ALMA MATER STUDIORUM – UNIVERSITÀ DI BOLOGNA

Corso di Laurea Magistrale in Ingegneria e Scienze Informatiche

Modelli per l'analisi dello sprint planning di Scrum

Tesi di laurea in:
Project Management

Relatore
Marco Antonio Boschetti

Candidato
Daniel Capannini

I Sessione di Laurea
Anno Accademico 2024-2025

Abstract

La presente tesi analizza modelli matematici per la pianificazione degli sprint in progetti software basati su metodologie Agile, con focus sul framework Scrum. Il lavoro formula il problema di assegnazione delle user story agli sprint, incorporando vincoli di capacità, dipendenze tra storie e obiettivi di massimizzazione del valore utente e minimizzazione del rischio.

Vengono esaminati modelli esistenti in letteratura, ne vengono identificate le limitazioni e si propongono nuove formulazioni che integrano funzioni obiettivo avanzate, sensibili ai vincoli di dipendenza.

I risultati dimostrano che tali vincoli influenzano significativamente la pianificazione degli sprint e che le funzioni obiettivo che li considerano generano piani mediamente meno rischiosi rispetto a quelle che li ignorano. Inoltre, i nuovi modelli proposti consentono di variare la dimensione degli sprint per adattarsi alle caratteristiche del progetto e supportano la ripianificazione in caso di cambiamenti durante l'esecuzione.

Nah, i'd win

Contents

List of Figures

List of Listings

Chapter 1

Introduzione

I tradizionali approcci di sviluppo software mostrano limiti crescenti nel rispondere alle esigenze dinamiche del business moderno, e numerosi studi empirici suggeriscono che l'agilità rappresenta una delle soluzioni più promettenti per superarli [Abrahamsson et al., 2003b, Dybå and Dingsøyr, 2008]. I metodi Agile, tra cui *Scrum* ed *eXtreme Programming* (XP), si fondano sui dodici principi del Manifesto Agile [Beck et al., 2001b] e sono ormai adottati da un numero sempre maggiore di aziende per rendere lo sviluppo software più rapido, adattivo e indirizzato al valore. In particolare, il metodo Scrum è stato ampiamente discusso in letteratura [Schwaber, 1995], che ne descrive le idee fondamentali e il ciclo di vita operativo.

Un elemento comune alle metodologie Agile è la progettazione e implementazione incrementale, in cui il software è suddiviso in funzionalità specifiche (*user stories*) e, ad ogni iterazione (*sprint*), il team è chiamato a fornire il set di *user stories* che massimizza il valore per l'utente, soddisfacendo al tempo stesso vincoli di durata, dipendenze e relazioni tra le storie. Secondo il principio della *team awareness*, la pianificazione dello sprint si basa sulla condivisione e la mediazione delle stime fornite dai membri del team riguardo alla complessità delle storie, utilità, rischio di mancata consegna e dipendenze. anticipare user story ad alto valore per il cliente porta a risultati significativi per l'utente, rafforzando la consapevolezza del team; analogamente, anticipare user story critiche consente di limitare rischi di impatto tardivo, bilanciando però una maggiore probabilità di ritardi nelle

fasi iniziali. Il contributo di [Cohn, 2004a] offre indicazioni pratiche per valutare complessità e valore delle *user stories*.

Una pianificazione di sprint efficace è unanimemente riconosciuta come fattore chiave per il successo di un progetto agile [Cohn, 2005]. L'efficacia del piano dipende sia dalla precisione delle stime (aspetto legato all'esperienza del team), sia dalla capacità di considerare correttamente variabili e vincoli progettuali: quest'ultimo obiettivo si traduce in un problema di ottimizzazione la cui complessità cresce con la dimensione del progetto, e il cui mancato raggiungimento può generare inefficienze, extracosti e ritardi.

Questa tesi prende come riferimento il lavoro [Boschetti et al., 2014], il quale proponeva dei modelli euristici per la pianificazione degli sprint, dal quale si parte per valutare l'efficacia dei modelli proposti e analizzare come il variare delle proprietà dei progetti influenzi le prestazioni degli algoritmi. Successivamente, si introducono nuove funzioni obiettivo da integrare nei modelli, al fine di analizzare il loro impatto sulla qualità delle soluzioni ottenute. Infine, si propongono nuovi modelli per migliorare ulteriormente la qualità delle soluzioni e rendere più flessibile la parametrizzazione di user story e sprint all'interno delle varie istanze.

Chapter 2

Descrizione del problema

Poiché la soddisfazione del cliente rappresenta uno dei principali indicatori di successo di un progetto, i metodi agili si propongono di rispondere in modo più efficace alle esigenze degli utenti, riducendo i tempi di consegna e aumentando la flessibilità del processo di sviluppo. Accelerare il time-to-market consente infatti di ridurre la pressione competitiva, mentre la flessibilità permette di adattarsi rapidamente sia ai progressi tecnologici sia ai cambiamenti nei requisiti degli utenti. Per conseguire tali obiettivi, l'approccio agile adotta una serie di pratiche complementari, tra cui lo sviluppo incrementale e iterativo, il coinvolgimento continuo degli utenti, la consapevolezza del team, e una documentazione leggera ma efficace. Dall'esperienza dei professionisti del settore sono nati diversi framework agili basati su questi principi. Tra questi, Scrum e gli approcci ibridi ,che combinano pratiche di Scrum ed eXtreme Programming (XP), risultano i più diffusi nelle aziende: Scrum si concentra sugli aspetti organizzativi e di gestione del processo, mentre XP introduce pratiche tecniche specifiche, come la programmazione in coppia, per migliorare la qualità e l'efficienza dello sviluppo del codice.

Il ciclo di vita di un progetto Scrum è suddiviso in varie fasi. Durante la definizione delle user story, il team di sviluppo e gli utenti collaborano per definire la struttura generale del sistema e individuare un insieme di funzionalità. Una user story rappresenta una funzionalità di piccole dimensioni ma di elevato valore per l'utente [Cohn, 2004b], descritta in modo semplice e sintetico. Essa costituisce un vincolo "leggera", che può essere ulteriormente approfondita attraverso un

dialogo continuo con l'utente, ma deve al contempo contenere informazioni sufficienti a permettere al team di stimarne la complessità di sviluppo e pianificarne l'implementazione in modo efficace.

Durante la fase di stima delle user story, a ciascuna storia vengono attribuiti due principali parametri: l'utilità e la complessità. Le valutazioni possono essere effettuate sulla base di esperienze pregresse, pareri di esperti o tramite tecniche basate sul consenso come il planning poker. L'utilità rappresenta il valore per l'azienda percepito dall'utente che ha definito la user story. In alcuni casi è sufficiente stabilire un ordine di priorità tra le storie, mentre in altri l'utilità viene quantificata mediante un punteggio numerico. La complessità di sviluppo, invece, viene espressa in story point, un'unità adimensionale che viene preferita rispetto a misure di tempo o risorse per ridurre la soggettività e favorire la comparabilità tra le stime. L'assegnazione degli story point è effettuata dai membri del team in base alla loro esperienza, alla conoscenza del dominio e alle caratteristiche specifiche del progetto. Sono utilizzabili diverse tecniche di dimensionamento [Cohn, 2005] tra cui l'utilizzo di scale numeriche, come la sequenza di Fibonacci, intervalli di valori o l'adozione di sistemi che si basano su categorie qualitative, come l'utilizzo delle taglie delle magliette.

Durante la fase di prioritizzazione delle user story, il team assegna a ciascuna storia un livello di priorità basato principalmente sulla sua utilità e sul rischio associato, oltre a identificare le eventuali dipendenze tra le diverse storie. Vi sono due categorie di fattori di rischio, vi sono le storie critiche: si tratta di user story che esercitano un forte impatto sulle altre e rappresentano componenti fondamentali del progetto. Una scelta errata nell'assegnazione o nella stima di queste storie può compromettere il successo complessivo dello sprint e dei cicli successivi, creando effetti a cascata su altre dipendenze. E vi sono le storie incerte, caratterizzate da un elevato grado di difficoltà nella stima del tempo e dello sforzo richiesto. Questa incertezza nasce da potenziali imprevisti, come cambiamenti nei requisiti espressi dall'utente, problemi tecnici inaspettati durante l'implementazione, o complessità che emergeranno solo durante lo sviluppo. Entrambe le categorie di rischio vengono generalmente rappresentate tramite valori numerici. Le dipendenze, invece,

esprimono vincoli di sequenza nello sviluppo, indicando che una user story può essere avviata solo dopo il completamento di una o più storie precedenti. Sebbene i metodi agili tendano a limitare le dipendenze per mantenere elevata la flessibilità del progetto, alcune di esse risultano inevitabili e devono essere rispettate per garantire la coerenza del processo di sviluppo.

L'elenco delle user story, una volta definite le priorità, costituisce il product backlog, che viene poi suddiviso in sprint durante la fase di pianificazione dello sprint. Gli sprint devono avere una durata breve e costante, solitamente compresa tra una e le quattro settimane, in modo da assicurare un feedback rapido e continuo da parte degli utenti. L'assegnazione delle user story agli sprint si basa su tre elementi fondamentali: la velocità di sviluppo del team, la complessità delle singole story e le possibili correlazioni o affinità tra le story. La velocità di sviluppo rappresenta il numero stimato di story point, che rappresenta l'unita minima per stimare lo sforzo necessario a completare una user story o un'attività, che il team è in grado di completare giornalmente. Questo valore viene utilizzato per calcolare la capacità dello sprint, ovvero il numero massimo di story point che il team è in grado di erogare durante l'intera durata dello sprint. Infine, l'affinità tra story si riferisce alle correlazioni logiche o funzionali tra user story che, se incluse nello stesso sprint, generano un valore maggiore per l'utente finale. Quando story correlate vengono realizzate insieme, gli utenti percepiscono più chiaramente il valore complessivo della funzionalità erogata e possono sperimentare un'esperienza più coerente e completa. Considerare l'affinità nella pianificazione permette quindi di massimizzare l'utilità percepita e di ottimizzare l'impatto di ogni rilascio incrementale.

Durante la fase di sviluppo dello sprint gli sviluppatori eseguono un'analisi dettagliata delle user story selezionate e producono lo sprint backlog, ovvero l'elenco operativo delle attività da completare. Successivamente, ogni membro del team si occupa di un sottoinsieme di story, seguendo l'intero ciclo di progettazione, implementazione e test. Al termine dello sprint si svolge la sprint review, durante la quale gli utenti e gli stakeholder esaminano le story completate per verificare che le funzionalità implementate corrispondano effettivamente ai requisiti richiesti.

Le story che superano la revisione vengono considerate completate e consegnate agli utenti come incremento del prodotto, mentre quelle non approvate vengono reinserite nel product backlog. In questa fase conclusiva, il team conduce anche una retrospettiva, analizzando i problemi riscontrati durante lo sprint e le soluzioni adottate, al fine di identificare opportunità di miglioramento per le iterazioni successive. Questo può includere, ad esempio, l'aggiornamento delle stime sulla base dell'esperienza maturata. Se dal feedback degli utenti emergono nuove user story o modifiche significative a quelle esistenti, oppure se le stime vengono riviste in modo sostanziale, viene eseguita una nuova fase di prioritizzazione del backlog prima dell'avvio dello sprint successivo, garantendo così che la pianificazione rimanga sempre allineata alle reali esigenze del progetto.

Il problema della pianificazione dello sprint, dato un insieme di user story e un insieme di sprint, consiste nell'allocare ogni story a uno sprint in modo da soddisfare tutti i vincoli relativi alla capacità dello sprint e alle dipendenze tra le storie, ottimizzando obiettivi come la soddisfazione del cliente e la gestione del rischio. Gli obbiettivi per soddisfare il cliente sono:

- Fornire prima le user story con maggiore utilità per aumentare la consapevolezza e la fiducia dell'utente.
- Includere storie affini nello stesso sprint per aumentarne il valore per gli utenti.

Mentre gli obbiettivi per la gestione del rischio sono:

- Promuovere le user story critiche per identificare tempestivamente problemi o effetti collaterali, evitando impatti negativi tardivi.
- Distribuire le user story incerte in diversi sprint e posticipandole, riducendo il rischio che più incertezze si concentrino nello stesso sprint, minimizzando la probabilità di ritardi nella consegna e garantendo una maggiore stabilità e prevedibilità del ciclo di sviluppo.

Il problema della pianificazione dello sprint può essere formulato come un Generalized Assignment Problem (GAP) con vincoli aggiuntivi. Questo tipo di prob-

lema consiste nell'assegnare un insieme di compiti a un insieme di risorse con l'obiettivo di ottimizzare una funzione (nel nostro caso, saranno varie per poter confrontare i loro risultati), rispettando i limiti di capacità di ciascuna risorsa e soddisfacendo vincoli aggiuntivi che impongono condizioni specifiche sulle assegnazioni.

Nel contesto della pianificazione agile, gli sprint rappresentano le risorse (o "contenitori"), mentre le user story costituiscono i compiti (o elementi) da assegnare. Ogni user story è caratterizzata da due attributi fondamentali: gli story point, che misurano il peso dell'elemento in termini di sforzo e complessità richiesti, e l'utilità, che ne rappresenta il valore per il progetto e per l'utente finale. La capacità dello sprint corrisponde al limite di capacità della risorsa ed è determinata dal numero massimo di story point che il team può completare, calcolato in base alla durata prevista dello sprint e alla velocità di sviluppo del team stesso. Questa capacità rappresenta quindi il vincolo principale che regola quante e quali user story possono essere assegnate a ciascun sprint.

Per quanto riguarda la gestione del rischio, nella formulazione del vincolo di capacità, gli story point delle user story vengono aumentati in proporzione alla loro incertezza. Questo meccanismo penalizza l'inclusione di due o più storie incerte nello stesso sprint, favorendo una loro distribuzione equilibrata su sprint diversi e riducendo così il rischio di ritardi nella consegna.

È importante sottolineare che il Problema di Assegnazione Generalizzato appartiene alla classe dei problemi NP-hard, il che significa che trovare una soluzione ottima richiede tempi computazionali che crescono esponenzialmente con le dimensioni del problema, rendendo necessario l'impiego di algoritmi euristici o metaeuristici per progetti di dimensioni reali.

Chapter 3

Stato dell'arte

I principi Agile hanno registrato una diffusione crescente nell'ultimo decennio, stimolando lo sviluppo di numerosi approcci metodologici proposti tanto dai professionisti quanto dalla comunità di ricerca [Abrahamsson et al., 2003a]. Una revisione sistematica presentata in letteratura [Dybå and Dingsøyr, 2008] confronta diversi metodi Agile analizzandone sia le caratteristiche organizzative che quelle tecniche, evidenziando in particolare la crescente adozione delle pratiche Scrum e XP nei contesti industriali. L'approccio Scrum è approfondito ulteriormente in [Schwaber, 1997], dove vengono descritti i fondamenti metodologici e il ciclo di vita caratteristico, contribuendo così alla comprensione dei meccanismi e delle peculiarità che contraddistinguono questo framework.

Per quanto concerne la stima del valore del software, la letteratura propone molteplici approcci senza tuttavia convergere verso un modello condiviso. Nel contesto più ampio dell'ingegneria del software basata sul valore, in [Rönkkö et al., 2009] viene introdotta una matrice di decomposizione del valore che integra tre aspetti del software (tecnologia, progettazione e artefatto) con tre componenti del valore (valore intrinseco, esternalità e valore dell'opzione), fornendo in ciascuna cella domande specifiche che guidano gli analisti nell'interpretazione delle diverse combinazioni. Alternativamente, in [Park and Park, 2004] viene proposto un modello per esprimere il valore del software in termini monetari, sfruttando la relazione tra fattori tecnologici e di mercato. Una sintesi più completa è fornita da [Khurum et al., 2013], che distinguono quattro prospettive di valore: finanziaria,

cliente, processo aziendale interno e innovazione e apprendimento. Gli autori sviluppano una Software Value Map (SVM) in cui ciascuna prospettiva è scomposta nelle sue componenti e sottocomponenti principali, offrendo agli analisti uno strumento per costruire una comprensione condivisa del valore. Sulla base della SVM, propongono inoltre un modello pratico articolato in tre fasi per ottenere la stima finale dell'utilità: selezione dello scenario caratterizzante il progetto specifico, identificazione del pattern di componenti di valore che meglio si adatta allo scenario selezionato, e definizione di una valutazione quantitativa del pattern attraverso la stima delle componenti individuate.

Nell'ampio contesto della pianificazione dei progetti, la maggior parte degli sforzi di ricerca degli ultimi anni si è concentrata sullo sviluppo di procedure esatte o euristiche per la generazione di una pianificazione di base praticabile in ambiente deterministico, con numerosi modelli e algoritmi proposti in letteratura come documentato nella panoramica di [Kolisch and Padman, 2001]. La complessità aumenta significativamente quando un team deve pianificare simultaneamente più progetti, situazione frequente nella pratica che richiede approcci specifici come evidenziato in [Platje et al., 1994]. In questo scenario, [De Boer, 1998] distingue due livelli di pianificazione: il primo livello, denominato pianificazione approssimativa della capacità, affronta il problema di pianificazione a medio termine, mentre il secondo livello, chiamato pianificazione di progetto con vincoli di risorse, si occupa della pianificazione operativa a breve termine. Questo approccio a due livelli, esplorato anche in [Gademann and Schutten, 2005], utilizza una scomposizione top-down delle attività in grandi pacchetti di lavoro per ridurre la complessità della pianificazione a medio termine.

Secondo le classificazioni dei problemi di pianificazione di progetto proposte in [Herroelen et al., 1999] e [Brucker et al., 1999], il problema affrontato può essere classificato come vincolato dalle risorse, con risorse rinnovabili (quali la manodopera) disponibili periodo per periodo. Analogamente al modello PERT/CPM di base, vengono considerate precedenze di tipo finish-to-start con sfasamento temporale nullo e non è consentita alcuna prelazione sulle attività. La funzione obiettivo adottata differisce tuttavia dall'approccio tradizionale: anziché minimizzare la du-

rata complessiva del progetto, si persegue una misura di completamento incrementale che mira a massimizzare il valore aziendale percepito dagli utenti, riflettendo una prospettiva orientata al valore piuttosto che alla sola efficienza temporale.

Il problema della pianificazione ha acquisito crescente interesse nell'ambito delle metodologie iterative e incrementali. In [Denne and Cleland-Huang, 2003] viene proposta una strategia di sviluppo software basata su fattori finanziari, applicabile in contesti iterativi, che genera una sequenza ottimale di requisiti massimizzando nel tempo il valore attuale netto, ovvero una combinazione di ricavi, costi e rischi di ciascun requisito. Gli autori presentano due strategie risolutive: un algoritmo greedy che seleziona il successivo requisito da soddisfare considerando quelli senza precursori non soddisfatti e con il massimo valore attuale netto, e un approccio look-ahead che estende la strategia greedy analizzando sottoinsiemi di sequenze di precedenza redditizie.

Un modello specificamente orientato ai metodi agili è descritto in [Szöke, 2011], dove viene presentato un modello concettuale per la pianificazione dei rilasci e sviluppato un modello di ottimizzazione finalizzato all'assegnazione dei requisiti alle diverse iterazioni di un rilascio, con l'obiettivo di massimizzare l'utilità complessiva fornita pur tenendo conto delle precedenze e delle condizioni di accoppiamento. In questo contesto, gli autori illustrano un algoritmo branch-and-bound per la risoluzione del modello, che incorpora esplicitamente la gestione del rischio. Un ulteriore contributo rilevante nel contesto Agile è stato quello di [van Valkenhoef et al., 2011], focalizzato primariamente sulla gestione del rischio e dell'incertezza nei progetti XP. L'approccio proposto stima la velocità di sviluppo del team e considera più set di user story con rilevanza decrescente secondo la classificazione MoSCoW:

- set *must have* (requisiti indispensabili)
- set *should have* (requisiti importanti)
- set *could have* (requisiti desiderabili)

L'obiettivo è assegnare ciascuna user story al set più appropriato, massimizzando l'utilità complessiva e rispettando le precedenze e gli accoppiamenti tra le user story

mediante un algoritmo branch-and-bound per identificare la soluzione ottimale. Tuttavia, il numero limitato di set considerati determina un piano a grana grossa che necessita di successivo raffinamento per ottenere una pianificazione operativa, ad esempio suddividendo i set secondo limiti di budget o frammentando le user story in attività più granulari.

In un precedente lavoro è stato inoltre proposto un approccio per la pianificazione degli sprint nei progetti di data warehouse agile [Golfarelli et al., 2012]. Il modello presentato in quel contributo si distingue per una struttura più semplice rispetto a quella qui descritta, non includendo storie forzate e adottando una modellazione dell'accoppiamento meno espressiva; inoltre, il problema della ripianificazione non viene affrontato in quella formulazione.

Nei contesti reali, l'ambiente di progetto difficilmente può essere considerato deterministico a causa della varietà di eventi imprevisti che possono verificarsi e dell'imprecisione intrinseca delle stime, rendendo necessarie politiche di gestione del cambiamento. Sebbene nessuno dei lavori precedentemente menzionati si occupi specificamente di gestione del cambiamento, un approccio rilevante in questa direzione è proposto in [Greer and Ruhe, 2004], orientato a contesti iterativi e incrementali. In questo framework, un piano di rilascio include diversi incrementi e in ogni fase un insieme di requisiti viene assegnato agli incrementi attuali e futuri in modo da restituire il miglior compromesso tra le priorità degli stakeholder e i vincoli di sviluppo, quali capacità di incremento, precedenze e condizioni di accoppiamento. Il modello è formalizzato come un problema dello zaino multiplo e risolto mediante un algoritmo genetico. Per gestire il cambiamento è implementa una strategia parziale di ripianificazione: a ogni incremento sono consentiti nuovi requisiti e modifiche alle priorità o ai vincoli, e una nuova soluzione viene generata ex novo.

Nel contesto specifico della pianificazione Scrum, in [Li et al., 2010] viene fornito un modello di ottimizzazione, basato sulla formulazione del problema dello zaino, per la pianificazione a singola iterazione che seleziona i requisiti in modo da massimizzando il profitto dell'iterazione successiva e rispettando i vincoli di sviluppo. L'evoluzione viene gestita consentendo modifiche ai parametri dopo

ogni iterazione e valutandone l'impatto sul modello, con una nuova soluzione per l'iterazione successiva generata da zero incorporando modifiche e storie aggiuntive. Un approccio più sofisticato è rappresentato dalla pianificazione bi-obiettivo proposta in [Saliu and Ruhe, 2007], in cui l'iterazione successiva viene pianificata considerando l'impatto di nuovi requisiti o modifiche sul sistema esistente dal punto di vista aziendale e di sviluppo. Viene generato un insieme di piani alternativi, ciascuno dei quali riflette una diversa importanza relativa degli aspetti aziendali e di implementazione; il piano ottimale viene quindi selezionato come quello che meglio soddisfa un gruppo di interdipendenze, denominate SD-coupling, tra i requisiti identificati attraverso l'analisi di impatto. Questo approccio permette di bilanciare in modo esplicito le esigenze di business con le complessità tecniche derivanti dall'evoluzione del sistema.

Sebbene questi approcci affrontino specificamente le problematiche relative all'incertezza e alla ripianificazione, nessuno di essi si occupa di evitare che il nuovo piano interferisca con quello precedente, aspetto che costituisce uno dei contributi distintivi di questo lavoro. Per identificare contributi in questa direzione, è necessario esaminare l'area di ricerca sulla pianificazione in condizioni di incertezza, i cui sforzi si sono concentrati su due fronti principali: la pianificazione proattiva e la pianificazione reattiva [Demeulemeester and Herroelen, 2002].

La pianificazione proattiva, o robusta, si concentra sullo sviluppo di una schedule di base che incorpora un certo grado di anticipazione della variabilità durante l'esecuzione del progetto al fine di proteggere la pianificazione stessa da perturbazioni. Ad esempio, in [Herroelen et al., 2002] viene definita una pianificazione iniziale aggressiva a cui vengono successivamente aggiunti buffer di risorse e tempo che proteggono i percorsi critici. Oltre alle regole empiriche, come la regola del dimensionamento del buffer al 50%, metodi più sofisticati per il dimensionamento dei buffer sono discussi in [Newbold, 1998].

La schedulazione reattiva si riferisce invece alle modifiche che potrebbero dover essere apportate alla schedulazione durante l'esecuzione del progetto e può basarsi su diverse strategie. Da un lato, l'approccio reattivo può utilizzare tecniche semplici volte a ripristinare rapidamente la coerenza della schedulazione: ad esempio, la regola dello spostamento a destra proposta in [Sadeh et al., 1993] pos-

ticipa tutte le attività interessate dall'interruzione della schedulazione. D'altro canto, un approccio di schedulazione reattiva può comportare una rischedulazione completa delle attività rimanenti. In caso di rischedulazione, la nuova schedulazione può differire notevolmente da quella di base, circostanza non auspicabile poiché annullerebbe gli impegni precedentemente stabiliti generando costi aggiuntivi, tensioni e insoddisfazione sia da parte dei clienti che dei membri del team. Per questo motivo, gli approcci di rischedulazione naïve si basano spesso su euristiche che effettuano riorganizzazioni locali dei piani. In alternativa, la rischedulazione può adottare una strategia di minima perturbazione che mira alla stabilità ex post, basandosi su algoritmi esatti o subottimali il cui obiettivo è la minimizzazione di una funzione delle differenze tra gli istanti di inizio di ciascuna attività nella schedulazione nuova e in quella originale [Sakkout and Wallace, 2000], oppure la minimizzazione del numero di attività da svolgere in sprint diversi [Alagöz and Azizoğlu, 2003].

Chapter 4

Formulazione matematica

In questa sezione definiamo matematicamente il problema di pianificazione agile. Consideriamo l'insieme $U = \{1, \ldots, n\}$ degli indici delle n user story da realizzare durante il progetto. Ogni user story $j \in U$ è caratterizzata dai seguenti attributi principali:

- u_j: l'utilità, che rappresenta il valore apportato dalla user story;
- r_j^{cr}: il rischio critico, che misura l'impatto potenziale della story sulle altre;
- r_j^{unc}: l'incertezza, che quantifica la difficoltà di stima dovuta a possibili imprevisti;
- p_j: la complessità, espressa in story point, che indica lo sforzo necessario per completarla.

Definiamo inoltre $Y_j \subseteq U$ come l'insieme delle user story affini alla story j, ovvero quelle che, se assegnate allo stesso sprint, generano un valore aggiunto. Per ognuna di esse, a_j rappresenta l'incremento di utilità ottenuto per ogni user story affine assegnata allo stesso sprint. Nel caso in cui $Y_j = \emptyset$, poniamo $a_j = 0$.

Per gestire le dipendenze tra user story, distinguiamo due sottoinsiemi U^{OR} e U^{AND}, rispettivamente contenenti le user story con dipendenze di tipo OR e AND.

- Per ogni $j \in U^{OR}$, indichiamo con $D_j^{OR} \subseteq U$ l'insieme delle user story dalle quali j dipende secondo una logica OR: almeno una delle storie in D_j^{OR} deve

essere assegnata allo stesso sprint di j o a uno sprint precedente affinché j possa essere eseguita.

- Per ogni $j \in U^{AND}$, definiamo $D_j^{AND} \subseteq U$ come l'insieme delle user story da cui j dipende in modalità AND: tutte le storie in D_j^{AND} devono essere assegnate allo stesso sprint di j o a sprint precedenti.

Consideriamo infine l'insieme $S = \{1, \ldots, m\}$ degli m sprint pianificati per il progetto, dove ogni sprint $i \in S$ ha una capacità massima p_i^{max}, espressa in story point, che rappresenta il limite di lavoro che il team può completare durante quello sprint.

Le variabili decisionali che modellano il problema sono:

- $x_{ij} \in \{0, 1\}$, variabile binaria che vale 1 se la user story j viene assegnata allo sprint i, 0 altrimenti;

- $y_{ij} \in \mathbb{Z}_{\geq 0}$, variabile intera non negativa che indica il numero di user story affini a j (cioè appartenenti a Y_j) assegnate allo stesso sprint i.

Il modello di programmazione lineare intera mista è:

$$z_P = \max \sum_{k=1}^{m} \sum_{i=1}^{k} \sum_{j=1}^{n} u_j \left(r_j^{cr} x_{ij} + a_j y_{ij} \right) \tag{4.1}$$

$$s.t. \sum_{j=1}^{n} p_j r_j^{un} x_{ij} \leq p_i^{max}, \qquad i \in S \tag{4.2}$$

$$\sum_{i=1}^{m} x_{ij} = 1, \qquad j \in U \tag{4.3}$$

$$\sum_{k=1}^{i} \sum_{z \in D_j^{OR}} x_{kz} \geq x_{ij}, \qquad i \in S, j \in U^{OR} \tag{4.4}$$

$$\sum_{k=1}^{i} \sum_{z \in D_j^{AND}} x_{kz} \geq x_{ij} |D_j^{AND}|, \quad i \in S, j \in U^{AND} \tag{4.5}$$

$$y_{ij} \leq \sum_{k \in Y_j} x_{ik}, \qquad i \in S, j \in U \tag{4.6}$$

$$y_{ij} \leq |Y_j| x_{ij}, \qquad i \in S, j \in U \tag{4.7}$$

$$x_{ij} \in \{0, 1\}, \qquad i \in S, j \in U \tag{4.8}$$

$$y_{ij} \geq 0, \qquad i \in S, j \in U \tag{4.9}$$

La funzione obiettivo (4.1) ha lo scopo di massimizzare l'utilità complessiva del progetto agile. L'utilità base u_j di ciascuna user story j viene incrementata considerando due fattori principali:

1. il **rischio di criticità** r_j^{cr}, che assegna un peso maggiore alle storie critiche e ne incentiva l'esecuzione anticipata negli sprint iniziali, riducendo il rischio di impatti tardivi sul progetto;

2. l'**incremento di affinità** a_j, che aumenta il valore percepito dall'utente quando storie correlate vengono sviluppate congiuntamente nello stesso sprint, migliorando la coerenza funzionale del rilascio.

