Università degli Studi di Genova
Facoltà di Ingegneria Informatica
Corso di Laurea Magistrale in Logistica e Produzione

Sviluppo, analisi e confronto di formulazioni matematiche per l'energy-efficient scheduling per problemi con macchine parallele

Relatori:
Prof. Massimo Paolucci
Dott. Roberto Ronco

Candidato:
Stefano Lavaggi

Anno Accademico 20/21
14/06/2022

Abstract

Il raggiungimento dell'efficienza energetica nella produzione manifatturiera è diventato una questione ben nota negli ultimi anni a causa dei presenti problemi ambientali e dell'obbligo di transizione verso ad un modello più sostenibile. Per alleviare i carichi della rete elettrica durante le ore di punta, in molti paesi del mondo è stata implementata la tariffazione dell'energia elettrica in base all'orario di utilizzo, denominata *Time-of-Use Tariffs Policy*. Questa strategia incoraggia a deferire l'utilizzo dell'energia elettrica dai periodi di picco a quelli di basso carico al fine di ridurre i costi sia per chi consuma e sia per chi fornisce energia. Questa tesi ha l'obiettivo di sviluppare ed analizzare il problema bi-obiettivo di "energy-efficient scheduling" con macchine parallele non correlate in uno schema di tariffazione Time-of-Use, andando ad offrire nuovi spunti per la comunità scientifica su questo particolare tema. La politica tariffaria presa in esame comporta ad una partizione dell'orizzonte temporale in un insieme di fasce orarie, ciascuna associata ad un costo che diventa parte del problema di ottimizzazione. Gli obiettivi del problema sono due: minimizzare il costo totale dell'energia consumata e la minimizzare il tempo totale di completamento, programmando in modo appropriato i lavori in modo che non si superi una scadenza di produzione predeterminata. Per risolverlo si presentano quattro formulazioni matematiche basate sulla programmazione intera mista, differenziate dal modo in cui le fasce orarie vengono gestite nella formulazione e successivamente si analizzano le loro prestazioni. I risultati ottenuti potrebbero dimostrarsi utili

nell'industria manifatturiera, poiché i decisori potranno usufruire dei modelli forniti in questa tesi per ottenere un compromesso tra produttività e sostenibilità.

Indice

Elenco delle figure

Elenco delle tabelle

Capitolo 1

Introduzione

1.1 Sostenibilità

Uno degli aspetti che le compagnie manifatturiere devono prendere in considerazione è la sostenibilità. Questa cambiamento è spinto da motivazioni di grande rilievo come: una sempre maggiore preoccupazione verso l'ambiente, la diminuzione della disponibilità delle risorse non rinnovabili, l'introduzione di leggi più restringenti, la preferenza dei consumatori per prodotti rispettosi dell'ambiente e costi energetici in crescita. [1].

Non esiste alcuna definizione consolidata né per "Sostenibilità", né per "Produzione sostenibile" [2], per cui questa sezione è dedicata all'analisi generale di cosa si intenda per sostenibilità e la relazione di quest'ultima con la produzione manifatturiera. All'inizio degli anni '70, la sostenibilità cominciò a divenire oggetto d'interesse per ricercatori, governi e imprese. Uno dei primi studi fu "Limit to Growth" pubblicato da Meadows et al. nel 1972 [3]. Gli autori studiarono diversi possibili scenari futuri considerando differenti fattori, come: l'impatto della crescita della popolazione mondiale, l'industrializzazione, l'inquinamento e il crollo della quantità di alimenti pro capite. Le conclusioni del rapporto furono che se l'attuale tasso di crescita della popolazione, dell'industrializzazione, dell'inquinamento, della produzione di cibo

e dello sfruttamento delle risorse restasse inalterato si raggiungerebbero i limiti dello sviluppo su questo pianeta alla metà del 21esimo secolo. Tutto ciò comporterebbe ad un improvviso ed incontrollabile declino della capacità industriale e della popolazione. Per evitare che ciò accada i tassi di sviluppo devono esser modificati, difatti nel loro studio Meadows et al. hanno dimostrato che abbassando i tassi di crescita si sarebbe giunti ad una condizione di stabilità ecologica ed economica, sostenibile anche nel lontano futuro. Questo studio fu uno dei primi a sottolineare l'importanza e la necessità di un cambiamento verso la sostenibilità, portandola ad essere uno dei temi più discussi negli anni avvenire. Nel 1987 una sotto-organizzazione delle Nazioni Unite (ONU), precisamente la "United Nations Brundtland Commission", presentò uno studio contenente una delle prime definizioni di sostenibilità approvata dalla comunità scientifica:

> "Development which meets the needs of current generations without compromising the ability of future generations to meet their own needs" [4]

Più avanti nel 1997 ,nel primo rapporto di sostenibilità della compagnia petrolifera anglo-olandese Shell, venne utilizzando il concetto di "Triple Bottom Line (TBL)" sviluppato da Elkington (1994) [5] per espandere la definizione di sostenibilità utilizzata fino a quel momento. Nel particolare si esaminò la sostenibilità sotto tre aspetti: economico, sociale e ambientale (vedi Fig. 1.1):

- L'aspetto economico nella produzione si traduce con la produzione del numero massimo di prodotti con il minimo utilizzo di risorse. Ciò è essenziale perché se una produzione non è competitiva, l'azienda avrà grandi probabilità di non sopravvivere nel futuro.

- Per quando riguarda l'aspetto sociale si intendono le condizioni di lavoro per i dipendenti, la soddisfazione dei clienti e anche l'impatto nella società.

- Infine l'aspetto ambientale si può esprimere con l'impatto della produzione sull'ambiente, per esempio la composizione dell'energia utilizzata (se ricavata o no da fonti rinnovabili) e l'utilizzo di materiali riciclati e/o riciclabili durante il processo di produzione.

Ciò significa che qualsiasi processo per esser dichiarato sostenibile dovrà esser compatibile e in equilibro con tutte e tre le dimensioni.

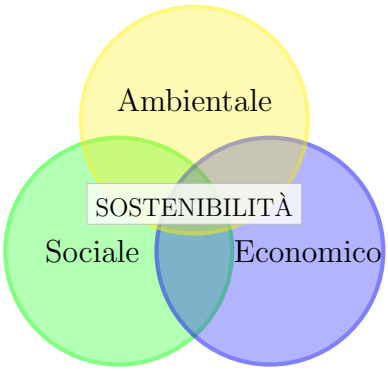


Figura 1.1: Rappresentazione grafica del concetto di sostenibilità utilizzando la definizione di "Triple Bottom Line"

1.2 Produzione sostenibile

Seguendo la definizione di sostenibilità esposta in precedenza, non vi sono dubbi che la produzione sia importante per gli ambiti sociale ed economico. Tuttavia non si può negare il grande impatto negativo che può avere sull'ambiente. Infatti l'estrazione di materie prime e il grosso dispendio energetico hanno avuto sicuramente un'influenza negativa [6]. Parlando in cifre, nel 2020 la produzione industriale rappresenta ben il 38% (156 EJ) del consumo totale di energia nel mondo [7]. Una diminuzione o un utilizzo intelligente del consumo energetico comporterebbe sicuramente effetti ambientali positivi, ma anche ad un risparmio economico che, dato l'ordine di grandezza di cui si parla, risulterebbe notevole.

Per stabilire cosa si intenda con il termine "produzione sostenibile" può esser utilizzata la definizione proposta dal US EPA ("United States Environmental Protection Agency") che risulta ampiamente accettata dalla comunità scientifica [8]:

> "Sustainable manufacturing is the creation of manufactured products through economically-sound processes that minimize negative environmental impacts while conserving energy and natural resources [9]"

Si nota che l'attenzione viene portata sull'uso efficiente delle risorse e sulla minimizzazione dei molteplici impatti ambientali, che sono causati dalle emissioni inquinanti e dall'uso di materiali tossici durante il ciclo di produzione [8].

Quando si parla di produzione sostenibile, il ruolo della tecnologia deve essere obbligatoriamente preso in considerazione, soprattutto per il suo utilizzo nello sviluppo di approcci sostenibili alla produzione [6]. Anche la ricerca, al pari della tecnologia, ha un ruolo fondamentale, in particolare per fornire soluzioni innovative per tutte le componenti della produzione: prodotti, processi, sistemi di produzione, organizzazione industriale e nuovi modelli di business. Date tutte le complessità coinvolte, si può dedurre che lo sviluppo della sostenibilità e della produzione sostenibile è un lungo processo che durerà anni, ma che deve esser attuato.

1.2.1 Pianificazione della produzione

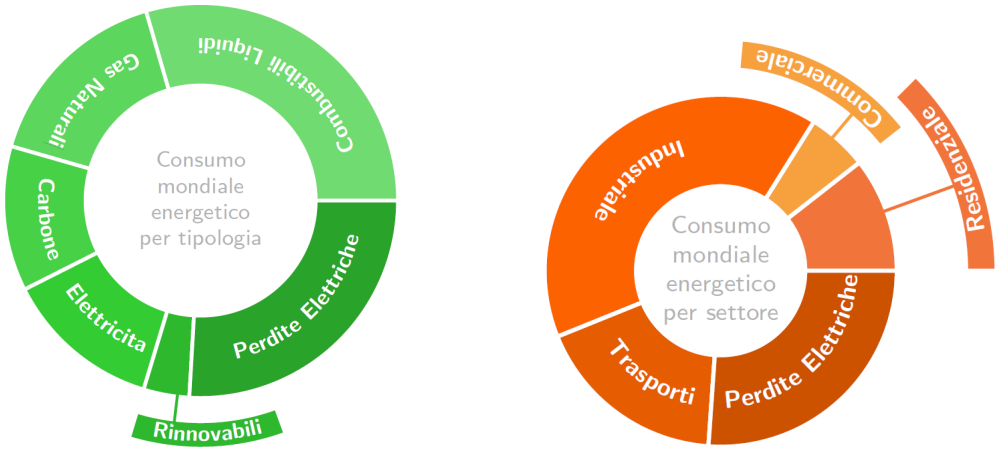
Nel ciclo della produzione uno dei passi fondamentali è la pianificazione. Il "production planning" è il processo di stesura di un piano per garantire un corretto ed efficiente flusso di produzione, il tutto secondo degli obiettivi fissati. Generalmente, a livello operativo, questi obiettivi sono concentrati sul raggiungimento di una qualità desiderata del prodotto tramite il minor costo possibile di produzione [10]. Da ciò si può comprendere quanto il planning sia

un compito molto complesso, difatti include molti problemi di decisione come: dimensione dei lotti, gestione delle scorte, scheduling di produzione, ecc. Gli sforzi per sviluppare sistemi di pianificazione sostenibili devono considerare le problematiche a tutti i livelli di pianificazione (strategico, tattico e operativo) e non solo focalizzarsi solamente su di uno di essi. Al momento vi sono diversi studi su tutti e tre i livelli, ma la maggior parte sono incentrati sul livello strategico [2]. Il motivo principale di questa enfasi sui livelli strategici è che gran parte degli sforzi verso la sostenibilità sono guidati dai livelli più alti della dirigenza delle aziende. Nonostante ciò bisogna ricordare che il tema della sostenibilità, se preso seriamente in considerazione, richiede sforzi e cambiamenti in tutti e tre i livelli di pianificazione.

1.3 Il ruolo dell'energia

Secondo l'International Energy Outlook [11] è stato previsto che il consumo energetico mondiale aumenterà da 601.5 YBTU (British Thermal Unit) nel 2020 a 886.3 YBTU nel 2050, una crescita del 47,3% in 30 anni. Parallelamente a questa crescita vi sarà anche un aumento dell'emissione di gas serra, dato che gli impianti di produzione energetica sono una delle maggiori fonti di emissione. Visti questi dati sicuramente il problema dell'energia e del suo utilizzo in modo ottimale sono una delle maggiori preoccupazioni pubbliche mondiali degli ultimi anni.

A causa di diversi fattori, come l'aumento del costo dell'energia e l'emanazione di leggi sempre più stringenti, il tema energetico è diventato una delle sfide essenziali per le compagnie produttive, portando ad uno sviluppo della gestione delle operazioni in modo che risulti "energy-aware". In ambito manifatturiero la ricerca di un ottimo utilizzo dell'energia disponibile è sicuramente un'operazione che può esser definita sostenibile, per via dell'attenzione agli aspetti ambientale e sociale. Una innovazione o cambiamento attento ai consumi energetici applicata ad un livello strategico può richiede-



(a) Per tipologia di fonte energetica

(b) Per tipologia settore

Figura 1.2: Suddivisione del consumo mondiale energetico per settore di consumo e per tipologia di fonte [12]

re grandi investimenti economici, i quali possono risultare grossi oneri per un'azienda manifatturiera. Tuttavia una produzione può affrontare il tema del consumo energetico anche al livello operativo. Difatti operazioni come: pianificazione del sequenziamento dei lavori, gestione delle scorte, dimensionamento dei lotti, non richiedono elevati costi di investimento, essendo variazioni della sola organizzazione. Differenti studi descrivono tecniche di risparmio energetico durante lo scheduling, come lo spegnimento delle macchine, per risparmiare energia non destinato alla produzione, una selezione ottimale delle velocità di produzione della macchine, ecc... Pertanto risulta fondamentale che i gestori di impianti controllino i propri processi produttivi in modo efficiente dal punto di vista energetico [13] per compiere dei passi avanti verso la sostenibilità ed una produzione sostenibile. Oltre a chi ne consuma, l'energia deve esser gestita in modo ottimale anche da chi la produce. I fornitori per indurre i propri clienti ad impiegare l'energia disponibile in modo da diminuirne lo spreco. Il modo più diffuso per attuare ciò è l'applicazione e l'utilizzo di "demand response program (DR)", cioè poli-

tiche di gestione della domanda. Generalmente un programma DR consiste nella modifica del prezzo dell'elettricità in risposta alla quantità richiesta dai clienti[14]. Essi vengono stabiliti dai fornitori di energia e non sono altro che dei tariffari che hanno l'obiettivo di motivare gli utenti a programmare i loro consumi in base al cambiamento del prezzo dell'elettricità [15]. Un programma DR ben strutturato può far ridurre i picchi di carico presenti durante una giornata (vedi Fig. 1.3), in modo da rendere la distribuzione dell'energia più omogenea durante la giornata.

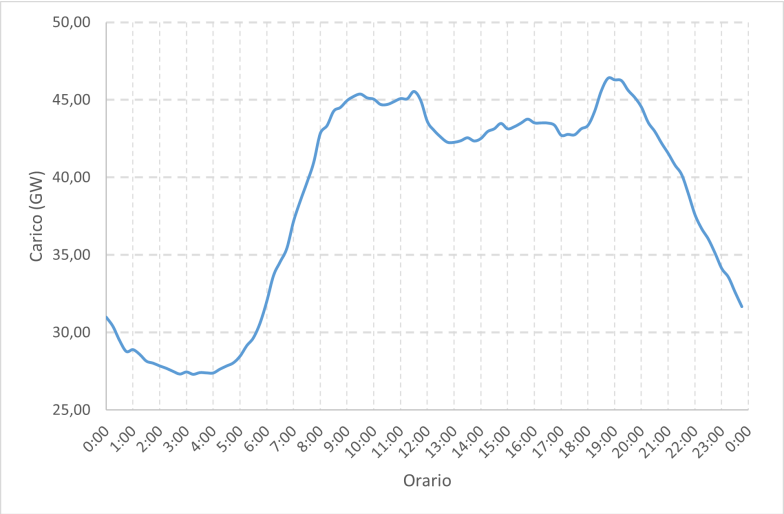


Figura 1.3: Esempio di andamento giornaliero della curva di carico in Italia, dato del 18/03/2022 [16]

1.3.1 Time-Of-Use (TOU)

Uno dei programmi DR più studiati in ambito accademico è il "Time-Of-Use pricing scheme", che risulta anche uno dei più utilizzati realmente per via della sua semplicità ed economicità di implementazione [17]. In sintesi l'idea del TOU consiste nell'attuare prezzi bassi durante le ore non di punta, mentre penalizza le ore di carico con prezzi più alti (vedi Fig. 1.4).

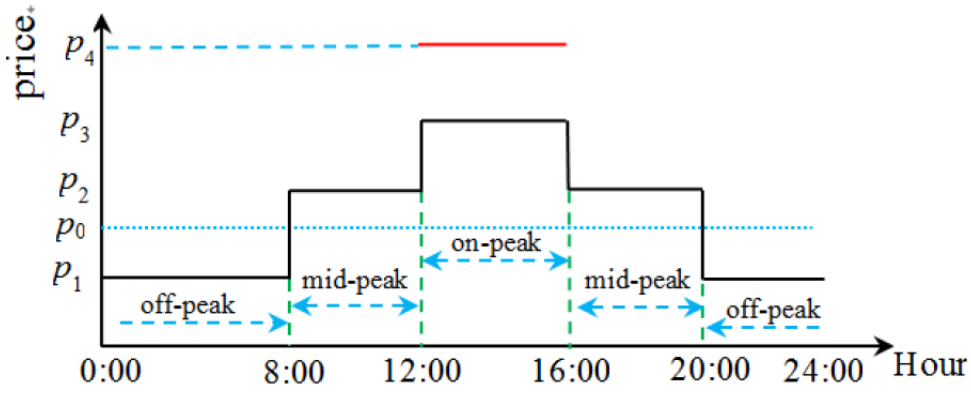
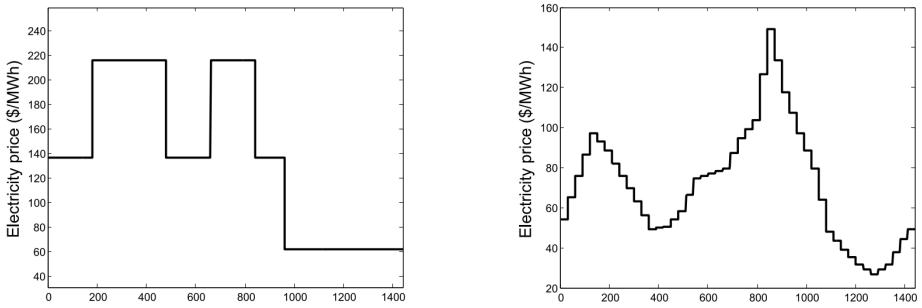


Figura 1.4: Esempio di andamento del prezzo secondo la politica TOU [18]



(a) Cambio prezzo poche volte al giorno (b) Cambio prezzo ogni mezz'ora/ora

Figura 1.5: Due esempi di profili del prezzo per kWh con politica TOU differenziati dalle quantità di cambi di prezzo attuati giornalmente

Tali strutture tariffarie possono incoraggiare consumatori di elevate quantità di energia elettrica a spostare il loro uso dai periodi di punta a quelli medi o non di punta, poiché l'utilizzo dell'elettricità nei picchi viene addebitato a un prezzo più alto.

Le politiche di prezzo TOU variano notevolmente in base ai diversi luoghi/paesi. Tuttavia, si possono generalmente dividere in due categorie [19]: quelle con prezzi che cambiano solo poche volte nell'arco di una giornata (vedi Fig.1.5a), e quelle con prezzi che fluttuano frequentemente per brevi periodi (ogni ora/mezz'ora, vedi Fig.1.5b).

Come esempio rappresentativo della politica TOU si riporta lo standard di tariffazione TOU commerciale di Pechino nella Tabella 1.1, in questo caso si può notare che il prezzo cambia poche volte durante la giornata.

Tabella 1.1 Esempio di variazione giornaliera delle tariffe Time-of-Use a Pechino

	High-Peak	Peak	Mid-peak	Off-peak
Periodo	11:00-13:00	10:00-15:00	7:00-10:00	23:00-7:00
	20:00-21:00	18:00-21:00	15:00-18:00	
			21:00-23:00	
Prezzo (CNY/kWh)	1.4409	1.3222	0.8395	0.3818
1 CNY = 0.1416 EUR				

Come si può osservare, il prezzo dell'ora di punta (1,4409 CNY/kWh) può essere quasi quattro volte superiore a quello delle ore non di punta (0,3818 CNY/kWh). Una variazione così ampia suggerisce grandi opportunità di risparmio per le aziende manifatturiere ad alta intensità energetica.

1.4 Motivazioni

Questa tesi di ricerca si pone l'obiettivo presentare ed analizzare nuove formulazioni per il problema di job scheduling con la considerazione delle tariffe orarie. Questo tema è ancora poco discusso e studiato in ambito accademico e quindi questa tesi può presentare nuovi spunti per la ricerca verso direzioni inesplorate con l'obiettivo di una produzione sostenibile. Oltre all'ambito accademico il tema della schedulazione ha una grande applicazione nel mondo industriale, di conseguenza chi gestisce impianti di produzione potrebbe servirsi delle formulazioni proposte per ottenere un bilanciamento tra produttività e sostenibilità.

