pisa, (2015).