Per ogni user story j, il numero di storie affini che vengono incluse nello sprint

i è quantificato dalla variabile y_{ij}, calcolata come:

$$y_{ij} = \sum_{k \in Y_j} x_{ik},$$

dove $y_{ij} = 0$ quando $x_{ij} = 0$, ovvero quando la story j non è assegnata allo sprint i. Trattandosi di un problema di massimizzazione, i vincoli (4.6) e (4.7) garantiscono il corretto calcolo delle variabili y_{ij} senza la necessità di imporre vincoli espliciti di integralità su di esse.

I **vincoli di capacità** (4.2) assicurano che la complessità totale delle user story assegnate a ciascuno sprint, misurata in story point, non ecceda la capacità massima disponibile $p_i^{\max}$ per quello sprint, rispettando così i limiti operativi del team.

I **vincoli di assegnazione** (4.3) impongono che ogni user story sia assegnata esattamente a uno e un solo sprint, garantendo che nessuna storia venga trascurata o duplicata nella pianificazione.

Le **relazioni di dipendenza** tra user story sono modellate attraverso i vincoli (4.4) e (4.5), che regolano l'ordine di esecuzione delle storie in base ai loro prerequisiti:

- per le dipendenze di tipo **OR**, il vincolo (4.4) permette l'assegnazione della story j allo sprint i solamente se *almeno una* delle user story appartenenti all'insieme D_j^{OR} è stata completata in uno sprint precedente o nello stesso sprint;

- per le dipendenze di tipo **AND**, il vincolo (4.5) richiede invece che *tutte* le user story contenute nell'insieme D_j^{AND} siano state completate entro lo sprint i, assicurando che tutti i prerequisiti necessari siano soddisfatti entro la sprint in cui sarà eseguita la user story j.

4.1 Funzioni obiettivo

Per valutare approcci alternativi alla funzione obiettivo originale (4.1), sono state introdotte cinque nuove funzioni obiettivo, progettate per catturare diversi aspetti

critici della pianificazione degli sprint. La motivazione principale di questa estensione risiede nella natura multi-obiettivo intrinseca del problema di sprint planning: oltre alla massimizzazione del valore di business, i team di sviluppo devono simultaneamente gestire l'incertezza nelle stime, rispettare le dipendenze tecniche, e ottimizzare l'utilizzo della capacità disponibile. Le funzioni proposte permettono di esplorare sistematicamente come diverse priorità strategiche influenzino la qualità e la fattibilità delle soluzioni ottenute.

Le nuove funzioni obiettivo sono definite come segue:

$$z_P^{\text{cr+mod}} = \max \sum_{k=1}^{m} \sum_{i=1}^{k} \sum_{j=1}^{n} u_j \left(r_j^{cr} x_{ij} + a_j y_{ij} \right) dp_{ij} + ub_{ij} \tag{4.10}$$

$$z_P^{\text{unc}} = \max \sum_{k=1}^{m} \sum_{i=1}^{k} \sum_{j=1}^{n} u_j \left(r_j^{unc} x_{ij} + a_j y_{ij} \right) \tag{4.11}$$

$$z_P^{\text{unc+mod}} = \max \sum_{k=1}^{m} \sum_{i=1}^{k} \sum_{j=1}^{n} u_j \left(r_j^{unc} x_{ij} + a_j y_{ij} \right) dp_{ij} + ub_{ij} \tag{4.12}$$

$$z_P^{\text{load}} = \max \sum_{k=1}^{m} \sum_{i=1}^{k} \sum_{j=1}^{n} p_j \left(r_j^{unc} x_{ij} + a_j y_{ij} \right) \tag{4.13}$$

$$z_P^{\text{load+mod}} = \max \sum_{k=1}^{m} \sum_{i=1}^{k} \sum_{j=1}^{n} p_j \left(r_j^{unc} x_{ij} + a_j y_{ij} \right) dp_{ij} + ub_{ij} \tag{4.14}$$

I nuovi parametri introdotti sono:

- $dp_{ij} \in [0.25, 1.0]$: *development penalty*, coefficiente che quantifica il grado di completamento delle dipendenze della user story j prima dello sprint i. Assume valore 1.0 se tutte le dipendenze sono state completate, e decade progressivamente fino a 0.25 se nessuna dipendenza è stata ancora eseguita, questo coefficente è calcolato come (4.15).

- $ub_{ij} \geq 0$: *urgency bonus*, bonus proporzionale al numero di user story che hanno j come dipendenza e sono pianificate negli sprint successivi. Vale 0 se j non ha dipendenze in uscita, questo coefficente è calcolato come (4.16).

$$dp_{ij} = 0.25 + 0.75 \cdot \frac{\text{completed dependencies of } j \text{ before sprint } i}{\text{total dependencies of } j} \tag{4.15}$$

$$ub_{ij} = \sum_{k=1}^{n} u_k \cdot \mathbb{I}(j \in D_k^{OR} \cup D_k^{AND}) \tag{4.16}$$

4.1.1 Obiettivi delle diverse funzioni

Ciascuna funzione obiettivo persegue una specifica strategia di pianificazione:

Funzione $z_P^{\mathbf{cr+mod}}$ (Criticità con modificatori) (4.10)**.** Rappresenta la formulazione originale del problema, che massimizza il valore di business ponderato per la criticità r_j^{cr} delle user story. I modificatori dp_{ij} e ub_{ij} penalizzano l'assegnazione precoce di story con dipendenze non soddisfatte e premiano le story "critiche" che sbloccano molte altre attività. **Obiettivo**: bilanciare valore di business e rispetto delle dipendenze architetturali.

Funzione $z_P^{\mathbf{unc}}$ (Incertezza senza modificatori) (4.11)**.** Incorpora il parametro r_j^{unc}, che quantifica l'incertezza nelle stime di sviluppo della user story j (tipicamente derivata dalla deviazione standard delle stime del team). Questa funzione privilegia l'assegnazione precoce delle user story più incerte, garantendo un buffer temporale sufficiente per gestire ritardi o scoperte impreviste durante l'implementazione. **Obiettivo**: riduzione del rischio di schedule slippage concentrando le attività ad alto rischio nei primi sprint.

Funzione $z_P^{\mathbf{unc+mod}}$ (Incertezza con modificatori) (4.12)**.** Estende (4.11) integrando i modificatori dp_{ij} e ub_{ij} per combinare la gestione dell'incertezza con il rispetto delle dipendenze. **Obiettivo**: bilanciare il rischio di schedule slippage e il rispetto delle dipendenze.

Funzione $z_P^{\mathbf{load}}$ (Carico di lavoro senza modificatori) (4.13)**.** Utilizza il peso p_j (espresso in story points, giorni-persona, o altra metrica di effort) per massimizzare il carico di lavoro complessivo assegnato agli sprint. Questa funzione mira a saturare la capacità disponibile, riducendo gli sprechi di risorse e accelerando

il completamento del progetto. **Obiettivo**: massimizzazione del throughput e riduzione della durata totale del progetto.

Funzione $z_P^{\mathbf{load+mod}}$ (Carico di lavoro con modificatori) (4.14)**.** Combina l'ottimizzazione del carico con i modificatori per evitare che la saturazione degli sprint violi le dipendenze architetturali. **Obiettivo**: throughput elevato mantenendo la fattibilità tecnica della pianificazione.

4.1.2 Ruolo dei parametri modificatori

La presenza o assenza dei modificatori dp_{ij} e ub_{ij} definisce due varianti per ciascuna strategia:

Varianti senza modificatori ((4.11**), (**4.13**)).** Rappresentano le formulazioni "pure" che ottimizzano esclusivamente il criterio principale (incertezza o carico) senza considerazioni esplicite sulle dipendenze. Queste varianti sono utili come baseline per valutare l'impatto dei modificatori, ma possono generare soluzioni che violano frequentemente i vincoli di precedenza, richiedendo correzioni successive.

Varianti con modificatori ((4.10**), (**4.12**), (**4.14**)).** Integrano due meccanismi di guida della ricerca: **Development penalty (dp_{ij})** e **Urgency bonus (ub_{ij})**. Questi modificatori agiscono come forze di attrazione o repulsione durante l'assegnazione delle user story agli sprint, influenzando la direzione della ricerca verso soluzioni più coerenti con le dipendenze tecniche e le priorità di business.

I risultati computazionali di confronto sono riportati nella Sezione 6.4.

4.2 Procedure di riduzione

Le procedure di riduzione mirano a rafforzare i vincoli di capacità (4.2) modificando le capacità degli sprint o i pesi $pr_j = p_j r_j^{un}$ delle user story. Approcci simili sono utilizzati per problemi di packing presenti in [Boschetti et al., 2002, Boschetti and Mingozzi, 2003, Boschetti and Montaletti, 2010].

4.2.1 Modifica delle capacità degli sprint

Se non esiste alcuna combinazione di user story che saturi esattamente la capacità p_i^{max} dello sprint $i \in S$, è possibile ridurre tale capacità rimuovendo i "story point inutilizzabili"" senza alterare il valore ottimale del problema. La nuova capacità può essere trovata risolvendo il problema di Subset Sum:

$$p_i^{max} = \max \left\{ \sum_{k \in U} pr_k \xi_k : \sum_{k \in U} pr_k \xi_k \leq p_i^{max}, \quad \xi_j \in \{0,1\},\, j \in U \right\}$$

dove $pr_j = p_j r_j^{un}$ è il peso effettivo associato alla user story j. Il problema di Subset Sum può essere efficacemente risolto utilizzando una procedura di programmazione dinamica.

4.2.2 Modifica dei pesi delle user story

Un ulteriore miglioramento consiste nell'aumentare il peso pr_j di una user story $j \in U$, mantenendo la fattibilità dello sprint che la contiene. Per ogni sprint $i \in S$, definiamo il valore p''_{ij}:

$$p''_{ij} = \max \left\{ \sum_{h \in U} pr_h \xi_h : \sum_{h \in U} pr_h \xi_h \leq p_i^{max}, \quad \xi_j = 1, \quad \xi_k \in \{0,1\},\, k \in U \setminus \{j\} \right\}$$

ossia la massima somma di pesi che include necessariamente la storia j senza superare la capacità dello sprint. In tal modo, è possibile aggiornare il peso di j come:

$$pr_j := pr_j + (p_i^{max} - p''_{ij}).$$

Per massimizzare il numero di storie a cui aggiornare il peso, si propone di ordinare le user story in ordine decrescente di peso, ovvero:

$$pr_1 \geq pr_2 \geq \cdots \geq pr_n.$$

Queste tecniche di riduzione sfruttano problemi classici come il Subset Sum, risolvibili con algoritmi di programmazione dinamica, contribuendo a semplificare il modello e a ridurre i tempi di calcolo necessari per la soluzione ottimale.

4.2.3 Formulazione matematica alternativa

Nella formulazione presentata in precedenza le capacità degli sprint sono fisse e rappresentate dai parametri p_i^{max}. Si è tuttavia valutato che invece di fornire una capacità fissa per ogni sprint, potrebbe essere vantaggioso permettere al modello di variare la capacità di ogni sprint, accorpando più sprint insieme, al fine di massimizzare l'utilizzo della capacità e migliorare la fattibilità delle soluzioni. Da questa considerazione nasce una formulazione matematica alternativa, che si distingue dalla formulazione originale dalla sostituzione dei vincoli (4.2) con i vincoli (4.18), in cui p^{sp} rappresenta la capacità fissa di ogni settimana (o altro periodo che costituisce una durata minima standard) e ξ_i è una variabile intera che rappresenta il numero di settimane che verranno accorpate nello sprint i. Questa variazione permette al modello di variare la capacita di ogni sprint, accorpando più settimane insieme, al fine di massimizzare l'utilizzo della capacità e migliorare la fattibilità delle soluzioni.

$$z_P = \max \sum_{k=1}^{m} \sum_{i=1}^{k} \sum_{j=1}^{n} u_j \left(r_j^{cr} x_{ij} + a_j y_{ij} \right) \tag{4.17}$$

$$s.t. \sum_{j=1}^{n} p_j r_j^{un} x_{ij} - p^{sp} \xi_i \leq 0, \qquad i \in S \tag{4.18}$$

$$\sum_{i=1}^{m} x_{ij} = 1, \qquad j \in U \tag{4.19}$$

$$\sum_{k=1}^{i} \sum_{z \in D_j^{OR}} x_{kz} \geq x_{ij}, \qquad i \in S, j \in U^{OR} \tag{4.20}$$

$$\sum_{k=1}^{i} \sum_{z \in D_j^{AND}} x_{kz} \geq x_{ij} |D_j^{AND}|, \quad i \in S, j \in U^{AND} \tag{4.21}$$

$$y_{ij} \leq \sum_{k \in Y_j} x_{ik},\ i \in S, j \in U \tag{4.22}$$

$$y_{ij} \leq |Y_j| x_{ij},\ \ i \in S, j \in U \tag{4.23}$$

$$x_{ij} \in \{0, 1\},\ \ i \in S, j \in U \tag{4.24}$$

$$y_{ij} \geq 0,\qquad i \in S, j \in U \tag{4.25}$$

Chapter 5

Euristiche per la pianificazione degli sprint

5.1 Euristiche greedy e di scambio del lavoro precedente

In [Boschetti et al., 2014] sono state proposte due euristiche greedy per costruire una pianificazione e una procedura di post ottimizzazione basata su scambi locali. In questa sezione ne riprendiamo brevemente il funzionamento, poiché tali euristiche costituiscono la base anche per il lavoro sviluppato in questa tesi.

5.1.1 GreedyHeuristic: costruzione sprint-per-sprint

L'idea di base della *GreedyHeuristic* è la seguente: il piano viene costruito iterativamente, ottimizzando uno sprint per volta in ordine cronologico (prima lo sprint 1, poi lo sprint 2, e così via). A ogni passo, fissato uno sprint $i \in S$ e l'insieme F_i delle user story già assegnate agli sprint precedenti, si risolve il sottoproblema

(SP_i):

$$(SP_i) \quad z_{SP_i} = \max \sum_{j=1}^{n} (m-i+1) u_j \left(r_j^{cr} x_{ij} + a_j y_{ij} \right) \tag{5.1}$$

$$s.t. \sum_{j=1}^{n} p_j r_j^{un} x_{ij} \leq p_i^{max} \tag{5.2}$$

$$\sum_{k=1}^{i} \sum_{z \in D_j^{OR}} x_{kz} \geq x_{ij} - |D_j \cap F_i|, \qquad j \in U^{OR} \tag{5.3}$$

$$\sum_{k=1}^{i} \sum_{z \in D_j^{AND}} x_{kz} \geq x_{ij} |D_j| - |D_j \cap F_i|, \quad j \in U^{AND} \tag{5.4}$$

$$y_{ij} \leq \sum_{k \in Y_j} x_{ik}, \qquad j \in U \tag{5.5}$$

$$y_{ij} \leq |Y_j| x_{ij}, \qquad j \in U \tag{5.6}$$

$$x_{ij} \in \{0,1\}, \qquad j \in U \setminus F_i \tag{5.7}$$

$$x_{ij} = 0, \qquad j \in F_i \tag{5.8}$$

$$y_{ij} \geq 0, \qquad i \in S, j \in U \tag{5.9}$$

dove F_i è l'insieme delle storie già assegnate agli sprint $1, \ldots, i-1$ (con $F_1 = \emptyset$).

Listing 5.1: GreedyHeuristic

```
1
2  Set F_1 = ∅, i = 1
3
4  WHILE (F_i ≠ U and i ≤ m):
5      Compute the optimal solution x* of subproblem SP_i
6      Set F_{i+1} = F_i ∪ {j ∈ U : x*_ij = 1}
7      Set i = i + 1
```

Pseudocodice della GreedyHeuristic. Questa procedura è concettualmente greedy perché, a ogni passo, prende la "miglior scelta locale" per lo sprint corrente

(risolvendo SP_i) e non rivede le decisioni sugli sprint precedenti.

5.1.2 QuickGreedyHeuristic: versione veloce basata su knapsack

L'euristica *GreedyHeuristic*, riassunta in Figura 5.1, è estremamente rapida se eseguita una sola volta, ma il suo costo computazionale può diventare rilevante quando deve essere richiamata ripetutamente. In particolare come mostrato in [Boschetti et al., 2014], per alcune istanze di grandi dimensioni, una singola esecuzione di *GreedyHeuristic* richiede più di un secondo; di conseguenza, ripeterla per migliaia di iterazioni comporta un tempo totale dell'ordine di migliaia di secondi, imputabile unicamente alla componente greedy.

Per ridurre questo onere, proponiamo una variante accelerata dell'euristica greedy, denominata *QuickGreedyHeuristic*. L'idea di base consiste nel rilassare il sottoproblema SP_i rimuovendo i vincoli (5.3) (5.6). Il sottoproblema risultante, indicato con SP'_i, si riconduce a un problema di knapsack che può essere risolto in modo efficiente tramite programmazione dinamica. Tale rilassamento introduce tuttavia due criticità:

- in assenza dei vincoli di collegamento (5.5) (5.6), le variabili $\{y_{ij}\}$ diventano indipendenti dalle decisioni di assegnazione;
- l'eliminazione dei vincoli di dipendenza (5.3) (5.4) può determinare violazioni delle relazioni di precedenza tra le user story.

Per risolvere il primo problema, il knapsack SP'_i viene risolto ignorando le variabili $\{y_{ij}\}$, che sono poi valutate a posteriori a partire dalla soluzione $\{x_{ij}\}$ di SP'_i. In particolare, per ogni sprint i e user story j, poniamo

$$y_{ij} = \begin{cases} \sum_{k \in Y_j} x_{ik} & \text{se } x_{ij} = 1, \\ 0 & \text{altrimenti.} \end{cases}$$

In questo modo, le variabili y_{ij} riflettono il numero di user story "di supporto" selezionate nello sprint i a fronte della scelta di j.

Per quanto riguarda le dipendenze, la soluzione del knapsack rilassato SP'_i può violare i vincoli (5.3) (5.4). Fissato lo sprint corrente i e una user story j che viola almeno un vincolo di dipendenza, *QuickGreedyHeuristic* adotta una delle seguenti quattro strategie di recupero dell'ammissibilità:

i **Esclusione di** j. Si proibisce l'uso di j nello sprint i fissando $x_{ij} = 0$ e si riottimizza il knapsack SP'_i.

ii **Inserimento mirato di una dipendenza**. Si seleziona una user story $j' \in D_j^{OR}$ (oppure $j' \in D_j^{AND}$, a seconda del vincolo violato), non presente nella soluzione corrente di SP'_i, che massimizza il rapporto

$$\frac{u_{j'} r_{j'}^{cr}}{p_{j'} r_{j'}^{un}},$$

si fissa $x_{ij'} = 1$ e si riottimizza SP'_i.

iii **Inserimento completo per vincoli AND**. Solo nel caso di dipendenze di tipo AND, si fissano in soluzione tutte le user story $j' \in D_j^{AND}$ non selezionate (ponendo $x_{ij'} = 1$) e si riottimizza SP'_i; per le dipendenze di tipo OR si applica invece la strategia (ii).

iv **Rinforzo progressivo dei profitti**. Per ogni user story j si introduce un coefficiente κ_j che moltiplica il profitto in ogni knapsack SP'_i. Per ogni $j' \in D_j^{OR} \setminus F_i$ (oppure $j' \in D_j^{AND} \setminus F_i$, a seconda del vincolo violato) con $x_{ij'} = 0$ nella soluzione corrente, si aumenta $\kappa_{j'}$ secondo una delle due regole:

(a) $\kappa_{j'} = \rho\, \kappa_{j'}$,

(b) $\kappa_{j'} = \kappa_{j'}^2$.

Successivamente, si riavvia l'euristica greedy dal primo sprint, ponendo $i = 1$ e $F_1 = \emptyset$.

Listing 5.2: QuickGreedyHeuristic

```
Set z_star = -infinity

FOR each Strategy s = 1, 2, 3, 4:
    Set z_temp = 0
    Set F_1 = empty
    Set i = 1
    Set Iter = 0
    WHILE (F_i != U AND i <= m):
        REPEAT:
            Compute optimal solution x_temp of knapsack
               SP_i_prime

            IF (x_temp violates some dependency):
                Apply strategy s
                IF (s == 4):
                    Set z_temp = 0
                    Set F_1 = empty
                    Set i = 1
                    Set Iter = Iter + 1
        UNTIL (x_temp feasible OR SP_i_prime infeasible OR
           Iter > MaxIter)
        IF (SP_i_prime infeasible OR Iter > MaxIter):
            Set z_temp = -infinity
            Set i = m + 1
        ELSE:
            z_temp = z_temp + value(SP_i)
            F_{i+1} = F_i union { j in U with x_temp[i,j] =
               1 }
            i = i + 1
    IF (z_star < z_temp):
        z_star = z_temp
        x_star = x_temp
END FOR
```

La Figura 5.2 riassume il funzionamento dell'algoritmo *QuickGreedyHeuristic*. Per ciascuna strategia $s = 1, 2, 3, 4$, l'algoritmo costruisce iterativamente una soluzione, aggiornando lo sprint corrente i, l'insieme delle user story già assegnate agli sprint precedenti (F_i) e il valore complessivo z' della soluzione ottenuta. Alla fine, viene mantenuta la migliore soluzione trovata $(\mathbf{x}^*, z^*)$.

In termini di proprietà, la strategia (i) garantisce sempre la convergenza a una soluzione ammissibile, poiché ogni violazione viene gestita tramite esclusione della user story responsabile. Al contrario, le strategie (ii) e (iii), in particolare la terza, possono generare istanze di knapsack non ammissibili (ad esempio quando la capacità di sprint è insufficiente per ospitare tutte le dipendenze forzate). In tali casi, se SP'_i risulta non ammissibili, l'algoritmo interrompe il trattamento dello sprint i e passa all'iterazione successiva della procedura greedy. La strategia (iv) può richiedere un numero elevato di iterazioni prima di raggiungere l'ammissibilità; per questo motivo, viene imposto un numero massimo di iterazioni *MaxIter*, oltre il quale il tentativo con la strategia (iv) viene interrotto.

Nella fase sperimentale, si è fissato $MaxIter = 1000$. La strategia (iv) è stata applicata una volta utilizzando la regola (b), inizializzando i coefficienti a $\kappa_j = 1.025$ per ogni $j \in U$, e due volte utilizzando la regola (a), con $\kappa_j = 1$ per ogni $j \in U$ e valori di ρ pari a 2 e 5. La scelta di κ_j e ρ riflette un compromesso tra tempo di raggiungimento della ammissibilità e qualità della soluzione: se i profitti crescono troppo rapidamente, si ottiene spesso una soluzione ammissibile in poche iterazioni, ma potenzialmente di bassa qualità, poiché le user story "rinforzate" vengono anticipate ai primi sprint senza considerare adeguatamente la loro utilità reale; al contrario, se i profitti aumentano troppo lentamente, il numero di iterazioni necessario per ottenere una soluzione ammissibile può diventare eccessivo.

5.1.3 ExchangeHeuristic: miglioramento locale per scambi

Sia *GreedyHeuristic* sia *QuickGreedyHeuristic* possono terminare senza individuare una soluzione ammissibile, poiché tutti gli sprint risultano esaminati (cioè $i > m$) ma non tutte le user story sono assegnate agli sprint disponibili (ovvero $F_i \neq U$). Inoltre, anche quando *QuickGreedyHeuristic* produce una soluzione ammissibile, tale soluzione può in genere essere ulteriormente migliorata: le strategie di recupero

della ammissibilità rispetto ai vincoli di dipendenza, infatti, possono condurre a configurazioni non localmente ottimali.

In [Boschetti et al., 2014] viene proposta *ExchangeHeuristic*, descritta in Figura 5.3. Data una soluzione $\mathbf{x}'$ da migliorare, la procedura tenta sistematicamente di scambiare user story tra coppie di sprint, ripetendo il processo fino a quando avviene almeno uno scambio migliorativo. In particolare, *ExchangeHeuristic* considera tre tipi di mosse:

1-0: spostamento di una user story $j \in U$ eseguita nello sprint i' verso uno sprint i;

1-1: scambio di una user story $j \in U$ eseguita nello sprint i con una user story $j' \in U$ eseguita nello sprint i';

2-1: scambio di due user story $j, j' \in U$ eseguite nello sprint i con una user story $j'' \in U$ eseguita nello sprint i'.

Uno scambio viene effettivamente applicato solo se è contemporaneamente *fattibile* e *profittevole*. La fattibilità richiede che, dopo lo scambio, i vincoli di capacità degli sprint coinvolti rimangano soddisfatti e che non si generino violazioni delle dipendenze tra user story; la profittabilità richiede invece che il valore complessivo della funzione obiettivo aumenti.

L'algoritmo *ExchangeHeuristic* può essere esteso includendo mosse di scambio più complesse (ad esempio scambi k-ℓ con $k, \ell > 2$). Tuttavia, l'incremento di complessità computazionale associato a tali estensioni tende a non essere compensato da miglioramenti significativi della qualità della soluzione, motivo per cui in questo lavoro ci si limita alle mosse 1-0, 1-1 e 2-1 descritte sopra.

Listing 5.3: ExchangeHeuristic

```
Repeat
    // Apply 1-0 exchanges
    For sprint i = 1..m-1:
        For sprint i' = i+1..m:
            For each story j with x'[i',j] = 1:
                If moving j from i' to i is feasible and
                   profitable:
                    x'[i',j] = 0
                    x'[i,j]   = 1
    // Apply 1-1 exchanges
    For sprint i = 1..m-1:
        For sprint i' = i+1..m:
            For each pair j, j' with x'[i,j]=1 and x'[i',j
               ']=1:
                If exchanging j and j' is feasible and
                   profitable:
                    x'[i,j]   = 0
                    x'[i',j'] = 0
                    x'[i,j']  = 1
                    x'[i',j]  = 1
    // Apply 2-1 exchanges
    For sprint i = 1..m:
        For sprint i' = 1..m:
            For each triple j, j', j'' with x'[i,j]=x'[i,j
               ']=x'[i',j'']=1:
                If exchanging j with {j',j''} is feasible
                   and profitable:
                    x'[i,j]   = 0
                    x'[i,j']  = 0
                    x'[i',j'']= 0
                    x'[i',j]   = 1
                    x'[i',j']  = 1
                    x'[i,j'']  = 1
Until no exchanges occur.
```

5.2 A Lagrangian Heuristic

La letteratura propone numerose euristiche basate su metodi di decomposizione e rilassamento lagrangiano. In [Boschetti et al., 2014] si descrive una procedura euristica che sfrutta il rilassamento lagrangiano del modello, combinato con un algoritmo del subgradiente e con le euristiche greedy introdotte in precedenza.

Il rilassamento lagrangiano si ottiene dualizzando i vincoli (4.3)–(4.7) mediante i vettori di penalità $\{\lambda_j\}$, $\{\lambda_{ij}^{OR}\}$, $\{\lambda_{ij}^{AND}\}$, $\{\lambda_{ij}^{Y1}\}$ e $\{\lambda_{ij}^{Y2}\}$. Le penalità λ_j, $j \in U$, sono non vincolate, mentre le restanti sono non positive, così da interpretare le violazioni dei vincoli corrispondenti come costi aggiuntivi nel problema rilassato. Il problema lagrangiano risultante è:

$$(LR) \qquad z_{LR}(\boldsymbol{\lambda}) = \max \sum_{k=1}^{m}\sum_{i=1}^{k}\sum_{j=1}^{n}\big(u'_{ij}(\boldsymbol{\lambda})x_{ij} + u''_{ij}(\boldsymbol{\lambda})y_{ij}\big) + \sum_{j=1}^{n}\lambda_j \tag{5.10}$$

$$\text{s.t.} \ \sum_{j=1}^{n} p_j r_j^{un} x_{ij} \le p_i^{\max}, \qquad i \in S \tag{5.11}$$

$$x_{ij} \in \{0,1\}, \qquad i \in S, j \in U \tag{5.12}$$

$$0 \le y_{ij} \le |Y_j|, \qquad i \in S, j \in U \tag{5.13}$$

dove le *utilità penalizzate* $u'_{ij}(\boldsymbol{\lambda})$ e $u''_{ij}(\boldsymbol{\lambda})$ sono definite da

$$\begin{aligned} u'_{ij}(\boldsymbol{\lambda}) &= u_j r_j^{cr} - \lambda_j + \left(\lambda_{ij}^{OR} - \sum_{k=i}^{m}\sum_{j' \in \bar{D}_j^{OR}} \lambda_{kj'}^{OR}\right) + \\ &\quad + \left(|D_j^{AND}|\lambda_{ij}^{AND} - \sum_{k=i}^{m}\sum_{j' \in \bar{D}_j^{AND}} \lambda_{kj'}^{AND}\right) - \lambda_{ij}^{Y1} - |Y_j|\lambda_{ij}^{Y2} \\ u''_{ij}(\boldsymbol{\lambda}) &= u_j a_j + \lambda_{ij}^{Y1} + \lambda_{ij}^{Y2}, \end{aligned} \tag{5.14}$$

con $\bar{D}_j^{OR} = \{j' \in U : j \in D_{j'}^{OR}\}$ e $\bar{D}_j^{AND} = \{j' \in U : j \in D_{j'}^{AND}\}$.