La tesi è così strutturata, nel capitolo 1 si è presentato cosa si intende per sostenibilità e cosa è possibile fare per produrre in modo sostenibile. Inoltre si è delineato il ruolo dell'energia durante il processo di fabbricazione e sono state illustrate le politiche utilizzate per spingere i consumatori ad un utilizzo più sostenibile, con particolare attenzione alla politica Time-of-Use (TOU). Nel capitolo 2 si presenta in generale il problema dello scheduling, descrivendone le sue caratteristiche e le notazioni utilizzate. Successivamente si pone l'attenzione sul tema principale della tesi: l'energy efficient scheduling, approfondendo l'argomento con una dettagliata analisi dello stato dell'arte attuale. Nel capitolo 3, il fulcro della tesi, si analizzano una ad una le quattro formulazioni matematiche descrivendone tutti i loro dettagli. Infine nell'ultimo capitolo, il 4, si espone come sono stati implementati i quattro modelli e si commentano i risultati ottenuti.

Capitolo 2

Introduzione allo Scheduling

La programmazione dei lavori è un processo decisionale che viene utilizzato in molte industrie manifatturiere, ha un ruolo importante nella maggior parte dei sistemi di produzione, ma risulta significativo anche in ambiti di trasporto e distribuzione e in altri tipi di industrie e di servizi.

In generale possiamo descrivere il problema dello scheduling nell'assegnamento di risorse alle attività produttive in determinati periodi di tempo con l'obiettivo di ottimizzare uno o più obiettivi [20].

In linea di principio lo scheduling implica tre decisioni distinte [21]:

- Assegnamento: attribuire i lavori alle macchine che forniscono la tipologia di elaborazione richiesto;
- Sequenziamento: stabilire l'ordine di elaborazione per i lavori assegnati a ciascuna macchina;
- Tempistica: specificare gli istanti di tempo in cui deve iniziare l'elaborazione di ciascun lavoro sulla macchina assegnata.

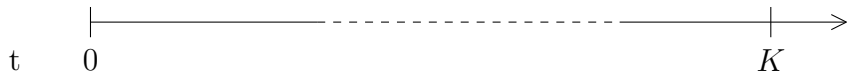
Le risorse e le attività (chiamate il più delle volte "job" o "tasks") possono assumere molte forme diverse, per esempio le risorse possono essere macchine in un impianto manifatturiero, squadre/gruppi di lavoratori in un

cantiere, piste di un aeroporto, unità di elaborazione in ambito informatico e così via. Invece le attività possono esser intese come operazioni in un processo di produzione, decolli e atterraggi in un aeroporto, fasi di un progetto di costruzione, esecuzioni di programmi di un computer e via dicendo... I job possono presentare diverse caratteristiche/proprietà, come un certo livello di priorità, uno "start-time" (istante di inizio), "set-up time" (tempo di configurazione), una "due date" (data di scadenza), ecc. ecc... Anche gli obiettivi di un problema di scheduling possono assumere molte forme diverse ad esempio si può cercare la minimizzazione del tempo di completamento ("makespan") oppure la minimizzazione del numero di attività completate dopo la loro due date.

Questo capitolo sarà così strutturato: nella sezione 2.1 vi è una introduzione sulle caratteristiche e sulle notazioni utilizzate in un problema di scheduling, con un approfondimento sulle possibili casistiche del problema. Nella sezione successiva, la 2.2, vi è un approfondimento sul problema sul quale sarà focalizzata questa tesi, partendo da una aggiunta di notazioni nella sottosezione 2.2.1 per poi inquadrare il tema con uno studio sullo stato dell'arte, suddiviso nelle casistiche del problema rispettivamente nelle sezioni 2.2.2 e 2.2.3.

2.1 Caratteristiche e notazione

In tutti i problemi di programmazione il numero di job e il numero di macchine vengono assunti come finiti. Il numero dei job è denotato con N, mentre il numero delle macchine da M [20]. Oltre a questi due numeri finiti solitamente si aggiunge anche un numero, indicato con K, che indica la lunghezza dell'orizzonte temporale disponibile per pianificare lo scheduling. Se definito significa che l'orizzonte temporale risulta finito. L'orizzonte temporale può essere rappresentato in modo continuo o in modo discreto, nella maggior parte dei problemi che verranno studiati è considerato in modo discreto. Quando



(a) Orizzonte temporale finito e continuo

(b) Orizzonte temporale finito e discreto

Figura 2.1: Esempio di due orizzonti temporali finiti

risulta discreto esso viene suddiviso in K istanti discreti chiamati "time slot" (vedi Fig. 2.1).

Conseguentemente, prendendo come esempio un'orizzonte temporale discreto, grazie ai tre numeri naturali interi N, M e K si definiscono i tre insiemi:

- $\mathcal{J} = \{1, 2, \ldots, N\}$ l'insieme degli N job da eseguire
- $\mathcal{H} = \{1, 2, \ldots, M\}$ l'insieme delle M macchine
- $\mathcal{T} = \{1, 2, \ldots, K\}$ l'insieme dei K time slot che costituiscono l'orizzonte temporale

Un altra caratteristica è che ad ogni job è attributo un numero intero positivo denominato "processing time" $p_{jh} \leq K, j \in \mathcal{J}, h \in \mathcal{H}$, che corrisponde alla durata dell'esecuzione del job $j \in \mathcal{J}$ sulla macchina $h \in \mathcal{H}$. Nel caso in cui il processing time non dipenda da h (caso macchine identiche in parallelo Pm o problema con una sola macchina) si scrive solamente p_j.

Le differenti caratteristiche dei job, delle macchine e del problema possono esser descritte tramite l'utilizzo dello schema di classificazione a 3-campi ("3-field problem classification" [22]): la terna $\alpha \mid \beta \mid \gamma$.

- Il campo α descrive l'ambiente della macchina.

- Il campo β fornisce dettagli sui vincoli e sulle restrizioni del problema, può contenere nessuna voce, una singola voce o più voci.

- Infine il campo γ descrive l'obiettivo da essere minimizzato, può contenere una o più voci, nel caso di più voci avremo la definizione di un problema multi-obiettivo.

Questa modalità descrittiva delle caratteristiche del problema segue ciò che è presentato in *Scheduling* di Pinedo [20].

2.1.1 Campo α

Vediamo le possibili casistiche sull'ambiente delle macchine:

Macchina singola (1): Il caso di una macchina singola è il più semplice di tutti i possibili ambienti macchina ed è un caso speciale di tutti gli altri ambienti più complessi.

Macchine identiche in parallelo (Pm): Vi sono M macchine in parallelo aventi tutti la stessa velocità (per quello chiamate identiche), per cui il processing-time di un job non dipende in che macchina viene eseguito: $p_j, j \in \mathcal{J}$.

Macchine uniformi in parallelo (Qm): Vi sono M macchine in parallelo aventi diverse velocità. La velocità della macchina h è indicata come v_h. Un job $j \in \mathcal{J}$ è caratterizzato da un processing-time p_j, se j viene eseguito sulla macchina $h \in \mathcal{H}$ allora il tempo di esecuzione sarà p_j/v_h. É possibile ricondursi al caso precedente avendo tutte le macchine con la stessa velocità, $v_h = 1$ per tutte le macchine, ottenendo così $p_{jh} = p_j$.

Macchine non correlate in parallelo (Rm): Questo caso è un ulteriore generalizzazione del caso precedente. Ci sono M macchine non correlate in parallelo. La macchina $h \in \mathcal{H}$ può eseguire il job j alla velocità v_{jh}. Il tempo d'esecuzione per il job j sulla macchina h sarà così definito: $p_{jh} = p_j/v_{jh}$. Per ricondursi alla casistica Qm le velocità delle macchine devono essere

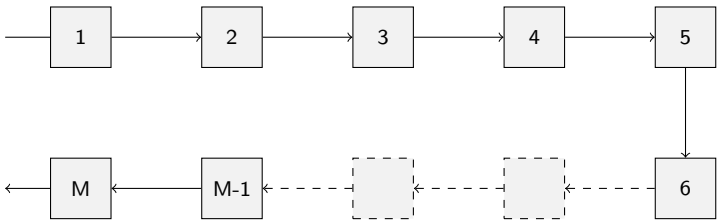


Figura 2.2: Rappresentazione del *Flow Shop*: le frecce rappresentano il percorso obbligato che ogni job deve seguire; le macchine sono raffigurate con i quadrati numerati da 1 a M. Si nota che ogni job passa attraverso ogni macchina solo una volta. Un job uscito dalla macchina M può considerarsi terminato.

indipendenti dal job, ottenendo $v_{jh} = v_j$, esattamente come la definizione del caso precedente.

Flow Shop (Fm): Vi sono M macchine in serie. Ogni job $j \in \mathcal{J}$ deve esser processato su ognuna delle M macchine e tutti i job devono seguire lo stesso ordine di esecuzione, ovvero devono essere processati sulla macchina 1, poi sulla macchina 2, e così via... Dopo il completamento dell'esecuzione su una macchina il job andrà in coda per la prossima macchina (vedi Fig. 2.2). Le code possono essere gestite secondo diverse modalità, per esempio FIFO ("First In First Out"), LIFO ("Last In First Out'"), ecc. ecc...

Job Shop (Jm): Un officina ("job-shop") con M macchine nella quale ogni job j ha da seguire il suo predeterminato ordine di esecuzione, cioè un ordine di passaggi tra le macchine presenti. Solitamente ogni macchina compie una funzione specifica.

2.1.2 Campo β

Il secondo campo indica le caratteristiche dei job, si elencano le proprietà più diffuse ed utilizzate:

Preemption ($pmtn$): Indica l'ammissione di "preemption", ovvero l'esecuzione del job può esser frammentata ("job-splitting"), come visibile nella

(a) Preemptive (b) Non - preemptive

Figura 2.3: Esempio per rappresentare la differenza tra uno scheduling preemptive e non-preemptive

figura 2.3. L'esecuzione di un job può quindi esser interrotta prima del suo completamento e ripresa successivamente. Se nel campo β non viene indicato *pmtn* allora non è ammessa la *preemption*.

Relazione di precedenza (*prec*): Viene specificata una relazione di precedenza tra i job. Si richiede l'esecuzione completa di un o più job prima di iniziare il processo di un altro job. Se *prec* non è indicato i job non sono soggetti a vincoli di precedenza.

Release-time (r_j): Il job $j \in \mathcal{J}$ non può esser processato prima del suo "release time": r_j. Come in precedenza se non è presente r_j nel campo allora i job possono esser processati in qualsiasi istante. Le due date (date di scadenza) contrariamente ai release-time non vengono indicate nel campo β, seppur di simile significato, perché la tipologia di funzione obiettivo riporterà indicazioni sufficienti se vi sarà o no la presenza di due date [20].

Produzione per lotti ($batch(b)$): Una macchina può eseguire al massimo un numero b di job contemporaneamente, permettendo una produzione a lotti. Il processing-time del batch sarà caratterizzato dal job con il processing-time più grande. Se $b = 1$ ci si riconduce facilmente allo scheduling senza l'utilizzo di lotti ("batch"), invece se $b = \infty$ significa che non vi sono limiti riguardo al numero di job che la macchina può processare in contemporanea.

On/Off (on/off): Le macchine sono sempre state considerate disponibili a processare un job se non occupate in una esecuzione di un altro job. Se invece nel campo β è presente on/off le macchine possono avere due stati: accesa o spenta, naturalmente quando una macchina risulta spenta non è disponibile per l'esecuzione di qualsiasi job. Esempio vedi Fig. 2.4.

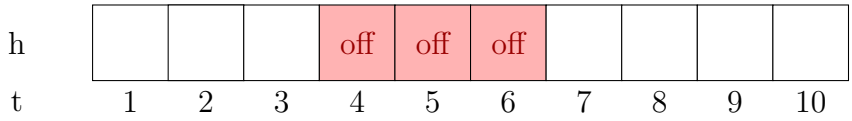


Figura 2.4: Esempio di programmazione con una macchina avente degli istanti di spegnimento: durante i time slot $\in \{4, 5, 6\}$ non è possibile eseguire nessun job

Peso (w_j): Ad ogni job $j \in \mathcal{J}$ viene associato un peso w_j. Può esser utilizzato come indice di importanza per i job, ad esempio dati due job $j, j' \in \mathcal{J}, j \neq j'$ con $w_j > w_{j'}$ allora il job j risulta più importante del job j' e la sua esecuzione probabilmente risulterà effettuata prima del job j'.

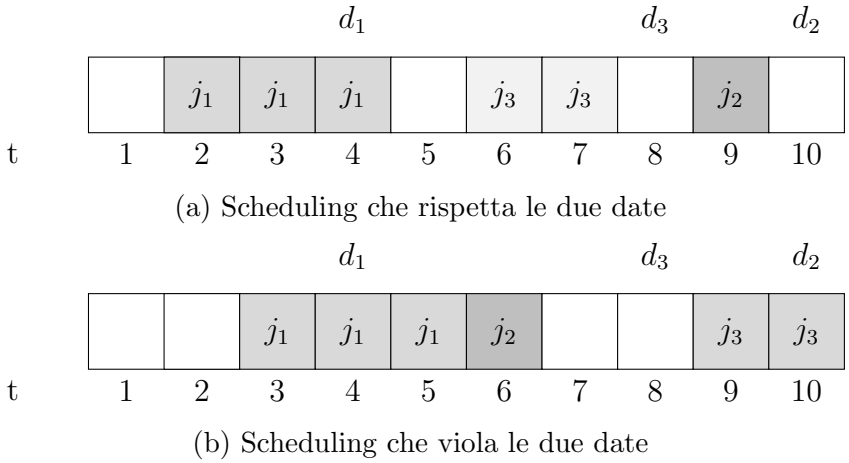
Permutazioni ($prmu$): Un vincolo che può apparire nella casistica Flow Shop (Fm) è che le code precedenti ad ogni macchina operino secondo la disciplina FIFO, implicando che l'ordine (o permutazione) in cui i lavori passano attraverso le macchine sia lo stesso per ogni step del processo.

Per completezza e comprensione delle due date si rappresenta la loro applicazione tramite la figura 2.5, nella quale si trovano due esempi di scheduling esplicativi.

2.1.3 Campo γ

Il terzo campo è dedicato al criterio di ottimizzazione scelto. Nella maggior parte dei problemi di scheduling consiste nella minimizzazione di una funzione, ma è anche possibile trovarne più di uno, in quel caso saranno risulteranno problemi multi-obiettivo. Si elencano una parte delle possibili funzioni obiettivo:

Makespan (C_{max}): Il "makespan" viene definito come il tempo che intercorre tra l'inizio della lavorazione del primo job e il completamento dell'ultimo job. $C_{max} = \max(C_1, C_2, \ldots, C_N)$ con $C_1, \ldots, C_N$ "completion-time" (tempo di completamento) di ogni job j. La minimizzazione del makespan



(a) Scheduling che rispetta le due date

(b) Scheduling che viola le due date

Figura 2.5: Esempi di scheduling con la presenza di due date: Nel caso (a) tutti e tre i job rispettano le loro due date, mentre nel caso (b) si hanno due job che non le rispettano: il job j_1 termina la sua esecuzione all'istante 5 quando la sua due date era fissata all'istante 4, mentre il job j_3 termina all'istante 10 non rispettando così la sua due date stabilita all'istante 8

implica un buon utilizzo delle macchine [20]. Nell'esempio mostrato nella figura 2.6 vi è il calcolo del makespan in uno scheduling.

Total flow time (F): Il "flow time" F_j di un job j è definito come la differenza tra il primo istante (start-time s_j) e l'ultimo istante d'esecuzione del job sulla macchina h: $F_j = C_j - s_j$. Di conseguenza il total flow time risulta: $F = \sum_{j \in \mathcal{J}} F_j$. Viene utilizzato quando le macchine sono uniformi o non correlate, perché il processing time del job j dipenderà dalla macchina in cui viene eseguito.

Maximum job lateness (L_{max}): Se in un problema sono prese in considerazione le due date allora è possibile definire la *lateness* del job j. Essa si determina come la differenza tra il l'istante in cui termina il processo del job j e la sua due date: $L_j = C_j - d_j$ (vedi Fig. 2.7a). Date $(L_1, \ldots, L_N)$ si definisce la massima lateness come $L_{max} = \max(L_1, \ldots, L_N)$, che misura la peggior violazione delle due date.

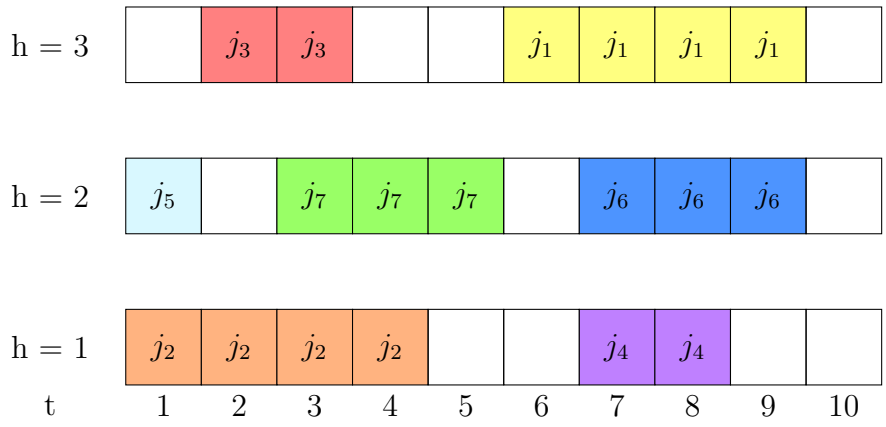


Figura 2.6: Esempio di calcolo del makespan: Si presenta uno scheduling di 7 job su 3 macchine. I completion time dei vari job sono: $C_1 = 9$, $C_2 = 4$,$C_3 = 3$, $C_4 = 8$, $C_5 = 1$, $C_6 = 9$, $C_7 = 5$. Il makespan si ottiene da $\max\{9, 4, 3, 8, 1, 9, 5\}$, quindi risulta $C_{\max} = 9$.

Maximum job tardiness(T_{max}): La *tardiness* del job j è così definita:

$$T_j = \max(C_j - d_j, 0) = \max(L_j, 0)$$

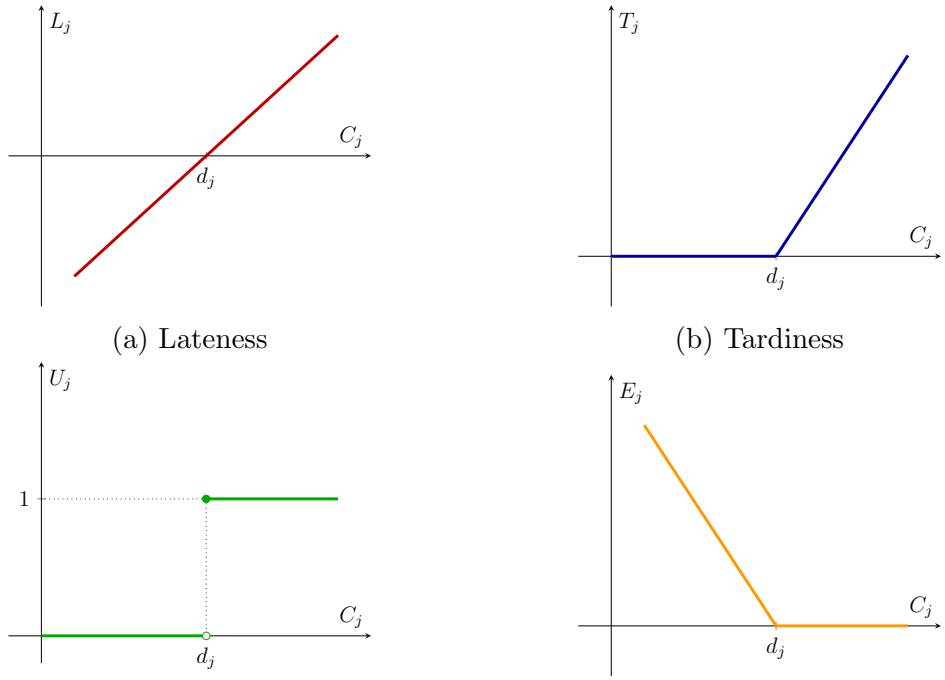
Come si può notare dalla figura 2.7 la differenza tra la lateness (2.7a) e la *tardiness* (2.7b) è che quest'ultima non è mai negativa. Date $(T_1, \ldots, T_N)$ si definisce la massima "tardiness" come $T_{max} = \max(T_1, \ldots, T_N)$, ed è pari al più grande ritardo accumulato da un job tra tutti gli eventuali ritardi.