Grazie alla struttura dei vincoli di capacità, il problema LR si decompone in

$2m$ sottoproblemi indipendenti, due per ogni sprint $i \in S$:

$$(LR_i^1) \qquad z_{LR_i^1}(\boldsymbol{\lambda}) = \max \sum_{j=1}^{n} u'_{ij}(\boldsymbol{\lambda}) x_{ij} \tag{5.15}$$

$$\text{s.t.} \sum_{j=1}^{n} p_j r_j^{un} x_{ij} \leq p_i^{\max} \tag{5.16}$$

$$x_{ij} \in \{0,1\}, \qquad j \in U \tag{5.17}$$

e

$$(LR_i^2) \qquad z_{LR_i^2}(\boldsymbol{\lambda}) = \max \sum_{j=1}^{n} u''_{ij}(\boldsymbol{\lambda}) y_{ij} \tag{5.18}$$

$$\text{s.t. } 0 \leq y_{ij} \leq |Y_j|,\ j \in U \tag{5.19}$$

Il sottoproblema LR_i^1 è un problema di knapsack, mentre LR_i^2 è risolvibile per ispezione, assegnando $y_{ij} = |Y_j|$ se $u''_{ij}(\boldsymbol{\lambda}) > 0$ e $y_{ij} = 0$ altrimenti. Il valore ottimo del rilassamento lagrangiano è quindi

$$z_{LR}(\boldsymbol{\lambda}) = \sum_{i=1}^{m} (m - i + 1)\big(z_{LR_i^1}(\boldsymbol{\lambda}) + z_{LR_i^2}(\boldsymbol{\lambda})\big) + \sum_{j=1}^{n} \lambda_j, \tag{5.20}$$

che fornisce un upper bound valido per il problema originale P. La ricerca del vettore di penalità $\boldsymbol{\lambda}^*$ che minimizza $z_{LR}(\boldsymbol{\lambda})$ porta al *Lagrangiano Duale* $z_{LR}(\boldsymbol{\lambda}^*) = \min_{\boldsymbol{\lambda}}\{z_{LR}(\boldsymbol{\lambda})\}$, affrontato qui tramite un algoritmo del subgradiente.

Sia $(\boldsymbol{x}, \boldsymbol{y})$ la soluzione, di valore $z_{LR}(\boldsymbol{\lambda})$, ottenuta risolvendo LR a una iterazione del subgradiente. I moltiplicatori lagrangiani vengono aggiornati come segue:

$$\begin{aligned}
\lambda_j &= \lambda_j + \alpha g_j, && j \in U \\
\lambda_{ij}^{OR} &= \max\{0, \lambda_{ij}^{OR} + \alpha g_{ij}^{OR}\}, && i \in S, j \in U^{OR} \\
\lambda_{ij}^{AND} &= \max\{0, \lambda_{ij}^{AND} + \alpha g_{ij}^{AND}\}, && i \in S, j \in U^{AND} \\
\lambda_{ij}^{Y1} &= \max\{0, \lambda_{ij}^{Y1} + \alpha g_{ij}^{Y1}\}, && i \in S, j \in U \\
\lambda_{ij}^{Y2} &= \max\{0, \lambda_{ij}^{Y2} + \alpha g_{ij}^{Y2}\}, && i \in S, j \in U
\end{aligned} \tag{5.21}$$

dove α è la lunghezza del passo lungo la direzione di ricerca data dal subgradiente

$\boldsymbol{g}$, le cui componenti sono

$$\begin{aligned}
g_j &= \sum_{i=1}^{m} x_{ij} - 1, && j \in U \\
g_{ij}^{OR} &= \sum_{k=1}^{i} \sum_{z \in D_j^{OR}} x_{kz} - x_{ij}, && i \in S, j \in U^{OR} \\
g_{ij}^{AND} &= \sum_{k=1}^{i} \sum_{z \in D_j^{AND}} x_{kz} - x_{ij}|D_j^{AND}|, && i \in S, j \in U^{AND} \\
g_{ij}^{Y1} &= \sum_{k \in Y_j} x_{ik} - y_{ij}, && i \in S, j \in U \\
g_{ij}^{Y2} &= |Y_j| x_{ij} - y_{ij}, && i \in S, j \in U
\end{aligned} \tag{5.22}$$

La regola per il passo adottata negli esperimenti è

$$\alpha = \beta \frac{0.1\, z_{LR}(\boldsymbol{\lambda})}{\|\boldsymbol{g}\|_2^2},$$

dove β è inizializzato a un valore dipendente dal problema (nel nostro caso, $\beta = 3$) e viene ridotto moltiplicandolo per 0.85 se, dopo un numero prefissato di iterazioni (10), il valore $z_{LR}(\boldsymbol{\lambda})$ non migliora. Il numero massimo di iterazioni è fissato a 5000 e il metodo viene interrotto anticipatamente se, in un intervallo di 50 iterazioni, il valore di $z_{LR}(\boldsymbol{\lambda})$ non si riduce di almeno lo 0.01%.

La procedura complessiva, denominata *LagrangianHeuristic*, è riassunta in Figura 5.4. A ogni iterazione del subgradiente si risolve LR per un dato vettore di penalità $\boldsymbol{\lambda}$, ottenendo il valore $z_{LR}(\boldsymbol{\lambda})$ e aggiornando il miglior upper bound z_{LR}^*. Successivamente, si calcola una soluzione euristica $\boldsymbol{x}'$ tramite *QuickGreedyHeuristic*, utilizzando le utilità penalizzate definite in (5.14), e la si migliora mediante *ExchangeHeuristic*; il valore z' della soluzione così ottenuta sostituisce il miglior valore corrente z^* se $z' > z^*$. Inoltre, se la condizione $\gamma z^* \leq z'$ è soddisfatta, si genera una seconda soluzione euristica $\boldsymbol{x}''$ con la più costosa *GreedyHeuristic*, sempre basata sulle utilità penalizzate, e il corrispondente valore z'' viene confrontato con z^* per un eventuale aggiornamento.

Listing 5.4: Algorithm LagrangianHeuristic

```
Set lambda = 0
Set z_best = -INF
Set z_LR_best = +INF

REPEAT
    Compute z_LR(lambda) by solving the LagrangianProblemLR
    IF z_LR(lambda) < z_LR_best THEN
        z_LR_best = z_LR(lambda)
    ENDIF
    Compute a heuristic solution x_prime with
       QuickGreedyHeuristic
        using penalized utilities
    Improve x_prime with ExchangeHeuristic
    Let z_prime = value of x_prime
    IF gamma * z_best <= z_prime THEN
        Compute a heuristic solution x_second with
           GreedyHeuristic
            using penalized utilities
        Let z_second = value of x_second

        IF z_best < z_second THEN
            z_best = z_second
            x_best = x_second
        ENDIF
    ENDIF
    IF z_best < z_prime THEN
        z_best = z_prime
        x_best = x_prime
    ENDIF
    Update penalties lambda

UNTIL subgradient stopping conditions are satisfied
```

Il parametro γ controlla la frequenza di invocazione di *GreedyHeuristic*: ponendo $\gamma = 1$ si esegue questa procedura solo quando $\boldsymbol{x}'$ rappresenta la migliore soluzione trovata finora, mentre per $\gamma < 1$ *GreedyHeuristic* viene attivata anche quando z' è "sufficientemente vicino" a z^*, cioè entro una distanza percentuale $100(1-\gamma)$. Al termine di *LagrangianHeuristic* si dispone di una soluzione ammissibile di valore z^* e di un upper bound z^*_{LR} derivante dal rilassamento lagrangiano, che consente di stimare la distanza massima della soluzione euristica dall'ottimo del problema originale.

5.3 Aggiornamenti ai metodi di cutting plane

I metodi *cutting plane* costituiscono una tecnica fondamentale nella programmazione lineare intera mista (MILP), utilizzata per rafforzare iterativamente il rilassamento lineare continuo mediante l'aggiunta di disuguaglianze valide (*tagli*) che eliminano soluzioni frazionarie non ammissibili per il problema discreto. Nel contesto del problema di assegnazione sprint-user story, i tagli di precedenza OR/AND garantiscono che, per eseguire una user story j nello sprint i, almeno una (OR) oppure tutte (AND) le sue dipendenze siano state completate negli sprint precedenti o quello corrente ($i_1 \leq i$).

Durante la fase di sperimentazione ci si è resi conto di alcuni possibili limiti presenti nella componente di branch-and-cut dell'euristica matheuristic proposta da [Boschetti et al., 2014]. In particolare, l'implementazione originale della funzione `CuttingPlaneHeu` presentava: assenza di controllo sulla profondità dell'albero di branch-and-bound e mancanza di meccanismi di deduplicazione.

La nuova implementazione `CuttingPlaneHeu_new` introduce i seguenti potenziamenti:

- **Limitazione alla profondità dell'albero**: i tagli vengono generati esclusivamente per nodi con depth ≤ 3, concentrando lo sforzo computazionale dove l'impatto sul bound è massimo, cercando quindi di limitare l'uso di risorse computazionali per l'esecuzione di tagli di poco conto. La adozione di depth ≤ 3 è dovuta a un'analisi sperimentale che ha mostrato come la maggior parte dei miglioramenti significativi del bound si ottenga nei

primi livelli dell'albero di branch-and-bound e che la profondita media fosse Avg_depth = 2.7.

- **Generazione multi-cut**: ogni callback può richiamare la funzione che aggiunge i tagli fino a 10 volte, accelerando la scoperta dei tagli da inserire.

- **Deduplicazione mediante hashing FNV-1a**: un checksum a 64 bit sulle colonne non nulle previene la presenza di tagli duplicati.

- **Tolleranza di violazione ridotta**: $\epsilon = 10^{-6}$ per una maggiore selettività.

Listing 5.5: Algoritmo multi-cut per la callback di generazione tagli

```
myHeucutcallbackNew(env, cbdata, wherefrom, cbhandle):
    if FlagLP or depth > 3:
        return

    Get node solution x[0..numcols-1]

    numCutsAdded = 0
    FOR iter = 1 TO 10:
        if NOT OR_AND_InequalitiesNew(cutinfo):
            break

        Add cut with CPX_USECUT_FILTER
        numCutsAdded++

    if numCutsAdded > 0:
        useraction_p = CPX_CALLBACK_SET
```

La funzione `OR_AND_InequalitiesNew` implementa correttamente la generazione dei tagli di precedenza considerando la dimensione temporale.

Listing 5.6: Generazione corretta dei tagli di precedenza OR-AND

```
OR_AND_InequalitiesNew(cutinfo):
    FOR j IN user_stories:
        FOR i IN sprints:
            if x[i*n+j] < 1e-9:
                continue

            // Precedenza OR: almeno UNA dipendenza
               completata
            lhs = sum_{i1<=i, u in UOR[j]} x[i1*n+u]
            if lhs - x[i*n+j] < -1e-6:
                Build cut: sum_{i1<=i, u in UOR[j]} x[i1*n+u
                   ] >= x[i*n+j]
                if checksum != last_cut_checksum:
                    return VIOLATED

            // Precedenza AND: TUTTE le dipendenze
               completate
            lhs = sum_{i1<=i, u in UAND[j]} x[i1*n+u]
            if lhs - |UAND[j]|*x[i*n+j] < -1e-6:
                Build cut: sum_{i1<=i, u in UAND[j]} x[i1*n+
                   u]
                                >= |UAND[j]|*x[i*n+j]
                if checksum != last_cut_checksum:
                    return VIOLATED

    return NO_VIOLATION
```

Chapter 6

Risultati ottenuti

Tutti gli algoritmi descritti nei capitoli precedenti sono stati implementati in C++ all'interno di Microsoft Visual Studio 2024. L'esecuzione degli esperimenti è stata condotta su un computer con processore Intel core i5-1135G7 a 2.4 GHz e 16 GB di RAM, con sistema operativo Windows 11. Nei test computazionali è stato fissato un limite di tempo di 300 secondi per tutti i risultati riportati di seguito, per cui quando IBM Ilog Cplex non trova una soluzione ammissibile per un'istanza entro il limite di tempo assegnato, riportiamo il carattere "–" nelle colonne z.

6.1 Dati utilizzati

Per i test computazionali sono stati utilizzati dati sintetici. Le istanze devono presentare determinate caratteristiche per poter essere rappresentative di scenari reali e mettere alla prova gli algoritmi in modo efficace. Devono presentare una varietà di dimensioni (numero di user story e sprint), tipi e quantità di dipendenze, presenza o assenza di affinità tra user story, e diversi livelli di complessità e utilità. Inoltre, è importante includere istanze che riflettano le sfide tipiche dei progetti software, come la gestione di vincoli stringenti di capacità degli sprint o la presenza di dipendenze complesse tra le user story.

6.1.1 Generazione di istanze di test

I dataset utilizzati per la valutazione sperimentale degli algoritmi proposti in [Boschetti et al., 2014] presentano alcune limitazioni significative: in particolare, mancano completamente i parametri relativi al rischio (r_j^{cr}) e all'incertezza (r_j^{unc}) delle user story, essendo stati concepiti prima dell'introduzione di queste metriche nella modellazione del problema. Questa carenza rende impossibile una valutazione empirica significativa delle nuove funzioni obiettivo introdotte nella Sezione 4.1, che si basano proprio su questi parametri per guidare le decisioni di pianificazione.

Per ovviare a tale limitazione, è stato sviluppato un generatore sintetico di istanze che permette il controllo completo sui seguenti parametri configurabili:

- numero di user story (n) e numero di sprint (m);
- range di utilità delle user story: $u_j \in [u_{\min}, u_{\max}]$;
- range di rischio (criticità): $r_j^{cr} \in [r_{\min}^{cr}, r_{\max}^{cr}]$;
- range di incertezza: $r_j^{unc} \in [r_{\min}^{unc}, r_{\max}^{unc}]$;
- range di story point (peso): $p_j \in [p_{\min}, p_{\max}]$;
- range di capacità degli sprint: $c_i \in [c_{\min}, c_{\max}]$;
- probabilità di dipendenza (ρ_{dep}): probabilità che una user story presenti vincoli di precedenza;
- numero massimo di dipendenze per user story ($d_{\max}$).

Algoritmo di generazione. Il generatore opera in due fasi sequenziali. Nella prima fase, vengono campionati casualmente da distribuzioni uniformi i parametri numerici di ciascuna user story (u_j, r_j^{cr}, r_j^{unc}, p_j) e la capacità di ogni sprint (c_i). Nella seconda fase, viene costruito il grafo delle dipendenze: per ogni user story j, con probabilità ρ_{dep} vengono estratti casualmente fino a $d_{\max}$ predecessori dall'insieme delle user story già generate, verificando l'assenza di cicli mediante visita topologica. Questo garantisce che il grafo delle dipendenza risultante sia un DAG (Directed Acyclic Graph), condizione necessaria per l'ammissibilità del problema.

6.1.2 Benchmark suite generato

Utilizzando il generatore descritto, sono state create 25 istanze sintetiche organizzate in 5 gruppi tematici, ciascuno composto da 5 progetti con caratteristiche dimensionali e strutturali omogenee:

Gruppo 1 (istanze small). Progetti con 25 user story distribuite su 9 sprint, con al massimo 3 dipendenze per user story. Queste istanze rappresentano team Scrum di piccole dimensioni che lavorano su progetti di breve durata (circa 2-3 mesi).

Gruppo 2 (istanze medium). Progetti con 60 user story distribuite su 30 sprint, con al massimo 3 dipendenze per user story. Simulano progetti di media complessità con cicli di sviluppo di 6-12 mesi.

Gruppo 3 (istanze large). Progetti con 120 user story distribuite su 30 sprint, con al massimo 2 dipendenze per user story ma densità di story per sprint più elevata. Rappresentano progetti complessi con backlog ampio ma orizzonte temporale contenuto.

Gruppo 4 (istanze extra-large). Progetti con 300 user story distribuite su 110 sprint, con al massimo 2 dipendenze per user story. Queste istanze modellano progetti enterprise di lunga durata (oltre 2 anni) con grafi di dipendenza molto ampi.

Gruppo 5 (istanze Fibonacci). Progetti con 60 user story distribuite su 25 sprint, con al massimo 3 dipendenze per user story. La caratteristica distintiva di questo gruppo è l'uso della sequenza di Fibonacci (1, 2, 3, 5, 8, 13, 21) per gli story point, in conformità alle linee guida ufficiali dello Scrum framework, che raccomandano l'uso di scale non lineari per riflettere l'incertezza crescente nelle stime di effort elevato.

Distribuzione degli story point. Nei gruppi 1–4, gli story point sono espressi come giorni-uomo e seguono una distribuzione uniforme nell'intervallo $[p_{\min}, p_{\max}]$,

modellando una situazione realistica in cui le user story hanno complessità variabile. Nel gruppo 5, gli story point sono campionati uniformemente dall'insieme discreto $\{1, 2, 3, 5, 8, 13, 21\}$, rispecchiando la pratica industriale di Planning Poker e della stima relativa. Questa distinzione permette di valutare la robustezza degli algoritmi rispetto a diverse convenzioni di stima adottate dai team di sviluppo.

6.1.3 Caratteristiche strutturali delle istanze

La Tabella 6.1 riporta le principali metriche strutturali delle 25 istanze generate, dove:

- n: numero totale di user story;
- m: numero di sprint pianificati;
- n_{aff}: numero di user story coinvolte in almeno un vincolo di affinità;
- $|U^{OR}|$: numero totale di vincoli di precedenza OR nel grafo;
- $|U^{AND}|$: numero totale di vincoli di precedenza AND nel grafo;
- $l_{\max}$: lunghezza massima di una catena di dipendenze (profondità del DAG);
- $d_{\max}$: grado in-degree massimo nel grafo delle dipendenze.

Table 6.1: Instances generated with new data generator

Group	**Proj. Name**	n	m	n_{aff}	$\lvert U^{OR}\rvert$	$\lvert U^{AND}\rvert$	l_{max}	d_{max}
			Gruppo 1					
instance_09_01		25	9	11	2	3	3	3
instance_09_02		25	9	11	3	5	3	4
instance_09_03		25	9	12	2	6	3	5
instance_09_04		25	9	13	3	3	3	3
instance_09_05		25	9	13	3	5	3	4
			Gruppo 2					
instance_30_01		60	30	14	9	9	3	6
instance_30_02		60	30	14	6	10	3	6
instance_30_03		60	30	14	7	8	3	4
instance_30_04		60	30	11	5	7	3	5
instance_30_05		60	30	17	5	7	3	3
			Gruppo 3					
instance_30b_01		120	30	14	18	12	2	3
instance_30b_02		120	30	17	12	19	2	5
instance_30b_03		120	30	16	13	22	2	3
instance_30b_04		120	30	14	16	25	2	5
instance_30b_05		120	30	12	21	18	2	4
			Gruppo 4					
instance_110_01		300	110	16	41	42	2	6
instance_110_02		300	110	17	48	52	2	4
instance_110_03		300	110	16	50	46	2	5
instance_110_04		300	110	17	42	58	2	5
instance_110_05		300	110	12	40	52	2	5
			Gruppo 5					
instance_f_01		60	25	14	10	12	3	5
instance_f_02		60	25	16	9	12	3	6
instance_f_03		60	25	12	5	11	3	3
instance_f_04		60	25	14	10	4	3	4
instance_f_05		60	25	14	15	6	3	4

Le istanze mostrano una varietà significativa sia in termini dimensionali (da 25 a 300 user story) sia in termini di complessità del grafo delle dipendenze (profondità massima delle catene compresa tra 2 e 3, grado massimo in-degree tra 3 e 6). Questa eterogeneità permette di testare la scalabilità e la robustezza degli algoritmi proposti su un ampio spettro di scenari operativi realistici.

6.2 KPI di valutazione

Per valutare in modo sistematico le prestazioni degli algoritmi e l'efficacia delle funzioni obiettivo proposte, sono stati adottati due insiemi complementari di indicatori: i KPI computazionali, che misurano l'efficienza dell'ottimizzazione, e i KPI di qualità della pianificazione, che valutano l'idoneità pratica delle soluzioni nel contesto Agile.

6.2.1 KPI computazionali

I seguenti indicatori, già utilizzati in [Boschetti et al., 2014], permettono di confrontare le prestazioni algoritmiche e la convergenza verso l'ottimalità:

$Z_{\mathbf{OPT}}$ Valore della funzione obiettivo nella soluzione ottima certificata da IBM ILOG CPLEX. Costituisce il benchmark di riferimento per valutare la qualità delle soluzioni euristiche.

$Z_{\mathbf{HEU}}$ Valore della funzione obiettivo nella migliore soluzione euristica individuata dall'algoritmo matheuristic entro il limite temporale imposto.

Time (s) Tempo computazionale totale espresso in secondi, dall'inizializzazione dell'algoritmo fino al termine dell'esecuzione (che sia per raggiungimento dell'ottimo certificato sia per time limit). Riflette la scalabilità pratica dell'approccio.

6.2.2 KPI di qualità della pianificazione

Per valutare l'idoneità delle soluzioni nel contesto della pianificazione Agile reale, sono stati introdotti nuovi indicatori che misurano aspetti strategici e operativi non catturati dalla sola funzione obiettivo:

Efficienza temporale e utilizzo delle risorse.

N.S.O. *Numero Sprint Occupati*: conta gli sprint effettivamente utilizzati nella soluzione (quelli con almeno una user story assegnata). Valori bassi indicano consegna rapida del progetto, concentrando il lavoro in meno iterazioni e

riducendo l'overhead di gestione. Un numero elevato può segnalare frammentazione eccessiva o sottoutilizzo della capacità.

A.S.U. (%) *Media Utilizzo Sprint*: percentuale media di capacità c_i effettivamente sfruttata negli sprint occupati:

$$\text{A.S.U.} = \frac{100}{|\{i : \exists j, x_{ij} = 1\}|} \sum_{i:\exists j, x_{ij}=1} \frac{\sum_j p_j x_{ij}}{c_i}$$

Valori elevati (prossimi al 100%) indicano saturazione efficiente della capacità disponibile, minimizzando gli sprechi. Valori troppo bassi suggeriscono che la pianificazione lascia risorse inutilizzate.

Distribuzione del rischio e dell'incertezza.

D.R. *Deviazione Standard del Rischio*: quantifica la variabilità del rischio medio aggregato $\bar{r}_i^{cr} = \frac{1}{|\{j:x_{ij}=1\}|} \sum_j r_j^{cr} x_{ij}$ tra gli sprint occupati:

$$\text{D.R.} = \sqrt{\frac{1}{|\{i : \exists j, x_{ij} = 1\}|} \sum_{i:\exists j, x_{ij}=1} (\bar{r}_i^{cr} - \bar{r}^{cr})^2}$$

dove $\bar{r}^{cr}$ è il rischio medio globale. Bassa deviazione indica distribuzione equilibrata del rischio tra gli sprint, evitando concentrazioni pericolose o sprint eccessivamente sbilanciati.

D.U. *Deviazione Standard dell'Incertezza*: analoga a D.R., misura la variabilità dell'incertezza media $\bar{r}_i^{unc}$ tra gli sprint. Bassa variabilità favorisce gestione prevedibile e uniforme delle attività ad alta incertezza, evitando sprint particolarmente volatili.

R.MIN *Rischio Minimo*: identifica il valore minimo di $\bar{r}_i^{cr}$ tra tutti gli sprint occupati. Valori troppo bassi possono segnalare sprint composti esclusivamente da user story "sicure", indicando potenziale sottoutilizzo delle capacità del team di gestire attività complesse.

R.MAX *Rischio Massimo*: identifica il valore massimo di $\bar{r}_i^{cr}$ tra tutti gli sprint occupati. Valori eccessivamente elevati segnalano sprint critici con espo-

sizione al rischio potenzialmente ingestibile, che richiedono attenzione particolare dal team e dagli stakeholder.

U.MIN *Incertezza Minima*: valore minimo di $\bar{r}_i^{unc}$ tra gli sprint occupati. Garantisce che ogni sprint contenga user story con un livello minimo di incertezza, evitando sprint composti solo da attività banali o completamente comprese.

U.MAX *Incertezza Massima*: valore massimo di $\bar{r}_i^{unc}$ tra gli sprint occupati. Identifica l'esposizione all'incertezza nel caso peggiore, essenziale per quantificare la robustezza della pianificazione rispetto a imprevisti e variazioni nelle stime.

Progressione del valore consegnato.

H.U.S. *Half Utility Sprint*: indice dello sprint in cui viene raggiunto almeno il 50% dell'utilità totale cumulata:

$$\text{H.U.S.} = \min \left\{ k : \sum_{i=1}^{k} \sum_{j=1}^{n} u_j x_{ij} \geq \frac{1}{2} \sum_{j=1}^{n} u_j \right\}$$

Questo indicatore misura il *time-to-half-value*, ovvero la rapidità con cui il progetto inizia a consegnare valore significativo agli stakeholder. Valori bassi (sprint iniziali) indicano pianificazioni che prioritizzano il rilascio anticipato di feature ad alto valore, conformemente ai principi Agile di early value delivery.

6.2.3 Interpretazione congiunta dei KPI

I KPI computazionali e i KPI di qualità della pianificazione forniscono prospettive complementari: i primi valutano l'efficienza matematica dell'ottimizzazione, mentre i secondi misurano la traducibilità pratica delle soluzioni in piani di sprint realistici e bilanciati. Una soluzione ideale dovrebbe eccellere in entrambe le dimensioni, presentando gap di ottimalità ridotti (qualità computazionale) e distribuzione equilibrata di rischio, incertezza e valore (qualità gestionale). L'analisi comparativa delle funzioni obiettivo nella Sezione 6.4 utilizza sistematicamente questi indicatori per identificare i trade-off tra le diverse strategie di pianificazione.

6.3 Confronto tra il metodo euristico originale e la versione con modifiche ai tagli

Nella tabella 6.2 sono riportati i confronti tra il metodo euristico originale e la versione aggiornata che incorpora le modifiche descritte nella sezione 5.3. I risulati mostrano che la versione aggiornata dell'euristica riporta la stessa soluzione di quella originale ma con un aumento dei tempi di esecuzione. Le modifiche ai tagli non hanno migliorato i tempi di calcolo come previsto, determinando anzi un peggioramento delle prestazioni computazionali senza benefici sulla qualità della soluzione.

Table 6.2: new heuristic results

	Heuristic		Improved Heuristic	
Istanza	Zheu	Time	Zheu	Time
instance_09_01	22686.6	0.03	22686.6	0.05
instance_09_02	21740.6	0.03	21740.6	0.05
instance_09_03	21159.2	0.03	21159.2	0.06
instance_09_04	15175.4	0.04	15175.4	0.05
instance_09_05	16450.1	0.06	16450.1	0.06
instance_30_01	123352.1	0.10	123352.1	0.25
instance_30_02	135481.2	0.10	135481.2	0.22
instance_30_03	125447.2	0.14	125447.2	0.22
instance_30_04	120783.4	0.20	120783.4	0.26
instance_30_05	119839.7	0.28	119839.7	0.33
instance_30b_01	267873.5	0.10	267873.5	0.31
instance_30b_02	279010.9	0.14	279010.9	0.26
instance_30b_03	268060.4	0.06	268060.4	0.27
instance_30b_04	247839.4	0.15	247839.4	0.46
instance_30b_05	259333.8	0.11	259333.8	0.31
instance_110_01	2050139.1	1.15	2050139.1	1.78
instance_110_02	2082565.4	1.05	2082565.4	1.52
instance_110_03	2086918.6	0.99	2086918.6	1.69
instance_110_04	2148900.6	0.87	2148900.6	1.83
instance_110_05	2015801.2	0.99	2015801.2	1.99
instance_f_01	120411.2	0.04	120411.2	0.11
instance_f_02	127857.5	0.04	127857.5	0.13
instance_f_03	124444.8	0.04	124444.8	0.11
instance_f_04	127565.4	0.03	127565.4	0.11
instance_f_05	121055.0	0.04	121055.0	0.10

6.4 Confronto funzioni obiettivo

In questa sezione vengono confrontati i risultati ottenuti utilizzando le due diverse funzioni obiettivo descritte nella sezione 4.1. Si userà come riferimento la tabella 6.3 la quale riporta i risultati ottenuti con la funzione obiettivo (4.1) e nelle tabelle 6.4, 6.5 che riportano i risultati delle metriche calcolate su tali risultati.