Weighted number of tardy job ($\sum w_j U_j$) Per contare il numero di job in ritardo va definita la funzione "Unity penalty" (penalità unitaria):

$$U_j = \begin{cases} 1 & \text{se} \quad C_j > d_j \\ 0 & \text{altrimenti} \end{cases}$$

Rappresentata graficamente nella figura 2.7c. Calcolando $\sum_{j \in \mathcal{J}} U_j$ si ottiene esattamente il numero di job completati in ritardo. Ponendola come obiettivo di un problema di minimizzazione si otterrebbe come risultato uno scheduling

con il numero minore di job in ritardo possibile. Se è pure definito il peso per ogni job (w_j) allora possiamo definire la "Weighted number of tardy job" come segue: $\sum_{j \in \mathcal{J}} w_j U_j$. Minimizzando questa funzione si riduce al minimo il peso totale dei job completati in ritardo.



(a) Lateness

(b) Tardiness

(c) Penalità unitaria

(d) Earliness

Figura 2.7: Raffigurazione di quattro funzioni dipendenti dalla due date

Tutte le funzioni obiettivo considerate in precedenza sono considerate come "regular performance measures" (misure di prestazioni/parametri regolari) [20]. Una regular performance measure è una funzione che risulta non-decrescente in $C_1, \ldots, C_N$. I ricercatori oltre allo studio di questa tipologia di funzioni, si sono dedicati anche alle funzioni che non risultano "regular", per esempio se un job $j \in \mathcal{J}$ ha una due date d_j, può esser introdotto ed associata al job una penalità di "earliness", definendo earliness di

un job $j \in \mathcal{J}$ (vedi Fig.2.7d):

$$E_j = \max(d_j - C_j, 0)$$

Il valore E_j aumenta se il job j viene eseguito in anticipo alla sua due date, più sarà in anticipo più la penalità aumenta, difatti E_j risulta non-crescente in C_j. Naturalmente anche una funzione obiettivo contenente sia earliness che tardiness:

$$\sum_{j \in \mathcal{J}} E_j + \sum_{j \in \mathcal{J}} T_j$$

risulta non regular.

Come verrà descritto successivamente esistono anche funzioni obiettivo specifiche ad un certa tipologia di problema, generalmente create e pensate ad-hoc per una tipologia problema di scheduling.

2.2 Energy-efficient Scheduling

Oltre alle funzioni obiettivo descritte nella sezione precedente possono esser presenti funzioni obiettivo specifiche all'applicazione del problema di scheduling. Per quanto riguarda il settore manifatturiero uno degli aspetti di maggior rilievo è l'impiego di energia/elettricità durante il processo di produzione, basta pensare che durante la produzione le macchine rimangono per la maggior parte del tempo "idle" (inattive), ma consumando ugualmente circa l'80% di energia [23]. Un corretto impiego delle macchine è sicuramente un enorme potenziale di risparmio energetico. L' ottimizzazione dello scheduling era raramente considerato come uno strumento adatto per migliorare l'efficienza energetica, ma negli ultimi anni molti ricercatori usando approcci innovativi di programmazione hanno dimostrato che lo scheduling energeticamente efficiente è un modo efficace per ridurre il consumo di energia, persino con pochi investimenti di capitale [24][25]. Di conseguenza il numero di pubblicazioni e studi sullo scheduling efficiente dal punto di vista energetico (*Energy Efficient Scheduling* - EES) è in crescita [23, 26, 27]. Per conseguire un EES sono stati studiati e presentati diversi meccanismi di risparmio energetico, ad esempio il più semplice è lo spegnimento della macchina quando è in attesa per troppo tempo, o in attesa di un certo numero di job in modo da rilasciarli tutti insieme per la produzione per lotti.

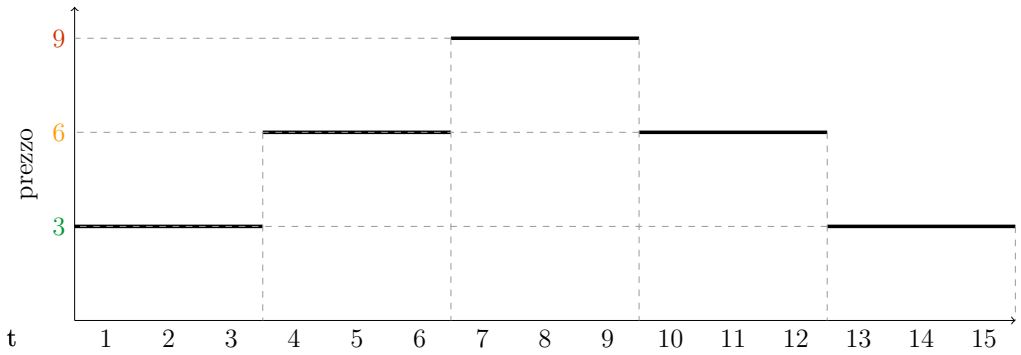
Oltre a questi meccanismi uno degli aspetti importanti da monitorare è la gestione del carico. Infatti, come presentato nel capitolo (1), i fornitori d'energia per gestire ed ottenere un corretto profilo di carico hanno presentato ed attuato diverse politiche. Una delle più famose, ma soprattutto una dei più studiate ed approfondite in ambito accademico [28](presentata in precedenza 1.3.1), è fondata sul tariffario basato sul "Time-Of-Use (TOU)". La sua applicazione può migliorare le "abitudini elettriche" degli utenti, riducendo così il carico della rete elettrica e la presenza di picchi di consumo. In contesto manifatturiero una risposta all'applicazione delle tariffe basate sul "Time-Of-

Use" consiste nella ri-schedulazione della produzione, per non andare incontro ad un aumento di costi per via di produzioni avvenute durante i picchi di carino. Il problema affrontato in questa tesi sarà esattamente di questa tipologia: un problema bi-obiettivo di scheduling con l'applicazione delle tariffe TOU.

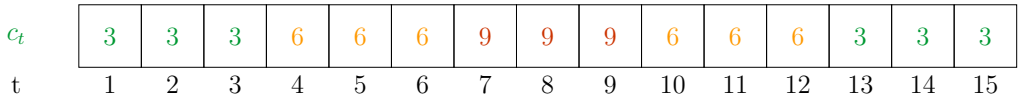
Nelle sezione successiva (2.2.1) verranno elencate le notazioni utilizzate per la definizione del problema e la sua struttura, mentre nelle sezioni 2.2.2 e 2.2.3 verranno presentati ed analizzati i diversi studi presenti su questo problema di programmazione sia con una macchina e sia con più di una macchina. Entrambi le sottosezioni 2.2.2 e 2.2.3 sono basate sull'indagine fatta da Catanzaro et al. [29].

2.2.1 Notazioni del problema

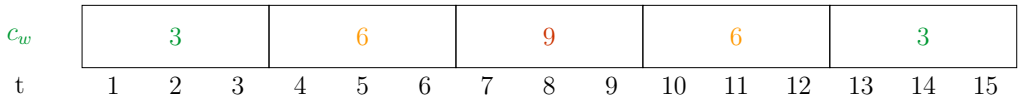
La struttura del problema affrontato risulterà simile al classico problema di scheduling, quindi con l'assegnamento di risorse ai job in determinati periodi di tempo e con l'obiettivo di minimizzare due funzioni [20] dove si terrà conto di requisiti temporali e, in questo caso, anche dei costi energetici. L'applicazione di tariffari basati sul TOU si traduce con la partizione dell'orizzonte temporale in time slot, dove ad ogni time slot viene associato un costo non negativo chiamato *TOU cost* [30] o *TOU price* [31], esattamente come nella definizione del tariffario "Time-Of-Use". La somma dei *TOU cost* durante i quali vi è associata l'esecuzione dei job viene nominata *Total Energy Cost (TEC)* (costo energetico totale), questa somma verrà utilizzata come funzione obiettivo con lo scopo di minimizzarla. Il più delle volte quest'ultima verrà utilizzata in aggiunta ad una delle funzioni obiettivo *regular*, elencate in precedenza (2.1.3), come il makespan, creando così un problema multi-obiettivo.



(a) Esempio rappresentativo dell'andamento del prezzo dell'energia in base alla politica TOU



(b) Applicazione dei time slot



(c) Applicazione dei time period

Figura 2.8: Esempio di applicazione dei *Time cost* (2.8b) e i *Time period cost* (2.8c) partendo da un tariffario TOU definito su un orizzonte temporale finito

Come detto in precedenza in questa tipologia di problemi l'orizzonte temporale viene partizionato, sono due le tipologie di suddivisione più diffuse. Nella prima si utilizza il concetto di time slot, quindi un orizzonte temporale di lunghezza K viene suddiviso in K slot ed ad ogni slot si associa un costo energetico denominato $c_t, \forall t \in \mathcal{T}$. Invece nella seconda il l'orizzonte temporale non è più suddiviso in slot di lunghezza unitaria, ma in "time period" [28]. Un time period corrisponde ad una porzione dell'orizzonte temporale e la somma delle lunghezze dei time period deve corrispondere a K. Ogni oriz-

zonte temporale può esser suddiviso in D time period ed ad ognuno di essi viene associato un costo energetico, caratteristico di quel periodo, denominato $c_k, \forall k \in \{1, 2, \ldots, D\}$. Nella figura 2.8 viene mostrato un esempio delle due tipologie di discretizzazione applicate con un tariffario TOU definito.

Oltre al costo per slot time o time period, si definisce anche il costo di utilizzazione della macchina, cioè un costo fisso dovuto all'utilizzo della macchina, questo costo viene definito: $u_h, \forall h \in \mathcal{H}$. Il Total Energy Cost dipenderà sia in che slot time/time period verranno processati i job e sia che macchine sono state utilizzate, ma verrà poi definito in base alla casistica di problema presa in considerazione durante l'analisi dello stato dell'arte, essendo presente qualche differenza tra la casistica di scheduling con macchina singola e la casistica con macchine multiple.

2.2.2 Analisi stato dell'arte TOU Scheduling con macchina singola

In questa sezione si presenta una classificazione delle pubblicazioni e studi sullo scheduling con l'applicazione delle tariffe TOU utilizzando la "three-field notation" [22] presentata precedentemente. Tramite la tabella 2.1 si mostrano le notazioni che verranno utilizzate per la classificazione, ricordando che il campo α caratterizza l'ambiente delle macchine, β descrive i vincoli e le restrizioni sui job e infine γ denota la funzione obiettivo. Verrà discusso lo sviluppo del TOU scheduling partendo dal caso a macchina singola (Pm) per poi passare al caso di macchine multiple (Qm e Rm), riassunte con le tabelle 2.2 e 2.3.

Tabella 2.1 Notazioni utilizzate per i problemi di scheduling seguendo la "three-field notation"

Campo	Caratteristiche	Significato
α	1	singola macchina
	Pm	m macchine identiche parallele
	Qm	m macchine con velocità differenti
	Rm	m macchine non omogenee
	Fm	Flow-shop con m macchine
	Jm	Job-shop con m macchine
β	r_j	release time
	d_j	due date
	w_j	pesi/importanza
	prmp	preemption
	batch(b)	produzione per lotti
	prmu	permutazioni
	prec	ordini di precedenza
	on/off	macchine con on/off
γ	C_{max}	makespan
	L_{max}	massima lateness
	T_{max}	massima tardiness
	$\sum F_j$	total flow time
	$\sum w_j C_j$	total weighted completion time
	$\sum w_j T_j$	total weighted tardiness
	$\sum w_j U_j$	weighted number of tardy jobs

Avendo una sola macchina disponibile per ottenere uno scheduling è possibile enunciare in modo generale il problema in questo modo:

> Ogni job $j \in \mathcal{J}$ deve esser assegnato ad un sotto insieme $\mathcal{T}_j \subseteq \mathcal{T}, |\mathcal{T}_j| = p_j$, con l'obiettivo di minimizzare il Total Energy Consumption (TEC). Che il questo caso può esser definito cosi:

$$TEC = \sum_{j \in \mathcal{J}} \sum_{t \in \mathcal{T}_j} c_t$$

Uno dei primi studi sul TOU scheduling con macchina singola fu proposto da Wan e Qi [32], loro hanno studiato cinque versioni differenti del problema, differenziate dal tipo di funzione obiettivo scelta. Nel particolare sono state prese in considerate: *total flow time*,*maximum lateness*, *maximum tardiness*, *weighted number of tardy job* e *total tardiness*. Formalmente gli autori hanno presentato e analizzato i seguenti cinque problemi (utilizzando *three-field notation*): $1|\ |\sum_{j\in\mathcal{J}} C_j + TEC$, $1|\ |L_{max} + TEC$, $1|\ |T_{max} + TEC$, $1|\ |\sum_{j\in\mathcal{J}} w_j U_j + TEC$ e infine $1|\ |\sum_{j\in\mathcal{J}} T_j + TEC$. Wan e Qi hanno proposto, sotto specifiche assunzioni, algoritmi in tempo polinomiale, per esempio: considerando il problema $1|\ |\sum_{j\in\mathcal{J}} C_j + TEC$, avente i time slot costs in ordine non-crescente rispetto a $t \in \mathcal{T}$, allora è presente almeno uno scheduling ottimale nel quale i lavori seguano l'ordine "Shortest Processing Time First (SPT)"[1]. Partendo da questa proprietà gli autori dimostrano che il problema $1|\ |\sum_{j\in\mathcal{J}} C_j + TEC$ è risolvibile in $O(K^2)$ con l'assunzione di time slot costs non-crescenti rispetti a $t \in \mathcal{T}$. Inoltre nella pubblicazione viene dimostrato che tutti e cinque i problemi risultano NP-hard [33] tramite

[1]La regola SPT ordina i lavori in ordine crescente secondo il loro processing time: presi due job qualsiasi j e j' con $j < j'$ allora $p_j < p_{j'}$. Ogni volta che si libera una macchina, inizia l'elaborazione del lavoro più breve pronto in quel momento

una riduzione del "3-partition problem"[2].

Una modalità diffusa per considerare simultaneamente sia il costo delle time slot e le funzioni obiettivo di scheduling tradizionali è quello di minimizzare la somma ponderata dei due obiettivi, questo ragionamento viene applicato da Zhong e Liu [34]. Gli autori considerano un problema di scheduling in cui l'obiettivo è quello di minimizzare una combinazione lineare del makespan e dei costi totali degli slot di tempo (TEC). Essi dimostrano che il problema è fortemente NP-hard e analizzano un caso speciale con costi di slot di tempo non crescenti, esattamente come visto in [32].

Kulkarni e Munagala [35] hanno studiato le varianti dinamiche di tali problemi, dove i job vengono rilasciati nel tempo e le decisioni di programmazione devono essere prese online con l'obiettivo di minimizzare una semplice combinazione lineare del "total weighted completion time" e del TEC: $\sum_{j\in\mathcal{J}} w_j(C_j - r_j) + TEC$.

Penn e Raviv [36] invece hanno considerato due varianti principali del problema: minimizzazione dei costi e la massimizzazione del profitto. Nel problema di minimizzazione dei costi, l'insieme dei job da elaborare è definito e l'obiettivo è quello di pianificare tutti i lavori entro un orizzonte di pianificazione $(1, \ldots, K)$ in modo da minimizzare il costo totale, mentre nel problema di massimizzazione del profitto, si deve selezionare un insieme di lavori da elaborare in modo che il profitto totale sia massimizzato. I casi generali dei problemi di minimizzazione dei costi e di massimizzazione dei profitti in cui le preemptions sono proibite sono fortemente NP-hard, per

[2]Dato un insieme A di numeri interi positivi, determinare se può essere partizionato in tre sottoinsiemi disgiunti che abbiano tutti la stessa somma.

Per esempio:

$$A = \{7, 3, 2, 1, 5, 4, 8\}$$

Possiamo partizionare A in tre partizioni, ognuna delle quali ha una somma di 10.

$$A1 = \{7, 3\} \quad A2 = \{5, 4, 1\} \quad A3 = \{8, 2\}$$

questo mostrano che alcuni casi speciali con tempi di elaborazione identici possono essere risolti da algoritmi efficienti ($1|p_j = 1|TEC$ risolvibile in $O(K^3)$). Inoltre gli autori hanno preso in considerazione anche la possibilità di preemption, dimostrando che esiste un algoritmo che risolve il problema $1|prmp, d_j, r_j|TEC$ in tempo $O(\log_2 K(K + N^2 \log_2 K))$, mentre per il problema $1|prmp|TEC$ hanno provato che è risolvibile in $O(K \log_2 K + N)$.

Che et al. [37] hanno sviluppato un modello MILP a tempo continuo e hanno proposto un'euristica greedy per il problema: $1|\ \ |TEC$. Nel loro studio l'orizzonte temporale non viene suddiviso in time slot di lunghezza unitaria, ma in time period. Nell'euristica proposta, i job sono inseriti nei periodi di tempo disponibili uno dopo l'altro in ordine non crescente secondo i loro consumi di elettricità e ogni job viene inserito nel periodo (o nei periodi) con il minimo costo di elettricità costo dell'elettricità, risolvendo così il problema in tempo $O(N^2K^2)$. Inoltre viene analizzato un caso di studio reale di un'azienda cinese, il quale rivela che il costo totale dell'elettricità può essere ridotto di circa il 30% utilizzando il loro algoritmo.

L'articolo pubblicato da Che et al. [38] considera un problema di scheduling a macchina singola con meccanismo di spegnimento per minimizzare sia il consumo totale di energia che il massimo ritardo, rappresentato da $1|on/off|TEC,\ T_{max}$. L'obiettivo è quello di trovare una sequenza ottimale di lavori e determinare se sulla macchina deve essere eseguito lo spegnimento tra il processo di due lavori consecutivi. Per formulare il problema, viene sviluppato un modello di programmazione lineare misto-intero (MILP) avente due obiettivi. Poi viene proposto un algoritmo basato sul metodo ϵ - constraint [39] per risolvere il problema e per ottenere la frontiera di Pareto.

Tabella 2.2 Suddivisione per classi dei problemi di scheduling con singola macchina suddivisi

Classe	Problema	Autori
Scheduling con una macchina e un solo obiettivo	$1\|\ \ \|\sum_{j\in\mathcal{J}} C_j + TEC$	Wan e Qi, 2010
	$1\|\ \ \|L_{max} + TEC$	
	$1\|\ \ \|T_{max} + TEC$	
	$1\|\ \ \|\sum_{j\in\mathcal{J}} w_j U_j + TEC$	
	$1\|\ \ \|\sum_{j\in\mathcal{J}} T_j + TEC$	
	$1\|q_j, s_j \in \mathbb{R}_{0^+}, w_j\|TEC$	Fang et al., 2016
	$1\|prmp, q_j, s_j \in \mathbb{R}_+\|TEC$	
	$1\|prmp, speed, q_j, s_j \in \mathbb{R}_{0^+}, w_j\|TEC$	
	$1\|prmp, w_j\|\sum_{j\in\mathcal{J}} w_j C_j + TEC$	Chen et al., 2021
	$1\|prmp, d_j, r_j\|TEC$	Penn e Raviv, 2021
	$1\|prmp, r_j, w_j\|\sum_{j\in\mathcal{J}} w_j(C_j - r_j) + TEC$	Kulkarni e Munagala, 2013
	$1\|s_j \in \mathbb{R}_{0^+}\|TEC$	Chen e Zhang, 2018
	$1\|q_j, s_j \in \mathbb{R}_{0^+}\|TEC$	Che et al., 2016
	$1\|batch(b)\|TEC$	Cheng et al., 2014
		Cheng, Junheng et al., 2016
	$1\|on/off\|TEC$	Shrouf et al., 2014
		Aghelinejad et al., 2016
		Aghelinejad et al., 2018
		Aghelinejad et al., 2019
	$1\|on/off, prmp\|TEC$	Aghelinejad et al., 2017
	$1\|on/off, seq, q_j\|TEC$	Aghelinejad et al., 2018
	$1\|on/off, speed, seq, q_j\|TEC$	
Scheduling multi obiettivo con una sola macchina	$1\|prmp\|C_{max}, TEC$	Rubaiee e Yildirim, 2019
	$1\|prmp\|C_max, TEC$	Rubaiee et al., 2019
	$1\|batch(b)\|C_{max}, TEC$	Chen e Zhang, 2018
		Chen et al., 2021
		Zhang et al., 2018
		Wu et al., 2021
	$1\|batch(b), r_j\|C_{max}, TEC$	Zhou et al., 2020

Un passo avanti nello studio di questo campo è stato realizzato da Fang et al. [40] studiando problemi con job che hanno richieste di potenza $q_j, j \in \mathcal{J}$. Dal punto di vista della produzione, l'introduzione delle richieste di potenza riflette la necessità di suddividere in due i lavori: quelli ad alto dispendio energetico e non. Se un job $j \in \mathcal{J}$ è processato durante un periodo di time slot $\mathcal{T}_j \in \mathcal{T}$, il costo dell'esecuzione del job j risulta $q_j \sum_{t \in \mathcal{T}_j} c_t$. Gli autori hanno considerato il problema $1|prmp, q_j, s_j \in \mathbb{R}_+|TEC$ e hanno dimostrato che può esser risolto con un algoritmo greedy che sfrutta il fatto che, in una scheduling ottimale, i job ad alto dispendio energetico sono associati a fasce orarie con un costo inferiore. Oltre a questo problema, Fang et al. hanno approfondito e studiato il problema $1|prmp, speed, q_j, s_j \in \mathbb{R}_{0^+}, w_j|TEC$, dove le diverse parti di un job possono esser processato a diverse velocità. Nello specifico se una parte o l'intera elaborazione di un job $j \in \mathcal{T}$ avviene in time slot $t \in \mathcal{T}$ a velocità v_t con carico di lavoro $w_{j,t}$, allora il processing-time di j in t è $p_{j,t} = w_{j,t}/v_t$, e la sua richiesta energetica risulta $q_j = v_t^\delta$ in t, per qualche costante fissata $\delta > 0$. Gli autori hanno fornito un'espressione esplicita per il carico di lavoro e per la velocità di ogni lavoro ad ogni $t \in \mathcal{T}$ formulando il problema come un problema a funzione convessa, inoltre hanno presentato un algoritmo che lo risolve il tempo polinomiale.