Table 6.3: risultati con funzione obiettivo (4.1)

	CPLEX			Heuristic			Lagrangian Heuristic		
Istanza	Zopt	Gap	Time	Zheu	Time	GHeu	Zheu	Time	GHeu
instance_09_01	23409.9	0.01	0.23	22686.6	0.03	4.66	22686.6	0.23	4.66
instance_09_02	22540.6	0.01	0.42	21740.6	0.02	3.68	21740.6	0.22	3.68
instance_09_03	21779.1	0.01	0.30	21159.2	0.02	2.93	21159.2	0.32	2.93
instance_09_04	18125.7	0.01	0.11	15175.4	0.03	18.04	15175.4	0.33	18.04
instance_09_05	18160.2	0.00	0.38	16450.1	0.03	11.35	16450.1	0.16	11.35
instance_30_01	125477.6	1.02	300.10	123352.1	0.10	2.17	123352.1	3.01	2.17
instance_30_02	143586.5	0.29	300.07	135481.2	0.12	6.28	135481.2	4.00	6.28
instance_30_03	134896.3	0.01	217.66	125447.2	0.14	8.61	125447.2	3.03	8.61
instance_30_04	119857.3	0.01	291.01	120783.4	0.14	0.42	120783.4	4.00	0.42
instance_30_05	134534.8	3.38	300.05	119839.7	0.21	11.19	119839.7	2.46	11.19
instance_30b_01	271716.6	0.05	300.12	267873.5	0.18	1.97	267873.5	18.34	1.97
instance_30b_02	287801.6	0.08	300.12	279010.9	0.21	3.81	279010.9	14.45	3.81
instance_30b_03	271547.9	0.10	300.11	268060.4	0.15	2.54	268060.4	22.44	2.54
instance_30b_04	247552.9	0.48	300.13	247839.4	0.26	2.01	247839.4	27.66	2.01
instance_30b_05	254136.5	0.26	300.12	259333.8	0.20	-0.64	259333.8	23.21	-0.64
instance_110_01	1920308.4	3.91	301.25	2050139.1	1.78	-2.66	2050139.1	300.18	-2.66
instance_110_02	1937904.6	6.61	300.93	2082565.4	1.75	-5.34	2082565.4	300.13	-5.34
instance_110_03	2011959.1	1.82	301.11	2086918.6	1.60	-0.33	2086918.6	300.72	-0.33
instance_110_04	2060624.0	5.00	300.91	2148900.6	1.84	-2.89	2148900.6	300.26	-2.89
instance_110_05	2053093.2	2.16	301.12	2015801.2	1.85	-1.83	2015801.2	300.02	-1.83
instance_f_01	129829.8	0.00	0.22	120411.2	0.27	7.38	120411.2	23.69	7.38
instance_f_02	134864.3	0.01	53.07	127857.5	0.31	5.20	127857.5	23.25	5.20
instance_f_03	131870.5	0.00	0.77	124444.8	0.36	5.91	124444.8	14.66	5.91
instance_f_04	130108.4	0.01	40.23	127565.4	0.27	1.96	127565.4	12.34	1.96
instance_f_05	137811.6	0.01	19.82	121055.0	0.29	12.16	121055.0	17.91	12.16

Table 6.4: KPI con funzione obiettivo (4.1)

	CPLEX			Heuristic			Lagrangian Heuristic		
Istanza	N.S.O.	A.S.U.	H.U.S.	N.S.O.	A.S.U.	H.U.S.	N.S.O.	A.S.U.	H.U.S.
instance_09_01	5	95.59%	2	5	95.38%	2	5	95.38%	2
instance_09_02	7	84.68%	2	7	84.20%	2	7	84.20%	2
instance_09_03	6	88.89%	2	6	90.71%	2	6	90.71%	2
instance_09_04	5	86.65%	2	5	86.10%	2	5	86.10%	2
instance_09_05	5	98.34%	2	6	81.43%	2	6	81.43%	2
instance_30_01	23	95.94%	9	24	91.64%	8	24	91.64%	8
instance_30_02	21	93.09%	7	21	92.89%	6	21	92.89%	6
instance_30_03	21	95.87%	8	22	90.88%	7	22	90.88%	7
instance_30_04	23	96.12%	9	24	91.59%	8	24	91.59%	8
instance_30_05	21	95.91%	7	21	96.09%	7	21	96.09%	7
instance_30b_01	21	96.16%	6	21	96.24%	6	21	96.24%	6
instance_30b_02	19	95.29%	5	19	95.27%	5	19	95.27%	5
instance_30b_03	20	99.20%	7	21	94.30%	5	21	94.30%	5
instance_30b_04	22	97.72%	7	22	97.67%	6	22	97.67%	6
instance_30b_05	23	95.06%	6	23	95.06%	6	23	95.06%	6
instance_110_01	103	91.35%	32	95	98.74%	29	95	98.74%	29
instance_110_02	94	91.49%	31	88	97.58%	29	88	97.58%	29
instance_110_03	92	92.68%	29	87	98.08%	27	87	98.08%	27
instance_110_04	101	90.10%	32	92	98.62%	28	92	98.62%	28
instance_110_05	96	92.06%	30	90	98.73%	27	90	98.73%	27
instance_f_01	9	92.56%	2	9	92.10%	2	9	92.10%	2
instance_f_02	10	93.61%	3	10	93.12%	2	10	93.12%	2
instance_f_03	9	95.56%	2	9	95.57%	2	9	95.57%	2
instance_f_04	10	93.14%	2	10	93.17%	2	10	93.17%	2
instance_f_05	9	95.09%	3	9	94.16%	2	9	94.16%	2

Table 6.5: KPI con funzione obiettivo (4.1)

	CPLEX						Heuristic						Lagrangian Heuristic					
Istanza	D.R.	R.MAX	R.MIN	D.U.	U.MAX	U.MIN	D.R.	R.MAX	R.MIN	D.U.	U.MAX	U.MIN	D.R.	R.MAX	R.MIN	D.U.	U.MAX	U.MIN
instance_09_01	2.2	10.6	5.1	2.1	10.1	4.3	2.5	11.3	4.7	2.0	10.4	4.9	2.5	11.3	4.7	2.0	10.4	4.9
instance_09_02	2.3	9.2	1.8	2.0	7.7	1.6	2.9	11.7	1.8	2.5	10.3	1.6	2.9	11.7	1.8	2.5	10.3	1.6
instance_09_03	3.6	13.5	1.9	4.3	15.7	1.7	4.3	15.1	2.3	4.5	16.4	3.5	4.3	15.1	2.3	4.5	16.4	3.5
instance_09_04	3.1	11.2	3.0	3.0	11.1	3.4	4.2	12.4	2.6	4.2	13.2	2.7	4.2	12.4	2.6	4.2	13.2	2.7
instance_09_05	2.0	10.1	4.0	2.1	9.8	3.8	3.3	11.8	1.1	2.9	11.0	1.1	3.3	11.8	1.1	2.9	11.0	1.1
instance_30_01	1.1	6.4	2.5	0.9	5.9	2.7	1.3	6.6	1.1	1.1	5.9	1.4	1.3	6.6	1.1	1.1	5.9	1.4
instance_30_02	1.5	7.6	1.6	1.1	6.1	1.2	1.4	7.2	1.4	1.1	6.5	2.0	1.4	7.2	1.4	1.1	6.5	2.0
instance_30_03	1.5	6.5	1.2	1.1	5.8	1.7	1.9	9.0	1.0	1.4	6.5	1.7	1.9	9.0	1.0	1.4	6.5	1.7
instance_30_04	1.3	7.8	2.4	1.1	7.5	2.5	1.5	7.5	1.1	1.1	6.6	1.7	1.5	7.5	1.1	1.1	6.6	1.7
instance_30_05	1.3	6.8	1.8	1.2	6.4	1.9	1.4	7.8	2.4	1.1	6.6	2.2	1.4	7.8	2.4	1.1	6.6	2.2
instance_30b_01	4.2	20.0	2.7	3.0	15.5	3.3	4.4	21.8	2.7	3.3	17.9	3.4	4.4	21.8	2.7	3.3	17.9	3.4
instance_30b_02	4.5	19.9	1.7	4.2	19.2	1.5	5.5	28.2	1.4	5.0	25.5	1.6	5.5	28.2	1.4	5.0	25.5	1.6
instance_30b_03	4.1	23.6	4.7	3.2	19.8	5.1	4.8	22.0	1.5	4.0	19.9	1.6	4.8	22.0	1.5	4.0	19.9	1.6
instance_30b_04	3.1	19.0	4.7	2.8	18.4	5.2	3.5	19.7	3.3	2.8	17.5	4.5	3.5	19.7	3.3	2.8	17.5	4.5
instance_30b_05	4.8	24.4	1.3	4.2	21.7	1.5	4.9	24.5	1.0	3.9	19.4	2.0	4.9	24.5	1.0	3.9	19.4	2.0
instance_110_01	1.6	9.9	1.4	1.1	7.4	1.7	1.5	9.2	2.2	1.0	7.1	2.7	1.5	9.2	2.2	1.0	7.1	2.7
instance_110_02	1.6	9.8	1.2	1.3	8.1	1.1	1.6	10.9	1.1	1.3	8.2	1.8	1.6	10.9	1.1	1.3	8.2	1.8
instance_110_03	1.7	10.1	1.1	1.3	7.9	1.4	1.7	10.8	1.1	1.1	7.3	1.8	1.7	10.8	1.1	1.1	7.3	1.8
instance_110_04	1.7	9.4	1.1	1.2	7.4	1.8	1.7	10.4	2.1	1.1	7.7	2.7	1.7	10.4	2.1	1.1	7.7	2.7
instance_110_05	1.5	8.6	1.3	1.3	7.6	1.1	1.6	9.0	2.3	1.2	8.2	2.8	1.6	9.0	2.3	1.2	8.2	2.8
instance_f_01	8.6	26.7	1.5	8.2	25.7	1.7	10.8	36.6	1.5	10.8	36.7	1.7	10.8	36.6	1.5	10.8	36.7	1.7
instance_f_02	5.6	19.7	2.4	4.6	17.2	2.8	7.6	30.0	3.2	6.7	26.3	2.9	7.6	30.0	3.2	6.7	26.3	2.9
instance_f_03	9.8	35.2	2.3	8.4	30.7	3.2	9.4	34.0	2.3	8.2	30.5	3.4	9.4	34.0	2.3	8.2	30.5	3.4
instance_f_04	8.7	33.2	2.3	8.2	31.8	3.3	9.8	36.4	2.5	9.3	35.3	2.5	9.8	36.4	2.5	9.3	35.3	2.5
instance_f_05	6.0	21.8	2.7	5.5	20.7	3.2	8.6	29.1	2.7	8.1	27.2	3.0	8.6	29.1	2.7	8.1	27.2	3.0

Come si può notare nella tabella 6.3 nella colonna del tempo di esecuzione relativa alla esecuzione di CPLEX, già dal secondo gruppo di istanze il tempo di esecuzione è sempre al limite massimo di 300 secondi, questo indica che CPLEX non riesce a trovare la soluzione ottima in tempi ragionevoli.

6.4.1 **Funzione obiettivo** (4.10)

Nella tabella 6.6 vi è riportato il confronto tra la funzione obiettivo (4.1) e la funzione obiettivo (4.10) in termini di tempo di esecuzione. Come si può notare, esistono differenze nei tempi di esecuzione tra le due funzioni obiettivo. Non considerando CPLEX (che, come già detto, tende a saturare il tempo massimo di esecuzione), i tempi dell'algoritmo euristico con la funzione obiettivo (4.10) risultano il 16,3% più veloci rispetto a quelli della funzione obiettivo (4.1). Al contrario, i tempi dell'euristica lagrangiana variano solo del 2,2%, sempre a favore della funzione obiettivo (4.10). Per quanto riguarda i KPI delle due funzioni obiettivo si possono notare delle differenze minime, tra le tabelle 6.4 e 6.7 si può notare che vi sono alcune istanze per cui i KPI N.S.O. e H.U.S. risultano inferiri per le istanze delle funzione obiettivo (4.10), perciò questo indica che anticipare se possibile la risoluzione dei vincoli porta poi successivamente a velocizzare il progetto. Con-

frontando le tabelle 6.5 e 6.10, si nota che i valori di D.R. e R_{MAX} sono leggermente maggiori per la funzione obiettivo (4.1). In particolare, D.R. della funzione (4.1) è il 9,8% superiore rispetto a quella della funzione (4.10), mentre R_{MAX} risulta del 13,2% maggiore. Questo indica che anticipare la risoluzione dei vincoli porta a sprint mediamente meno rischiosi.

Table 6.6: confronto tempi di esecuzione tra le due funzioni obiettivo (4.1) e (4.10)

	Funzioni obiettivo(4.1)			Funzioni obiettivo(4.10)		
Istanza	CPLEX Time	Heuristic Time	Lagrangian Heuristic Time	CPLEX Time	Heuristic Time	Lagrangian Heuristic Time
instance_09_01	0.23	0.03	0.23	0.13	0.02	0.16
instance_09_02	0.42	0.02	0.22	0.43	0.02	0.38
instance_09_03	0.30	0.02	0.32	0.26	0.02	0.28
instance_09_04	0.11	0.03	0.33	0.06	0.03	0.22
instance_09_05	0.38	0.03	0.16	0.30	0.02	0.14
instance_30_01	300.10	0.10	3.01	300.09	0.09	2.10
instance_30_02	300.07	0.12	4.00	300.11	0.10	3.34
instance_30_03	217.66	0.14	3.03	200.22	0.13	2.40
instance_30_04	291.01	0.14	4.00	88.79	0.12	2.63
instance_30_05	300.05	0.21	2.46	300.12	0.13	2.82
instance_30b_01	300.12	0.18	18.34	300.10	0.15	20.06
instance_30b_02	300.12	0.21	14.45	300.10	0.18	15.56
instance_30b_03	300.11	0.15	22.44	300.10	0.23	21.00
instance_30b_04	300.13	0.26	27.66	300.12	0.16	26.84
instance_30b_05	300.12	0.20	23.21	300.14	0.14	23.19
instance_110_01	301.25	1.78	300.18	301.18	1.68	300.44
instance_110_02	300.93	1.75	300.13	300.92	1.48	300.73
instance_110_03	301.11	1.60	300.72	301.07	1.34	300.73
instance_110_04	300.91	1.84	300.26	302.31	1.65	300.26
instance_110_05	301.12	1.85	300.02	301.10	1.53	300.34
instance_f_01	0.22	0.27	23.69	6.08	0.28	25.06
instance_f_02	53.07	0.31	23.25	76.14	0.26	21.04
instance_f_03	0.77	0.36	14.66	73.87	0.25	13.35
instance_f_04	40.23	0.27	12.34	61.72	0.26	11.22
instance_f_05	19.82	0.29	17.91	25.08	0.29	17.36

Table 6.7: KPI con funzione obiettivo (4.10)

	CPLEX			Heuristic			Lagrangian Heuristic		
Istanza	N.S.O.	A.S.U.	H.U.S.	N.S.O.	A.S.U.	H.U.S.	N.S.O.	A.S.U.	H.U.S.
instance_09_01	5	96.05%	2	5	95.4%	2	5	95.4%	2
instance_09_02	7	84.17%	3	7	84.7%	2	7	84.7%	2
instance_09_03	6	89.71%	3	6	89.2%	2	6	89.2%	2
instance_09_04	5	86.65%	2	5	86.5%	2	5	86.5%	2
instance_09_05	6	80.57%	2	6	80.1%	2	6	80.1%	2
instance_30_01	23	95.31%	9	24	91.4%	9	24	91.4%	9
instance_30_02	21	93.10%	7	21	93.2%	6	21	93.2%	6
instance_30_03	21	95.80%	8	22	91.8%	8	22	91.8%	8
instance_30_04	23	96.08%	10	24	91.0%	8	24	91.0%	8
instance_30_05	21	96.26%	9	21	95.6%	8	21	95.6%	8
instance_30b_01	21	96.14%	6	21	96.1%	6	21	96.1%	6
instance_30b_02	19	95.28%	6	19	95.7%	5	19	95.7%	5
instance_30b_03	20	99.20%	7	21	94.7%	6	21	94.7%	6
instance_30b_04	22	97.70%	8	22	97.7%	7	22	97.7%	7
instance_30b_05	23	94.97%	7	23	95.7%	6	23	95.7%	6
instance_110_01	101	93.18%	32	95	98.7%	29	95	98.7%	29
instance_110_02	92	93.86%	32	88	97.5%	30	88	97.5%	30
instance_110_03	89	95.55%	30	87	98.0%	28	87	98.0%	28
instance_110_04	98	92.80%	34	93	97.6%	30	93	97.6%	30
instance_110_05	96	92.19%	29	90	98.7%	29	90	98.7%	29
instance_f_01	9	91.96%	3	9	91.4%	2	9	91.4%	2
instance_f_02	10	93.61%	3	10	93.4%	3	10	93.4%	3
instance_f_03	9	95.66%	2	9	95.5%	2	9	95.5%	2
instance_f_04	10	93.13%	2	10	93.3%	2	10	93.3%	2
instance_f_05	9	95.10%	3	9	94.4%	2	9	94.4%	2

Table 6.8: KPI con funzione obiettivo (4.10)

	CPLEX						Heuristic						Lagrangian Heuristic					
Istanza	D.R.	R.MAX	R.MIN	D.U.	U.MAX	U.MIN	D.R.	R.MAX	R.MIN	D.U.	U.MAX	U.MIN	D.R.	R.MAX	R.MIN	D.U.	U.MAX	U.MIN
instance_09_01	2.7	11.3	4.6	2.6	10.3	4.0	2.5	11.3	4.3	2.4	10.4	3.8	2.5	11.3	4.3	2.4	10.4	3.8
instance_09_02	2.4	9.2	1.8	2.2	8.0	1.6	2.8	11.1	1.8	2.3	8.9	1.6	2.8	11.1	1.8	2.3	8.9	1.6
instance_09_03	2.4	10.5	3.1	2.9	12.3	2.7	3.6	13.5	3.1	4.0	14.9	2.7	3.6	13.5	3.1	4.0	14.9	2.7
instance_09_04	3.1	11.2	3.0	3.0	11.1	3.4	3.1	11.2	3.0	3.0	11.1	3.4	3.1	11.2	3.0	3.0	11.1	3.4
instance_09_05	2.9	10.1	1.2	2.8	9.8	1.2	3.4	10.1	1.2	2.9	9.8	1.2	3.4	10.1	1.2	2.9	9.8	1.2
instance_30_01	1.3	6.3	1.1	1.1	6.4	1.8	1.3	6.6	1.1	1.3	7.1	1.4	1.3	6.6	1.1	1.3	7.1	1.4
instance_30_02	1.6	7.6	1.3	1.3	6.1	1.5	1.4	7.4	1.0	1.2	6.1	1.2	1.4	7.4	1.0	1.2	6.1	1.2
instance_30_03	1.4	6.5	1.2	1.1	5.8	1.7	1.4	6.8	1.2	1.2	6.3	1.7	1.4	6.8	1.2	1.2	6.3	1.7
instance_30_04	1.4	7.8	1.6	1.2	7.5	1.4	1.3	6.7	1.1	1.2	6.7	1.7	1.3	6.7	1.1	1.2	6.7	1.7
instance_30_05	1.5	8.3	2.3	1.1	7.0	3.1	1.5	8.1	1.4	1.0	6.4	1.7	1.5	8.1	1.4	1.0	6.4	1.7
instance_30b_01	3.8	16.2	2.3	2.9	14.4	3.3	4.0	18.9	2.7	3.0	15.1	3.3	4.0	18.9	2.7	3.0	15.1	3.3
instance_30b_02	4.1	17.8	1.7	3.9	17.1	1.5	4.3	18.5	1.4	4.0	18.2	1.6	4.3	18.5	1.4	4.0	18.2	1.6
instance_30b_03	3.5	17.0	5.2	2.8	15.8	4.3	4.6	18.5	1.5	3.7	15.8	1.6	4.6	18.5	1.5	3.7	15.8	1.6
instance_30b_04	1.9	11.9	5.6	1.4	11.0	6.0	3.1	18.6	3.4	2.6	17.2	4.8	3.1	18.6	3.4	2.6	17.2	4.8
instance_30b_05	4.4	20.7	1.8	3.8	19.1	1.2	4.3	17.3	1.0	3.8	18.3	2.0	4.3	17.3	1.0	3.8	18.3	2.0
instance_110_01	1.5	8.9	1.8	1.1	6.9	1.7	1.5	8.9	2.4	1.1	7.1	2.6	1.5	8.9	2.4	1.1	7.1	2.6
instance_110_02	1.5	10.1	1.1	1.2	7.3	1.6	1.6	8.8	1.1	1.1	7.6	1.8	1.6	8.8	1.1	1.1	7.6	1.8
instance_110_03	1.4	9.0	2.3	1.0	7.0	3.0	1.5	10.8	1.1	1.0	6.8	1.8	1.5	10.8	1.1	1.0	6.8	1.8
instance_110_04	1.4	8.5	2.1	1.1	6.9	2.7	1.6	10.5	1.1	1.2	7.5	1.5	1.6	10.5	1.1	1.2	7.5	1.5
instance_110_05	1.7	9.7	1.2	1.4	7.8	1.2	1.6	9.9	2.5	1.3	8.7	2.7	1.6	9.9	2.5	1.3	8.7	2.7
instance_f_01	6.2	18.0	1.5	6.0	18.0	1.7	8.9	29.5	1.5	8.9	29.2	1.7	8.9	29.5	1.5	8.9	29.2	1.7
instance_f_02	5.6	19.7	2.4	4.6	17.2	2.8	6.1	21.9	2.5	5.3	18.3	2.9	6.1	21.9	2.5	5.3	18.3	2.9
instance_f_03	7.3	25.3	2.3	6.6	22.7	2.7	8.0	27.3	2.4	7.2	24.5	2.8	8.0	27.3	2.4	7.2	24.5	2.8
instance_f_04	7.9	30.3	2.3	7.6	29.8	3.3	9.3	35.4	2.3	8.8	34.3	3.6	9.3	35.4	2.3	8.8	34.3	3.6
instance_f_05	7.0	23.7	2.7	6.3	20.9	3.2	7.4	22.8	2.6	6.9	22.1	3.3	7.4	22.8	2.6	6.9	22.1	3.3

6.4.2 Funzione obiettivo (4.11) e (4.12)

I risultati ottenuti utilizzando le funzioni obiettivo (4.11) e (4.12), come riportato nella tabella 6.9 che indica i tempi di esecuzione delle 2 funzioni obiettivo, mostrano (come nella relazione tra le funzioni obiettivo (4.1) e (4.10)) che vi è un discostamento per quanto riguarda i tempi di esecuzione dell'algoritmo euristico, in cui la funzione obiettivo (4.12) risulta più veloce del 15,1% rispetto alla sua controparte (4.11), mentre per quanto riguarda l'euristica lagrangiana i tempi di esecuzione non si discostano in modo significativo. Al contrario di quello che si è notato nel confronto tra le funzioni obiettivo (4.1) e (4.10), in questo caso il confronto dei KPI N.S.O. e H.U.S. tra le due funzioni obiettivo risulta per alcune istanze favorevole alla funzione obiettivo (4.11) in quanto riesce a concludere le istanze con un numero leggermente minore di sprint rispetto alla funzione obiettivo (4.12). Per quanto riguarda i KPI D.U. e U.MAX si può notare confrontando le tabelle 6.11 e 6.13 che la funzione obiettivo (4.11) risulta avere un valore del D.U. del 8.1% maggiore rispetto a quello della funzione obiettivo (4.12), mentre per il KPI U.MAX risulta essere maggiore del 12.3%, indicando che vi sono presenti in media sprint maggiormente incerti.

Table 6.9: confronto tempi di esecuzione tra le due funzioni obiettivo (4.11) e (4.12)

	Funzioni obiettivo(4.11)			Funzioni obiettivo(4.12)		
Istanza	CPLEX Time	Heuristic Time	Lagrangian Heuristic Time	CPLEX Time	Heuristic Time	Lagrangian Heuristic Time
instance_09_01	0.12	0.02	0.14	0.03	0.02	0.16
instance_09_02	0.39	0.02	0.24	0.26	0.02	0.34
instance_09_03	0.36	0.02	0.41	0.46	0.03	0.41
instance_09_04	0.09	0.04	0.27	0.03	0.02	0.27
instance_09_05	0.24	0.02	0.15	0.42	0.02	0.15
instance_30_01	300.09	0.09	2.11	300.07	0.09	2.52
instance_30_02	300.10	0.13	3.37	128.92	0.10	3.57
instance_30_03	300.10	0.15	2.39	271.07	0.13	2.37
instance_30_04	232.82	0.13	2.59	287.18	0.10	2.67
instance_30_05	300.11	0.15	2.27	300.05	0.12	2.00
instance_30b_01	300.13	0.23	16.53	300.12	0.27	17.70
instance_30b_02	300.12	0.15	14.26	300.10	0.16	16.10
instance_30b_03	300.09	0.28	19.72	300.10	0.15	17.05
instance_30b_04	300.11	0.21	24.39	300.16	0.15	23.99
instance_30b_05	300.12	0.21	22.86	300.12	0.19	25.55
instance_110_01	301.24	1.72	300.47	301.20	1.61	300.01
instance_110_02	300.98	1.66	300.42	300.94	1.52	300.01
instance_110_03	301.12	1.43	300.71	301.10	1.46	300.60
instance_110_04	300.97	1.81	300.19	300.94	1.91	300.60
instance_110_05	301.10	1.64	300.52	301.10	1.59	300.27
instance_f_01	4.45	0.27	22.54	24.06	0.28	24.49
instance_f_02	177.38	0.31	21.26	223.67	0.27	24.92
instance_f_03	1.68	0.28	13.01	1.01	0.25	13.77
instance_f_04	75.31	0.39	12.89	0.84	0.29	12.01
instance_f_05	26.09	0.32	17.52	14.03	0.26	17.30

Table 6.10: KPI con funzione obiettivo (4.11)

	CPLEX			Heuristic			Lagrangian Heuristic		
Istanza	N.S.O.	A.S.U.	H.U.S.	N.S.O.	A.S.U.	H.U.S.	N.S.O.	A.S.U.	H.U.S.
instance_09_01	5	95.62%	2	5	95.32%	2	5	95.32%	2
instance_09_02	7	84.42%	3	7	84.47%	2 xt	7	84.47%	2
instance_09_03	6	88.89%	2	6	88.76%	2	6	88.76%	2
instance_09_04	5	86.65%	2	5	86.86%	2	5	86.86%	2
instance_09_05	6	81.67%	2	6	80.77%	2	6	80.77%	2
instance_30_01	23	95.50%	9	24	91.51%	8	24	91.51%	8
instance_30_02	20	98.03%	7	21	93.89%	6	21	93.89%	6
instance_30_03	21	95.93%	7	21	95.85%	7	21	95.85%	7
instance_30_04	23	95.89%	9	24	91.24%	7	24	91.24%	7
instance_30_05	21	96.12%	7	22	91.05%	7	22	91.05%	7
instance_30b_01	21	96.26%	6	21	96.22%	6	21	96.22%	6
instance_30b_02	19	95.27%	5	19	95.28%	4	19	95.28%	4
instance_30b_03	20	99.20%	6	21	94.31%	5	21	94.31%	5
instance_30b_04	22	97.62%	7	22	97.68%	6	22	97.68%	6
instance_30b_05	23	95.01%	7	23	95.02%	6	23	95.02%	6
instance_110_01	102	92.13%	31	95	98.69%	28	95	98.69%	28
instance_110_02	92	98.13%	29	87	98.66%	29	87	98.66%	29
instance_110_03	92	92.29%	29	87	98.13%	27	87	98.13%	27
instance_110_04	100	91.01%	34	92	98.66%	30	92	98.66%	30
instance_110_05	96	91.82%	29	90	98.72%	28	90	98.72%	28
instance_f_01	9	92.25%	2	9	92.11%	2	9	92.11%	2
instance_f_02	10	93.65%	3	10	93.72%	2	10	93.72%	2
instance_f_03	9	95.45%	2	9	95.36%	2	9	95.36%	2
instance_f_04	10	93.10%	2	10	93.23%	2	10	93.23%	2
instance_f_05	9	95.08%	3	9	94.13%	2	9	94.13%	2

Table 6.11: KPI con funzione obiettivo (4.11)