Anche Rubaiee e Yildirim [50] hanno considerato un problema di scheduling preemptive. A differenza degli autori citati in precedenza hanno introdotto, assieme al TEC, un secondo obiettivo che risulta in conflitto con il TEC: la minimizzazione del makespan. Rubaiee e Yildirim hanno fornito una formulazione MILP e implementazione adattata dell'algoritmo "Ant Colony Optimization (ACO)". Oltre al problema presentato dove è permessa la preemption, una parte degli stessi autori, Rubaiee et al., hanno studiato il problema di scheduling non preemptive rappresentato da un problema multi-obiettivo, cercando di minimizzare la total tardiness e il TEC, Per soddisfare questi obiettivi hanno sviluppato nuovi algoritmi genetici, nel particolare il modello è stato risolto attraverso algoritmi genetici (GA) basati su *domi-*

nance rank (GA-1), *weighted sum aggregation* (GA-2), *dominance ranking procedure and crowding distance comparison* (GA-3) e infine un approcio euristico (GA-4).

Chen et al. [41] hanno focalizzato i loro studi su un problema bi-obiettivo, nel particolare hanno presentato i problemi: $1|prmp|\sum_{j\in\mathcal{J}} C_j + TEC$ (e il caso con più macchine non omogenee: $Rm|prmp|\sum_{j\in\mathcal{J}} C_j + TEC$) e $1|prmp, seq, w_j|\sum_{j\in\mathcal{J}} w_j C_j + TEC$. Gli autori hanno dimostrato l'esistenza di un algoritmo che risolve il primo e il secondo problema in tempo polinomiale, mentre per il terzo un algoritmo di approssimazione (*polynomial-time approximation scheme* PTAS).

Chen e Zhang nella pubblicazione [30] hanno studiato una classe di problemi di scheduling di un insieme di job su una singola macchina sotto le tariffe TOU. L'obiettivo è quello di minimizzare TEC soggetto anche a criteri di scheduling "regular". Per ciascuno dei problemi della classe, stabiliscono la sua fattibilità computazionale. Per quelli che sono fattibili forniscono un algoritmo efficiente, mentre per quelli che non fattibili forniscono un algoritmo pseudo-polinomiale o uno schema di approssimazione in tempo polinomiale. Per esempio Chen e Zhang hanno presentato una FPTAS ("fully polynomial-time approximation scheme") per il problema $1|s_j \in \mathbb{R}_{0^+}|TEC$, grazie alla definizione di "valley" per un insieme di time slot, che corrisponde al time slot con il costo TOU minore rispetto a quelli adiacenti.

Nelle pubblicazioni più recenti sono stati presi in considerazione sempre di più problemi di scheduling multi-obiettivo, così da avvicinarsi sempre di più alla realtà industriale dove sono presenti più obiettivi in conflitto. Un articolo che utilizza questa metodologia è stato pubblicato da Cheng et al. [42]. Questo articolo studia un problema di scheduling per lotti bi-obiettivo a macchina singola con l'applicazione della politica di prezzo TOU: il primo obiettivo è quello di minimizzare il makespan e il secondo è quello di minimizzare i costi totali di elettricità (TEC). Il problema considerato è prima formulato con un modello di programmazione non lineare, il quale analizzan-

do le proprietà viene linearizzato. Il suo corrispondente linearizzato viene utilizzato per applicare il metodo "ϵ-constraint" per ottenere la frontiera di Pareto. Successivamente Cheng, Junheng et al. in una pubblicazione seguente [43] mostra un miglioramento di ciò che aveva pubblicato in precedenza [42]. Presenta il problema con un modello di programmazione lineare intera (MILP) bi-obiettivo. Successivamente, il MILP è semplificato analizzando le sue proprietà. Poi, un metodo "ϵ-constraint" è utilizzato ed adattato per ottenere il la frontiera di Pareto. Infine, una soluzione preferibile è raccomandata ai decisori tramite un approccio basato sulla logica fuzzy. I risultati di calcolo su istanze generate casualmente mostrano l'efficacia dell'approccio proposto, migliorando così da un punto di vista computazionale ciò che aveva mostrato in [42].

Rimanendo sempre con l'utilizzo della produzione per lotti (batch) è presente un altro studio scritto da Zhou et al. [54] dove oltre l'utilizzo dei lotti considera anche le release-date. Nello specifico viene sviluppata una formulazione matematica per ottimizzare TEC e il makespan. A causa della complessità computazionale, viene sviluppato un algoritmo meta-euristico per trovare la frontiera di Pareto. Inoltre vengono presentate due euristiche presentate per raggruppare i lavori in lotti e due approcci diversi per migliorare il costo totale dell'elettricità. Le performance del modello proposto e degli algoritmi sono valutate attraverso ampi esperimenti numerici.

L'articolo pubblicato da Zhang et al. affronta un problema di scheduling a lotti su macchina singola, sempre con la presenza di TOU-cost, con gli obiettivi di minimizzare il costo totale dell'energia (TEC) e il tempo di esecuzione, risultando anch'esso un problema bi-obiettivo. Nella pubblicazione per prima cosa vengono anche calcolati e dichiarati i limiti inferiori e superiori sul numero di lotti, successivamente viene proposto un modello di programmazione lineare misto-integrale a tempo continuo, che migliora un modello a tempo discreto esistente in letteratura ≪Bi-objective optimization of a single machine batch scheduling problem with energy cost considera-

tion» di Wang et al. [55]. Due euristiche migliorate vengono proposte sulla base del modello migliorato. Gli esperimenti computazionali (su istanze di grandi dimensioni) dimostrano che il modello migliorato e l'euristica possono essere eseguiti centinaia di volte più velocemente rispetto a quelli esistenti.

Sullo stesso problema, $1|batch(b)|C_{max}, TEC$, si presenta un articolo di Wu et al. [53]. L'idea di Wu et al. è quella di trasformare il problema bi-obiettivo in una serie di problemi mono-obiettivo che vengono risolti con un'euristica per ottenere un fronte di Pareto approssimativo. In particolare, per ogni problema mono-obiettivo trasformato, vengono proposti due nuovi algoritmi euristici che risolvono una serie di "problemi dello zaino" e "dello zaino 0-1" [3]. I risultati computazionali su 145 istanze di benchmark mostrano che gli algoritmi proposti sono abbastanza efficienti e che con un massimo di 200 lotti gli algoritmi sono in grado di trovare soluzioni di Pareto di alta qualità.

Un altra casistica particolare presa in considerazione in letteratura è la possibilità di spegnere ed accendere le macchine tramite il meccanismo *on/off*. Oltre ai due stati accesa o spenta è possibile considerare anche quando la macchina è in attesa, di conseguenza gli stati risultano:

- "In esecuzione"
- "In attesa"
- "Spenta"

[3]Dato un insieme di elementi, ciascuno con un peso e un valore, determinare il numero di ogni elemento da includere in una zaino in modo che il peso totale sia inferiore o uguale a un dato limite (capacità dello zaino) e il valore totale sia il più grande possibile. In maniera più formale il problema si presenta così:

- N oggetti caratterizzati da un peso w_i e un valore c_i
- W peso massimo supportato dallo zaino
- variabile decisionale x_i che esprime se l'oggetto è inserito o no nello zaino

Con funzione obiettivo da massimizzare: $\sum_{i=1}^{N} c_i x_i$ e vincolo: $\sum_{i=1}^{N} w_i x_i \leq W$.

Nel caso $x_i = \{0, 1\} \forall i = 1, \ldots, N$ si parla di problema dello zaino 0-1, dove quindi un oggetto può esserci o non esserci senza nessuna ripetizione.

Una macchina nello stato "In attesa" consuma energia, portando così ad un costo, ma comunque è pronta per l'esecuzione di un job appena gli verrà assegnato. Invece se la macchina risulta "Spenta" allora il suo consumo energetico è nullo, ma naturalmente non può processare alcun job, deve esser portata prima nello stato "In attesa" richiedendo così un consumo d'energia. I primi studiosi che hanno approfondito questa casistica sono Shrouf et al., hanno pubblicato l'articolo ≪Optimizing the production scheduling of a single machine to minimize total energy consumption costs≫ [44] dove viene considerato il problema $1|on/off|TEC$, presentando sia una formulazione MILP e sia un algoritmo genetico. Il loro lavoro ha ispirato altri a pubblicare studi ed articoli sul medesimo argomento, come Aghelinejad et al. Aghelinejad et al. ha pubblicato diversi articoli su questa tipologia di problema. Partendo presentando una formulazione MILP del problema $1|on/off|TEC$ in [45], migliorato con l'applicazione di limiti inferiori in [47] e presentando un' euristica e un algoritmo genetico in [46]. Aghelinejad et al. hanno preso in considerazione anche il problema con la presenza di preemption, $1|on/off, pmrp|TEC$ nella pubblicazione [48], questo presupposto ha permesso lo sviluppo di un algoritmo, basato sulla programmazione dinamica, che risolve il problema in tempo polinomiale, precisamente in $O(K^3)$. In [49] hanno ulteriormente esteso il loro studio, andando a considerare le richieste di potenza dei vari job in una versione del problema avente velocità variabile e dove i job sono già sequenziati. Il problema può esser rappresentato formalmente: $1|on/off, speed, seq, q_j|TEC$. Anche in questo caso hanno adattato la programmazione dinamica per sviluppare un algoritmo per questo problema. Nello specifico grazie al loro algoritmo il problema viene risolto in $O(K^2V(K+V))$, dove per V si intende il numero di velocità di esecuzione distinte della macchina.

2.2.3 Analisi stato dell'arte TOU scheduling con macchine multiple

Nella sottosezione precedente sono stati elencati ed esplicitati gli studi e gli articoli più citati e significativi per un problema di scheduling energeticamente efficiente dove vi era presente una sola macchina disponibile per lo scheduling. Questa classe di problemi può esser generalizzata andando a considerare la possibilità di presenza di più macchine parallele, sia che esse siano identiche, uniformi o non correlate tra di loro. In questo caso il problema più generale possibile può esser così enunciato:

> Ogni job $j \in \mathcal{J}$ deve esser assegnato ad un sottoinsieme $\mathcal{T}_j \subseteq \mathcal{T}, |\mathcal{T}_j| = p_{jh}$ su una macchina $h \in \mathcal{H}$, con l'obiettivo di minimizzare il Total Energy Consumption (TEC). Indicando con $\mathcal{J}_h$ l'insieme dei job eseguiti sulla macchina h, il TEC dello scheduling può esser definito nel modo seguente:
>
> $$TEC = \sum_{h \in \mathcal{H}} u_h \sum_{j \in \mathcal{J}_h} \sum_{t \in \mathcal{T}_j} c_t$$
>
> Da notare che il problema è facilmente riducibile alla formulazione con una sola macchina ponendo $u_h = 1, \quad \forall h \in \mathcal{H}$.

Si parte dalla casistica in cui sono considerate macchine parallele identiche: Pm. Uno degli articoli più significativi risulta quello di Wang et al. [56]. In [56] viene affrontato il problema bi-obiettivo $Pm| \quad |C_{max}, TEC$. Nella loro pubblicazione vengono presentati sia una formulazione MILP che un'euristica. L'euristica è basata sul metodo ϵ-constraint e viene utilizzata soprattutto per le istanze di piccola così da ottenere la frontiera di Pareto esatta. Mentre per le istanze di media e grande dimensione sono proposte due soluzioni: un altro metodo euristico con una strategia di ricerca locale e un'applicazione dell'algoritmo NSGA-II. Entrambi riescono ad ottenere una

buona approssimazione della frontiera in un tempo ragionevole. Da sottolineare che comunque tutte le loro soluzioni sono state testate e validate sia con istanze generate casualmente e sia con un caso di studio del mondo reale.

Il lavoro di Wang et al. è stato migliorato da Anghinolfi et al. in [21]. Nella loro pubblicazione hanno presentato una formulazione MILP, sempre bi-obiettiva, migliorata rispetto a Wang et al. e un'euristica denominata "Split-greedy Heuristic" le cui soluzioni sono successivamente migliorate con un algoritmo di ricerca locale, chiamato "Exchange Search".

Sempre rimanendo sul problema $Pm|\ \ |C_{max}, TEC$ è presente la pubblicazione di Zeng et al. [57]. Per prima cosa viene presentata una loro formulazione MILP del problema, successivamente tenendo conto di un solo obiettivo, minimizzare il TEC, viene proposto un "Greedy Insertion Algorithm", cioè un algoritmo presentato da Che et al. in [37] per il caso a singola macchina ed esteso in questo articolo per il caso Pm. Per la risoluzione del problema bi-obiettivo quindi viene utilizzata una struttura iterativa sull'algoritmo proposto. Nel particolare partendo da un limite inferiore del makespan calcolato in precedenza, viene fissato il makespan e calcolato il TEC con il makespan fissato, nell'iterazione successiva si effettua la stessa operazione ma con un makespan maggiore. Con questa metodologia viene così trovata una frontiera di Pareto quasi ottimale per il problema bi-obiettivo. La struttura viene testate con calcolo su istanze reali e generate casualmente, dimostrando che l'approccio proposto è abbastanza efficiente e può trovare fronti di Pareto di alta qualità per problemi di grandi dimensioni con fino a 5000 job.

In [58] viene espanso il problema studiato da Wang et al. e da Anghinolfi et al. andando però a considerare uno scheduling per batch con release-times, formalmente $Pm|batch(b), r_j|C_{max}, TEC$. Presentano un modello MILP, ma considerando che il problema è fortemente NP-hard, è proposto un algoritmo di evoluzione differenziale multi-obiettivo per risolvere efficacemente il problema su larga scala. Le prestazioni dell'algoritmo proposto sono valuta-

te confrontandolo con il ben noto algoritmo NSGA-II e un altro algoritmo di ottimizzazione multi-obiettivo (AMGA). I risultati sperimentali mostrano che l'algoritmo proposto si comporta meglio di NSGA-II e AMGA in termini di qualità e di distribuzione della soluzione.

Sulla stessa tipologia di problema visto in precedenza si trova il lavoro di Rocholl et al. [59], ma a differenza di [58] assieme al TEC viene considerato il total weighted tardiness: $\sum_{j\in\mathcal{J}} T_j$; ottenendo così il problema: $Pm|batch(b), r_j|\sum_{j\in\mathcal{J}} T_j, TEC$. Viene formulato un modello MILP, ma vengono analizzato anche un caso speciale, dove tutti i job hanno la stessa dimensione, tutte le due date sono nulle e i job sono disponibili fin da subito. Questo caso porta ad un MILP più trattabile. Inoltre vengono progettate tre euristiche basate su algoritmi genetici di raggruppamento, che a loro volta sono incorporate in una struttura basata sul NSGA-II. Vengono condotti esperimenti computazionali basati su istanze del problema generate casualmente e viene dimostrato che le soluzioni ottenute sono di ottima "qualità".

Sempre considerando la produzione per lotti si trova il lavoro di Jia et al. [60]. Per risolvere il problema bi-obiettivo viene proposto un algoritmo basato su "ant colony optimization algorithm (ACO)". L'algoritmo proposto viene confrontato con gli algoritmi multi-obiettivi esistenti attraverso ampi esperimenti di simulazione. I risultati sperimentali indicano che l'algoritmo proposto da Jia et al. supera tutti gli algoritmi confrontati, specialmente per problemi su larga scala.

[61] Come mostrato nel capitolo precedente la macchine possono anche esser considerate differenti tra di loro, non più identiche (Pm). Nel caso in cui vengano considerate Qm, cioè uniformi, non sono presenti numerosi studi su questa casistica. Gli unici reperiti sono due: ≪A multi-objective evolutionary algorithm based on adaptive clustering for energy-aware batch scheduling problem≫ di Qian et al. e ≪Genetic algorithms with greedy strategy for green batch scheduling on non-identical parallel machines≫ di Tan et al. Nel lavoro di Qian et al. il problema risulta multiobiettivo:

$Qm|batch(b), r_j, p_j|C_{max}, TEC$.

A causa della complessità NP-hard del problema, viene proposto un algoritmo evolutivo multi-obiettivo basato sul "adaptive clustering", dove il metodo di clustering incorporato è utilizzato per estrarre informazioni sulla distribuzione delle soluzioni nello spazio di ricerca.

Invece per quanto riguarda nel lavoro di Tan et al. [62] si ha lo studio del un problema $Qm|batch(b)|TEC$ quindi con un solo obiettivo: minimizzazione del TEC. Per risolvere il problema, oltre alla formulazione MILP, sono proposti due tipi di algoritmi genetici: uno a popolazione singola ("Single Population Genetic Algorithms" - SPGA) e un altro algoritmo genetico a popolazione multipla ("Multi-Population Genetic Algorithm" - MPGA). Negli algoritmi, i job vengono suddivisi in lotti e poi assegnati alle macchine in modo casuale. Una strategia greedy è progettata per decidere la sequenza di produzione e l'istante di inizio di elaborazione dei lotti. Inoltre, viene proposta una strategia di aggiustamento autoadattabile ("Self-adaptive parameter adjustment") per migliorare l'adattabilità dell'algoritmo. Sono stati condotti esperimenti di calcolo per valutare le prestazioni degli algoritmi: su piccole istanze, sia SPGA che MPGA possono raggiungere risultati approssimativi rispetto a quelli ottenuti dal modello MILP, e possono anche raggiungere un TEC minore su istanze grandi con tempo minore di calcolo.

Uno dei primi lavori che esamina il problema con la presenza di m macchine parallele non correlate tra di loro cercando di minimizzare la somma di due obiettivi è quello di Moon et al. [63]. In questo caso i due obiettivi sono i classici makespan e TEC. Il problema formalizzato risulta così: $Rm| \quad |KC_{max} + TEC$. Viene presentata una formulazione MILP per il problema ed inoltre gli autori presentano un algoritmo genetico per risolvere i problemi computazionali della formulazione MILP. L'algoritmo genetico pensato da Moon et al. non può garantire l'ottimalità delle soluzioni. Per questo Cheng et al. nella loro pubblicazione [64] forniscono un modello MILP così da poter risolvere esattamente il problema. In aggiunta viene mostrato un

netto miglioramento dal punto di vista computazionale. Sempre su questo problema, $Rm|\ \ |KC_{max} + TEC$, Kurniawan et al. [65] hanno proseguito la ricerca di un algoritmo meta-euristico efficace, proponendo anche loro un algoritmo genetico che si basa sul progetto Moon et al.