	CPLEX						Heuristic						Lagrangian Heuristic					
Istanza	D.R.	R.MAX	R.MIN	D.U.	U.MAX	U.MIN	D.R.	R.MAX	R.MIN	D.U.	U.MAX	U.MIN	D.R.	R.MAX	R.MIN	D.U.	U.MAX	U.MIN
instance_09_01	2.6	11.2	4.3	2.4	10.4	3.8	2.5	10.8	4.6	2.6	10.3	4.0	2.5	10.8	4.6	2.6	10.3	4.0
instance_09_02	2.1	7.9	1.6	2.4	8.0	1.1	2.6	9.7	1.6	2.6	9.2	1.1	2.6	9.7	1.6	2.6	9.2	1.1
instance_09_03	3.6	13.5	1.9	4.3	15.7	1.7	3.6	13.5	1.9	4.0	14.9	1.7	3.6	13.5	1.9	4.0	14.9	1.7
instance_09_04	3.1	11.2	3.0	3.0	11.1	3.4	4.0	12.4	2.5	4.1	13.2	3.0	4.0	12.4	2.5	4.1	13.2	3.0
instance_09_05	3.3	10.5	1.3	3.1	9.9	1.1	3.6	13.1	1.2	3.4	12.5	1.2	3.6	13.1	1.2	3.4	12.5	1.2
instance_30_01	1.4	7.9	1.1	1.1	6.3	1.8	1.5	8.8	1.3	1.0	6.4	1.4	1.5	8.8	1.3	1.0	6.4	1.4
instance_30_02	1.1	7.3	2.4	1.1	7.0	2.4	1.2	6.2	2.3	1.1	6.6	2.3	1.2	6.2	2.3	1.1	6.6	2.3
instance_30_03	1.5	8.5	2.1	1.2	7.0	2.6	1.5	6.4	2.1	1.2	6.9	2.5	1.5	6.4	2.1	1.2	6.9	2.5
instance_30_04	1.2	8.1	2.3	1.1	6.9	2.5	1.1	6.1	1.4	1.1	5.4	1.4	1.1	6.1	1.4	1.1	5.4	1.4
instance_30_05	1.2	6.7	2.7	1.1	6.0	2.5	1.3	6.5	1.2	1.3	6.7	1.0	1.3	6.5	1.2	1.3	6.7	1.0
instance_30b_01	3.6	19.1	3.6	3.4	18.0	2.5	4.0	20.2	3.4	3.7	19.7	2.9	4.0	20.2	3.4	3.7	19.7	2.9
instance_30b_02	4.1	19.5	3.6	4.2	20.3	2.4	4.8	21.8	1.4	4.8	22.9	1.6	4.8	21.8	1.4	4.8	22.9	1.6
instance_30b_03	3.5	20.3	4.5	3.4	20.3	4.5	4.5	22.6	2.0	4.4	22.4	1.1	4.5	22.6	2.0	4.4	22.4	1.1
instance_30b_04	2.8	18.4	5.2	2.8	18.9	5.8	3.6	21.5	5.5	3.8	23.0	4.8	3.6	21.5	5.5	3.8	23.0	4.8
instance_30b_05	4.4	24.1	1.3	4.4	25.2	1.5	4.5	23.4	1.8	4.7	26.0	1.2	4.5	23.4	1.8	4.7	26.0	1.2
instance_110_01	1.3	7.3	1.2	1.2	7.4	1.9	1.3	8.2	2.1	1.1	7.3	2.8	1.3	8.2	2.1	1.1	7.3	2.8
instance_110_02	1.4	9.3	1.1	1.3	8.4	1.2	1.4	8.8	2.5	1.1	7.7	3.1	1.4	8.8	2.5	1.1	7.7	3.1
instance_110_03	1.5	8.6	1.2	1.2	7.3	1.8	1.4	9.3	1.1	1.3	8.4	1.2	1.4	9.3	1.1	1.3	8.4	1.2
instance_110_04	1.4	8.2	1.4	1.3	8.0	1.2	1.2	7.6	2.4	1.1	8.0	2.7	1.2	7.6	2.4	1.1	8.0	2.7
instance_110_05	1.4	8.2	1.6	1.2	7.2	1.1	1.5	8.7	2.6	1.3	8.1	2.6	1.5	8.7	2.6	1.3	8.1	2.6
instance_f_01	6.9	24.7	1.5	6.9	25.5	1.7	10.8	37.3	1.5	11.3	39.2	1.7	10.8	37.3	1.5	11.3	39.2	1.7
instance_f_02	5.1	18.2	2.4	4.7	17.6	2.8	7.3	28.3	3.1	6.6	25.1	2.6	7.3	28.3	3.1	6.6	25.1	2.6
instance_f_03	9.0	32.7	2.6	8.2	30.0	2.9	9.0	33.3	2.4	8.3	30.9	2.8	9.0	33.3	2.4	8.3	30.9	2.8
instance_f_04	7.8	28.2	2.3	8.1	29.8	3.3	8.9	32.4	2.3	8.9	33.5	3.7	8.9	32.4	2.3	8.9	33.5	3.7
instance_f_05	6.4	22.4	2.7	6.4	24.4	3.2	8.2	29.4	2.7	8.5	31.1	3.0	8.2	29.4	2.7	8.5	31.1	3.0

Table 6.12: KPI con funzione obiettivo (4.12)

	CPLEX			Heuristic			Lagrangian Heuristic		
Istanza	N.S.O.	A.S.U.	H.U.S.	N.S.O.	A.S.U.	H.U.S.	N.S.O.	A.S.U.	H.U.S.
instance_09_01	6	79.77%	2	5	95.32%	2	5	95.32%	2
instance_09_02	7	84.42%	3	7	84.56%	2	7	84.56%	2
instance_09_03	6	89.71%	3	6	89.64%	2	6	89.64%	2
instance_09_04	5	86.65%	2	5	86.65%	2	5	86.65%	2
instance_09_05	6	80.57%	2	6	80.58%	2	6	80.58%	2
instance_30_01	23	95.38%	9	24	91.29%	9	24	91.29%	9
instance_30_02	20	98.03%	7	21	93.59%	6	21	93.59%	6
instance_30_03	21	96.05%	8	21	95.80%	8	21	95.80%	8
instance_30_04	23	96.22%	10	24	90.84%	8	24	90.84%	8
instance_30_05	21	95.75%	8	21	95.84%	9	21	95.84%	9
instance_30b_01	21	96.23%	6	21	96.28%	6	21	96.28%	6
instance_30b_02	19	95.28%	6	19	95.28%	5	19	95.28%	5
instance_30b_03	20	99.20%	7	21	94.30%	6	21	94.30%	6
instance_30b_04	22	97.75%	8	22	97.66%	7	22	97.66%	7
instance_30b_05	23	95.04%	7	23	94.98%	6	23	94.98%	6
instance_110_01	100	94.15%	33	96	97.70%	29	96	97.70%	29
instance_110_02	92	93.60%	33	87	98.61%	30	87	98.61%	30
instance_110_03	90	94.47%	29	87	98.15%	28	87	98.15%	28
instance_110_04	98	92.89%	33	92	98.64%	30	92	98.64%	30
instance_110_05	90	98.70%	29	90	98.70%	29	90	98.70%	29
instance_f_01	9	92.24%	3	9	91.62%	2	9	91.62%	2
instance_f_02	10	93.65%	3	10	93.23%	3	10	93.23%	3
instance_f_03	9	95.06%	2	9	95.26%	2	9	95.26%	2
instance_f_04	10	92.62%	2	10	93.00%	2	10	93.00%	2
instance_f_05	9	95.12%	3	9	94.23%	3	9	94.23%	3

Table 6.13: KPI con funzione obiettivo (4.12)

	CPLEX						Heuristic						Lagrangian Heuristic					
Istanza	D.R.	R.MAX	R.MIN	D.U.	U.MAX	U.MIN	D.R.	R.MAX	R.MIN	D.U.	U.MAX	U.MIN	D.R.	R.MAX	R.MIN	D.U.	U.MAX	U.MIN
instance_09_01	3.1	10.8	1.8	3.0	10.2	1.3	2.5	10.8	4.6	2.6	10.3	4.0	2.5	10.8	4.6	2.6	10.3	4.0
instance_09_02	2.1	7.9	1.6	2.4	8.0	1.1	2.3	9.7	1.8	2.3	9.1	1.6	2.3	9.7	1.8	2.3	9.1	1.6
instance_09_03	2.4	10.5	3.1	2.9	12.3	2.7	3.5	13.5	3.1	4.0	14.9	2.7	3.5	13.5	3.1	4.0	14.9	2.7
instance_09_04	3.1	11.2	3.0	3.0	11.1	3.4	3.1	11.2	3.0	3.0	11.1	3.4	3.1	11.2	3.0	3.0	11.1	3.4
instance_09_05	2.9	10.1	1.2	2.8	9.8	1.2	3.3	10.1	1.2	2.9	9.8	1.2	3.3	10.1	1.2	2.9	9.8	1.2
instance_30_01	1.4	6.9	1.1	1.2	6.2	1.8	1.4	6.8	1.1	1.2	6.2	1.4	1.4	6.8	1.1	1.2	6.2	1.4
instance_30_02	1.1	7.3	2.4	1.1	7.0	2.4	1.2	7.3	1.6	1.2	7.1	1.2	1.2	7.3	1.6	1.2	7.1	1.2
instance_30_03	1.3	6.5	2.4	1.0	6.3	2.8	1.3	6.6	2.0	1.1	6.3	2.8	1.3	6.6	2.0	1.1	6.3	2.8
instance_30_04	1.2	7.8	2.4	1.1	7.5	2.8	1.4	6.7	1.1	1.1	5.8	1.7	1.4	6.7	1.1	1.1	5.8	1.7
instance_30_05	1.4	8.3	2.3	1.2	7.0	2.6	1.1	6.5	2.5	1.1	6.6	2.6	1.1	6.5	2.5	1.1	6.6	2.6
instance_30b_01	3.5	16.4	3.6	3.2	15.5	3.0	3.6	18.3	3.6	3.3	15.8	2.5	3.6	18.3	3.6	3.3	15.8	2.5
instance_30b_02	3.7	17.6	3.6	3.8	18.0	2.4	4.5	23.2	1.4	4.6	23.6	1.6	4.5	23.2	1.4	4.6	23.6	1.6
instance_30b_03	3.2	17.3	4.5	2.9	15.8	4.5	3.9	18.2	2.0	3.7	16.5	1.1	3.9	18.2	2.0	3.7	16.5	1.1
instance_30b_04	1.9	13.6	6.0	1.8	12.9	5.7	2.7	15.9	4.4	2.6	15.8	4.9	2.7	15.9	4.4	2.6	15.8	4.9
instance_30b_05	4.2	22.5	1.4	4.3	23.2	1.5	4.2	17.6	1.8	4.0	18.5	1.2	4.2	17.6	1.8	4.0	18.5	1.2
instance_110_01	1.3	8.1	2.3	1.0	6.9	2.6	1.4	8.2	1.6	1.2	8.0	1.1	1.4	8.2	1.6	1.2	8.0	1.1
instance_110_02	1.4	10.2	1.2	1.1	7.9	1.8	1.4	9.2	2.4	1.2	9.1	2.7	1.4	9.2	2.4	1.2	9.1	2.7
instance_110_03	1.4	8.8	1.1	1.1	7.4	1.2	1.3	8.7	1.1	1.0	7.8	1.2	1.3	8.7	1.1	1.0	7.8	1.2
instance_110_04	1.3	7.8	1.4	1.2	7.1	1.9	1.3	8.6	2.6	1.1	7.8	2.8	1.3	8.6	2.6	1.1	7.8	2.8
instance_110_05	1.5	9.0	2.3	1.2	8.1	2.8	1.5	9.0	2.3	1.2	8.1	2.8	1.5	9.0	2.3	1.2	8.1	2.8
instance_f_01	6.5	21.0	1.5	6.5	22.2	1.7	8.5	26.3	1.5	9.1	27.4	1.7	8.5	26.3	1.5	9.1	27.4	1.7
instance_f_02	5.1	18.4	2.4	4.7	17.1	2.8	6.1	21.1	3.3	5.7	19.2	2.5	6.1	21.1	3.3	5.7	19.2	2.5
instance_f_03	7.2	23.1	2.4	6.6	21.0	2.8	7.4	24.7	2.4	7.2	22.7	2.8	7.4	24.7	2.4	7.2	22.7	2.8
instance_f_04	8.1	29.4	2.7	8.1	30.2	3.1	8.1	29.8	2.5	8.2	30.4	3.6	8.1	29.8	2.5	8.2	30.4	3.6
instance_f_05	7.0	22.4	2.7	7.0	22.2	3.2	7.6	23.1	2.6	7.7	24.1	3.0	7.6	23.1	2.6	7.7	24.1	3.0

6.4.3 Funzione obiettivo (4.13) e (4.14)

I risultati ottenuti risolvendo le istanze con le funzioni obiettivo (4.13) e (4.14) per quanto riguarda i tempi di esecuzione come mostrato nella tabella 6.14 dimostrando che la funzione obiettivo (4.14) risulta essere più veloce del 14.6% se si usa il metodo euristico e del 16.4% se si usa l'euristica Lagrangiana rispetto alla funzione obiettivo (4.13). Osservando i KPI nelle tabelle 6.15 e 6.17 si nota una differenza in alcune istanze sostanziale per quanto riguarda il valore di H.U.S. che risulta essere più basso utilizzando la funzione obiettivo (4.13) rispetto alla funzione obiettivo (4.14), mentre il valore di N.S.O. risulta essere pressoché uguale, Questo indica che, per questa tipologia di funzione obiettivo, considerare i parametri dp_{ij} e ub_{ij} conduce a risultati inefficienti in termini di utilità per l'utente. Osservando invece le tabelle 6.16 e 6.18 si nota come i valori di D.R., D.U. R.MAX e U.MAX risultano essere più bassi utilizzando la funzione obiettivo (4.14) di un valore compreso tra 19.8% e 21.5% rispetto alla funzione obiettivo (4.13), questo indica che la funzione obiettivo (4.14) riesce a distribuire meglio sia il rischio che l'incertezza lungo i vari sprint rispetto alla funzione obiettivo (4.13).

Table 6.14: confronto tempi di esecuzione tra le due funzioni obiettivo (4.13) e (4.14)

	Funzioni obiettivo(4.13)			Funzioni obiettivo(4.14)		
Istanza	CPLEX Time	Heuristic Time	Lagrangian Heuristic Time	CPLEX Time	Heuristic Time	Lagrangian Heuristic Time
instance_09_01	0.05	0.02	0.14	0.06	0.03	0.26
instance_09_02	0.72	0.03	0.23	0.39	0.03	0.42
instance_09_03	0.25	0.02	0.39	0.35	0.04	0.42
instance_09_04	0.06	0.02	0.28	0.04	0.03	0.50
instance_09_05	0.30	0.02	0.14	0.37	0.10	0.27
instance_30_01	300.05	0.12	2.14	300.04	0.08	2.43
instance_30_02	300.07	0.11	3.39	300.17	0.11	2.03
instance_30_03	300.16	0.18	2.41	300.07	0.10	1.99
instance_30_04	300.07	0.12	2.37	300.06	0.09	2.20
instance_30_05	300.10	0.15	2.43	300.05	0.09	2.76
instance_30b_01	300.09	0.22	20.01	300.11	0.16	7.52
instance_30b_02	300.12	0.21	18.55	300.11	0.11	9.22
instance_30b_03	300.11	0.27	17.10	300.11	0.15	10.44
instance_30b_04	300.10	0.20	24.32	300.11	0.13	9.71
instance_30b_05	300.34	0.21	31.59	300.23	0.14	7.29
instance_110_01	301.17	2.12	300.30	301.20	1.43	156.91
instance_110_02	300.92	1.87	300.49	300.96	1.40	199.44
instance_110_03	301.06	3.50	300.45	301.12	2.97	244.32
instance_110_04	300.99	3.79	300.21	300.93	1.78	231.82
instance_110_05	301.83	2.62	300.30	301.90	1.48	274.63
instance_f_01	300.63	0.31	25.16	39.65	0.76	23.86
instance_f_02	300.44	0.40	22.89	69.11	0.29	22.24
instance_f_03	300.58	0.63	14.82	15.99	0.44	14.67
instance_f_04	300.48	0.66	12.78	135.48	0.63	10.77
instance_f_05	300.55	0.77	18.89	4.86	0.54	18.46

Table 6.15: KPI con funzione obiettivo (4.13)

	CPLEX			Heuristic			Lagrangian Heuristic		
Istanza	N.S.O.	A.S.U.	H.U.S.	N.S.O.	A.S.U.	H.U.S.	N.S.O.	A.S.U.	H.U.S.
instance_09_01	5	95.62%	2	5	95.32%	2	5	95.32%	2
instance_09_02	7	84.42%	3	7	84.47%	2 xt	7	84.47%	2
instance_09_03	6	88.89%	2	6	88.76%	2	6	88.76%	2
instance_09_04	5	86.65%	2	5	86.86%	2	5	86.86%	2
instance_09_05	6	81.67%	2	6	80.77%	2	6	80.77%	2
instance_30_01	23	95.50%	9	24	91.51%	8	24	91.51%	8
instance_30_02	20	98.03%	7	21	93.89%	6	21	93.89%	6
instance_30_03	21	95.93%	7	21	95.85%	7	21	95.85%	7
instance_30_04	23	95.89%	9	24	91.24%	7	24	91.24%	7
instance_30_05	21	96.12%	7	22	91.05%	7	22	91.05%	7
instance_30b_01	21	96.26%	6	21	96.22%	6	21	96.22%	6
instance_30b_02	19	95.27%	5	19	95.28%	4	19	95.28%	4
instance_30b_03	20	99.20%	6	21	94.31%	5	21	94.31%	5
instance_30b_04	22	97.62%	7	22	97.68%	6	22	97.68%	6
instance_30b_05	23	95.01%	7	23	95.02%	6	23	95.02%	6
instance_110_01	102	92.13%	31	95	98.69%	28	95	98.69%	28
instance_110_02	92	98.13%	29	87	98.66%	29	87	98.66%	29
instance_110_03	92	92.29%	29	87	98.13%	27	87	98.13%	27
instance_110_04	100	91.01%	34	92	98.66%	30	92	98.66%	30
instance_110_05	96	91.82%	29	90	98.72%	28	90	98.72%	28
instance_f_01	9	92.25%	2	9	92.11%	2	9	92.11%	2
instance_f_02	10	93.65%	3	10	93.72%	2	10	93.72%	2
instance_f_03	9	95.45%	2	9	95.36%	2	9	95.36%	2
instance_f_04	10	93.10%	2	10	93.23%	2	10	93.23%	2
instance_f_05	9	95.08%	3	9	94.13%	2	9	94.13%	2

Table 6.16: KPI con funzione obiettivo (4.13)

	CPLEX						Heuristic						Lagrangian Heuristic					
Istanza	D.R.	R.MAX	R.MIN	D.U.	U.MAX	U.MIN	D.R.	R.MAX	R.MIN	D.U.	U.MAX	U.MIN	D.R.	R.MAX	R.MIN	D.U.	U.MAX	U.MIN
instance_09_01	2.6	11.2	4.3	2.4	10.4	3.8	2.5	10.8	4.6	2.6	10.3	4.0	2.5	10.8	4.6	2.6	10.3	4.0
instance_09_02	2.1	7.9	1.6	2.4	8.0	1.1	2.6	9.7	1.6	2.6	9.2	1.1	2.6	9.7	1.6	2.6	9.2	1.1
instance_09_03	3.6	13.5	1.9	4.3	15.7	1.7	3.6	13.5	1.9	4.0	14.9	1.7	3.6	13.5	1.9	4.0	14.9	1.7
instance_09_04	3.1	11.2	3.0	3.0	11.1	3.4	4.0	12.4	2.5	4.1	13.2	3.0	4.0	12.4	2.5	4.1	13.2	3.0
instance_09_05	3.3	10.5	1.3	3.1	9.9	1.1	3.6	13.1	1.2	3.4	12.5	1.2	3.6	13.1	1.2	3.4	12.5	1.2
instance_30_01	1.4	7.9	1.1	1.1	6.3	1.8	1.5	8.8	1.3	1.0	6.4	1.4	1.5	8.8	1.3	1.0	6.4	1.4
instance_30_02	1.1	7.3	2.4	1.1	7.0	2.4	1.2	6.2	2.3	1.1	6.6	2.3	1.2	6.2	2.3	1.1	6.6	2.3
instance_30_03	1.5	8.5	2.1	1.2	7.0	2.6	1.5	6.4	2.1	1.2	6.9	2.5	1.5	6.4	2.1	1.2	6.9	2.5
instance_30_04	1.2	8.1	2.3	1.1	6.9	2.5	1.1	6.1	1.4	1.1	5.4	1.4	1.1	6.1	1.4	1.1	5.4	1.4
instance_30_05	1.2	6.7	2.7	1.1	6.0	2.5	1.3	6.5	1.2	1.3	6.7	1.0	1.3	6.5	1.2	1.3	6.7	1.0
instance_30b_01	3.6	19.1	3.6	3.4	18.0	2.5	4.0	20.2	3.4	3.7	19.7	2.9	4.0	20.2	3.4	3.7	19.7	2.9
instance_30b_02	4.1	19.5	3.6	4.2	20.3	2.4	4.8	21.8	1.4	4.8	22.9	1.6	4.8	21.8	1.4	4.8	22.9	1.6
instance_30b_03	3.5	20.3	4.5	3.4	20.3	4.5	4.5	22.6	2.0	4.4	22.4	1.1	4.5	22.6	2.0	4.4	22.4	1.1
instance_30b_04	2.8	18.4	5.2	2.8	18.9	5.8	3.6	21.5	5.5	3.8	23.0	4.8	3.6	21.5	5.5	3.8	23.0	4.8
instance_30b_05	4.4	24.1	1.3	4.4	25.2	1.5	4.5	23.4	1.8	4.7	26.0	1.2	4.5	23.4	1.8	4.7	26.0	1.2
instance_110_01	1.3	7.3	1.2	1.2	7.4	1.9	1.3	8.2	2.1	1.1	7.3	2.8	1.3	8.2	2.1	1.1	7.3	2.8
instance_110_02	1.4	9.3	1.1	1.3	8.4	1.2	1.4	8.8	2.5	1.1	7.7	3.1	1.4	8.8	2.5	1.1	7.7	3.1
instance_110_03	1.5	8.6	1.2	1.2	7.3	1.8	1.4	9.3	1.1	1.3	8.4	1.2	1.4	9.3	1.1	1.3	8.4	1.2
instance_110_04	1.4	8.2	1.4	1.3	8.0	1.2	1.2	7.6	2.4	1.1	8.0	2.7	1.2	7.6	2.4	1.1	8.0	2.7
instance_110_05	1.4	8.2	1.6	1.2	7.2	1.1	1.5	8.7	2.6	1.3	8.1	2.6	1.5	8.7	2.6	1.3	8.1	2.6
instance_f_01	6.9	24.7	1.5	6.9	25.5	1.7	10.8	37.3	1.5	11.3	39.2	1.7	10.8	37.3	1.5	11.3	39.2	1.7
instance_f_02	5.1	18.2	2.4	4.7	17.6	2.8	7.3	28.3	3.1	6.6	25.1	2.6	7.3	28.3	3.1	6.6	25.1	2.6
instance_f_03	9.0	32.7	2.6	8.2	30.0	2.9	9.0	33.3	2.4	8.3	30.9	2.8	9.0	33.3	2.4	8.3	30.9	2.8
instance_f_04	7.8	28.2	2.3	8.1	29.8	3.3	8.9	32.4	2.3	8.9	33.5	3.7	8.9	32.4	2.3	8.9	33.5	3.7
instance_f_05	6.4	22.4	2.7	6.4	24.4	3.2	8.2	29.4	2.7	8.5	31.1	3.0	8.2	29.4	2.7	8.5	31.1	3.0

Table 6.17: KPI con funzione obiettivo (4.14)

	CPLEX			Heuristic			Lagrangian Heuristic		
Istanza	N.S.O.	A.S.U.	H.U.S.	N.S.O.	A.S.U.	H.U.S.	N.S.O.	A.S.U.	H.U.S.
instance_09_01	6	79.15%	2	5	95.11%	2	5	95.11%	2
instance_09_02	7	84.27%	3	7	84.56%	3	7	84.56%	3
instance_09_03	6	88.80%	3	6	88.81%	3	6	88.81%	3
instance_09_04	5	87.18%	2	5	87.20%	2	5	87.20%	2
instance_09_05	6	80.63%	3	5	98.32%	3	5	98.32%	3
instance_30_01	23	95.55%	11	23	95.43%	11	23	95.43%	11
instance_30_02	21	93.32%	10	21	93.26%	10	21	93.26%	10
instance_30_03	21	95.96%	9	21	95.58%	9	21	95.58%	9
instance_30_04	23	96.05%	11	23	95.89%	10	23	95.89%	10
instance_30_05	21	96.00%	11	21	96.16%	11	21	96.16%	11
instance_30b_01	21	96.25%	8	21	96.24%	8	21	96.24%	8
instance_30b_02	19	95.29%	8	19	95.28%	8	19	95.28%	8
instance_30b_03	20	99.18%	9	20	99.19%	8	20	99.19%	8
instance_30b_04	22	97.70%	11	22	97.70%	10	22	97.70%	10
instance_30b_05	22	99.31%	9	22	99.34%	8	22	99.34%	8
instance_110_01	100	93.99%	43	95	98.65%	41	95	98.65%	41
instance_110_02	95	98.65%	41	87	98.69%	37	87	98.69%	37
instance_110_03	91	93.62%	40	86	99.33%	37	86	99.33%	37
instance_110_04	99	91.88%	43	93	97.72%	41	93	97.72%	41
instance_110_05	94	94.44%	43	90	98.80%	40	90	98.80%	40
instance_f_01	9	90.55%	4	9	90.70%	4	9	90.70%	4
instance_f_02	10	93.79%	4	10	93.78%	4	10	93.78%	4
instance_f_03	9	95.72%	4	9	95.70%	4	9	95.70%	4
instance_f_04	10	93.88%	4	10	93.59%	4	10	93.59%	4
instance_f_05	9	95.13%	4	9	95.20%	4	9	95.20%	4

Table 6.18: KPI con funzione obiettivo (4.14)

	CPLEX						Heuristic						Lagrangian Heuristic					
Istanza	D.R.	R.MAX	R.MIN	D.U.	U.MAX	U.MIN	D.R.	R.MAX	R.MIN	D.U.	U.MAX	U.MIN	D.R.	R.MAX	R.MIN	D.U.	U.MAX	U.MIN
instance_09_01	3.6	10.6	1.3	3.3	10.1	1.3	3.1	11.3	3.4	2.9	10.7	3.2	3.1	11.3	3.4	2.9	10.7	3.2
instance_09_02	2.3	7.5	1.1	2.0	8.0	2.0	2.5	10.2	1.9	2.4	9.7	1.5	2.5	10.2	1.9	2.4	9.7	1.5
instance_09_03	1.8	8.6	3.4	2.2	9.7	3.5	3.2	10.6	1.5	3.6	12.7	1.8	3.2	10.6	1.5	3.6	12.7	1.8
instance_09_04	2.6	10.1	2.9	2.2	10.5	3.8	2.7	10.2	2.9	2.4	10.5	3.8	2.7	10.2	2.9	2.4	10.5	3.8
instance_09_05	3.0	9.9	1.8	2.9	9.3	1.5	1.7	9.9	4.5	1.4	9.2	5.0	1.7	9.9	4.5	1.4	9.2	5.0
instance_30_01	1.3	7.0	2.5	0.9	5.7	2.6	1.4	6.8	2.0	0.9	5.4	1.9	1.4	6.8	2.0	0.9	5.4	1.9
instance_30_02	1.4	7.9	1.3	1.1	5.8	1.8	1.7	8.0	1.7	1.3	5.8	1.9	1.7	8.0	1.7	1.3	5.8	1.9
instance_30_03	1.4	6.6	1.3	0.9	5.8	2.0	1.4	7.5	1.3	1.1	6.1	2.0	1.4	7.5	1.3	1.1	6.1	2.0
instance_30_04	1.3	7.3	1.9	1.1	7.0	2.0	1.2	6.3	1.7	0.8	5.9	2.0	1.2	6.3	1.7	0.8	5.9	2.0
instance_30_05	1.3	7.0	2.4	1.0	6.0	2.6	1.5	8.1	2.2	1.1	6.9	2.6	1.5	8.1	2.2	1.1	6.9	2.6
instance_30b_01	3.1	15.4	1.3	2.4	13.7	1.9	3.4	15.5	1.5	2.3	11.8	1.9	3.4	15.5	1.5	2.3	11.8	1.9
instance_30b_02	3.1	15.3	3.1	2.5	15.2	3.6	3.7	15.6	1.4	2.9	14.5	1.9	3.7	15.6	1.4	2.9	14.5	1.9
instance_30b_03	2.5	16.4	5.2	1.9	15.1	6.9	3.1	16.0	4.3	2.1	13.7	5.1	3.1	16.0	4.3	2.1	13.7	5.1
instance_30b_04	2.4	13.1	3.0	2.1	12.5	4.0	2.2	12.4	4.3	2.0	12.9	5.7	2.2	12.4	4.3	2.0	12.9	5.7
instance_30b_05	2.6	14.8	3.8	1.9	12.4	5.5	3.1	15.2	3.9	2.2	13.1	5.8	3.1	15.2	3.9	2.2	13.1	5.8
instance_110_01	1.5	8.1	1.6	1.1	7.2	1.5	1.7	9.6	1.1	1.2	7.6	2.0	1.7	9.6	1.1	1.2	7.6	2.0
instance_110_02	1.7	9.6	1.1	1.2	7.6	2.0	1.8	11.2	2.1	1.3	8.5	2.7	1.8	11.2	2.1	1.3	8.5	2.7
instance_110_03	1.5	11.9	1.4	1.1	7.9	1.8	1.6	11.4	2.5	1.1	8.3	2.8	1.6	11.4	2.5	1.1	8.3	2.8
instance_110_04	1.6	10.5	1.5	1.1	6.8	1.8	1.7	10.6	1.1	1.2	7.6	2.0	1.7	10.6	1.1	1.2	7.6	2.0
instance_110_05	1.4	8.6	1.4	1.2	7.9	1.9	1.7	9.4	1.9	1.3	7.8	2.0	1.7	9.4	1.9	1.3	7.8	2.0
instance_f_01	5.5	19.0	1.7	5.2	18.1	2.0	5.1	18.3	2.9	4.3	16.8	3.9	5.1	18.3	2.9	4.3	16.8	3.9
instance_f_02	4.3	17.5	5.3	2.8	14.4	5.7	5.5	21.1	2.8	4.0	18.1	3.8	5.5	21.1	2.8	4.0	18.1	3.8
instance_f_03	3.0	14.7	5.8	2.4	14.3	6.5	3.6	14.7	3.1	3.5	16.3	3.8	3.6	14.7	3.1	3.5	16.3	3.8
instance_f_04	5.1	21.5	2.7	4.7	21.0	3.7	6.7	26.1	2.5	5.9	23.9	3.7	6.7	26.1	2.5	5.9	23.9	3.7
instance_f_05	4.6	19.7	4.8	3.6	17.4	4.8	4.0	18.8	6.4	3.5	17.7	6.4	4.0	18.8	6.4	3.5	17.7	6.4

6.4.4 Confronto funzioni obiettivo

In questa sezione si confrontano i risultati presentati precedentemente delle diverse funzioni obiettivo considerando solo il metodo euristico.

Come si può osservare nella figura 6.1, tutte le funzioni obiettivo presentano tempi di calcolo simili per quanto riguarda i primi tre gruppi di istanze (instance_09_xx, instance_30_xx, instance_30b_xx). Per le istanze più grandi del gruppo 4 e quelle che presentano story point in base alle sequenza di Fibonacci (instance_110_xx e instance_f_xx) si nota che le funzioni obiettivo (4.13) e (4.14) impiegano rempi maggiori, confermando il trend già osservato tra (4.1) e (4.10), e tra (4.11) e (4.12). Questa differenza di tempi tra le funzioni obiettivo (4.13) e (4.14) e le altre è probabilmente dovuta al fatto che gli algoritmi risultano in difficoltà a selezionare la combinazione di user story migliori confrontando la quantità di story point che richiedono.

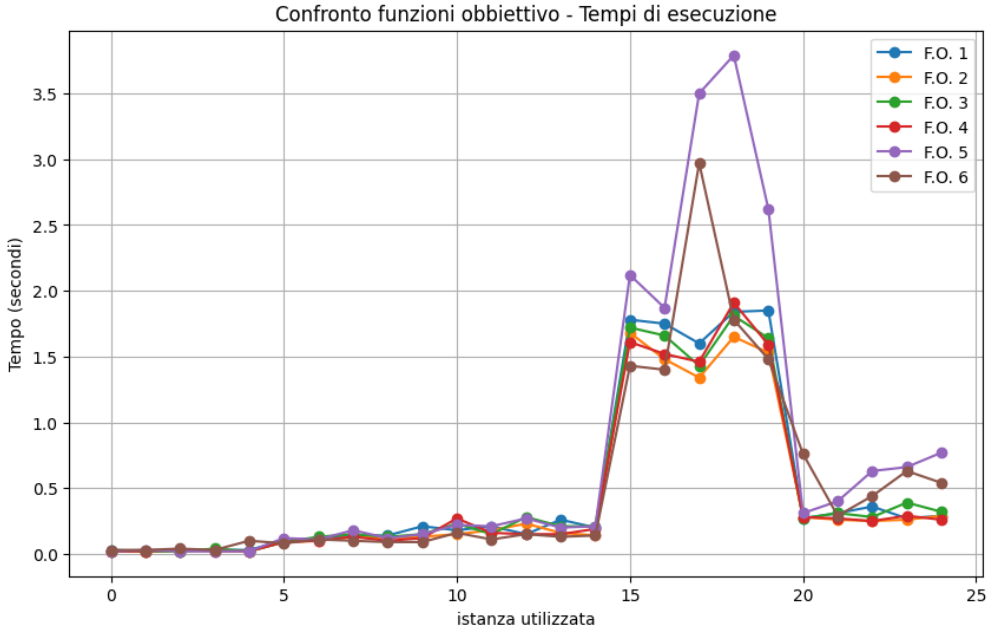

Figure 6.1: Tempo di calcolo per le diverse funzioni obiettivo

Analizzando la Figura 6.2, si nota che la percentuale di utilizzo degli sprint è simile per tutte le funzioni obiettivo. La funzione (4.14) si discosta più frequentemente dalle altre, ma solo per alcune istanze. Tale scostamento è dovuto alla capacità di questa funzione di utilizzare uno sprint in meno rispetto alle alternative: evitando sprint semi-vuoti, migliora la percentuale media di utilizzo nelle istanze favorevoli.

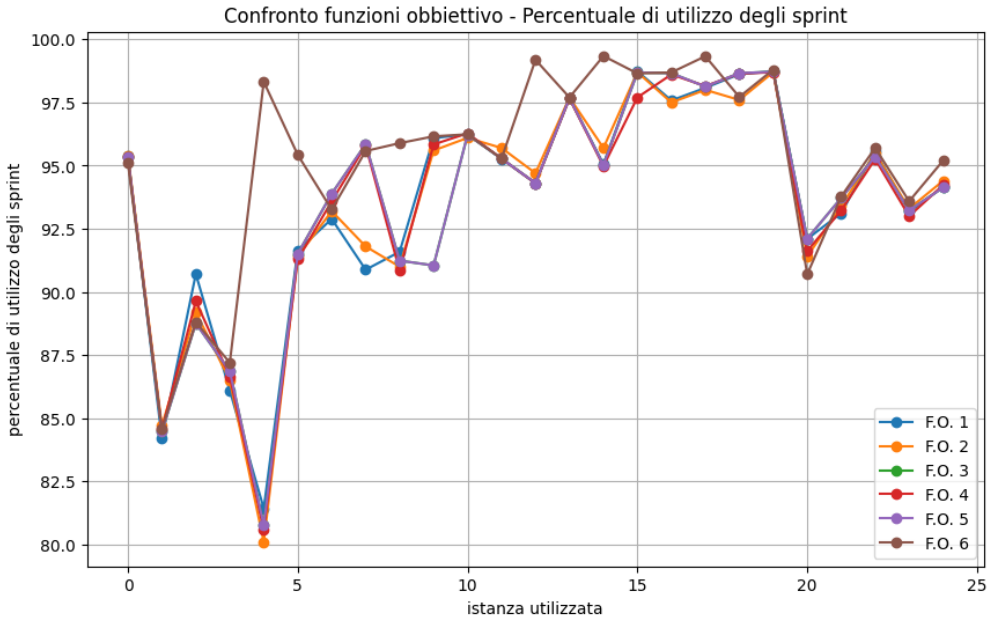

Figure 6.2: Tempo di calcolo per le diverse funzioni obiettivo

Nelle figure 6.3, 6.4, 6.5, 6.6 si può notare che in tutte e 4, le funzioni obiettivo (4.1), (4.11) e (4.13) risultano presentare valori simili tra loro e maggiori rispetto alle altre funzioni obiettivo, mentre la funzione obiettivo (4.14) risulta essere quella che presenta i valori più bassi in tutte e 4 le metriche considerate. Questo indica che la funzione obiettivo (4.14) è quella che riesce a bilanciare meglio il rischio e l'incertezza tra gli sprint rispetto alle altre funzioni obiettivo e che le funzioni obiettivo (4.10), (4.12) e (4.14) che presentano al loro interno i parametri dp_{ij} e ub_{ij}, risultano presentare valori minori rispetto alle altre, quindi riescono a distribuire meglio i rischi e le incertezze all'interno dei vari sprint. migliora questo paragrafo

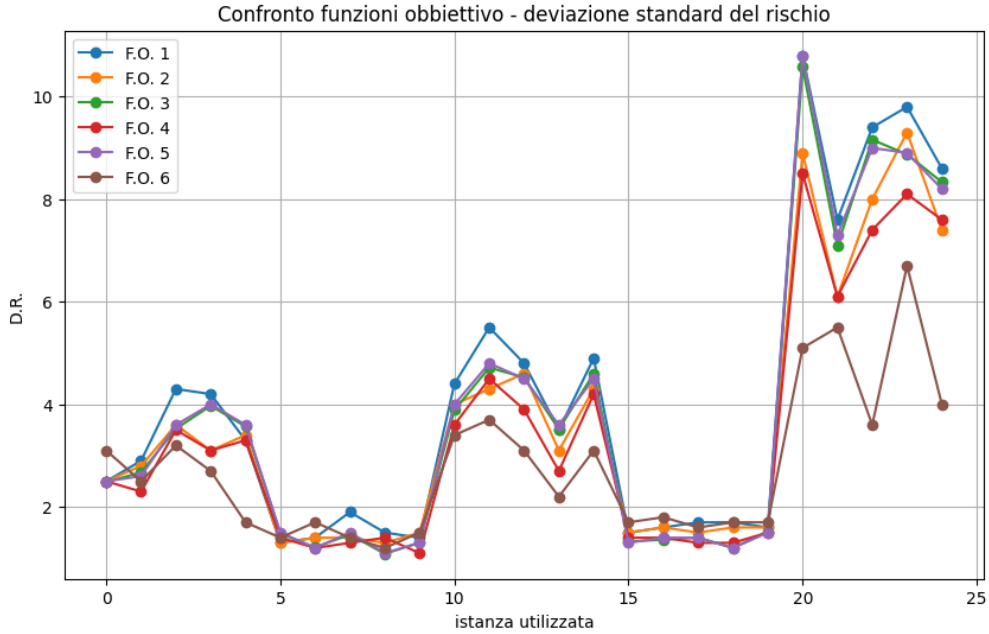

Figure 6.3: Deviazione standard del rischio per le diverse funzioni obiettivo

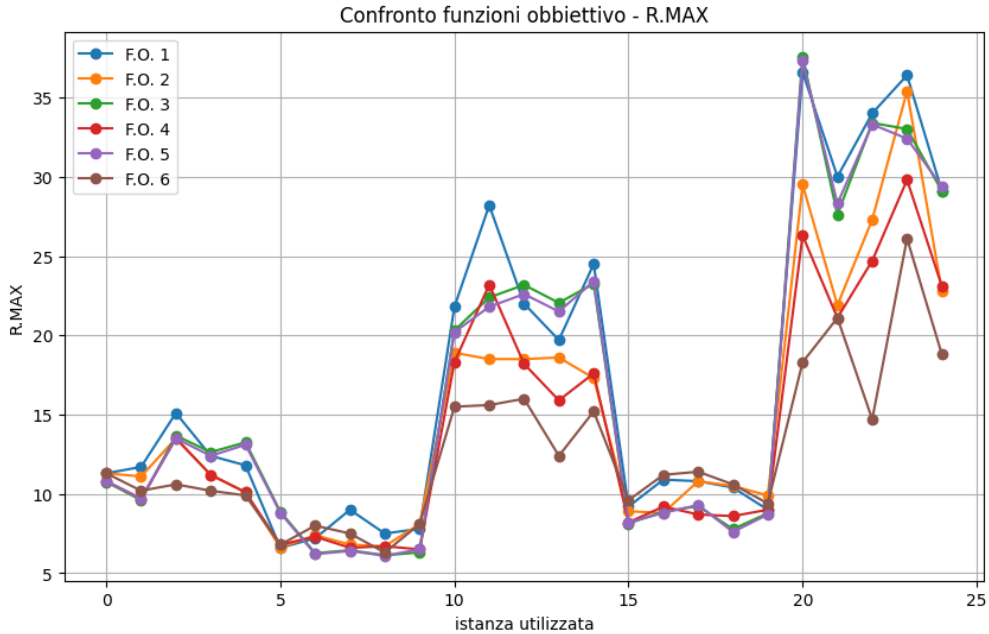

Figure 6.4: Valore massimo del rischio per le diverse funzioni obiettivo

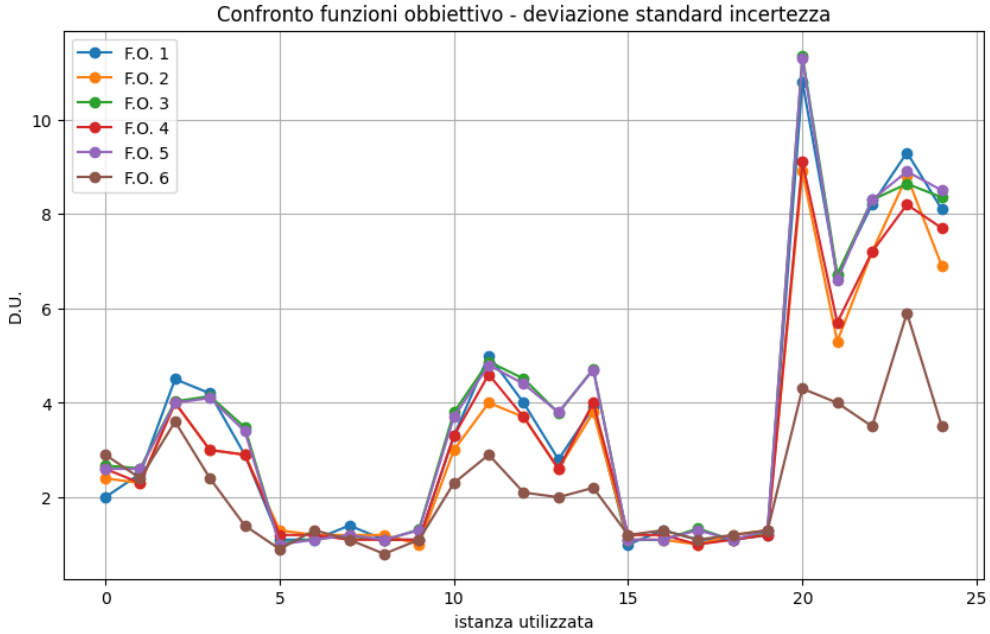

Figure 6.5: Deviazione standard dell'incertezza per le diverse funzioni obiettivo

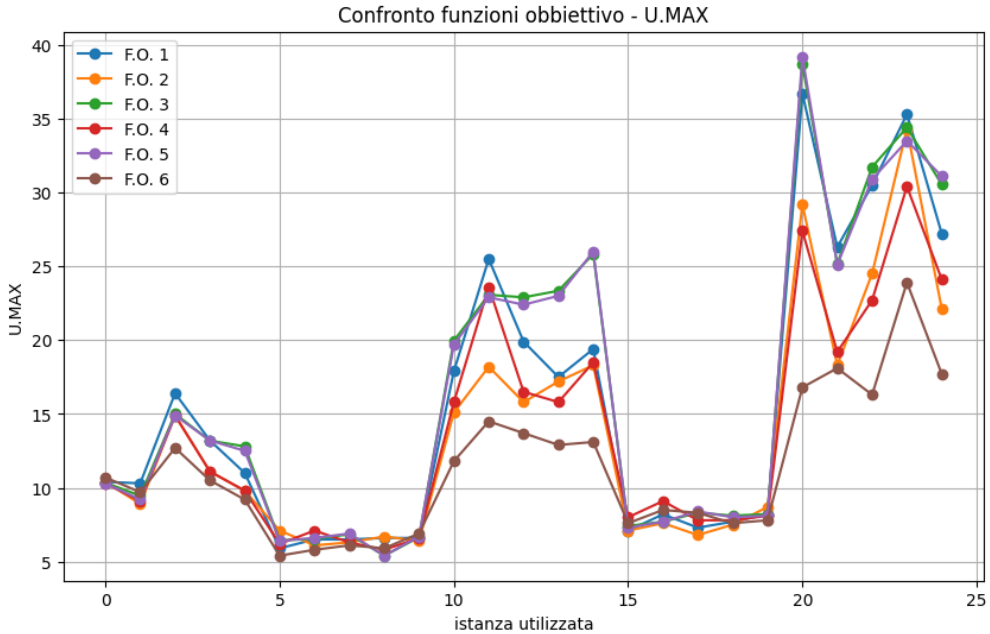

Figure 6.6: Valore massimo dell'incertezza per le diverse funzioni obiettivo

6.5 Variazione dei parametri

All'interno del progetto è stato condotto uno studio per valutare l'impatto di alcuni parametri sul progetto. In particolare, sono stati analizzati i seguenti aspetti:

- Effetto della riduzione del numero di sprint disponibili sul valore della soluzione e sui tempi di calcolo.
- La riduzione della capacità degli sprint e il suo impatto sulla fattibilità delle soluzioni.
- L'accorpamento di sprint una volta raggiunto un certo numero di sprint e il suo effetto sulla qualità della soluzione.
- Impatto del rilassamento del vincolo OR e AND tra le user story.

Per queste valutazioni si è scelto di utilizzare per tutti i test la funzione obiettivo (4.1) e per cui le considerazioni che seguono si basano sul confronto tra i risultati ottenuti modificando i parametri delle istanze e quelli ottenuti con i parametri originali riportati nelle tabelle 6.3, 6.4 e 6.5.

6.5.1 Riduzione del numero di sprint

La riduzione del numero di sprint disponibili indica una situazione in cui il team di sviluppo ha meno tempo a disposizione per completare il progetto, il che può portare a una maggiore pressione e a una maggiore difficoltà nel completare tutte le user story entro il tempo previsto se non all'impossibilità di completarlo. Nel nostro test è importante valutare come la riduzione del numero di sprint influisce sulla qualità della soluzione e sui tempi di calcolo. I test che sono stati condotti con una riduzione di 2 sprint rispetto al numero di sprint originale per ogni istanza, e i risultati ottenuti sono riportati nelle tabelle 6.19, 6.20 e 6.21. Si possono notare che le istanze *instance*_09_02, *instance*_09_03 e *instance*_110_05 presentano una soluzione con un valore della funzione obiettivo pari a $Zopt = 0$ per il metodo esatto; questo indica che per quelle istanze non si può avere una soluzione ammissibile riducendo il numero di sprint a disposizione, mentre le altre istanze si può notare che la riduzione del numero di sprint a disposizione non ha avuto nessun

impatto sulla soluzione ottenuta dai metodi euristici. Si può però notare una variazione nei tempi di calcolo, per il metodo esatto la riduzione del numero di sprint porta un risparmio del 61.01% di tempo di calcolo, mentre per il metodo euristico del 35.18% e per il metodo euristico Lagrangiano del 32.28%. Da quest'ultima considerazione si può notare come la riduzione del numero di sprint a disposizione porta a una diminuzione dei tempi di calcolo.

Table 6.19: KPI con riduzione di -2 sprint

	CPLEX		Heuristic		Lagrangian Heuristic	
Istanza	Zopt	Time	Zheu	Time	Zheu	Time
instance_09_01	23409.9	0.16	22686.6	0.02	22686.6	0.21
instance_09_02	0.0	0.01	21740.6	0.02	21434.2	0.32
instance_09_03	0.0	0.01	21159.2	0.02	20868.4	0.33
instance_09_04	18125.7	0.06	15175.4	0.03	15175.4	0.29
instance_09_05	18160.2	0.14	16450.1	0.04	16392.9	0.28
instance_30_01	125477.6	300.07	123352.1	0.09	123352.1	3.53
instance_30_02	143586.5	300.02	135481.2	0.09	135481.2	3.12
instance_30_03	134896.3	267.22	125447.2	0.11	125447.2	2.96
instance_30_04	119857.3	300.08	120783.4	0.11	120783.4	3.37
instance_30_05	134534.8	300.04	119839.7	0.16	119839.7	2.81
instance_30b_01	271716.6	300.12	267873.5	0.12	267873.5	8.49
instance_30b_02	287801.6	300.09	279010.9	0.15	279010.9	9.68
instance_30b_03	271547.9	300.11	268060.4	0.08	268060.4	11.46
instance_30b_04	247552.9	300.11	247839.4	0.23	247839.4	11.99
instance_30b_05	254136.5	300.13	259333.8	0.13	259333.8	11.88
instance_110_01	1920308.4	301.12	2050139.1	1.30	2050139.1	187.36
instance_110_02	1937904.6	300.86	2082565.4	1.16	2082565.4	225.93
instance_110_03	2011959.1	301.09	2086918.6	1.24	2086918.6	263.40
instance_110_04	2060624.0	300.91	2148900.6	1.20	2148900.6	250.08
instance_110_05	0.0	301.10	2015801.2	1.24	2015801.2	268.35
instance_f_01	129829.8	0.12	120411.2	0.05	120411.2	5.32
instance_f_02	134864.3	9.64	127857.5	0.07	127857.5	2.72
instance_f_03	131870.5	37.70	124444.8	0.06	124444.8	1.88
instance_f_04	130108.4	12.48	127565.4	0.06	127565.4	2.04
instance_f_05	137811.6	3.91	121055.0	0.06	121055.0	2.34

Table 6.20: KPI con riduzione di -2 sprint

	CPLEX			Heuristic			Lagrangian Heuristic		
Istanza	N.S.O.	A.S.U.	H.U.S.	N.S.O.	A.S.U.	H.U.S.	N.S.O.	A.S.U.	H.U.S.
instance_09_01	5	95.59%	2	5	95.38%	2	5	95.38%	2
instance_09_02	0	0.00%	None	5	95.07%	2	5	95.07%	2
instance_09_03	0	0.00%	None	5	92.43%	2	5	92.43%	2
instance_09_04	5	86.65%	2	5	86.10%	2	5	86.10%	2
instance_09_05	5	98.34%	2	5	95.65%	2	5	95.65%	2
instance_30_01	23	96.09%	9	24	91.64%	8	24	91.64%	8
instance_30_02	21	93.09%	7	21	92.89%	6	21	92.89%	6
instance_30_03	21	95.87%	8	22	90.88%	7	22	90.88%	7
instance_30_04	23	95.91%	8	24	91.59%	8	24	91.59%	8
instance_30_05	21	96.02%	7	21	96.09%	7	21	96.09%	7
instance_30b_01	21	96.16%	6	21	96.24%	6	21	96.24%	6
instance_30b_02	19	95.31%	5	19	95.27%	5	19	95.27%	5
instance_30b_03	20	99.19%	6	21	94.30%	5	21	94.30%	5
instance_30b_04	22	97.61%	8	22	97.67%	6	22	97.67%	6
instance_30b_05	23	95.05%	6	23	95.06%	6	23	95.06%	6
instance_110_01	103	91.19%	31	95	98.74%	29	95	98.74%	29
instance_110_02	94	91.66%	33	88	97.58%	29	88	97.58%	29
instance_110_03	90	94.30%	28	87	98.08%	27	87	98.08%	27
instance_110_04	98	92.95%	33	92	98.62%	28	92	98.62%	28
instance_110_05	0	0.00%	None	90	98.73%	27	90	98.73%	27
instance_f_01	9	92.44%	2	9	92.10%	2	9	92.10%	2
instance_f_02	10	93.61%	3	10	93.12%	2	10	93.12%	2
instance_f_03	9	95.66%	2	9	95.57%	2	9	95.57%	2
instance_f_04	10	93.14%	2	10	93.17%	2	10	93.17%	2
instance_f_05	9	95.09%	3	9	94.16%	2	9	94.16%	2

Table 6.21: KPI con riduzione di -2 sprint

	CPLEX						Heuristic						Lagrangian Heuristic					
Istanza	D.R.	R.MAX	R.MIN	D.U.	U.MAX	U.MIN	D.R.	R.MAX	R.MIN	D.U.	U.MAX	U.MIN	D.R.	R.MAX	R.MIN	D.U.	U.MAX	U.MIN
instance_09_01	2.2	10.6	5.1	2.1	10.1	4.3	2.5	11.3	4.7	2.0	10.4	4.9	2.5	11.3	4.7	2.0	10.4	4.9
instance_09_02	0.0	0.0	0.0	0.0	0.0	0.0	2.6	11.7	4.7	2.0	10.3	5.0	2.6	11.7	4.7	2.0	10.3	5.0
instance_09_03	0.6	4.8	3.5	0.7	4.7	3.2	4.5	15.1	2.3	4.7	16.4	3.6	4.5	15.1	2.3	4.7	16.4	3.6
instance_09_04	3.1	11.2	3.0	3.0	11.1	3.4	4.2	12.4	2.6	4.2	13.2	2.7	4.2	12.4	2.6	4.2	13.2	2.7
instance_09_05	2.0	10.1	4.0	2.1	9.8	3.8	2.8	11.8	3.8	2.2	11.0	4.7	2.8	11.8	3.8	2.2	11.0	4.7
instance_30_01	1.1	6.4	2.6	1.0	5.9	2.7	1.3	6.6	1.1	1.1	5.9	1.4	1.3	6.6	1.1	1.1	5.9	1.4
instance_30_02	1.5	7.6	1.6	1.1	6.1	1.2	1.4	7.2	1.4	1.1	6.5	2.0	1.4	7.2	1.4	1.1	6.5	2.0
instance_30_03	1.5	6.5	1.2	1.1	5.8	1.7	1.9	9.0	1.0	1.4	6.5	1.7	1.9	9.0	1.0	1.4	6.5	1.7
instance_30_04	1.3	8.1	2.1	1.0	6.9	2.7	1.5	7.5	1.1	1.1	6.6	1.7	1.5	7.5	1.1	1.1	6.6	1.7
instance_30_05	1.2	6.7	2.3	1.0	6.4	2.6	1.4	7.8	2.4	1.1	6.6	2.2	1.4	7.8	2.4	1.1	6.6	2.2
instance_30b_01	4.1	20.0	2.7	3.0	15.5	3.3	4.4	21.8	2.7	3.3	17.9	3.4	4.4	21.8	2.7	3.3	17.9	3.4
instance_30b_02	4.4	19.9	1.7	4.0	19.2	1.5	5.5	28.2	1.4	5.0	25.5	1.6	5.5	28.2	1.4	5.0	25.5	1.6
instance_30b_03	3.7	20.3	3.7	2.9	17.1	5.1	4.8	22.0	1.5	4.0	19.9	1.6	4.8	22.0	1.5	4.0	19.9	1.6
instance_30b_04	3.3	17.4	4.8	2.7	17.5	4.7	3.5	19.7	3.3	2.8	17.5	4.5	3.5	19.7	3.3	2.8	17.5	4.5
instance_30b_05	4.6	24.4	1.3	4.0	21.7	1.5	4.9	24.5	1.0	3.9	19.4	2.0	4.9	24.5	1.0	3.9	19.4	2.0
instance_110_01	1.6	10.4	1.2	1.1	7.4	1.7	1.5	9.2	2.2	1.0	7.1	2.7	1.5	9.2	2.2	1.0	7.1	2.7
instance_110_02	1.8	11.2	1.1	1.3	7.7	1.5	1.6	10.9	1.1	1.3	8.2	1.8	1.6	10.9	1.1	1.3	8.2	1.8
instance_110_03	1.6	9.9	1.1	1.1	7.2	1.2	1.7	10.8	1.1	1.1	7.3	1.8	1.7	10.8	1.1	1.1	7.3	1.8
instance_110_04	1.5	10.7	1.2	1.2	7.0	1.8	1.7	10.4	2.1	1.1	7.7	2.7	1.7	10.4	2.1	1.1	7.7	2.7
instance_110_05	0.0	0.0	0.0	0.0	0.0	0.0	1.6	9.0	2.3	1.2	8.2	2.8	1.6	9.0	2.3	1.2	8.2	2.8
instance_f_01	8.5	26.7	1.8	8.2	25.7	1.7	10.8	36.6	1.5	10.8	36.7	1.7	10.8	36.6	1.5	10.8	36.7	1.7
instance_f_02	5.6	19.7	2.4	4.6	17.2	2.8	7.6	30.0	3.2	6.7	26.3	2.9	7.6	30.0	3.2	6.7	26.3	2.9
instance_f_03	7.6	25.3	2.3	6.8	22.7	2.7	9.4	34.0	2.3	8.2	30.5	3.4	9.4	34.0	2.3	8.2	30.5	3.4
instance_f_04	8.7	33.2	2.3	8.2	31.8	3.3	9.8	36.4	2.5	9.3	35.3	2.5	9.8	36.4	2.5	9.3	35.3	2.5
instance_f_05	6.0	21.8	2.7	5.5	20.7	3.2	8.6	29.1	2.7	8.1	27.2	3.0	8.6	29.1	2.7	8.1	27.2	3.0

6.5.2 Riduzione della capacità degli sprint

In un progetto reale, è possibile che la capacità degli sprint venga ridotta a causa di imprevisti, come ad esempio malattie, ferie o altri impegni dei membri del team. Questo porta a un aumento dei tempi di completamento del progetto e in alcuni casi l'impossibilità al completamento nei tempi previsti. Per valutare l'impatto di una riduzione della capacità degli sprint sui KPI, abbiamo condotto un esperimento in cui abbiamo ridotto la capacità degli sprint del 20% i cui risultati si trovano nelle tabelle 6.22, 6.23 e 6.24. Si può notare come la riduzione della capacità dello sprint ha portato anche a una riduzione del valore della funzione obiettivo del 8% per il metodo esatto e del 6.1% per i metodi euristici si nota anche che il gruppo di istanze più grandi non è in grado di essere risolto a causa dell'insufficiente quantità di story point a disposizione. Il KPI N.S.O. è aumentato del 26.59% per il metodo esatto e del 25.08% per i metodi euristici, mentre per il KPI H.U.S. vi è stato un aumento del 26.67% per il metodo esatto e del 15.51% per i metodi euristici. Questo indicando come era presumibile che la riduzione della capacità degli sprint ha portato ad un aumento del numero degli sprint necessari a completare il progetto.