Tabella 2.3 Suddivisione per classe dei problemi di scheduling multi macchina

Classe	Problema	Autori
Scheduling con m macchine e un solo obiettivo	$Rm\|batch(b)\|TEC$	Tan et al., 2019
	$Rm\|\ \ |KC_{max} + TEC$	Moon et al., 2013
		Cheng et al., 2019
		Kurniawan et al., 2020
	$Rm\|\ \ |TEC$	Ding et al., 2016
		Cheng et al., 2018
		Che et al., 2017
		Saberi-Aliabad et al., 2019
Scheduling multi obiettivo con m macchine	$Pm\|\ \ |C_{max}, TEC$	Wang et al., 2018
		Anghinolfi et al., 2021
		Zeng et al., 2018
	$Pm\|batch(b), r_j\|C_{max}, TEC$	Zhou et al., 2018
	$Pm\|batch(b), r_j, s_j\|C_{max}, TEC$	Jia et al., 2017
	$Pm\|batch(b), r_j\|\sum_{j\in\mathcal{J}} T_j, TEC$	Rocholl et al., 2020
	$Qm\|batch(b), r_j, p_j\|C_{max}, TEC$	Qian et al., 2020
	$Rm\|\ \ \|\sum_{j\in\mathcal{J}} T_j, TEC$	Pan et al., 2018
	$Rm\|\ \ |C_{max}, TEC$	Pei et al., 2021

Per quanto riguarda la complessità di questo problema essa viene presa in considerazione nella pubblicazione [19] di Ding et al. dove dimostra che il problema $Rm|\ \ |TEC$ risulta NP-hard. Inoltre nello stesso articolo gli

autori presentano una formulazione MILP e, utilizzando la scomposizione di Dantzig–Wolfe, propongono un algoritmo "column generation", il quale viene testato tramite degli esperimenti computazionali condotti sotto diverse impostazioni di TOU e i risultati confermano l'effettiva efficacia dei metodi proposti. La formulazione presentata da Ding et al. viene migliorata da Cheng et al. [66], utilizzando meno variabili e dimostrando che la loro versione sia anche migliore dal punto di vista computazionale. Sulla stessa tipologia di problema si trovano i lavori di Che et al. [28] e Saberi-Aliabad et al. [67], entrambi con l'obiettivo di minimizzare un solo obiettivo: il TEC. Per prima cosa Che et al. costruiscono un modello di programmazione lineare integrale mista (MILP) a tempo continuo per il problema. Per affrontare problemi di grandi dimensioni, propongono poi un'euristica a due stadi. In particolare, nella prima fase, i lavori vengono assegnati a macchine con l'obiettivo di minimizzare il TEC permettendo la possibilità di scheduling preemptive. Nel secondo stadio, i lavori assegnati ad ogni macchina sono riprogrammati usando un'euristica di inserimento. Successivamente vengono presentati i risultati computazionali su un'istanza reale (dati ricavati da processo di tornitura) e su istanze di test casuali, e viene dimostrano che l'approccio MILP proposto è in grado di risolvere problemi di piccole dimensioni, mentre l'euristica a due stadi è appropriata per problemi di grandi dimensioni. Anche Saberi-Aliabad et al. presentano un modello MILP per il problema $Rm|\ \ |TEC$. Oltre al modello espongono delle regole di dominanza e delle "valid inequalities" per migliorare il tempo di calcolo del modello. I risultati dello studio hanno indicato che il modello proposto risulta migliore di quello degli altri studi correlati in letteratura. Oltre al modello MILP è stato proposto un algoritmo euristico per istanze di grandi dimensioni, che potrebbe risolvere le istanze fino a mille job e venti macchine.

Pan et al. [68] e Pei et al. [69] sono i primi che considerano questa tipologia di problema con due obiettivi da minimizzare. Nel caso di Pan et al. troviamo un problema bi-obiettivo, dove oltre al TEC viene si minimizza

anche tardiness totale. Per la risoluzione vengono proposte e confrontate due possibilità: l'utilizzo del metodo lessicografico oppure di un algoritmo competitivo imperialista ("Imperialist competitive algorithm" - ICA), arrivando a mostrare i promettenti vantaggi di ICA. Per quanto riguarda Pei et al. si tiene conto di: makespan e TEC. Nell'articolo viene proposto un modello non-lineare, sono presenti vincoli quadratici e vengono introdotti dei piani di taglio ad-hoc per restringere ulteriormente la regione ammissibile del problema. Partendo da questo problema Pei et al. trasformano il problema originale in diversi problemi a singola macchina, presentando così un rilassamento. Sulla base delle soluzioni ottimali dei problemi rilassati viene presentato un algoritmo approssimativo, che viene convalidato attraverso test su varie scale di istanze. Viene osservato che il divario tra l'algoritmo approssimato proposto e il problema non-lineare è per lo più entro il 4%.

Capitolo 3

Problema affrontato

Questo capitolo è il fulcro di questa tesi: viene esaminato un problema di scheduling con macchine non uniformi ("unrelated") con l'applicazione del tariffario Time-Of-Use (TOU), considerando due obiettivi: minimizzazione del makespan C_{max} e del Total Consumption Energy - TEC. Verranno presentate ed analizzate quattro formulazioni matematiche: partendo riadattando due modelli già presenti in letteratura stilati da Anghinolfi et al. [21] e Ronco [70], rispettivamente in «A bi-objective heuristic approach for green identical parallel machine scheduling» e «Exact and Heuristic Algorithms for Energy-Efficient Scheduling», per poi presentare due nuove formulazioni che ricalcano i modelli pubblicati da Che et al. in «Energy-conscious unrelated parallel machine scheduling under time-of-use electricity tariffs» [28] e da Saberi-Aliabad et al. in «Energy-Efficient Scheduling in an Unrelated Parallel-Machine Environment Under Time-Of-Use Electricity Tariffs» [67]. Al meglio delle nostre conoscenze ci sono quattro studi in letteratura che indagato lo stesso problema, riportati nella tabella 3.1. Quindi, l'obiettivo principale di questa tesi è proporre ulteriori metodi di soluzione portando così ad un ampliamento della ricerca su questo particolare problema di energy efficient scheduling. Inoltre è da sottolineare che il problema verrà osservato anche da un punto di vista pratico, infatti trovare tutte le soluzioni

Pareto-ottimali sarà un ulteriore scopo della tesi.

Tabella 3.1 Studi presenti sullo stesso problema e approccio proposto per la risoluzione

Classe	**Autori**	**Approccio proposto**
$Rm\|\quad\|C_{max}, TEC$	Ding et al., 2016	MILP & Algoritmo "Column Generation"
	Che et al., 2017	MILP & Euristica a due stadi
	Cheng et al., 2018	MILP
	Saberi-Aliabad et al., 2019	MILP & Euristica

Questo capitolo sarà organizzato nel modo seguente: nella sezione 3.1 viene definito il problema affrontato in modo specifico, comprendendo le ipotesi prese in considerazione, gli insiemi utilizzati e i parametri utilizzati. Nelle sezioni successive 3.2 e 3.3 vengono presentate le quattro formulazioni matematiche proposte, suddivise in due sezioni distinte perché sono basate su due concetti dell'orizzonte temporale differente.

3.1 Definizione del problema

Secondo la classificazione a 3-campi ("3-field problem classification" [22]) il problema preso in considerazione è: $Rm|\ \ |TEC, C_{\max}$, in letteratura chiamato "Bi-objective Unrelated Parallel Machine energy-efficient Scheduling with Time-of-Use prices Problem (BUPMSTP)" [70].

Nel problema saranno presi in considerazione:

- N job indipendenti tra di loro
- M macchine non correlate ("unrelated")
- Orizzonte temporale finito

Che portano alla definizione degli insiemi:

- $\mathcal{J} = \{1, 2, \ldots, N\}$
- $\mathcal{H} = \{1, 2, \ldots, M\}$

L'insieme temporale verrà definito successivamente in base alla tipologia di assunzione che verrà fatta: se l'orizzonte temporale sarà suddiviso in K time slot o D in time period.

Per quanto riguarda i job $j \in \mathcal{J}$ invece saranno disponibili dal primo istante, quindi non viene preso in considerazione il release time. Prendendo in considerazione macchine unrelated, per ogni job $j \in \mathcal{J}$ si definisce un processing time $p_{j,h}$ dipendente dalla macchina $h \in \mathcal{H}$ (vedi Fig.3.1), corrispondente ad al numero intero di time slot necessari per completare l'esecuzione del job j sulla macchina h.

Un assegnamento di un job $j \in \mathcal{J}$ ad un sottoinsieme $\mathcal{T}_j$ dell'orizzonte temporale di lunghezza pari alla durata del processing time $p_{j,h}$ corrisponde all'elaborazione di j da parte della macchina h per un periodo di lunghezza

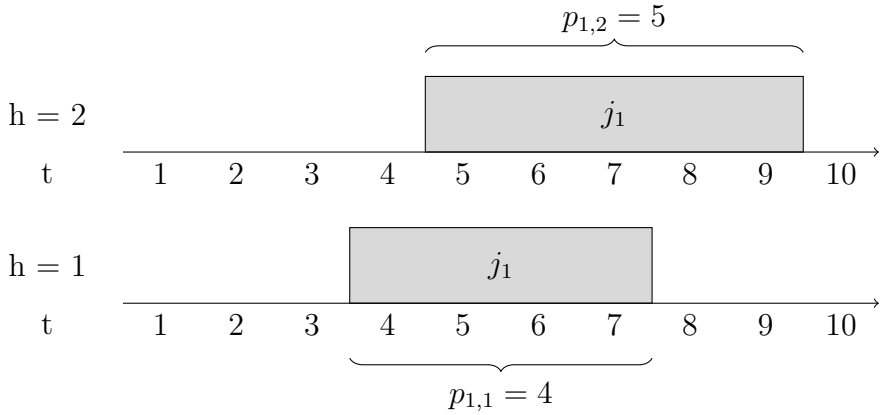


Figura 3.1: Esempio per rappresentare un processing time dipendente dalla macchina: il job j_1 se affidato alla macchina $h = 1$ la sua esecuzione richiederebbe 4 istanti di tempo, mentre se eseguito sulla macchina $h = 2$ il processo avrebbe una durata di 5 istanti di tempo.

$p_{j,h}$. L'assegnamento verrà definito tramite la tripla $(j, h, \mathcal{T}_j)$. Da questa definizione è possibile definire uno scheduling come un insieme di assegnamenti, precisamente:

$$\mathcal{S} = \{(j, h, \mathcal{T}_j) : h \in \mathcal{H}, |\mathcal{T}_j| = p_{j,h}, \forall j \in \mathcal{J}\} \tag{3.1}$$

Uno scheduling $\mathcal{S}$ per essere accettabile dovrà comprendere l'assegnamento di tutti i job $j \in \mathcal{J}$, i quali devono esser elaborati in sottoinsiemi distinti, cioè tali per cui non esistano intersezioni tra i vari $\mathcal{T}_j$, $\forall j$ sulla stessa macchina $h \in \mathcal{H}$. Quindi prendendo due assegnamenti (j, h_j, T_j) e $(j', h_{j'}, T_{j'})$ deve valere:

$$\mathcal{T}_j \cap \mathcal{T}_{j'} = \emptyset, \qquad \forall j, j' \in \mathcal{J} : j \neq j', h_j = h_{j'} \tag{3.2}$$

Comportando a definire uno scheduling accettabile nel modo seguente:

$$\begin{aligned} \mathcal{S} = \{(j, h, \mathcal{T}_j) : h \in \mathcal{H}, |\mathcal{T}_j| = p_{j,h}, \forall j \in \mathcal{J} \\ \wedge \mathcal{T}_j \cap \mathcal{T}_{j'} = \emptyset, \forall j, j' \in \mathcal{J} : j \neq j', h_j = h_{j'}\} \end{aligned} \tag{3.3}$$

Inoltre non essendo presa in considerazione la preemption tutti i sottoinsiemi devono comprendere insiemi di istanti dell'orizzonte temporale consecutivi. Tutte queste ipotesi,caratteristiche e dati saranno in comune con tutte le formulazioni proposte.

3.2 Formulazioni indicizzate ai time slot

Le prime formulazioni proposte utilizzano un orizzonte temporale suddiviso in time slot. Questo concetto è stato sfruttato in diversi studi, ma applicato a questa tipologia di problema lo si trova nella pubblicazione di Anghinolfi et al. in «A bi-objective heuristic approach for green identical parallel machine scheduling» e nello studio di Ronco «Exact and Heuristic Algorithms for Energy-Efficient Scheduling», precisamente nel loro caso è stato studiato il caso di macchine parallele identiche.

La partizione dell'orizzonte temporale si traduce con la definizione dell'insieme degli istanti di tempo: $\mathcal{T} = \{1, 2, \ldots, K\}$, che verrà utilizzato nella definizione delle due formulazioni presenti in questa sezione.

c_t	3	3	3	6	6	6	9	9	9	6	6	6	3	3	3
t	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Figura 3.2: Esempio di orizzonte temporale suddiviso in time slot

Suddividendo l'orizzonte temporale in time slot rende più facile ed intuitiva l'applicazione di un tariffario basato sul concetto TOU. Difatti, come mostrato nella figura 3.2, ad ogni time slot si associa un costo energetico denominato $c_t, \forall t \in \mathcal{T}$. Oltre al costo energetico si associa ad ogni macchina $h \in \mathcal{H}$ un tasso di consumo energetico caratteristico della macchina $u_h, \forall h$. Queste caratteristiche dei time slot e delle macchine verranno utilizzate per entrambe le formulazioni presenti in questa sezione.

Infine definendo:

$$p_j^{\max} = \max_{h \in \mathcal{H}} p_{j,h}, \qquad j \in \mathcal{J} \tag{3.4}$$

il processing time più grande del job $j \in \mathcal{J}$ su tutte le macchine $h \in \mathcal{H}$. Si può osservare che una condizione sufficiente affinché BUPMSTP, formulato con l'utilizzo dei time slot, ammetta almeno una soluzione accettabile è che

la somma di tutti i $p_j^{\max}$ dei job $j \in \mathcal{J}$ non superi il numero complessivo MK di slot disponibili per la schedulazione.

$$N \leq \sum_{j \in \mathcal{J}} p_j^{\max} \leq MK \tag{3.5}$$

con un minimo uguale ad N, ottenibile quando tutti i processing time massimi hanno durata equivalente ad 1: $\sum_{j \in \mathcal{J}} p_j^{\max} = N$, se $p_j^{\max} = 1, \forall j \in \mathcal{J}$.

3.2.1 Formulazione 1

Questa prima formulazione si basa sul modello presentato da Anghinolfi et al. in «A bi-objective heuristic approach for green identical parallel machine scheduling». La formulazione presentata era per il caso con macchine parallele identiche (Pm), quindi il processing time dei job non dipendeva dalla macchina in cui veniva eseguito. Con la semplice sostituzione di p_j in $p_{j,h}$ è stato possibile adattare il loro modello al caso Rm, utilizzando le medesime variabili decisionali e i medesimi vincoli.

Variabili

Per questa formulazione MIP ("Mixed Integer Programming") vengono definite le seguenti variabili:

$$X_{j,h,t} \in \{0,1\} \qquad j \in \mathcal{J}, h \in \mathcal{H}, t \in \mathcal{T} \tag{3.6}$$

$X_{j,h,t}$ assume il valore 1 se il processo del job j sulla macchina h parte all'istante t, altrimenti sarà uguale a 0.

$$C_{\max} = \max\{C_j, j \in \mathcal{J}\} \tag{3.7}$$

Il makespan viene definito come il massimo dei vari completion time dei job $C_j, j \in \mathcal{J}$, si ricorda l'esempio raffigurato nella figura 2.6 presente nel capitolo 2.

Formulazione

$$\min \quad C_{\max}, \tag{3.8}$$

$$\min \quad TEC, \tag{3.9}$$

soggetto ai vincoli:

$$TEC = \sum_{h \in \mathcal{H}} u_h \sum_{j \in \mathcal{J}} \sum_{t=1}^{K-p_{j,h}+1} X_{j,h,t} \left(\sum_{i=t}^{t+p_{j,h}-1} c_i \right) \tag{3.10}$$

$$\sum_{h \in \mathcal{H}} \sum_{t=1}^{K-p_{j,h}+1} X_{j,h,t} = 1, \quad j \in \mathcal{J}, \tag{3.11}$$

$$\sum_{j \in \mathcal{J}} \sum_{i=\max\{1,t-p_{j,h}+1\}}^{t} X_{j,h,i} \leq 1, \quad h \in \mathcal{H}, t \in \mathcal{T}, \tag{3.12}$$

$$\sum_{h \in \mathcal{H}} \sum_{t=1}^{K-p_{j,h}+1} (t + p_{j,h} - 1) X_{j,h,t} \leq C_{\max}, \quad j \in \mathcal{J}, \tag{3.13}$$

$$C_{\max} \leq K, \tag{3.14}$$

$$C_{\max} \geq 0, \quad TEC \geq 0, \quad X_{j,h,t} \in \{0,1\}, \\ j \in \mathcal{J}, h \in \mathcal{H}, t \in \mathcal{T}. \tag{3.15}$$

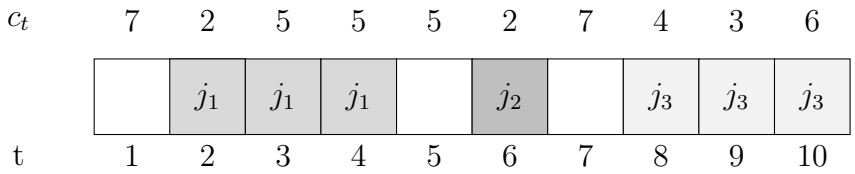
Gli obiettivi 3.8 e 3.9 rappresentano rispettivamente la minimizzazione del makespan e del TEC, che viene definito con la 3.10. Il vincolo 3.11 impone che ogni job $j \in \mathcal{J}$ inizi in un singolo time-slot su una sola macchina, oltre impone che tutti i job debbano iniziare in un istante che permetta la loro esecuzione entro K-esimo, ultimo, time slot. Il vincolo 3.12 evita l'esecuzione contemporanea di due job sulla stessa macchina. Nella parte

sinistra del vincolo 3.13 si definisce il completion time C_j ti ogni job j e che deve risultare al massimo uguale al makespan $C_{\max}$. Infine in 3.14 si dichiara che il makespan non possa risultare maggiore del numero di time slot, cioè K. Per quanto riguarda in 3.15 vengono definiti i domini delle variabili.

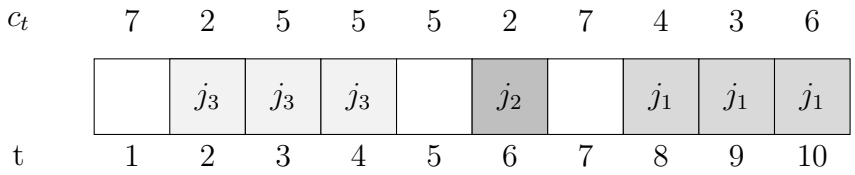
3.2.2 Formulazione 2

In ≪Exact and Heuristic Algorithms for Energy-Efficient Scheduling≫ di Ronco viene proposto un modello MIP per il problema $Pm|\ \ |TEC, C_{\max}$ con l'obiettivo di migliorare la formulazione proposta in ≪A bi-objective heuristic approach for green identical parallel machine scheduling≫. Difatti Ronco dimostra che la nuova formulazione sia decisamente migliore per quanto riguarda sia il numero di variabili che il numero di vincoli. Inoltre viene osservato che in caso di job aventi la stessa durata di processing time la formulazione 1 in diverse esecuzioni può presentare diversi scheduling ma con risultati equivalenti sia per il TEC che per il $C_{\max}$, come mostrato nell'esempio raffigurato nella figura 2.8.

L'idea che sta alla base della nuova formulazione è di non utilizzare variabili che sono indicizzate all'id del job, ma che siano indicizzate alla durata dei vari job. Ciò comporta che il risultato di questa formulazione non sarà un vero e proprio scheduling, infatti ogni soluzione "feasible" (fattibile) di questa formulazione definisce un insieme di scheduling equivalenti [70] (esempio in figura 3.4). Nel particolare questa formulazione garantisce che per ogni $Y_{d,h,t} = 1$ un job $j \in \mathcal{J}$ con tempo di elaborazione pari a d sia schedulato senza preemption negli slot $t, t+1, \ldots, t+d-1$ sulla macchina $h \in \mathcal{H}$. Per ottenere un possibile scheduling è necessario applicare un algoritmo di assegnamento, sulla base di quello presente in [70] è stato formulato uno per questa casistica Rm, ed è definito più avanti in Algoritmo 1.



(a) Primo risultato di scheduling



(b) Secondo risultato di scheduling

Figura 3.3: Esempio dei due possibili scheduling ottenibili con la stessa istanza: da dei dati identici si possono ottenere due scheduling differenti ma con TEC e $C_{\max}$ equivalenti. Infatti dall'esempio qua raffigurato il $C_{\max} = 10$ e il TEC vale $2+5+5+2+4+3+6 = 25$ per entrambi gli scheduling. Difatti si nota che scambiando i job j_1 e j_3 la soluzione non cambia.

Parametri

Per questo modello è necessario definire tre insiemi che verranno utilizzati per la formulazione matematica del problema. Precisamente si ha:

- $\boldsymbol{\mathcal{P}} = \{p_{11}, p_{12}, p_{13}, \ldots, p_{NM}\}$: insieme di tutti i processing time dei job su tutte le macchine.
- $\boldsymbol{\mathcal{P}}_h, \quad \forall h \in \boldsymbol{\mathcal{H}}$: insieme dei diversi processing time dei job sulla macchina h.
- $\boldsymbol{\mathcal{J}}_{d,h}, \quad \forall h \in \boldsymbol{\mathcal{H}}$: insieme dei job aventi processing time $p_{j,h} = d$ sulla macchina h.

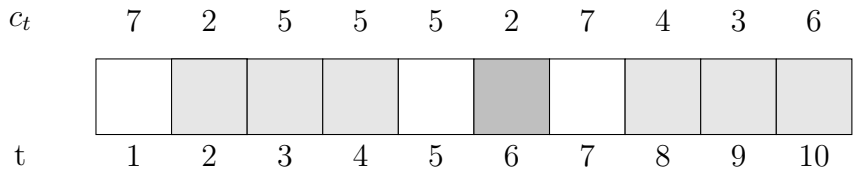


Figura 3.4: Esempio di risultato ottenibile con la risoluzione della formulazione 2. Il problema utilizzato in questo esempio è: programmare l'esecuzione di tre job: j_1, j_2, j_3 su un unica macchina in un orizzonte temporale di durata 10 istanti di tempo. Sapendo che $p_{1,1} = 3$, $p_{2,1} = 1$, $p_{3,1} = 3$

Per rendere la formulazione più compatta e chiara viene definito $b_{d,t}$:

$$b_{d,t} = \sum_{i=t}^{t+d-1} c_i, \qquad d \in \mathcal{P}, t = 1, \ldots, K - d + 1, \tag{3.16}$$

Con $b_{d,t}$ si va a definire il costo cumulativo associato al sottoinsieme di time slot consecutivi partendo da t fino ad arrivare a $t+d-1$. Conseguentemente un job $j \in \mathcal{J}$ con processing time $p_{j,h}$ sulla macchina $h \in \mathcal{H}$ se assegnato alla macchina h nel periodo $\{t, t+1, \ldots, t+d-1\}$ avrà un costo energetico pari a $u_h b_{d,t}$. Ricordando che con u_h si intende il tasso di consumo energetico delle macchina $h \in \mathcal{H}$.

Variabili

$$Y_{d,h,t} \in \{0, 1\} \qquad d \in \mathcal{P}_h, h \in \mathcal{H}, t \in \mathcal{T} \tag{3.17}$$

$Y_{d,h,t}$ assume il valore 1 se parte l'esecuzione al time slot t di un job di durata d sulla macchina h, altrimenti è nulla.

$$Z_{j,h} \in \{0, 1\} \qquad j \in \mathcal{J}, h \in \mathcal{H} \tag{3.18}$$

$Z_{j,h}$ è uguale ad 1 se il job j è assegnato alla macchina h, in caso contrario assume il valore 0.

Il $C_{\max}$ viene definito come nella 3.7.

Formulazione

$$\min C_{\max}, \tag{3.19}$$

$$\min TEC, \tag{3.20}$$

soggetto ai vincoli:

$$TEC = \sum_{h \in \mathcal{H}} u_h \sum_{d \in \mathcal{P}_h} \sum_{t=1}^{K-d+1} b_{d,t}\, Y_{d,h,t} \tag{3.21}$$

$$\sum_{h \in \mathcal{H}} Z_{j,h} = 1, \quad j \in \mathcal{J} \tag{3.22}$$

$$\sum_{h \in \mathcal{H}} \sum_{d \in \mathcal{P}_h} \sum_{t=1}^{K-d+1} Y_{d,h,t} = N \tag{3.23}$$

$$\sum_{t=1}^{K-d+1} Y_{d,h,t} = \sum_{j \in \mathcal{J}_{d,h}} Z_{j,h}, \quad h \in \boldsymbol{\mathcal{H}}, d \in \boldsymbol{\mathcal{P}}_h \tag{3.24}$$

$$\sum_{d \in \mathcal{P}_h} \sum_{i=\max\{1,t-d+1\}}^{t} Y_{d,h,i} \leq 1,\ h \in \boldsymbol{\mathcal{H}}, t \in \boldsymbol{\mathcal{T}} \tag{3.25}$$

$$(t + d - 1)\, Y_{d,h,t} \leq C_{\max},$$
$$d \in \boldsymbol{\mathcal{P}}_h, h \in \boldsymbol{\mathcal{H}}, t = 1, \ldots, K - d + 1 \tag{3.26}$$

$$C_{\max} \leq K \tag{3.27}$$

$$C_{\max} \geq 0 \quad TEC \geq 0,$$
$$Y_{d,h,t} \in \{0, 1\}, Z_{j,h} \in \{0, 1\}$$
$$h \in \boldsymbol{\mathcal{H}}, d \in \boldsymbol{\mathcal{P}}_h, t \in \boldsymbol{\mathcal{T}}, j \in \boldsymbol{\mathcal{J}} \tag{3.28}$$

Gli obiettivi naturalmente sono la minimizzazione del makespan 3.19 e del Total Consumption Energy 3.20, che viene definito nella 3.21. Il vincolo 3.22 costringe l'esecuzione di tutti i job $\in \mathcal{J}$, non lasciando così job non eseguiti. Il vincolo 3.24 mette in correlazione le variabili $Y_{d,h,t}$ e $Z_{j,h}$ imponendo

che il numero di start time per job di durata d sulla macchina h sia esattamente uguale al numero di job assegnati alla macchina h, si ricorda che $\mathcal{P}_h$ è l'insieme dei processing time differenti presenti sulla macchina h, mentre $\mathcal{J}_{d,h}$ è l'insieme dei job che hanno durata d sulla macchina h. Il vincolo 3.25 impone che non possano esserci sovrapposizioni tra i job, infatti impone che possa esserci al massimo uno start time di un job con processing time di lunghezza d su una macchina h nel periodo $\{\max(1, t-d+1), t\}$. Il vincolo 3.26 definisce il tempo di completamento dei job C_j, che deve essere minore o uguale al makespan $C_{\max}$. Per quanto riguarda il vincolo 3.27 non permette che il makespan possa superare il limite massimo dell'orizzonte temporale. Per concludere in 3.28 vengono definiti i domini delle variabili.

Algoritmo di assegnamento

Algoritmo 1 Genera scheduling

Input: Variabili $Y_{d,h,t}$ assegnate
Output: Uno scheduling $\mathcal{S}$ feasible

1: $\mathcal{S} \leftarrow \emptyset$
2: **for** $(\hat{d}, \hat{h}, \hat{t}) \in \{(d, h, t) : Y_{d,h,t} = 1, d \in \mathcal{P}_h, h \in \mathcal{H}, t \in \mathcal{T}\}$ **do**
3: Dato $j \in \mathcal{J}_{\hat{d},\hat{h}}$
4: $\mathcal{S} \leftarrow \mathcal{S} \cup \{j, \hat{h}, \{\hat{t}, \hat{t}+1, \ldots, \hat{t}+d-1\}\}$
5: $\mathcal{J}_{\hat{d},\hat{h}} \leftarrow \mathcal{J}_{\hat{d},\hat{h}} \setminus \{j\}$
6: **end for**
7: **return** $\mathcal{S}$

L'Algoritmo 1 genera un possibile scheduling partendo dall'insieme di possibili scheduling definiti da una soluzione della formulazione 2. Andando nel dettaglio alla riga 1 si inizializza uno scheduling con un insieme vuoto $\emptyset$. Successivamente nelle righe (3 - 6) per ogni (d, h, t) tali per cui vale $Y_{d,h,t} = 1$, un job j appartenente all'insieme $\mathcal{J}_{d,h}$ viene assegnato a d time slot consecutivi sulla macchina h partendo dall'istante t. Infine alla riga 7 lo scheduling effettivo con tutti job assegnati viene restituito.

3.3 Formulazioni indicizzate ai time period

Il problema può essere riformulato sfruttando una diversa suddivisione dell'orizzonte temporale, in questo caso si sceglie di utilizzare i time period, introdotti in 2.2.1. Questo concetto è stato presentato da Che et al. in «Energy-conscious unrelated parallel machine scheduling under time-of-use electricity tariffs» e da Saberi-Aliabad et al. in «Energy-Efficient Scheduling in an Unrelated Parallel-Machine Environment Under Time-Of-Use Electricity Tariffs». Nelle loro pubblicazioni vengono presentate delle formulazioni matematiche per questa tipologia di problema , $Rm| \quad |TEC$. Va sottolineato che le formulazioni di Che et al. e Saberi-Aliabad et al. non determinano esplicitamente né la sequenza né la tempistica dei job sulle macchine, trovano invece l'assegnazione dei job nei time period sulle diverse macchine che consente di generare una schedulazione fattibile tramite una procedura di post-elaborazione.

Prendendo in considerazione anche la minimizzazione del makespan e traendo spunto dalle loro ricerche sono state sviluppate due ulteriori formulazioni sul problema bi-obiettivo, precisamente $Rm| \quad |TEC, C_{\max}$. Per questa casistica gli insiemi dei job da schedulare e delle macchine risultano invariati rispetto alle formulazioni precedenti, mentre per quanto riguarda l'orizzonte temporale, di durata T, essendo suddiviso in time period risulta differente:

- $\mathcal{J} = \{1, 2, \ldots, N\}$
- $\mathcal{H} = \{1, 2, \ldots, M\}$
- $\mathcal{K} = \{1, 2, \ldots, D\}$

Ogni time period $k \in \mathcal{K}$ è caratterizzato da un istante d'inizio, denominato s_k e da un istante di fine s_{k+1} che combacia con l'istante d'inizio del time period $k + 1$. Da ciò è possibile ricavare e definire la durata di ogni time period $l_k = s_{k+1} - s_k, \ \forall k \in \mathcal{K}$ fissando $s_0 = 0$ e $s_{D+1} = T$. Inoltre, come

accadeva per i time slot, ogni time period viene caratterizzato da un costo energetico c_k, $\forall k \in \mathcal{K}$. Nella figura 3.5 vi è una rappresentazione di un orizzonte temporale suddiviso in time period e con i vari c_k associati ad ogni time period, naturalmente il prezzo seguirà una politica di prezzo basata sullo tariffario TOU.

c_k	$c_1 = 3$	$c_2 = 6$	$c_3 = 9$	$c_4 = 6$	$c_5 = 3$
k	1	2	3	4	5

Figura 3.5: Esempio di orizzonte temporale suddiviso in time period

Come le formulazioni precedenti ogni job è caratterizzato da un processing time $p_{j,h}$. In ≪Energy-conscious unrelated parallel machine scheduling under time-of-use electricity tariffs≫ di Che et al. ad ogni job si associa un consumo di energia per unità di tempo, denominato in letteratura "electricity consumption rate" $r_{j,h}$, dipendente dalla macchina, in questa formulazione si omette la dipendenza dalla macchina, facendo sì che $r_{j,h}$ coincida con u_h, già definito ed utilizzato nelle formulazioni precedenti.

$$u_h \qquad h \in \mathcal{H} \tag{3.29}$$

Rimangono invariate le ipotesi già utilizzate nelle formulazioni precedenti: ogni macchina può elaborare un solo lavoro alla volta e ogni lavoro deve essere elaborato da una sola macchina, non è consentita la preemption, tutti i job sono pronti e le macchine sono disponibili all'inizio dell'orizzonte temporale di produzione. Da sottolineare che i modelli su time slot sono basati su un orizzonte temporale suddiviso segmenti unitari vincolando così i processing time $p_{j,h}$ ad essere numeri interi, mentre in queste prossime formulazioni con l'utilizzo dei time period i processing time possono assumere valori reali positivi.

$$p_{j,h} \in \mathbb{R}^+ \qquad j \in \mathcal{J}, h \in \mathcal{H} \tag{3.30}$$

Utilizzando la definizione 3.4 definita in precedenza, si può osservare che una condizione sufficiente affinché BUPMSTP, in questo caso formulato con l'utilizzo dei time period, ammetta almeno una soluzione accettabile è che la somma di tutti i $p_j^{\max}$ dei job $j \in \mathcal{J}$ non superi il numero complessivo $M \sum_{k \in \mathcal{K}} l_k$, corrispondente alla somma delle durate di tutti i time period moltiplicata per il numero di macchine disponibili:

$$N \leq \sum_{j \in \mathcal{J}} p_j^{\max} \leq M \sum_{k \in \mathcal{K}} l_k \tag{3.31}$$

3.3.1 Formulazione 3

La prima formulazione sviluppata si basa sul modello presentato da Che et al. nella pubblicazione «Energy-conscious unrelated parallel machine scheduling under time-of-use electricity tariffs» [28]. L'obiettivo di Che et al. era di migliorare la formulazione proposta da Ding et al., che si ricorda essere una delle prime pubblicazioni su questo problema $Rm|\quad|TEC$, andandone a diminuire il numero di vincoli e il numero di variabili. Precisamente Ding et al. utilizza un totale di $NM(5D+1)$ variabili, mentre Che et al. ne utilizza $3NMD$, quindi $NM(2D-1)$ variabili in meno, mentre i vincoli definiti da Ding et al. risultano $5NMD + 2MD + N$ e quelli del modello diChe et al. sono $4NMD + 2MD + 2N - 4NM$, diminuendo sia il numero di vincoli che di variabili Che et al. rende così il modello decisamente più efficiente. In aggiunta in [28] viene presentata un'euristica a due stadi, per poi presentare un confronto computazionale tra il modello proposto da Ding et al., il suo modello e l'euristica.

Parametri

Vengono utilizzati tutti i parametri presentati nell'introduzione:

- $l_k,\ c_k,\ k \in \mathcal{K}$;

- $u_h, h \in \mathcal{H}$;
- $p_{j,h},\ j \in \mathcal{J},\ h \in \mathcal{H}$;

e in più si adopera:

$$\tau_{j,h} = \left\lfloor \frac{p_{j,h}}{\min_{k \in \mathcal{K}} l_k} \right\rfloor + 1 \qquad j \in \mathcal{J}, h \in \mathcal{H} \tag{3.32}$$

Con $\tau_{j,h}$ si definisce il limite superiore per il numero di periodi che possono esser assegnati al job $j \in \mathcal{J}$ sulla macchina $h \in \mathcal{H}$

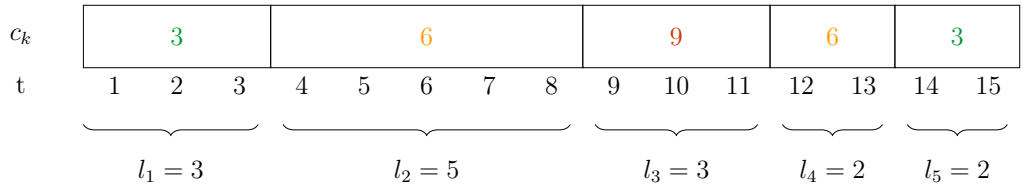


Figura 3.6: Esempio rappresentativo delle durate dei time period che dividono un orizzonte temporale

Variabili

$$0 \leq R_{j,h,k} \leq p_{j,h} \qquad j \in \mathcal{J}, h \in \mathcal{H}, k \in \mathcal{K} \tag{3.33}$$

Con $R_{j,h,k}$ si indica la quantità di processing time processata del job $j \in \mathcal{J}$ sulla macchina $h \in \mathcal{H}$ durante il periodo $k \in \mathcal{K}$ (vedi esempio a Fig.3.7).

$$X_{j,h,k} \in \{0, 1\} \qquad j \in \mathcal{J}, h \in \mathcal{H}, k \in \mathcal{K} \tag{3.34}$$

La variabile binaria $X_{j,h,k}$ vale 1 se il job $j \in \mathcal{J}$ è processato dalla macchina $h \in \mathcal{H}$ durante il periodo $k \in \mathcal{K}$. Varrà 0 in caso contrario.

$$V_{j,h} \in \{0, 1\} \qquad j \in \mathcal{J}, h \in \mathcal{H} \tag{3.35}$$

Viene utilizzata un altra variabile binaria, $V_{j,h}$, per l'associazione job-to-machine, infatti essa assumerà valore 1 quando il job $j \in \mathcal{J}$ è assegnato alla macchina $h \in \mathcal{H}$.

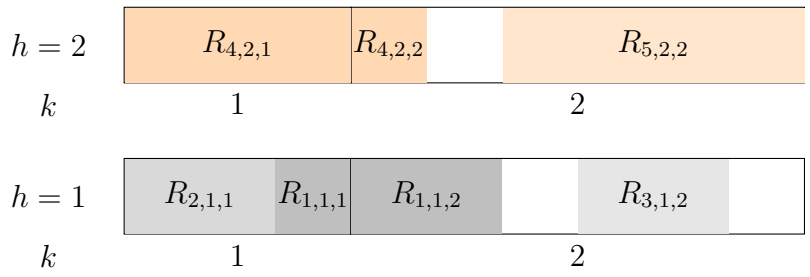


Figura 3.7: Esempio di possibile risultato ottenibile dalla risoluzione della formulazione 3 o 4 utilizzato per rappresentare le variabili $R_{j,h,k}$. Il problema utilizzato richiede la programmazione di 5 job con 2 macchine. Per rendere più chiara la funzione delle variabili $R_{j,h,k},\ j \in \mathcal{J}, h \in \mathcal{H}, k \in \mathcal{K}$, si prende d'esempio il job 1, il quale è caratterizzato da un processing time sulla macchina $h = 1$ pari a $p_{1,1} = 3$. Una porzione pari ad 1 unità di tempo viene elaborata durante il periodo $k = 1$, il che significa che $R_{1,1,1} = 1$. La porzione rimanente dovrà esser processata necessariamente (preemption non è possibile) durante il time period $k = 2$, ciò comporta che la variabile $R_{1,1,2} = 2$.

Tutte e tre le variabili $R_{j,h,k}$, $X_{j,h,k}$ e $V_{j,h}$ sono prese dal modello presentato da Che et al. L'unica differenza è in [28] veniva ulteriormente definita un'altra variabile, utilizzata per determinare quali job fossero elaborati a cavallo tra due time period. L'esclusione di essa fa sì che questo modello abbia un numero minore di variabili rispetto a quelle di Che et al.