Table 6.22: KPI con riduzione del 20% della capacita degli sprint

	CPLEX		Heuristic		Lagrangian Heuristic	
Istanza	Zopt	Time	Zheu	Time	Zheu	Time
instance_09_01	22272.3	0.16	20464.9	0.04	20464.9	0.18
instance_09_02	20329.6	0.52	20258.6	0.03	20258.6	0.28
instance_09_03	19788.9	0.61	19533.2	0.02	19533.2	0.19
instance_09_04	16492.0	0.32	15535.9	0.02	15535.9	0.20
instance_09_05	16924.4	0.41	15655.5	0.03	15655.5	0.12
instance_30_01	111229.0	300.09	111563.6	0.10	111563.6	1.82
instance_30_02	124759.6	300.07	127029.1	0.11	127029.1	2.61
instance_30_03	122446.4	300.11	121500.3	0.12	121500.3	2.55
instance_30_04	107280.6	300.07	107385.3	0.11	107385.3	1.97
instance_30_05	116941.2	300.06	114602.8	0.11	114602.8	2.32
instance_30b_01	255918.1	300.13	250412.3	0.14	250412.3	10.00
instance_30b_02	271364.9	300.09	259817.7	0.17	259817.7	9.94
instance_30b_03	254136.9	300.10	249184.5	0.15	249184.5	9.19
instance_30b_04	225431.3	300.10	230712.1	0.17	230712.1	9.02
instance_30b_05	235408.7	300.12	237726.0	0.14	237726.0	13.25
instance_110_01	0.0	1.91	1821352.4	1.32	1821352.4	121.78
instance_110_02	0.0	300.89	1886633.6	1.09	1886633.6	141.92
instance_110_03	0.0	302.38	1889067.2	1.17	1889067.2	192.12
instance_110_04	0.0	1.67	1940857.4	1.14	1940857.4	162.54
instance_110_05	0.0	301.18	1825225.6	1.19	1825225.6	174.35
instance_f_01	124678.1	33.01	115934.6	0.08	115934.6	5.09
instance_f_02	128333.6	300.08	120922.7	0.09	120922.7	4.52
instance_f_03	128739.8	50.27	121844.7	0.07	121844.7	1.41
instance_f_04	127241.2	300.10	123545.8	0.08	123545.8	1.23
instance_f_05	134046.1	83.93	118536.2	0.07	118536.2	3.58

Table 6.23: KPI con riduzione del 20% della capacita degli sprint

	CPLEX			Heuristic			Lagrangian Heuristic		
Istanza	N.S.O.	A.S.U.	H.U.S.	N.S.O.	A.S.U.	H.U.S.	N.S.O.	A.S.U.	H.U.S.
instance_09_01	7	85.22%	3	7	85.07%	2	7	85.07%	2
instance_09_02	8	90.90%	3	8	90.46%	3	8	90.46%	3
instance_09_03	8	86.48%	2	8	87.12%	2	8	87.12%	2
instance_09_04	6	89.39%	2	6	89.41%	2	6	89.41%	2
instance_09_05	7	83.74%	3	6	97.73%	2	6	97.73%	2
instance_30_01	30	92.59%	11	30	92.29%	9	30	92.29%	9
instance_30_02	27	92.67%	8	27	92.80%	8	27	92.80%	8
instance_30_03	27	94.15%	9	28	90.63%	8	28	90.63%	8
instance_30_04	29	93.67%	10	29	93.51%	9	29	93.51%	9
instance_30_05	27	91.80%	10	26	94.70%	8	26	94.70%	8
instance_30b_01	26	98.13%	7	27	94.77%	7	27	94.77%	7
instance_30b_02	24	96.66%	7	24	96.64%	5	24	96.64%	5
instance_30b_03	25	97.36%	8	25	97.36%	7	25	97.36%	7
instance_30b_04	28	94.16%	10	27	97.84%	8	27	97.84%	8
instance_30b_05	28	97.18%	8	28	97.26%	8	28	97.26%	8
instance_110_01	0	0.00%	None	110	98.63%	35	110	98.63%	35
instance_110_02	0	0.00%	None	110	96.88%	35	110	96.88%	35
instance_110_03	0	0.00%	None	109	97.68%	34	109	97.68%	34
instance_110_04	0	0.00%	None	110	98.55%	35	110	98.55%	35
instance_110_05	0	0.00%	None	110	98.56%	36	110	98.56%	36
instance_f_01	11	95.90%	3	12	87.65%	2	12	87.65%	2
instance_f_02	12	97.21%	3	12	97.33%	3	12	97.33%	3
instance_f_03	12	94.47%	3	13	86.31%	2	13	86.31%	2
instance_f_04	12	96.92%	2	13	89.74%	2	13	89.74%	2
instance_f_05	11	93.77%	4	11	93.54%	2	11	93.54%	2

Table 6.24: KPI con riduzione del 20% della capacita degli sprint

	CPLEX						Heuristic						Lagrangian Heuristic					
Istanza	D.R.	R.MAX	R.MIN	D.U.	U.MAX	U.MIN	D.R.	R.MAX	R.MIN	D.U.	U.MAX	U.MIN	D.R.	R.MAX	R.MIN	D.U.	U.MAX	U.MIN
instance_09_01	2.8	9.6	1.6	2.5	9.3	1.6	2.9	11.6	1.6	2.3	10.0	2.0	2.9	11.6	1.6	2.3	10.0	2.0
instance_09_02	1.2	6.6	3.1	1.2	6.3	2.7	2.3	10.5	3.0	1.9	9.1	3.1	2.3	10.5	3.0	1.9	9.1	3.1
instance_09_03	2.2	8.6	1.9	2.3	8.5	1.7	3.7	13.9	1.9	4.0	15.2	1.7	3.7	13.9	1.9	4.0	15.2	1.7
instance_09_04	2.0	9.0	2.7	1.9	9.5	3.2	3.3	12.1	2.6	3.1	12.4	2.7	3.3	12.1	2.6	3.1	12.4	2.7
instance_09_05	2.6	9.3	1.1	2.4	8.9	1.1	2.4	10.3	3.9	2.0	9.8	4.0	2.4	10.3	3.9	2.0	9.8	4.0
instance_30_01	1.1	6.7	1.1	0.9	5.8	1.4	1.2	5.5	1.1	0.9	4.8	1.7	1.2	5.5	1.1	0.9	4.8	1.7
instance_30_02	1.2	5.7	1.3	0.9	5.0	1.7	1.2	5.4	1.5	1.0	5.1	1.2	1.2	5.4	1.5	1.0	5.1	1.2
instance_30_03	1.4	6.7	1.0	1.0	5.7	1.7	1.5	7.2	1.0	1.1	4.9	1.3	1.5	7.2	1.0	1.1	4.9	1.3
instance_30_04	0.9	5.0	1.3	1.0	5.5	1.4	1.2	6.7	1.0	1.0	5.8	1.7	1.2	6.7	1.0	1.0	5.8	1.7
instance_30_05	1.1	6.9	1.5	0.9	5.5	1.7	1.2	8.1	1.8	0.8	5.8	1.9	1.2	8.1	1.8	0.8	5.8	1.9
instance_30b_01	3.5	16.6	2.7	2.7	15.0	3.4	3.6	17.0	1.7	2.7	14.1	1.9	3.6	17.0	1.7	2.7	14.1	1.9
instance_30b_02	3.7	19.5	3.1	3.4	17.8	2.5	4.0	22.0	3.2	3.6	20.4	3.0	4.0	22.0	3.2	3.6	20.4	3.0
instance_30b_03	3.3	18.4	2.9	2.6	15.4	3.4	3.8	18.7	2.7	3.2	17.4	3.0	3.8	18.7	2.7	3.2	17.4	3.0
instance_30b_04	2.7	17.9	1.1	2.4	16.8	1.5	2.7	16.2	3.3	2.3	14.8	4.3	2.7	16.2	3.3	2.3	14.8	4.3
instance_30b_05	3.7	20.9	2.6	3.2	18.7	3.1	4.2	24.2	2.3	3.4	19.7	3.4	4.2	24.2	2.3	3.4	19.7	3.4
instance_110_01	0.0	0.0	0.0	0.0	0.0	0.0	1.2	7.5	1.1	0.8	5.8	2.0	1.2	7.5	1.1	0.8	5.8	2.0
instance_110_02	0.0	0.0	0.0	0.0	0.0	0.0	1.4	9.1	1.1	0.9	5.9	1.5	1.4	9.1	1.1	0.9	5.9	1.5
instance_110_03	0.0	0.0	0.0	0.0	0.0	0.0	1.4	9.2	1.2	1.0	6.6	1.5	1.4	9.2	1.2	1.0	6.6	1.5
instance_110_04	0.0	0.0	0.0	0.0	0.0	0.0	1.3	9.0	2.2	0.8	6.0	2.3	1.3	9.0	2.2	0.8	6.0	2.3
instance_110_05	0.0	0.0	0.0	0.0	0.0	0.0	1.3	7.3	2.1	1.0	6.0	2.5	1.3	7.3	2.1	1.0	6.0	2.5
instance_f_01	6.4	21.7	1.8	6.2	22.2	1.7	8.7	32.2	1.6	8.5	31.6	1.5	8.7	32.2	1.6	8.5	31.6	1.5
instance_f_02	4.6	17.6	1.7	3.7	15.8	2.0	6.2	25.9	2.5	5.3	22.4	2.5	6.2	25.9	2.5	5.3	22.4	2.5
instance_f_03	7.2	28.6	1.3	6.4	25.4	1.2	7.4	27.3	1.0	6.7	24.7	1.1	7.4	27.3	1.0	6.7	24.7	1.1
instance_f_04	6.7	26.9	1.3	6.5	25.8	1.9	7.2	27.4	1.0	6.9	27.2	1.6	7.2	27.4	1.0	6.9	27.2	1.6
instance_f_05	6.0	21.3	1.4	5.7	19.6	1.8	6.7	24.0	2.6	6.3	23.1	2.7	6.7	24.0	2.6	6.3	23.1	2.7

6.5.3 Accorpamento di sprint

L'accorpamento di sprint consiste nell'unire più sprint in uno solo, riducendo così il numero totale di sprint e, di conseguenza, le riunioni di pianificazione e revisione. Questa pratica si rivela utile in due scenari principali:

- Dopo una fase iniziale del progetto, se si rileva che gli sprint sono troppo brevi per massimizzare l'efficacia del team.
- Quando il progetto, post-fase iniziale, appare sufficientemente stabile da permettere l'accorpamento senza rischi di modifiche significative al piano complessivo.

In questo modo, si ottimizza il flusso di lavoro mantenendo flessibilità. Per i test computazionali, si è deciso di accorpare gli sprint in gruppi di 2 a partire dalla metà del numero totale di sprint. Ad esempio, in un progetto di 8 sprint, si accorpano gli sprint 5 e 6 (ottenendo un nuovo sprint 5 con capacità $c_5 = c_5 + c_6$) e gli sprint 7 e 8 (ottenendo un nuovo sprint 6 con capacità $c_6 = c_7 + c_8$). I risultati dei test si trovano nelle tabelle 6.25, 6.26 e 6.27. Si può notare che l'accorpamento degli sprint non ha impattato sul valore ottenuto dalla funzione obiettivo rispetto alla

versione base; questo è dovuto al fatto che l'accorpamento non ha modificato la capacità totale disponibile, ma solo la distribuzione della stessa tra gli sprint. Vi sono però differenze sui KPI D.R. e D.U. che hanno avuto un aumento. Per quanto riguarda D.R. per la soluzione esatta l'aumento è stato del 2.59% mentre per le soluzioni euristiche è stato del 8.83%. Per quanto riguarda D.U. per la soluzione esatta l'aumento è stato del 3.11% mentre per le soluzioni euristiche è stato del 11.36%. Lo stesso trend si nota anche nei KPI R.MAX e U.MAX, con un aumento rispettivamente del 9.75% e del 11.02% per le soluzioni euristiche, mentre per la soluzione esatta l'aumento è stato del 3.01% per R.MAX e del 3.46% per U.MAX. Questo aumento è dovuto dal fatto che vi sono sprint più grandi che permettono di allocare più lavoro in un singolo sprint, ma questo comporta anche un aumento del rischio e dell'incertezza complessiva dello sprint.

Table 6.25: KPI con accorpamento di sprint

	CPLEX		Heuristic		Lagrangian Heuristic	
Istanza	Zopt	Time	Zheu	Time	Zheu	Time
instance_09_01	23409.9	0.22	22686.6	0.20	22686.6	0.31
instance_09_02	22074.6	0.36	21762.2	0.15	21762.2	0.46
instance_09_03	21546.1	0.20	21159.2	0.15	21159.2	0.42
instance_09_04	18125.7	0.13	15175.4	0.22	15175.4	0.33
instance_09_05	18160.2	0.35	16450.1	0.17	16450.1	0.31
instance_30_01	126054.2	300.10	123952.2	0.18	123952.2	3.07
instance_30_02	143748.2	241.66	135636.2	0.22	135636.2	4.13
instance_30_03	135112.3	176.26	125652.1	0.24	125652.1	2.85
instance_30_04	120546.9	113.24	121250.2	0.24	121250.2	2.93
instance_30_05	135370.7	300.03	120064.6	0.35	120064.6	2.78
instance_30b_01	271959.6	300.15	268115.1	0.34	268115.1	9.61
instance_30b_02	287751.8	300.12	279033.3	0.30	279033.3	6.14
instance_30b_03	271740.6	300.14	268235.7	0.35	268235.7	7.13
instance_30b_04	248591.0	300.13	248485.4	0.34	248485.4	9.25
instance_30b_05	255188.4	300.14	260002.3	0.35	260002.3	9.01
instance_110_01	1971326.1	301.28	2084806.1	1.46	2084806.1	172.64
instance_110_02	1953259.9	300.92	2102288.1	1.26	2102288.1	224.27
instance_110_03	2042298.7	301.03	2109849.9	1.20	2109849.9	219.81
instance_110_04	2089579.2	300.95	2176941.3	1.21	2176941.3	206.79
instance_110_05	1972880.6	301.11	2044539.5	1.33	2044539.5	244.09
instance_f_01	128597.5	0.35	120411.2	0.32	120411.2	4.07
instance_f_02	134864.3	10.20	127857.5	0.35	127857.5	3.17
instance_f_03	131870.5	24.00	124444.8	0.37	124444.8	2.05
instance_f_04	130108.4	16.82	127565.4	0.54	127565.4	1.98
instance_f_05	137811.6	6.59	121055.0	0.40	121055.0	3.73

Table 6.26: KPI con accorpamento di sprint

	CPLEX			Heuristic			Lagrangian Heuristic		
Istanza	N.S.O.	A.S.U.	H.U.S.	N.S.O.	A.S.U.	H.U.S.	N.S.O.	A.S.U.	H.U.S.
instance_09_01	5	95.59%	2	5	95.38%	2	5	95.38%	2
instance_09_02	6	87.74%	2	6	87.09%	2	6	87.09%	2
instance_09_03	6	84.78%	2	6	82.53%	2	6	82.53%	2
instance_09_04	5	86.65%	2	5	86.10%	2	5	86.10%	2
instance_09_05	5	98.34%	2	6	80.69%	2	6	80.69%	2
instance_30_01	20	97.00%	9	21	91.08%	8	21	91.08%	8
instance_30_02	19	96.65%	7	19	95.99%	6	19	95.99%	6
instance_30_03	19	97.07%	8	19	96.35%	7	19	96.35%	7
instance_30_04	20	96.53%	9	20	95.03%	8	20	95.03%	8
instance_30_05	19	97.15%	8	19	96.54%	7	19	96.54%	7
instance_30b_01	19	97.87%	6	19	97.67%	6	19	97.67%	6
instance_30b_02	18	97.11%	5	18	97.14%	5	18	97.14%	5
instance_30b_03	19	95.77%	6	19	96.01%	5	19	96.01%	5
instance_30b_04	20	94.67%	7	20	95.61%	6	20	95.61%	6
instance_30b_05	20	95.98%	7	20	96.40%	6	20	96.40%	6
instance_110_01	78	94.32%	34	76	98.68%	29	76	98.68%	29
instance_110_02	75	94.04%	32	72	98.86%	29	72	98.86%	29
instance_110_03	73	94.83%	30	72	98.55%	27	72	98.55%	27
instance_110_04	77	94.06%	30	75	98.28%	28	75	98.28%	28
instance_110_05	77	91.33%	30	74	97.95%	27	74	97.95%	27
instance_f_01	9	92.98%	2	9	92.10%	2	9	92.10%	2
instance_f_02	10	93.61%	3	10	93.12%	2	10	93.12%	2
instance_f_03	9	95.66%	2	9	95.57%	2	9	95.57%	2
instance_f_04	10	93.14%	2	10	93.17%	2	10	93.17%	2
instance_f_05	9	95.09%	3	9	94.16%	2	9	94.16%	2

Table 6.27: KPI con accorpamento di sprint

	CPLEX						Heuristic						Lagrangian Heuristic					
Istanza	D.R.	R.MAX	R.MIN	D.U.	U.MAX	U.MIN	D.R.	R.MAX	R.MIN	D.U.	U.MAX	U.MIN	D.R.	R.MAX	R.MIN	D.U.	U.MAX	U.MIN
instance_09_01	2.2	10.6	5.1	2.1	10.1	4.3	2.5	11.3	4.7	2.0	10.4	4.9	2.5	11.3	4.7	2.0	10.4	4.9
instance_09_02	2.1	9.6	4.5	1.6	8.9	4.5	2.4	11.7	4.7	1.9	10.3	5.0	2.4	11.7	4.7	1.9	10.3	5.0
instance_09_03	3.6	13.5	1.9	4.3	15.7	1.7	4.3	15.1	2.3	4.5	16.4	3.5	4.3	15.1	2.3	4.5	16.4	3.5
instance_09_04	3.1	11.2	3.0	3.0	11.1	3.4	4.2	12.4	2.6	4.2	13.2	2.7	4.2	12.4	2.6	4.2	13.2	2.7
instance_09_05	2.0	10.1	4.0	2.1	9.8	3.8	3.3	11.8	1.1	2.9	11.0	1.1	3.3	11.8	1.1	2.9	11.0	1.1
instance_30_01	1.4	7.5	2.7	1.4	7.6	2.7	1.2	6.6	1.3	1.4	6.8	1.4	1.2	6.6	1.3	1.4	6.8	1.4
instance_30_02	1.1	7.6	2.5	0.9	6.7	3.1	1.2	7.2	2.9	1.2	6.9	2.8	1.2	7.2	2.9	1.2	6.9	2.8
instance_30_03	1.3	6.5	2.1	1.1	6.5	2.8	1.5	9.0	2.4	1.2	7.3	2.5	1.5	9.0	2.4	1.2	7.3	2.5
instance_30_04	1.4	7.8	2.8	1.6	8.7	2.8	1.5	7.5	2.5	1.7	9.6	2.7	1.5	7.5	2.5	1.7	9.6	2.7
instance_30_05	1.4	7.1	2.3	1.3	8.4	2.8	1.4	7.8	2.4	1.3	7.9	2.2	1.4	7.8	2.4	1.3	7.9	2.2
instance_30b_01	3.7	20.0	5.9	2.6	15.5	6.2	4.1	21.8	4.3	3.0	17.9	5.4	4.1	21.8	4.3	3.0	17.9	5.4
instance_30b_02	4.1	19.8	6.0	3.8	19.3	5.6	5.3	28.2	4.9	4.7	25.5	5.3	5.3	28.2	4.9	4.7	25.5	5.3
instance_30b_03	3.7	20.3	5.5	2.8	17.1	5.6	4.7	22.0	3.2	3.7	19.9	5.0	4.7	22.0	3.2	3.7	19.9	5.0
instance_30b_04	3.7	19.0	4.8	3.2	18.4	5.8	4.0	19.7	3.4	3.2	17.5	5.3	4.0	19.7	3.4	3.2	17.5	5.3
instance_30b_05	5.0	24.4	3.2	4.4	21.7	4.6	5.1	24.5	3.5	4.2	19.4	5.0	5.1	24.5	3.5	4.2	19.4	5.0
instance_110_01	2.0	11.5	2.6	2.2	13.7	2.6	2.1	11.4	2.3	2.1	11.8	3.1	2.1	11.4	2.3	2.1	11.8	3.1
instance_110_02	2.2	13.4	1.7	2.4	12.9	1.5	2.3	13.2	2.3	2.4	13.2	2.7	2.3	13.2	2.3	2.4	13.2	2.7
instance_110_03	2.2	10.9	2.6	2.4	11.7	2.8	2.2	11.2	2.5	2.3	13.4	2.7	2.2	11.2	2.5	2.3	13.4	2.7
instance_110_04	2.3	12.2	2.5	2.4	11.9	2.4	2.5	13.0	1.1	2.4	12.7	2.0	2.5	13.0	1.1	2.4	12.7	2.0
instance_110_05	2.3	12.2	2.2	2.4	12.4	2.1	2.4	12.2	1.5	2.5	12.7	1.9	2.4	12.2	1.5	2.5	12.7	1.9
instance_f_01	8.8	28.9	1.6	8.4	28.2	2.0	10.8	36.6	1.5	10.8	36.7	1.7	10.8	36.6	1.5	10.8	36.7	1.7
instance_f_02	5.6	19.7	2.4	4.6	17.2	2.8	7.6	30.0	3.2	6.7	26.3	2.9	7.6	30.0	3.2	6.7	26.3	2.9
instance_f_03	7.6	25.3	2.3	6.8	22.7	2.7	9.4	34.0	2.3	8.2	30.5	3.4	9.4	34.0	2.3	8.2	30.5	3.4
instance_f_04	8.7	33.2	2.3	8.2	31.8	3.3	9.8	36.4	2.5	9.3	35.3	2.5	9.8	36.4	2.5	9.3	35.3	2.5
instance_f_05	6.0	21.8	2.7	5.5	20.7	3.2	8.6	29.1	2.7	8.1	27.2	3.0	8.6	29.1	2.7	8.1	27.2	3.0

6.5.4 Eliminazione vincoli AND e OR

L'eliminazione dei vincoli AND e OR in un progetto reale non sarebbe possibile, visto che risulta impossibile eliminare tutte le dipendenze tra le attività. Tuttavia si è voluto testare se questi vincoli incidono sulla soluzione e se sì in quale misura. I risultati ottenuti sono riportati nelle tabelle 6.28, 6.29 e 6.30. Si può notare che i risultati ottenuti risultano essere migliori rispetto a quelli ottenuti con i vincoli AND e OR, il valore della funzione obiettivo è aumentato del 42.65% per il metodo esatto e del 41.91% per i metodi euristici. I tempi di esecuzione sono diminuiti drasticamente, con un tempo medio di esecuzione diminuito del 61.81% per il metodo esatto, del 6.42% per il metodo euristico e del 67.11% per il metodo lagrangiano. Il miglioramento del valore della funzione obiettivo è dovuto al fatto che l'eliminazione dei vincoli permette di avere una maggiore flessibilità nella schedule delle attività, consentendo di sfruttare meglio le risorse disponibili e di ridurre i tempi di completamento del progetto. Inoltre, l'eliminazione dei vincoli AND e OR consente di ridurre il tempo di esecuzione perché una qualsiasi soluzione che soddisfa i vincoli sulla capacità degli sprint è una soluzione ammissibile per cui non saranno necessari cercare ulteriori soluzioni.

Table 6.28: KPI con Eliminazione vincoli AND e OR

	CPLEX		Heuristic		Lagrangian Heuristic	
Istanza	Zopt	Time	Zheu	Time	Zheu	Time
instance_09_01	45039.4	0.10	40865.4	0.06	40865.4	0.15
instance_09_02	46233.3	0.11	43929.3	0.04	43929.3	0.20
instance_09_03	41495.1	0.27	40156.1	0.02	40156.1	0.20
instance_09_04	34073.1	0.13	28217.5	0.02	28217.5	0.14
instance_09_05	35163.5	0.37	33706.2	0.02	33706.2	0.07
instance_30_01	185940.2	149.39	179213.4	0.13	179213.4	1.45
instance_30_02	208816.9	300.14	192310.0	0.13	192310.0	1.47
instance_30_03	196361.9	300.07	184020.7	0.16	184020.7	1.72
instance_30_04	178445.5	173.80	173434.7	0.15	173434.7	1.50
instance_30_05	194205.3	300.09	176435.6	0.21	176435.6	1.59
instance_30b_01	821619.3	300.11	805023.8	0.25	805023.8	4.14
instance_30b_02	862178.2	300.11	826366.0	0.24	826366.0	2.95
instance_30b_03	825585.9	300.11	787655.4	0.23	787655.4	3.76
instance_30b_04	795374.5	300.10	782301.5	0.38	782301.5	4.79
instance_30b_05	794461.8	300.09	785668.0	0.27	785668.0	3.68
instance_110_01	5683254.3	301.25	5614591.5	1.97	5614591.5	86.03
instance_110_02	5636912.3	300.97	5652307.0	1.65	5652307.0	111.38
instance_110_03	5681839.9	301.25	5634129.1	1.67	5634129.1	101.31
instance_110_04	5948998.9	301.02	5891549.2	2.08	5891549.2	74.29
instance_110_05	5529959.8	301.26	5533057.3	1.95	5533057.3	80.25
instance_f_01	130342.7	0.34	120454.0	0.06	120454.0	0.58
instance_f_02	136611.2	9.17	127857.5	0.07	127857.5	1.83
instance_f_03	132505.2	27.60	124484.6	0.05	124484.6	0.95
instance_f_04	130873.7	5.22	129644.7	0.06	129644.7	0.93
instance_f_05	138889.5	25.55	121783.5	0.05	121783.5	0.94

Table 6.29: KPI con Eliminazione vincoli AND e OR

	CPLEX			Heuristic			Lagrangian Heuristic		
Istanza	N.S.O.	A.S.U.	H.U.S.	N.S.O.	A.S.U.	H.U.S.	N.S.O.	A.S.U.	H.U.S.
instance_09_01	5	95.47%	2	5	95.53%	2	5	95.53%	2
instance_09_02	7	84.43%	2	7	84.51%	2	7	84.51%	2
instance_09_03	6	88.78%	2	6	90.71%	2	6	90.71%	2
instance_09_04	5	86.65%	2	5	86.97%	1	5	86.97%	1
instance_09_05	5	98.38%	2	6	80.68%	2	6	80.68%	2
instance_30_01	23	95.89%	9	23	95.48%	8	23	95.48%	8
instance_30_02	21	93.35%	6	21	92.65%	6	21	92.65%	6
instance_30_03	21	95.74%	8	22	90.73%	7	22	90.73%	7
instance_30_04	23	96.01%	8	24	91.58%	8	24	91.58%	8
instance_30_05	21	95.88%	7	22	91.51%	7	22	91.51%	7
instance_30b_01	21	96.14%	6	21	96.24%	6	21	96.24%	6
instance_30b_02	19	95.30%	5	19	95.27%	5	19	95.27%	5
instance_30b_03	20	99.17%	6	21	94.36%	5	21	94.36%	5
instance_30b_04	22	97.66%	6	22	97.67%	6	22	97.67%	6
instance_30b_05	23	95.00%	6	23	95.06%	6	23	95.06%	6
instance_110_01	96	97.66%	30	95	98.72%	29	95	98.72%	29
instance_110_02	88	97.52%	29	87	98.74%	29	87	98.74%	29
instance_110_03	89	95.67%	28	87	98.08%	27	87	98.08%	27
instance_110_04	94	96.85%	30	92	98.59%	28	92	98.59%	28
instance_110_05	92	96.15%	29	90	98.73%	27	90	98.73%	27
instance_f_01	9	92.46%	2	9	92.10%	2	9	92.10%	2
instance_f_02	10	93.64%	3	10	93.12%	2	10	93.12%	2
instance_f_03	9	95.60%	2	9	95.54%	2	9	95.54%	2
instance_f_04	10	93.22%	2	10	93.32%	2	10	93.32%	2
instance_f_05	9	94.34%	3	9	93.96%	2	9	93.96%	2