Formulazione

$$\min C_{\max} \tag{3.36}$$

$$\min TEC \tag{3.37}$$

soggetto ai vincoli:

$$TEC = \sum_{h \in \mathcal{H}} u_h \sum_{k \in \mathcal{K}} c_k \sum_{j \in \mathcal{J}} R_{j,h,k}, \tag{3.38}$$

$$\sum_{h \in \mathcal{H}} V_{j,h} = 1, \quad j \in \mathcal{J} \tag{3.39}$$

$$\sum_{h \in \mathcal{H}} \sum_{k \in \mathcal{K}} \frac{R_{j,h,k}}{p_{j,h}} = 1, \quad j \in \mathcal{J} \tag{3.40}$$

$$\sum_{j \in \mathcal{J}} R_{j,h,k} \leq l_k, \quad h \in \mathcal{H}, k \in \mathcal{K} \tag{3.41}$$

$$R_{j,h,k} \leq p_{j,h} X_{j,h,k}, \quad j \in \mathcal{J}, h \in \mathcal{H}, k \in \mathcal{K} \tag{3.42}$$

$$\sum_{g=k+2}^{K} X_{j,h,g} \leq (D-k-1)(1 - X_{j,h,k} + X_{j,h,k+1}),$$
$$j \in \mathcal{J}, h \in \mathcal{H}, k = 1, \ldots, D-2 \tag{3.43}$$

$$R_{j,h,k} \geq l_k(X_{j,h,k-1} + X_{j,h,k+1} - 1),$$
$$j \in \mathcal{J}, h \in \mathcal{H}, k = 2, \ldots, D-1 \tag{3.44}$$

$$\sum_{k \in \mathcal{K}} X_{j,h,k} \leq \tau_{j,h} V_{j,h}, \quad j \in \mathcal{J}, h \in \mathcal{H} \tag{3.45}$$

$$\sum_{k \in \mathcal{K}} R_{j,h,k} \leq p_{j,h} V_{j,h}, \quad j \in \mathcal{J}, h \in \mathcal{H} \tag{3.46}$$

$$TEC \geq 0, \tag{3.47}$$

$$R_{j,h,k} \in [0, p_{j,h}], j \in \mathcal{J}, h \in \mathcal{H}, k \in \mathcal{K} \tag{3.48}$$

$$X_{j,h,k} \in \{0,1\}, j \in \mathcal{J}, h \in \mathcal{H}, k \in \mathcal{K} \tag{3.49}$$

$$V_{j,h} \in \{0,1\}, j \in \mathcal{J}, h \in \mathcal{H} \tag{3.50}$$

Come per le formulazioni precedenti gli obiettivi sono sempre due: minimizzazione del makespan 3.36 e del total energy consumption (TEC) 3.37. Il TEC viene definito nell'equazione 3.38. Riguardo ai vincoli invece nella 3.39 viene imposto che ogni $j \in \mathcal{J}$ sia assegnato ad una sola macchina, evitando così che un job possa esser processato su diverse macchine. I vincoli 3.40 impone che ogni job $j \in \mathcal{J}$ venga processato per la sua intera durata. Nella

3.41 si impone che la durata di ogni periodo $k \in \mathcal{K}$, definita con l_k, non sia mai superata, nel senso che la quantità di processing time dei vari job assegnata alla macchina h durante il periodo k non superi l_k. I vincoli 3.42 definiscono un upper bound per $R_{j,h,k}$, imponendo che l'esecuzione di un job j sulla macchina h durante il periodo k è possibile solamente se la variabile di assegnamento $X_{j,h,k}$ ha valore uguale ad 1. In 3.43 si evita la preemption dei job $j \in \mathcal{J}$ su ogni macchina $h \in \mathcal{H}$, difatti impone che se un job è processato sulla macchina h durante il periodo k ($X_{j,h,k} = 1$) e non durante il periodo $k+1$ ($X_{j,h,k} = 0$) allora non potrà esser processato nei periodi successivi a $k+1$. Con i vincoli 3.44 viene imposto che se l'esecuzione di un job $j \in \mathcal{J}$ è assegnato sia al periodo $k-1$ che al periodo $k+1$ sulla macchina h, allora la quantità processata in k del job j ($R_{j,h,k}$) non dovrà esser minore della durata del periodo k, cioè l_k. I vincoli 3.45 definisco il limite superiore per il massimo numero di periodi durante il quale un job $j \in \mathcal{J}$, assegnato alla macchina $h \in \mathcal{H}$, può esser processato. Con i vincoli 3.46 si impone invece che il processing time accumulato durante tutti i periodi di un job $j \in \mathcal{J}$ sulla macchina $h \in \mathcal{H}$ può esser maggiore di 0, senza mai superare $p_{j,h}$, solo se j è assegnato alla macchina h, quindi quando $V_{j,h} = 1$. Infine in (3.47 - 3.50) vengono definiti i vari domini delle variabili utilizzate in questa formulazione.

3.3.2 Formulazione 4

Quest'ultima formulazione prende come esempio il modello presentato da Saberi-Aliabad et al. in «Energy-Efficient Scheduling in an Unrelated Parallel-Machine Environment Under Time-Of-Use Electricity Tariffs» [67]. L'obiettivo dello studio Saberi-Aliabad et al. era di presentare un modello migliore di quelli già presenti, ampliando così la ricerca e lo sviluppo in questo tema. Precisamente gli autori hanno sviluppato ed analizzato un nuovo modello MIP, con meno variabili e vincoli rispetto al modello di Che et al., ed un'euristica per la risoluzione di istanze più grandi.

Parametri

Tutti i parametri usati per questa formulazione corrispondono a quelli utilizzati nella (3.3.1), tranne che per $\tau_{j,h}$ (3.32), al suo posto si utilizza:

$$p_j^{max} = \max_{h \in \mathcal{H}} p_{j,h} \qquad j \in \mathcal{J} \tag{3.51}$$

p_j^{max} corrisponde al processing time più lungo per il job $j \in \mathcal{J}$ su tutte le macchine $h \in \mathcal{H}$.

Variabili

Esattamente come per 3.3.1 si definiscono $R_{j,h,k}$ e $V_{j,h}$ nel modo seguente:

$$0 \leq R_{j,h,k} \leq p_{j,h} \qquad j \in \mathcal{J}, h \in \mathcal{H}, k \in \mathcal{K} \tag{3.52}$$

$$V_{j,h} \in \{0,1\} \qquad j \in \mathcal{J}, h \in \mathcal{H} \tag{3.53}$$

Inoltre al posto della variabile $X_{j,h,k}$ utilizzata nella formulazione 3 (3.3.1), per l'associazione "job-to-period" si utilizza:

$$W_{j,k} \in \{0,1\} \qquad j \in \mathcal{J}, k \in \mathcal{K} \tag{3.54}$$

La variabile binaria $W_{j,k}$ avrà valore pari ad 1 se il job $j \in \mathcal{J}$ è processato durante il periodo $k \in \mathcal{K}$. In caso contrario assumerà il valore nullo. L'utilizzo di $W_{j,k}$ al posto di $X_{j,h,k}$ comporta ad un diminuzione del numero di variabili rispetto alla formulazione 3.

Formulazione

$$\min C_{\max} \tag{3.55}$$

$$\min TEC \tag{3.56}$$

soggetto ai vincoli:

$$TEC = \sum_{h\in\mathcal{H}} u_h \sum_{k\in\mathcal{K}} c_k \sum_{j\in\mathcal{J}} R_{j,h,k}, \tag{3.57}$$

$$\sum_{h\in\mathcal{H}} V_{j,h} = 1, \quad j \in \mathcal{J} \tag{3.58}$$

$$\sum_{k\in\mathcal{K}} R_{j,h,k} = p_{j,h} V_{j,h}, \quad j \in \mathcal{J}, h \in \mathcal{H} \tag{3.59}$$

$$\sum_{j\in\mathcal{J}} R_{j,h,k} \leq l_k, \quad h \in \mathcal{H}, k \in \mathcal{K} \tag{3.60}$$

$$\sum_{h\in\mathcal{H}} R_{j,h,k} \leq l_k W_{j,k}, \quad j \in \mathcal{J}, k \in \mathcal{K} \tag{3.61}$$

$$\sum_{h\in\mathcal{H}} R_{j,h,k} \geq l_k(W_{j,k-1} + W_{j,k+1} - 1),$$
$$j \in \mathcal{J}, k = 2, \ldots, D-1 \tag{3.62}$$

$$\sum_{h\in\mathcal{H}} R_{j,h,k} \geq W_{j,k}, \quad j \in \mathcal{J}, k \in \mathcal{K} \tag{3.63}$$

$$\sum_{g=k+2}^{K} W_{j,g} \leq (K-k-1)(1 - W_{j,k} + W_{j,k+1}),$$
$$j \in \mathcal{J}, h \in \mathcal{H}, \quad k = 1, \ldots, D-2 \tag{3.64}$$

$$\sum_{k\in\mathcal{K}} W_{j,k} \leq \frac{1}{\min\limits_{k\in\mathcal{K}} l_k} \sum_{h\in\mathcal{H}} p_{j,h} V_{j,h} + 2 \quad j \in \mathcal{J}, \tag{3.65}$$

$$\sum_{h\in\mathcal{H}} R_{j,h,k} \geq \sum_{h\in\mathcal{H}} \left(p_{j,h} V_{j,h} - \sum_{g=1}^{k-1} R_{j,h,g} \right)$$
$$- p_j^{max} \left(1 - W_{j,k} + \sum_{g=\min\{k+1,K\}}^{K} W_{j,g} \right), \tag{3.66}$$
$$j \in \mathcal{J}, k = 2, \ldots, D$$

$$W_{j,k} \geq W_{j,k-1} + W_{j,k+1} - 1,$$
$$j \in \mathcal{J}, k = 2, \ldots, D-1 \tag{3.67}$$

$$TEC \geq 0, \tag{3.68}$$

$$R_{j,h,k} \in [0, p_{j,h}], j \in \mathcal{J}, h \in \mathcal{H}, k \in \mathcal{K} \tag{3.69}$$

$$W_{j,k} \in \{0,1\}, j \in \mathcal{J}, k \in \mathcal{K} \tag{3.70}$$

$$V_{j,h} \in \{0,1\}, j \in \mathcal{J}, h \in \mathcal{H} \tag{3.71}$$

In 3.57 si definisce il TEC per questa formulazione. Successivamente in 3.58 si impone che ogni job $j \in \mathcal{J}$ sia assegnato ad una sola macchina (esattamente come in 3.39). I vincoli 3.59 garantiscono che se un job $j \in \mathcal{J}$ è assegnato alla macchina $h \in \mathcal{H}$ allora il processing time complessivo processato su h è uguale a $p_{j,h}$ solo se j è assegnato a h, quindi con $V_{j,h} = 1$. In caso contrario, $V_{j,h} = 0$ allora il processing time complessivo processato su h dovrà essere pari a 0. I vincoli 3.60 impongono che la somma dei processing time assegnati al periodo $k \in \mathcal{K}$ sulla macchina $h \in \mathcal{H}$ sia al massimo pari alla durata del periodo k: l_k. Il vincolo 3.61 impone che se un job $j \in \mathcal{J}$ è assegnato al periodo k, $W_{j,k} = 1$, allora il processing time cumulativo di j in k su tutte le macchine non deve superare l_k, la durata del periodo k, invece se il job j non è assegnato a k, $W_{j,k} = 0$, allora il processing time cumulativo di j in k su tutte le macchine dovrà esser pari a 0. I vincoli 3.62, esattamente come quelli in 3.44, impongono che il processing time di $j \in \mathcal{J}$ processato durante il periodo $k \in \mathcal{K}$ occupi tutto il periodo k se il job j è anche assegnato ai periodi $k-1$ e $k+1$. Con i vincoli 3.63 si impone che se un job $j \in \mathcal{J}$ è assegnato al periodo $k \in \mathcal{K}$, $W_{j,k} = 1$, allora il processing time cumulativo di j durante k su tutte le macchine dovrà essere almeno maggiore di 1. Con i vincoli 3.64 si garantisce la non preemption dei job sulle macchine, difatti viene imposto che se un job $j \in \mathcal{J}$ è assegnato al periodo $k \in \mathcal{K}$ e non al periodo $k+1$ allora la somma dei processing time processata nei periodi successivi a $k+1$ dovrà esser nulla. I vincoli 3.65 definiscono un limite superiore per il numero massimo di periodi assegnati all'esecuzione di $j \in \mathcal{J}$ sulla macchina $h \in \mathcal{H}$. Con $\min_{k \in \mathcal{K}} l_k$ si trova la più piccola durata dei vari periodi, per poi utilizzarla per dividere il processing time $p_{j,h}$ sulla macchina h assegnata per l'esecuzione del job j ottenendo così il numero massimo di periodi assegnati a j. Il vincolo 3.66 impone che

se un job $j \in \mathcal{J}$ è assegnato al periodo $k \in \mathcal{K}$, ma non al periodo $k+1$, allora la quantità di processing time processata durante k non potrà esser minore della quantità necessaria per terminare l'esecuzione del job j. Gli ultimi vincoli 3.67 impongono che se un job $j \in \mathcal{J}$ è assegnato al periodo $k-1$ e $k+1$ allora deve anche esser assegnato al periodo k. Infine (3.68) - (3.71) definiscono i domini delle variabili utilizzate per questa formulazione.

Capitolo 4

Implementazione e risultati computazionali

In questo capitolo vengono presentate le implementazioni e i risultati dei test effettuati per valutare in modo sperimentale e computazionale le formulazioni descritte nel Capitolo 3. Tutte le formulazioni sono state implementate in Java 16 con l'utilizzo delle API di CPLEX 22.1.0, mentre i test sperimentali sono stati eseguiti su un pc Windows 10 con una CPU AMD Ryzen 7 PRO 4650u (8 Core, 16 thread) e 24 GB di RAM.

Questo capitolo è così strutturato: nella sezione 4.1 si presentano le istanze utilizzate, descrivendo come sono state generate e come da esse sono stati ricavati tutti i parametri necessari per le quattro formulazioni. Invece nella sezione successiva 4.2 si introducono i due algoritmi esatti, rispettivamente nelle sottosezioni 4.2.1 e 4.2.2, che sono utilizzati per ottenere i punti della frontiera di Pareto, ovvero l'insieme delle soluzioni ottime del problema bi-obiettivo. Infine si presentano e discutono i risultati computazionali nella sezione 4.3 ottenuti sulle istanze precedentemente presentate.

4.1 Istanze e parametri

Una parte delle istanze impiegate nei test sono si basano su quelle utilizzate in [21] e in [70]. Si ricorda che il problema preso in considerazione in tali lavori era con macchine parallele identiche, quindi i processing time dei job p_j non dipendevano dalle macchine, e con l'utilizzo dei time slot. Ciò ha comportato a delle modifiche alle istanze per riadattarle al caso di macchine non omogenee. Oltre a queste sono state introdotte 30 ulteriori istanze, caratterizzate da *time period* più lunghi. Il totale di 60 istanze quindi è suddivisibile in 2 insiemi:

- Istanze ereditate da [21] e [70] e adattate al caso Rm:
  - Istanze con time period di corta e pseudo-casuale durata, numerate da 1 a 30;
- Nuove istanze:
  - Istanze con time period di durata fissata, numerate da 31 a 60;

I dati che caratterizzano il processo di generazione delle istanze comprendono i seguenti valori:

- N numero di job;
- M numero di macchine;
- il limite K dell'orizzonte temporale;
- D numero di time period (solo per istanze 31-60);

Tabella 4.1 Valori di N,M,N e D per ogni istanza utilizzata

Istanza	N	M	K	D	$\|\mathcal{P}\|$	p^{max}	Istanza	N	M	K	D	$\|\mathcal{P}\|$	p^{max}
1	5	3	50	37	4	4	**31**	50	5	300	5	8	8
2	5	3	100	71	4	4	**32**	50	5	300	10	8	8
3	10	3	50	42	4	4	**33**	50	5	300	15	8	8
4	10	3	100	79	4	4	**34**	100	5	300	5	8	8
5	15	3	50	33	4	4	**35**	100	5	300	10	8	8
6	15	3	100	77	4	4	**36**	100	5	300	15	8	8
7	20	3	50	31	4	4	**37**	150	5	300	5	8	8
8	20	3	100	73	4	4	**38**	150	5	300	10	8	8
9	25	3	50	38	4	4	**39**	150	5	300	15	8	8
10	25	3	100	81	4	4	**40**	200	5	300	5	8	8
11	5	4	50	38	4	4	**41**	200	5	300	10	8	8
12	5	4	100	73	4	4	**42**	200	5	300	15	8	8
13	10	4	50	39	4	4	**43**	50	10	300	5	8	8
14	10	4	100	79	4	4	**44**	50	10	300	10	8	8
15	15	4	50	37	4	4	**45**	50	10	300	15	8	8
16	15	4	100	69	4	4	**46**	100	10	300	5	8	8
17	20	4	50	39	4	4	**47**	100	10	300	10	8	8
18	20	4	100	60	4	4	**48**	100	10	300	15	8	8
19	25	4	50	40	4	4	**49**	150	10	300	5	8	8
20	25	4	100	69	4	4	**50**	150	10	300	10	8	8
21	5	6	50	37	4	4	**51**	150	10	300	15	8	8
22	5	6	100	85	4	4	**52**	200	10	300	5	8	8
23	10	6	50	35	4	4	**53**	200	10	300	10	8	8
24	10	6	100	77	4	4	**54**	200	10	300	15	8	8
25	15	6	50	40	4	4	**55**	50	20	300	5	8	8
26	15	6	100	70	4	4	**56**	50	20	300	10	8	8
27	20	6	50	38	4	4	**57**	50	20	300	15	8	8
28	20	6	100	71	4	4	**58**	100	20	300	5	8	8
29	25	6	50	32	4	4	**59**	100	20	300	10	8	8
30	25	6	100	78	4	4	**60**	100	20	300	15	8	8

Le istanze numerate da 1-30 sono date da un elemento (N, M, K) del prodotto cartesiano degli insiemi $N = \{5, 10, 15, 20, 25\},\ M = \{3, 4, 6\},\ K = \{50, 100\}$. In queste istanze la durata dei time period non era presa in considerazione per la loro generazione; difatti, come si può notare dalla tabella 4.1 (colonna D) tali istanze hanno una durata molto variabile e casuale. Invece nelle istanze 31-60 la durata dei time period viene decisa e fissata; infatti queste istanze sono date dagli elementi (N, M, K, D) del prodotto cartesiano degli insiemi: $N = \{50, 100, 150, 200\}, M = \{5, 10, 20\}, K = \{300\}$ e $D = \{5, 10, 15\}$, dove con D si intende il numero di time period.

Oltre a questi valori vengono generati pseudo-casualmente i parametri come il processing time $p_{j,h}, j \in \mathcal{J}, h \in \mathcal{H}$, i tassi di consumo energetico $u_h, \forall h$ e il costo energetico per ogni time slot $c_t, t \in \{1, 2, \ldots, K\}$. La generazione del valori è pseudo-casuale perché essi sono estratti da distribuzioni uniformi $U_{[a,b]}$, come dettagliato qui nel seguito:

- Istanze 1-30
  - $U_{[1,4]}$ per $p_{j,h}$;
  - $U_{[1,3]}$ per u_h;
  - $U_{[1,4]}$ per c_t;
- Istanze 31-60
  - $U_{[1,8]}$ per $p_{j,h}$;
  - $U_{[1,6]}$ per u_h;
  - $U_{[1,8]}$ per c_t;

Successivamente si determinano per ogni istanza gli insiemi necessari per la per la formulazione 2:

- $\mathcal{P}$: l'insieme delle diverse durate di tutti i processing time $p_{j,h}, \forall j \in \mathcal{J}, \forall h \in \mathcal{H}$;
- $\mathcal{P}_h$: l'insieme delle diverse durate dei processing time sulla macchina $h \in \mathcal{H}$;
- $\mathcal{J}_{d,h}$: l'insieme dei job aventi durata $d \in \mathcal{P}$ sulla macchina h.

Per le formulazioni 3 e 4 è necessario il calcolo delle durate dei periodi $l_k, k \in \mathcal{K}$ per le istanze 1-30 e l'assegnamento dei costi energetici a essi

$c_k, k \in \mathcal{K}$ per tutte le istanze. Il calcolo degli l_k consiste nel conteggio delle ripetizioni consecutive dei vari $c_t,\ t \in \{1, 2, \ldots, K\}$ durante tutto l'orizzonte temporale di durata K. Per chiarire si presenta un esempio prendendo i primi 15 time slot e i costi associati ad essi dell'istanza 1 (raffigurata in Fig.4.1). Partendo dal primo time slot si conteggia quante volte si ripresenta consecutivamente $c_t = 6$, al primo $c_t \neq 6$ si ferma il conteggio e successivamente si assegna ad l_1 il numero valore del conteggio, in questo caso $l_1 = 2$. Si itera il procedimento fino al termine dell'orizzonte temporale.

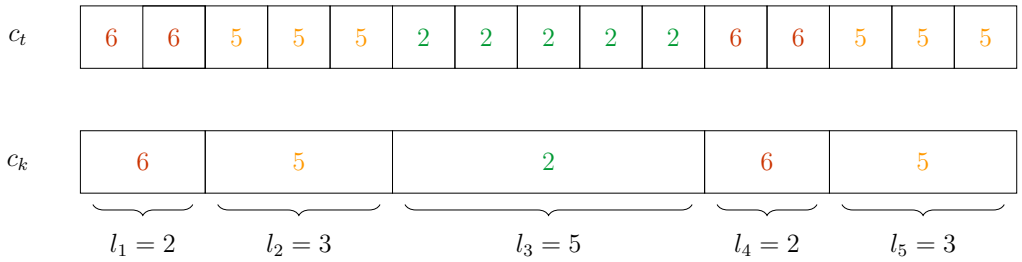


Figura 4.1: Esempio di calcolo degli $l_k, k \in \mathcal{K}$ e assegnamento dei $c_k, k \in \mathcal{K}$ dato un orizzonte temporale suddiviso in time slot e $c_t, t \in \mathcal{T}$

Oltre ai precedenti insiemi deve esser definito $p_j^{\max}$ per verificare la condizione sufficiente affinché BUPMSTP ammetta almeno una soluzione accettabile, sia se formulato con time slot (3.5) che con time period (3.31). Infine, sempre per le formulazioni 3 e 4, si devono determinare i valori di $\tau_{j,h} \forall j \in \mathcal{J}, h \in \mathcal{H}$ (3.32). Tutti questi insiemi e parametri sono ricavabili dai valori di $p_{j,h}, j \in \mathcal{J}, k \in \mathcal{K}$.