Table 6.30: KPI con Eliminazione vincoli AND e OR

	CPLEX						Heuristic						Lagrangian Heuristic					
Istanza	D.R.	R.MAX	R.MIN	D.U.	U.MAX	U.MIN	D.R.	R.MAX	R.MIN	D.U.	U.MAX	U.MIN	D.R.	R.MAX	R.MIN	D.U.	U.MAX	U.MIN
instance_09_01	2.3	10.8	4.4	2.2	10.2	4.3	3.0	13.0	4.7	2.3	11.4	4.9	3.0	13.0	4.7	2.3	11.4	4.9
instance_09_02	2.1	8.0	1.8	2.0	8.0	1.6	3.0	12.0	1.8	2.5	10.3	1.6	3.0	12.0	1.8	2.5	10.3	1.6
instance_09_03	3.3	12.3	1.9	3.7	13.4	1.7	4.3	15.1	2.3	4.5	16.4	3.5	4.3	15.1	2.3	4.5	16.4	3.5
instance_09_04	3.1	11.2	3.0	3.0	11.1	3.4	3.8	12.6	2.5	3.9	13.4	3.0	3.8	12.6	2.5	3.9	13.4	3.0
instance_09_05	2.1	9.7	4.5	1.7	9.9	5.0	4.1	13.1	1.2	3.6	12.4	1.2	4.1	13.1	1.2	3.6	12.4	1.2
instance_30_01	1.1	5.5	1.2	1.0	5.8	1.7	1.2	6.2	2.0	1.0	6.5	1.9	1.2	6.2	2.0	1.0	6.5	1.9
instance_30_02	1.4	7.0	1.0	1.2	6.5	1.5	1.6	7.2	1.3	1.2	6.5	1.8	1.6	7.2	1.3	1.2	6.5	1.8
instance_30_03	1.5	8.8	1.2	1.1	6.0	1.7	1.8	9.0	1.0	1.4	6.4	1.4	1.8	9.0	1.0	1.4	6.4	1.4
instance_30_04	1.3	7.2	1.1	0.9	5.9	2.0	1.3	7.5	1.1	1.1	6.6	1.7	1.3	7.5	1.1	1.1	6.6	1.7
instance_30_05	1.4	8.1	2.4	1.3	7.1	2.4	1.4	7.8	1.6	1.1	6.6	1.6	1.4	7.8	1.6	1.1	6.6	1.6
instance_30b_01	4.2	21.2	2.3	3.1	16.1	3.3	4.3	20.2	2.7	3.1	16.9	3.4	4.3	20.2	2.7	3.1	16.9	3.4
instance_30b_02	4.9	25.4	1.4	4.3	22.0	1.6	5.6	28.2	1.4	5.0	25.5	1.6	5.6	28.2	1.4	5.0	25.5	1.6
instance_30b_03	4.4	21.3	3.7	3.7	20.1	4.3	4.9	22.3	1.5	4.0	20.1	1.6	4.9	22.3	1.5	4.0	20.1	1.6
instance_30b_04	3.5	21.2	3.3	3.2	20.0	4.5	3.5	19.7	3.3	3.0	17.5	4.5	3.5	19.7	3.3	3.0	17.5	4.5
instance_30b_05	5.0	25.9	1.3	4.2	22.9	1.5	5.1	25.9	1.0	4.2	22.9	2.0	5.1	25.9	1.0	4.2	22.9	2.0
instance_110_01	1.4	9.2	2.3	1.1	7.4	2.6	1.6	10.7	2.1	1.1	7.3	2.7	1.6	10.7	2.1	1.1	7.3	2.7
instance_110_02	1.5	10.2	2.2	1.1	7.8	2.7	1.6	10.9	2.2	1.2	7.5	3.2	1.6	10.9	2.2	1.2	7.5	3.2
instance_110_03	1.7	10.3	1.5	1.1	7.4	1.5	1.5	10.8	1.1	1.0	8.0	1.8	1.5	10.8	1.1	1.0	8.0	1.8
instance_110_04	1.6	9.2	2.1	1.1	7.0	2.6	1.6	10.4	2.2	1.2	7.5	2.6	1.6	10.4	2.2	1.2	7.5	2.6
instance_110_05	1.6	8.9	2.3	1.2	7.5	2.6	1.6	9.4	2.3	1.2	8.3	2.8	1.6	9.4	2.3	1.2	8.3	2.8
instance_f_01	8.6	29.2	1.5	8.4	28.7	1.7	10.6	35.2	1.5	10.6	35.2	1.7	10.6	35.2	1.5	10.6	35.2	1.7
instance_f_02	6.1	24.4	2.9	5.3	21.5	2.7	7.6	30.0	3.2	6.7	26.3	2.9	7.6	30.0	3.2	6.7	26.3	2.9
instance_f_03	9.2	33.9	2.9	7.9	29.4	2.7	9.8	35.1	2.1	8.7	31.9	3.0	9.8	35.1	2.1	8.7	31.9	3.0
instance_f_04	8.9	33.7	2.3	8.5	33.2	3.6	9.7	36.4	2.3	9.4	35.3	2.5	9.7	36.4	2.3	9.4	35.3	2.5
instance_f_05	6.7	25.7	3.2	6.3	24.1	3.6	8.4	29.1	2.7	7.9	27.2	3.0	8.4	29.1	2.7	7.9	27.2	3.0

6.5.5 Risultati con formulazione matematica alternativa

Nelle tabelle 6.31 e 6.32 sono riportati i risultati ottenuti utilizzando la formulazione matematica della sezione 4.2.3 tramite il metodo esatto implementato in CPLEX e il metodo euristico. Come si può notare dai KPI N.S.O. e H.U.S. il metodo per massimmizzare la funzione obiettivo accorpa tutti gli sprint in uno unico. Questo è dovuto perché la formulazione matematica così espressa non pone vincoli per la dimensione dello sprint, né per il rischio né per l'incertezza massima che possono presentare in un unico sprint.

Table 6.31: KPI con formulazione matematica alternativa

	CPLEX		Heuristic	
Istanza	Zopt	Time	Zopt	Time
instance_09_01	26928.9	0.50	26928.9	0.05
instance_09_02	28491.3	0.54	28491.3	0.06
instance_09_03	25484.8	0.54	25484.8	0.06
instance_09_04	20503.8	0.50	20503.8	0.05
instance_09_05	21798.0	0.45	21798.0	0.07
instance_30_01	176685.0	0.52	176685.0	0.12
instance_30_02	183168.0	0.48	183168.0	0.11
instance_30_03	176618.0	0.51	176618.0	0.11
instance_30_04	167151.0	0.56	167151.0	0.10
instance_30_05	182553.0	0.48	182553.0	0.11
instance_30b_01	329604.0	0.60	329604.0	0.14
instance_30b_02	342816.0	0.70	342816.0	0.13
instance_30b_03	330453.0	0.62	330453.0	0.12
instance_30b_04	324516.5	0.63	324516.5	0.12
instance_30b_05	320931.0	0.72	320931.0	0.13
instance_110_01	2854456.0	2.38	2854456.0	0.40
instance_110_02	2837384.0	1.83	2837384.0	0.40
instance_110_03	2828331.0	2.15	2828331.0	0.38
instance_110_04	3000866.0	2.05	3000866.0	0.36
instance_110_05	2771178.7	2.21	2771178.7	0.31
instance_f_01	138936.7	0.76	138936.7	0.11
instance_f_02	149105.0	0.81	149105.0	0.11
instance_f_03	141372.5	0.88	141372.5	0.11
instance_f_04	138705.8	0.78	138705.8	0.10
instance_f_05	151400.8	0.86	151400.8	0.10

Table 6.32: KPI con formulazione matematica alternativa

	CPLEX			Heuristic		
Istanza	N.S.O.	A.S.U.	H.U.S.	N.S.O.	A.S.U.	H.U.S.
instance_09_01	1	95.47%	1	1	95.47%	1
instance_09_02	1	87.43%	1	1	87.43%	1
instance_09_03	1	95.78%	1	1	95.78%	1
instance_09_04	1	94.65%	1	1	94.65%	1
instance_09_05	1	98.38%	1	1	98.38%	1
instance_30_01	1	95.89%	1	1	95.89%	1
instance_30_02	1	98.35%	1	1	98.35%	1
instance_30_03	1	95.74%	1	1	95.74%	1
instance_30_04	1	96.01%	1	1	96.01%	1
instance_30_05	1	95.88%	1	1	95.88%	1
instance_30b_01	1	96.14%	1	1	96.14%	1
instance_30b_02	1	95.30%	1	1	95.30%	1
instance_30b_03	1	98.17%	1	1	98.17%	1
instance_30b_04	1	97.66%	1	1	97.66%	1
instance_30b_05	1	95.00%	1	1	95.00%	1
instance_110_01	1	97.66%	1	1	97.66%	1
instance_110_02	1	97.52%	1	1	97.52%	1
instance_110_03	1	95.67%	1	1	95.67%	1
instance_110_04	1	96.85%	1	1	96.85%	1
instance_110_05	1	96.15%	1	1	96.15%	1
instance_f_01	1	94.46%	1	1	94.46%	1
instance_f_02	1	95.64%	1	1	95.64%	1
instance_f_03	1	95.60%	1	1	95.60%	1
instance_f_04	1	96.22%	1	1	96.22%	1
instance_f_05	1	94.34%	1	1	94.34%	1

Chapter 7

Conclusioni

L'obiettivo della tesi era di raggiungere una migliore comprensione del problema di pianificazione degli sprint in un contesto di un progetto Scrum, al fine di testare l'efficacia di svariati metodi di risoluzione e proporre possibili funzioni obiettivo alternative, in grado di massimizzare l'utilità totale del progetto, tenendo conto del rischio e dell'incertezza associati alle user story.

7.1 Contributo della tesi

Il contributo principale della tesi consiste nella proposta e valutazione di nuove funzioni obiettivo, la cui analisi ha evidenziato che le funzioni obiettivo che incentivano l'inserimento precoce delle user story che sbloccano altre user story, tramite vincoli di dipendenza, tendono a fornire soluzioni paragonabili alle altre per numero di sprint utilizzati e utilità per il cliente, ma con sprint che risultano mediamente meno rischiosi e incerti.

Questo è dovuto al fatto che sbloccare tutte le user story il prima possibile permette al modello di distribuire in maniera ottimale il rischio e l'incertezza nei vari sprint disponibili.

Un altro contributo è stato l'analisi dell'impatto di possibili cambiamenti nelle istanze di progetto, come una diminuzione del numero di sprint disponibili o della capacità massima degli sprint, sulla pianificazione e sulla capacità dei metodi di risolvere il problema in tempi ragionevoli. I risultati mostrano che una riduzione del

numero di sprint, se il valore precedente era sovrastimato, non altera la soluzione; al contrario, una diminuzione eccessiva rende impossibile trovare una soluzione. La riduzione della capacità massima degli sprint, invece, comporta un aumento del numero di sprint utilizzati e un ritardo nella consegna delle user story al cliente. Inoltre, si è dimostrato che la presenza di vincoli di dipendenza tra le user story influisce sulla pianificazione del progetto e che la loro eliminazione riduce il numero di sprint necessari e i tempi di risoluzione.

7.2 Limiti dello studio

Vi sono alcuni limiti in questo studio. Le istanze che sono state utilizzate per fare i test sono sintetiche, per cui potrebbero non riflettere completamente la complessità e le dinamiche di un progetto reale. Inoltre, il miglioramento proposto in 5.3 non ha portato a miglioramenti nei risultati, probabilmente a causa del fatto che i tagli inseriti non sono stati sufficientemente efficaci nel ridurre lo spazio di ricerca, o che il metodo euristico non è stato in grado di sfruttare appieno i tagli inseriti, al fine di migliorare i tempi di risoluzione.

7.3 Sviluppi futuri

7.3.1 Miglioramento della Formulazione matematica 4.2.3

Come si è visto nei risultati 6.5.5 la formulazione matematica proposta nella sezione 4.2.3 non è in grado di produrre soluzioni con un numero di sprint maggiore di 1, a causa della mancanza di vincoli che limitano la dimensione dello sprint e il rischio e l'incertezza massima che possono essere presenti in un unico sprint. Un possibile sviluppo futuro consiste nell'introduzione di tali vincoli nella formulazione matematica, al fine di ottenere soluzioni più realistiche e utili per la pianificazione degli sprint. Una forma migliore potrebbe essere la seguente:

$$z_P = \max \sum_{k=1}^{m} \sum_{i=1}^{k} \sum_{j=1}^{n} u_j \left(r_j^{cr} x_{ij} + a_j y_{ij} \right) \tag{7.1}$$

$$s.t. \sum_{j=1}^{n} p_j r_j^{un} x_{ij} - p^{sp} \xi_i \leq 0, \qquad i \in S \tag{7.2}$$

$$\sum_{j=1}^{n} r_j^{cr} x_{ij} \leq \frac{\sum_{j=1}^{n} r_j^{cr}}{m} \psi, \qquad i \in S \tag{7.3}$$

$$\sum_{i=1}^{m} x_{ij} = 1, \qquad j \in U \tag{7.4}$$

$$\sum_{k=1}^{i} \sum_{z \in D_j^{OR}} x_{kz} \geq x_{ij}, \qquad i \in S, j \in U^{OR} \tag{7.5}$$

$$\sum_{k=1}^{i} \sum_{z \in D_j^{AND}} x_{kz} \geq x_{ij} |D_j^{AND}|, \quad i \in S, j \in U^{AND} \tag{7.6}$$

$$y_{ij} \leq \sum_{k \in Y_j} x_{ik}, \qquad i \in S, j \in U \tag{7.7}$$

$$y_{ij} \leq |Y_j| x_{ij}, \qquad i \in S, j \in U \tag{7.8}$$

$$x_{ij} \in \{0, 1\}, \qquad i \in S, j \in U \tag{7.9}$$

$$y_{ij} \geq 0, \qquad i \in S, j \in U \tag{7.10}$$

Dove con la funzione (7.3) si cerca di mantenere il rischio totale dello sprint i minore del rischio totale di tuttle le user story diviso il numero di sprint, trovando cosi una media di rischio da tenere in ogni sprint, moltiplicato per un parametro ψ il quale deve essere trovato per ogni istanza che si vuole analizzare, la sua presenza è dovuta perche ogni progetto presenta rischi diversi che possono variare sostanzialmete a seconda della tipologia di progetto, per esempoi progetti di ricerca rusultano essere molto più rischiosi di progetti di cui si ha già esperienza e si conoscono tutte le specifiche.

L'obiettivo ultimo di questa funzione è quello di concentrare le user story più rischiose all'inizio del progetto in sprint che hanno capacita pari a p^{sp} e successivamente la presenza di sprint più lunghi che comprendono le user story meno

rischiose del progetto. Ottenendo cosi una distribuzione del rischio più equilibrata nei vari sprint, e permettendo al team di affrontare prima le user story più rischiose, in modo da anticiparlo e poi poter proseguire con piú tranquillita nel progetto.

7.3.2 Formulazione matematica alternativa

Un promettente sviluppo futuro consiste in una riformulazione matematica alternativa del problema, basata su un approccio di tipo *pattern-based*. In tale approccio, anziché modellizzare direttamente l'assegnazione delle user story agli sprint, si considerano i possibili pattern di assegnazione delle user story agli sprint, selezionando quelli che massimizzano l'utilità totale.

In particolare, un sottoinsieme di user story $E \subseteq U$ potrebbe essere eseguito nello sprint $i \in S$ se e solo se soddisfa il vincolo di capacità dello sprint, ovvero

$$\sum_{j \in E} p_j r_j^{un} \leq p_i^{max}.$$

Tale sottoinsieme E verrebbe definito *pattern di sprint ammissibile*, o semplicemente *pattern*.

L'insieme indicizzato di tutti i pattern ammissibili sarebbe denotato con $\mathbb{S}$, ovvero

$$\mathbb{S} = \left\{ \ell : \sum_{j \in E_\ell} p_j r_j^{un} \leq p_i^{max}, E_\ell \subseteq U, i \in S \right\}.$$

Per un pattern $\ell \in \mathbb{S}$, $\pi(\ell)$ indicherebbe lo sprint corrispondente.

Si definirebbe inoltre $\mathbb{S}_j \subseteq \mathbb{S}$ come il sottoinsieme di pattern contenenti la user story $j \in U$, $\mathbb{S}'_i \subseteq \mathbb{S}$ come il sottoinsieme di pattern corrispondenti allo sprint $i \in S$, e $\mathbb{S}''_{ij} \subseteq \mathbb{S}$ come il sottoinsieme di pattern che corrispondono allo sprint $i \in S$ e contengono la user story $j \in U$.

Il nuovo modello di programmazione lineare mista intera è il seguente:

$$(P)\ z_P = \max \sum_{\ell \in \mathbb{S}} u_\ell x_\ell \tag{7.11}$$

$$s.t. \sum_{\ell \in \mathbb{S}_j} x_\ell = 1, \qquad j \in U \tag{7.12}$$

$$\sum_{\ell \in \mathbb{S}'_i} x_\ell \leq 1, \qquad i \in S \tag{7.13}$$

$$\sum_{k=1}^{i} \sum_{z \in D_j^{OR}} \sum_{\ell \in \mathbb{S}''_{kz}} x_\ell \geq \sum_{\ell \in \mathbb{S}''_{ij}} x_\ell, \qquad i \in S, j \in U^{OR} \tag{7.14}$$

$$\sum_{k=1}^{i} \sum_{z \in D_j^{AND}} \sum_{\ell \in \mathbb{S}''_{kz}} x_\ell \geq |D_j^{AND}| \sum_{\ell \in \mathbb{S}''_{ij}} x_\ell, \quad i \in S, j \in U^{AND} \tag{7.15}$$

$$x_\ell \in \{0, 1\}, \qquad \ell \in \mathbb{S} \tag{7.16}$$

dove $u_\ell = \sum_{j \in E_\ell} u_j \left(r_j^{cr} + a'_j y_{\ell j}\right)$ e $y_{\ell j}$ rappresenta il numero di storie in Y_j (storie affini alla storia j) incluse nel pattern ℓ.

L'utilità u_j di ogni storia j è incrementata dal rischio di criticità r_j^{cr}, favorendo così un posizionamento anticipato delle storie critiche, e dalle storie affini incluse nello stesso sprint, sommando la frazione $a'_j = \frac{a_j}{|Y_j|}$ per ciascuna di esse. Il numero di storie affini incluse nello sprint $\pi(\ell)$ è dato da

$$y_{\ell j} = |Y_j \cap E_\ell|,$$

se $j \in E_\ell$, altrimenti $y_{\ell j} = 0$.

La funzione obiettivo (7.11) massimizza l'utilità totale dei pattern selezionati nella soluzione ottimale.

I vincoli (7.12) garantiscono che ogni storia j sia inclusa esattamente in un pattern, mentre i vincoli (7.13) assicurano che per ogni sprint i sia selezionato al più un pattern.

Le dipendenze sono modellizzate dai vincoli (7.14) e (7.15). Per una dipendenza OR di una storia j, i vincoli (7.14) consentono l'assegnazione di j a un pattern dello sprint i solo se almeno una storia dell'insieme D_j^{OR} è assegnata a un pattern corrispondente a uno sprint $i' \leq i$. Analogamente, per una dipendenza AND, i

vincoli (7.15) richiedono che tutte le storie dell'insieme D_j^{AND} siano assegnate a pattern corrispondenti a sprint precedenti o uguali a i.

La rilassazione lineare di P si ottiene sostituendo i vincoli di integralità (7.16) con $x_\ell \geq 0$, $\forall \ell \in \mathbb{S}$.

Siano u_j, v_i, α_{ij}, e β_{ij} le variabili duali associate rispettivamente ai vincoli (7.12), (7.13), (7.14), e (7.15). Il duale è:

$$(D)\ z_D = \min \sum_{j \in U} u_j + \sum_{i \in S} v_i \tag{7.17}$$

$$s.t. \sum_{j \in E_\ell} u_j + v_{\pi(\ell)} + \sum_{j \in E_\ell} \sum_{j' \in \bar{D}_j^{OR}} \sum_{k=\pi(\ell)}^{m} \alpha_{kj'} + \sum_{j \in E_\ell} \sum_{j' \in \bar{D}_j^{AND}} \sum_{k=\pi(\ell)}^{m} \beta_{kj'} + - \sum_{j \in E_\ell \cap U^{OR}} \alpha_{\pi(\ell)j} - \sum_{j \in E_\ell \cap U^{AND}} |D_j^{AND}| \beta_{\pi(\ell)j} \geq u_\ell,\ \ell \in \mathbb{S} \tag{7.18}$$

$$u_j \text{ unconstrained}, \qquad j \in U \tag{7.19}$$

$$v_i \geq 0, \qquad i \in S \tag{7.20}$$

$$\alpha_{ij} \leq 0, \qquad i \in S, j \in U^{OR} \tag{7.21}$$

$$\beta_{ij} \leq 0, \qquad i \in S, j \in U^{AND} \tag{7.22}$$

Il duale è utilizzato per definire i *costi ridotti* dei pattern e il problema di pricing per generare dinamicamente i pattern candidati alla soluzione ottima. In questo modo si evita di generare tutti i pattern, potendo produrre solo un sottoinsieme ridotto sufficiente per dimostrare l'ottimalità della soluzione.

Bibliography

[Abrahamsson et al., 2003a] Abrahamsson, P., Warsta, J., Siponen, M., and Ronkainen, J. (2003a). New directions on agile methods: a comparative analysis. In *25th International Conference on Software Engineering, 2003. Proceedings.*, pages 244–254.

[Abrahamsson et al., 2003b] Abrahamsson, P., Warsta, J., Siponen, M. T., and Ronkainen, J. (2003b). New directions on agile methods: A comparative analysis. In *Proc. ICSE*, pages 244–254.

[Al-Zubaidi et al., 2018] Al-Zubaidi, W. H. A., Dam, H. K., Choetkiertikul, M., and Ghose, A. (2018). Multi-objective iteration planning in agile development. In *2018 25th Asia-Pacific Software Engineering Conference (APSEC)*, pages 484–493.

[Alagöz and Azizoğlu, 2003] Alagöz, O. and Azizoğlu, M. (2003). Rescheduling of identical parallel machines under machine eligibility constraints. *European Journal of Operational Research*, 149(3):523–532.

[Avella et al., 2010] Avella, P., Boccia, M., and Vasilyev, I. (2010). A computational study of exact kanpsack separation for the generalized assignment problem. *Computational Optimization and Applications*, 45:543–555.

[Beck et al., 2001a] Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., Kern, J., Marick, B., Martin, R. C., Mellor, S., Schwaber, K., Sutherland, J., and Thomas, D. (2001a). Manifesto for agile software development.

[Beck et al., 2001b] Beck, K. et al. (2001b). Manifesto for agile software development. http://agilemanifesto.org.

[Boschetti et al., 2002] Boschetti, M., Hadjinconstantinou, E., and Mingozzi, A. (2002). New upper bounds for the finite two-dimensional orthogonal non-guillotine cutting stock problem. *IMA Journal of Management Mathematics*, 13:95–119.

[Boschetti and Mingozzi, 2003] Boschetti, M. and Mingozzi, A. (2003). The two-dimensional finite bin packing problem. Part I: New lower bounds for the oriented case. *4OR*, 1:27–42.

[Boschetti et al., 2014] Boschetti, M. A., Golfarelli, M., Rizzi, S., and Turricchia, E. (2014). A lagrangian heuristic for sprint planning in agile software development. *Computers & Operations Research*, 43:116–128.

[Boschetti and Montaletti, 2010] Boschetti, M. A. and Montaletti, L. (2010). An exact algorithm for the two-dimensional strip-packing problem. *Operations Research*, 58(6):1774–1791.

[Brucker et al., 1999] Brucker, P., Drexl, A., Möhring, R., Neumann, K., and Pesch, E. (1999). Resource-constrained project scheduling: Notation, classification, models, and methods. *European Journal of Operational Research*, 112(1):3–41.

[Cohn, 2004a] Cohn, M. (2004a). *User Stories Applied: For Agile Software Development*. Addison-Wesley Professional.

[Cohn, 2004b] Cohn, M. (2004b). *User stories applied: For agile software development*. Addison-Wesley Professional.

[Cohn, 2005] Cohn, M. (2005). *Agile estimating and planning*. Pearson Education.

[De Boer, 1998] De Boer, R. (1998). *Resource-constrained multi-project management*. PhD thesis, PhD thesis, University of Twente, The Netherlands.

[Demeulemeester and Herroelen, 2002] Demeulemeester, E. L. and Herroelen, W. S. (2002). *Project scheduling: a research handbook*. Springer.

[Denne and Cleland-Huang, 2003] Denne, M. and Cleland-Huang, J. (2003). *Software by numbers: Low-risk, high-return development*. Prentice Hall Professional.

[Dybå and Dingsøyr, 2008] Dybå, T. and Dingsøyr, T. (2008). Empirical studies of agile software development: A systematic review. *Information & Software Technology*, 50(9-10):833–859.

[Dybå and Dingsøyr, 2008] Dybå, T. and Dingsøyr, T. (2008). Empirical studies of agile software development: A systematic review. *Information and Software Technology*, 50(9):833–859.

[Gademann and Schutten, 2005] Gademann, N. and Schutten, M. (2005). Linear-programming-based heuristics for project capacity planning. *IIE Transactions (Institute of Industrial Engineers)*, 37(2):153 – 165. Cited by: 31.

[Golfarelli et al., 2012] Golfarelli, M., Rizzi, S., and Turricchia, E. (2012). Sprint planning optimization in agile data warehouse design. In *Data Warehousing and Knowledge Discovery*, pages 30–41.

[Golfarelli et al., 2013] Golfarelli, M., Rizzi, S., and Turricchia, E. (2013). Multi-sprint planning and smooth replanning: An optimization model. *Journal of Systems and Software*, 86(9):2357–2370.

[Greer and Ruhe, 2004] Greer, D. and Ruhe, G. (2004). Software release planning: an evolutionary and iterative approach. *Information and Software Technology*, 46(4):243–253.

[Herroelen et al., 1999] Herroelen, W., Demeulemeester, E., and De Reyck, B. (1999). A classification scheme for project scheduling. In *Project scheduling: recent models, algorithms and applications*, pages 1–26. Springer.

[Herroelen et al., 2002] Herroelen, W., Leus, R., and Demeulemeester, E. (2002). Critical chain project scheduling: Do not oversimplify. *Project Management Journal*, 33(4):48–60.

[Khurum et al., 2013] Khurum, M., Gorschek, T., and Wilson, M. (2013). The software value map—an exhaustive collection of value aspects for the develop-

ment of software intensive products. *Journal of software: Evolution and Process*, 25(7):711–741.

[Kolisch and Padman, 2001] Kolisch, R. and Padman, R. (2001). An integrated survey of deterministic project scheduling. *Omega*, 29(3):249–272.

[Li et al., 2010] Li, C., van den Akker, M., Brinkkemper, S., and Diepen, G. (2010). An integrated approach for requirement selection and scheduling in software release planning. *Requirements Engineering*, 15(4):375 – 396. Cited by: 48; All Open Access, Hybrid Gold Open Access.

[Newbold, 1998] Newbold, R. C. (1998). *Project management in the fast lane: applying the theory of constraints*. CRC Press.

[Park and Park, 2004] Park, Y. and Park, G. (2004). A new method for technology valuation in monetary value: procedure and application. *Technovation*, 24(5):387–394.

[Platje et al., 1994] Platje, A., Seidel, H., and Wadman, S. (1994). Project and portfolio planning cycle: Project-based management for the multiproject challenge. *International Journal of Project Management*, 12(2):100–106.

[Rönkkö et al., 2009] Rönkkö, M., Frühwirth, C., and Biffl, S. (2009). Integrating value and utility concepts into a value decomposition model for value-based software engineering. In *International Conference on Product-Focused Software Process Improvement*, pages 362–374. Springer.

[Sadeh et al., 1993] Sadeh, N., Otsuka, S., and Schedlback, R. (1993). Predictive and reactive scheduling with the microboss production scheduling and control system. In *Proceedings of the IJCAI-93 workshop on knowledge-based production planning, scheduling and control*, pages 293–306.

[Sakkout and Wallace, 2000] Sakkout, H. E. and Wallace, M. (2000). Probe backtrack search for minimal perturbation in dynamic scheduling. *Constraints*, 5(4):359–388.

[Saliu and Ruhe, 2007] Saliu, M. O. and Ruhe, G. (2007). Bi-objective release planning for evolving software systems. In *Proceedings of the the 6th Joint Meeting of the European Software Engineering Conference and the ACM SIGSOFT Symposium on The Foundations of Software Engineering*, ESEC-FSE '07, page 105–114, New York, NY, USA. Association for Computing Machinery.

[Schwaber, 1995] Schwaber, K. (1995). SCRUM development process. In *Proc. OOPSLA*.

[Schwaber, 1997] Schwaber, K. (1997). Scrum development process. In Sutherland, J., Casanave, C., Miller, J., Patel, P., and Hollowell, G., editors, *Business Object Design and Implementation*, pages 117–134, London. Springer London.

[Szöke, 2011] Szöke, A. (2011). Conceptual scheduling model and optimized release scheduling for agile environments. *Information and Software Technology*, 53(6):574–591. Special Section: Best papers from the APSEC.

[van Valkenhoef et al., 2011] van Valkenhoef, G., Tervonen, T., de Brock, B., and Postmus, D. (2011). Quantitative release planning in extreme programming. *Information and Software Technology*, 53(11):1227–1235. AMOST 2010.

Acknowledgements

Vorrei ringraziare il mio relatore per il supporto e la guida durante lo sviluppo di questa tesi.

Ringrazio i miei familiari per il loro sostegno e incoraggiamento durante questo percorso, in particolare chi mi ha ospitato per anni.

Ringrazio i miei amici universitari per il supporto e la compagnia durante gli anni di studio: è stato un piacere dare gli esami con voi.

Infine, un grazie a tutti gli altri miei amici, da chi conosco da pochi mesi a chi conosco da più di venti anni: grazie per essere sempre presenti e per rendere la mia vita più leggera.