4.2 Implementazione

Il problema affrontato "Bi-objective Unrelated Parallel Machine energy-efficient Scheduling with Time-of-Use prices Problem (BUPMSTP)", è evidentemente un problema bi-obiettivo. Per trovare una soluzione a un problema multi-obiettivo possono essere utilizzati differenti metodi. In que-

sto caso la procedura utilizzata per la risoluzione del problema si basa sul "ϵ-constraint method" [39], già utilizzato per il caso a macchine parallele identiche in [21] e [70]. L'idea di base di questo metodo consiste nel minimizzare, o massimizzare, un solo dei due obiettivi presenti mentre l'altro obiettivo viene vincolato ad essere minore, o maggiore, a un valore fissato. Prima di definire l'algoritmo basato sul ϵ-constraint applicato al problema BUPMSTP si devono presentare le forme ridotte delle formulazioni 1 e 2. La formulazione ridotta consiste nel esplicitare il problema come se fosse a un solo obiettivo, precisamente andando a minimizzare solamente il TEC, scartando quindi l'obiettivo sul makespan.

La formulazione 1 ridotta si presenta così:

$$\min \quad TEC, \tag{4.1}$$

soggetto ai vincoli:

$$TEC = \sum_{h\in\mathcal{H}} u_h \sum_{j\in\mathcal{J}} \sum_{t=1}^{K-p_{j,h}+1} X_{j,h,t} \left(\sum_{i=t}^{t+p_{j,h}-1} c_i \right) \tag{4.2}$$

$$\sum_{h\in\mathcal{H}} \sum_{t=1}^{K-p_{j,h}+1} X_{j,h,t} = 1, \quad j \in \mathcal{J}, \tag{4.3}$$

$$\sum_{j\in\mathcal{J}} \sum_{i=\max\{1,t-p_{j,h}+1\}}^{t} X_{j,h,i} \leq 1, \quad h \in \mathcal{H}, t \in \mathcal{T}, \tag{4.4}$$

$$TEC \geq 0, \quad X_{j,h,t} \in \{0,1\}, \quad j \in \mathcal{J}, h \in \mathcal{H}, t \in \mathcal{T}. \tag{4.5}$$

Si sottolinea che oltre all'omissione dell'obiettivo di minimizzazione del makespan, sono stati rimossi anche i due vincoli relativi al completion time C_j dei vari job $j \in \mathcal{J}$ e al makespan della programmazione.

La formulazione 2 ridotta è composta da:

$$\min TEC, \tag{4.6}$$

soggetto ai vincoli:

$$TEC = \sum_{h \in \mathcal{H}} u_h \sum_{d \in \mathcal{P}_h} \sum_{t=1}^{K-d+1} b_{d,t}\, Y_{d,h,t} \tag{4.7}$$

$$\sum_{h \in \mathcal{H}} Z_{j,h} = 1, \quad j \in \mathcal{J} \tag{4.8}$$

$$\sum_{h \in \mathcal{H}} \sum_{d \in \mathcal{P}_h} \sum_{t=1}^{K-d+1} Y_{d,h,t} = N \tag{4.9}$$

$$\sum_{t=1}^{K-d+1} Y_{d,h,t} = \sum_{j \in \mathcal{J}_{d,h}} Z_{j,h}, \quad h \in \boldsymbol{\mathcal{H}}, d \in \boldsymbol{\mathcal{P}}_h \tag{4.10}$$

$$\sum_{d \in \mathcal{P}_h} \sum_{i=\max\{1,t-d+1\}}^{t} Y_{d,h,i} \leq 1,\ h \in \boldsymbol{\mathcal{H}}, t \in \boldsymbol{\mathcal{T}} \tag{4.11}$$

$$TEC \geq 0,$$

$$Y_{d,h,t} \in \{0,1\}, Z_{j,h} \in \{0,1\}$$

$$h \in \boldsymbol{\mathcal{H}}, d \in \boldsymbol{\mathcal{P}}_h, t \in \boldsymbol{\mathcal{T}}, j \in \boldsymbol{\mathcal{J}} \tag{4.12}$$

Anche in questa formulazione, come per la precedente, è stato rimosso il vincolo sul makespan (3.26). Per quanto riguarda le formulazioni 3 e 4 non è necessario definire nessuna forma ridotta perché in questo modello matematico non sono presenti vincoli che fanno riferimento alla lunghezza del makespan. L'unica operazione da effettuare è rimuovere l'obiettivo di minimizzazione del makespan (3.36 per la formulazione 3, 3.55 per la 4). Nelle prossime due sezioni verranno presentati ed esplicitati i due algoritmi, basati sul metodo ϵ-constraint, sia per le formulazioni indicizzate ai time slot (modelli 1 e 2), che per quelle indicizzate ai time period (modelli 3 e 4).

4.2.1 Algoritmo esatto per formulazioni indicizzate ai time slot

Per questa casistica vengono utilizzate le due formulazioni ridotte 4.1 e 4.6 presentate in precedenza.

Algoritmo 2 Algoritmo basato sul metodo ϵ-constraint per ottenere la frontiera di Pareto per le formulazioni 1 e 2

Input: Istanza $\mathcal{I}$ per il problema BPMSTP

Output: La frontiera Pareto $\mathcal{O}$: insieme delle soluzioni ottime per l'istanza $\mathcal{I}$

1: $\mathcal{O} \leftarrow \emptyset$
2: $\mathcal{S} \leftarrow \emptyset$
3: $\hat{K} \leftarrow K$
4: Calcola $K^{\text{lower bound}}$
5: **while** $\hat{K} \geq K^{\text{lower bound}}$ **do**
6: Risolvi la formulazione 1-2 ridotta
7: $\mathcal{S} \leftarrow$ Soluzione della formulazione ridotta
8: **if** $\mathcal{S} \neq$ feasible **then**
9: **break**
10: **end if**
11: Aggiorna $\mathcal{O} \leftarrow \mathcal{O} \cup \{\mathcal{S}\}$
12: $\hat{K} \leftarrow \hat{K} - 1$
13: **end while**
14: Rimuovi soluzioni dominate presenti in $\mathcal{O}$
15: **return** $\mathcal{O}$

Facendo riferimento allo pseudo codice dell'Algoritmo 2 se ne descrive nel seguito il funzionamento. L'esecuzione dell'algoritmo richiede un'istanza in input, mentre il risultato dell'esecuzione completa sarà la frontiera di Pareto $\mathcal{O}$, ovvero l'insieme di tutte le soluzioni non dominate. Per soluzione si

intende l'insieme di tutti i valori assegnati alle variabili dei modelli e i corrispondenti TEC e makespan. Nelle righe di preparazione (1 - 4) si assegnano ad $\boldsymbol{\mathcal{O}}$ e a $\boldsymbol{\mathcal{S}}$ l'insieme vuoto, a $\hat{K}$ la lunghezza dell'orizzonte temporale K e infine si calcola il limite inferiore per K, cioè il numero minimo di time slot necessari per ottenere una soluzione pari a: $K^{\text{lower bound}} = \frac{1}{M} \sum_{j \in \mathcal{J}} p_j^{\min}$. Nelle righe (5 - 13) è presente il vero e proprio Algoritmo basato sul metodo ϵ-constraint, difatti si fissa l'orizzonte temporale ad un massimo pari a $\hat{K}$ e si minimizza il solo obiettivo del TEC. Il ciclo while verrà effettuato finché la condizione $\hat{K} \geq K$ risulterà falsa, il che significa finché l'orizzonte temporale $\{1, 2, \ldots, K\}$ preso in considerazione risulta troppo piccolo per permettere di trovare una soluzione. Nelle righe (6 - 7) si risolve la formulazione MIP implementata e si assegna il risultato a $\boldsymbol{\mathcal{S}}$. Successivamente nelle righe (8 - 11) si controlla se il risultato ottenuto sia feasible o no, nel caso non lo fosse si interrompe il calcolo della frontiera e si restituisce ciò che fino a quel momento era stato calcolato. Invece se la soluzione risultasse feasible la si aggiunge all'insieme $\boldsymbol{\mathcal{O}}$, andando così ad aggiungere una nuova soluzione non dominata alla frontiera. Alla fine del ciclo, alla riga 12, si va ad aggiornare l'orizzonte temporale disponibile per la prossima iterazione, andando a sottrarre una unità di tempo, così nella iterazione successiva verrà calcolata una nuova soluzione ma che avrà un orizzonte temporale massimo disponibile minore. Alla fine del ciclo while si sarà ottenuto l'insieme di tutte le soluzioni trovate, per ottenere la frontiera di Pareto completa è necessario rimuovere dall'insieme $\boldsymbol{\mathcal{O}}$ le soluzioni dominate (riga 14). Effettuata l'operazione di rimozione è possibile restituire la esatta frontiera come stabilito nella 15.

4.2.2 Algoritmo esatto per formulazioni indicizzate ai time period

Algoritmo 3 Algoritmo basato sul metodo ϵ-constraint per ottenere la frontiera di Pareto per le formulazioni 3 e 4

Input: Istanza $\mathcal{I}$ per il problema BPMSTP

Output: La frontiera Pareto $\mathcal{O}$: insieme delle soluzioni ottime per l'istanza $\mathcal{I}$

1: $\mathcal{O} \leftarrow \emptyset$
2: $\mathcal{S} \leftarrow \emptyset$
3: $\hat{k} \leftarrow D$
4: **while** $M \sum_{k=1}^{\hat{k}} l_k \leq \sum_{j \in \mathcal{J}} p_j^{\min}$ **do**
5: Risolvi la formulazione 3-4
6: $\mathcal{S} \leftarrow$ Soluzione della formulazione
7: **if** $\mathcal{S} \neq$ feasible **then**
8: **break**
9: **end if**
10: Aggiorna $\mathcal{O} \leftarrow \mathcal{O} \cup \{\mathcal{S}\}$
11: **if** $l_{\hat{k}} > 1$ **then**
12: $l_{\hat{k}} \leftarrow l_{\hat{k}} - 1$
13: **else**
14: $\hat{k} \leftarrow \hat{k} - 1$
15: **end if**
16: **end while**
17: Rimuovi soluzioni dominate presenti in $\mathcal{O}$
18: **return** $\mathcal{O}$

Questo Algoritmo ha esattamente lo stesso scopo di quello presentato precedentemente (Algoritmo 2) ma è adattato per le formulazioni che si basano sui time period, quindi per le formulazioni 3 e 4. Il concetto alla base è il

medesimo, le uniche differenze sono sulla condizione del while e nell'aggiornamento dell'orizzonte temporale disponibile. Si ricorda che essendo presi in considerazione i time period in questo caso l'orizzonte temporale viene suddiviso in D time period, creando l'insieme dei time period: $\mathcal{K} = \{1, 2, \ldots, D\}$. Naturalmente viene presa in input un istanza $\mathcal{I}$ contenente tutti i dati e parametri necessari, successivamente, nelle righe (1 - 3) vengono inizializzate l'insieme delle soluzioni ottime $\mathcal{O}$, ovvero la frontiera di Pareto, l'insieme $\mathcal{S}$ contenente la soluzione trovata durante il ciclo while, che comprende i valori delle variabili ottenuti dalla risoluzione della formulazione, e infine l'iteratore $\hat{k}$ che indica l'ultimo time period, di durata $l_{\hat{k}}$ fino al quale è possibile trovare una soluzione. Ogni iterazione del ciclo while (righe 4 - 16) ha l'obiettivo di trovare una soluzione ottima da aggiungere alla frontiera di Pareto, come descritto nella riga 10, mentre nel caso la formulazione non porti ad una soluzione feasible si esce dal ciclo (vedi controllo alle righe 7 - 9) e si restituisce ciò che si era calcolato fino a quel momento. Dalla riga 11 alla 15 è presente l'aggiornamento dell'orizzonte temporale disponibile per la prossima iterazione. Nel dettaglio con la presenza dei time period l'aggiornamento deve esser svolto in questo modo: se l'ultimo time period $\hat{k}$ ha una durata $l_{\hat{k}}$ strettamente maggiore di 1 (riga 12) si diminuisce la sua durata di un'unità di tempo, invece nel caso avesse una durata minore di 1 si aggiorna l'iteratore $\hat{k}$, facendolo *puntare* al periodo precedente, ovvero $l_{\hat{k}-1}$ (riga 14). Alla fine del ciclo while nell'insieme $\mathcal{O}$ saranno presenti tutte le soluzioni ottime trovate, ma prima di ottenere l'esatta frontiera di Pareto è necessario eliminare le soluzioni $\in \mathcal{O}$ dominate, come riferito nella riga 17. Infine alla riga 18 si restituisce la frontiera completa.

4.3 Risultati

L'obiettivo dei due algoritmi esatti è ottenere l'insieme delle soluzioni ottime non dominate, ergo la frontiera di Pareto. Ogni punto appartenente alla

frontiera è rappresentato dalla coppia makespan e total cost energy, ($C_{\max}$, TEC), entrambi le quantità sono ottenute dallo scheduling conseguente dalla formulazione risolta. Per esempio si mostra la frontiera ottenuta dalla risoluzione dell'istanza 25:

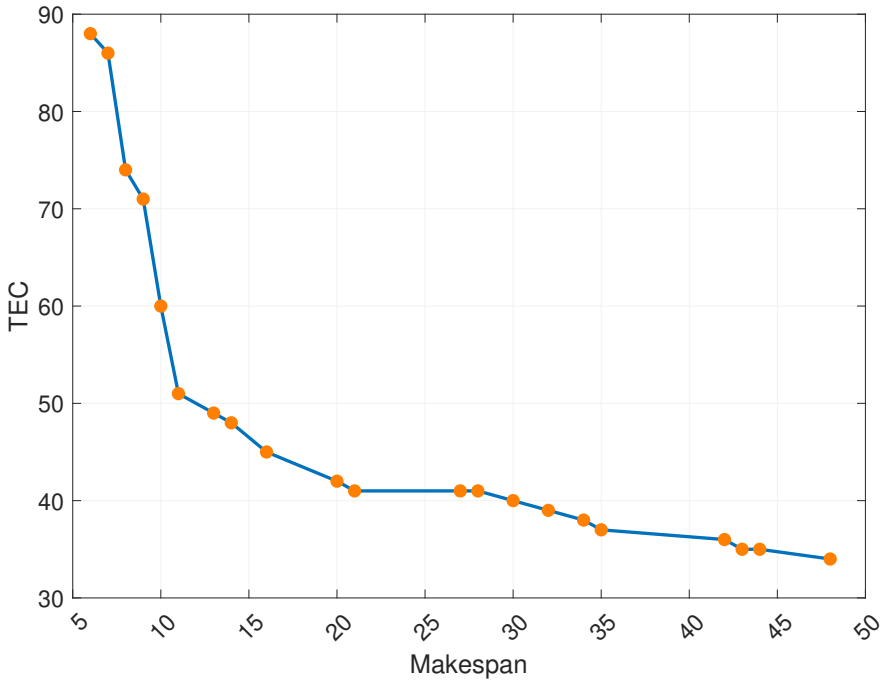


Figura 4.2: Fronte di Pareto ottenuto dalla risoluzione dell'istanza 25

Tutte le esecuzioni degli algoritmi per ogni istanza sono caratterizzate da un tempo limite pari a 3600 secondi (1 ora). Nel caso si superi il limite verrà restituito come risultato i punti della frontiera calcolati fino a quel momento. Per comparare le quattro formulazioni i due algoritmi esatti sono stati eseguiti con tutte le istanze presentate nella sezione 4.1. L'Algoritmo esatto 2 è stato utilizzato con le forme ridotte delle formulazioni 1 e 2, mentre l'Algoritmo 3 con le formulazioni 3 e 4. I tempi d'esecuzione sono raccolti nelle tabelle 4.2 e 4.3 e il loro andamento è rappresentato nelle figure 4.3 e 4.4.

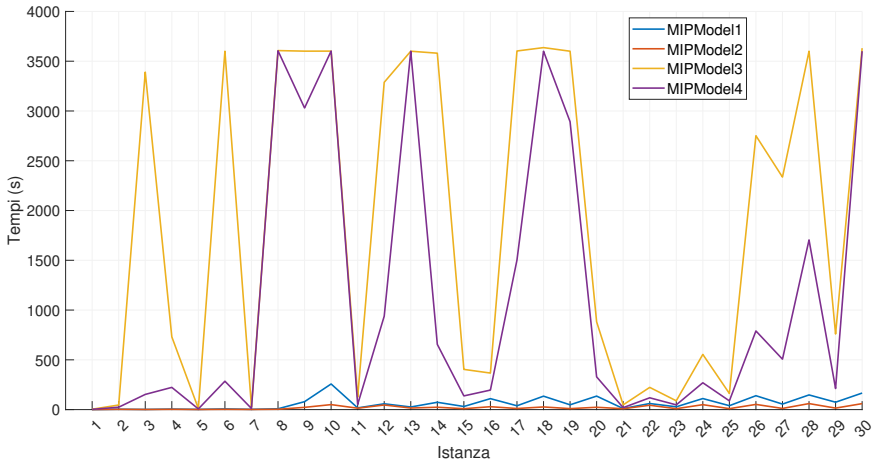


Figura 4.3: Grafico dei tempi di risoluzione delle istanze 1-30 riportati in secondi

Si suddividono le osservazioni in base all'insieme di istanze utilizzate. Partendo dalle prime 30 si può notare che le prime due formulazioni sono risultate le più performanti, nel particolare la seconda migliore della prima. In certi casi l'Algoritmo esatto 3 con i modelli 3 e 4 non è stato in grado di trovare una soluzione entro il tempo limite di 3600 secondi. Addirittura per il modello 3 nella metà delle istanze non è stata restituita una frontiera completa. Questo risultato non era inaspettato. Il motivo principale è la quantità di time period presenti nelle istanze 1-30, numero quasi pari alla durata dell'orizzonte temporale (vedi colonna D della tabella 4.1). Difatti i modelli 3 e 4 sono state formulati e pensati per esser più performanti con la presenza di time period di durata maggiore e non con durate quasi unitarie. Per questa ragione le quattro formulazioni vengono poi testate con le istanze 31-60, dove le durate dei periodi sono fissate e decisamente in numero minore rispetto alla lunghezza dell'orizzonte temporale.

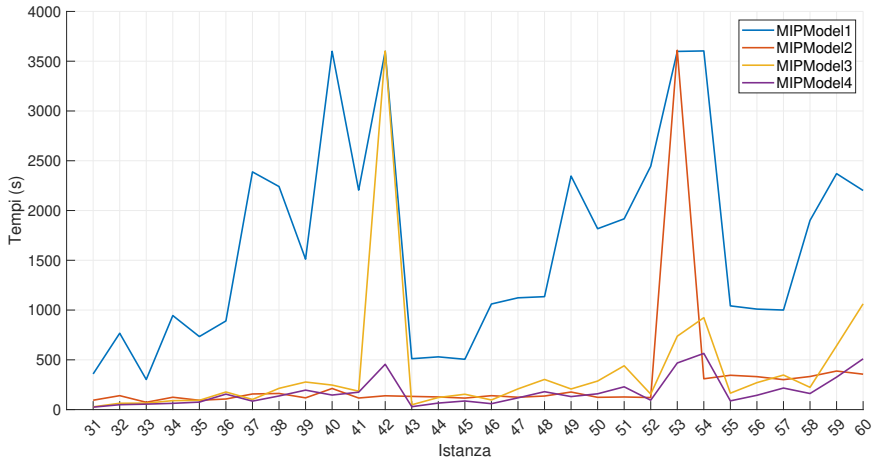


Figura 4.4: Grafico dei tempi di risoluzione delle istanze 31-60 riportati in secondi

Per le istanze 31-60, come si può notare dalla figura 4.4, è la formulazione 4 che risulta come la più performante per la maggior parte delle istanze. Precisamente su 21 istanze su 30 totali si posiziona come la più veloce a presentare una frontiera completa. Si nota che con un numero ridotto di time period ($D << K$) l'Algoritmo esatto 3 con la formulazione 4 non ha mai raggiunto il limite temporale, stessa cosa con la formulazione 3 esclusa l'istanza 42. In questo caso è addirittura la formulazione 1 a raggiungere più volte il limite. Sono da evidenziare gli ottimi risultati delle formulazioni 1 e 2 anche con queste istanze aventi un numero limitato di time period, in particolare i secondo modello, il quale dimostra che l'intuizione di indicizzare le variabili decisionali in base alle durate differenti dei job e non al loro id permette ottime prestazioni in tutte le istanze molto . Questi risultati permettono di comprendere la motivazione per cui sono state presentate quattro differenti formulazioni, confermando che in caso di un cambio frequente di prezzo durante l'orizzonte temporale (numero di time period elevato) la formulazione migliore risulta la numero 2, mentre in nel caso in cui il numero dei periodi sia decisamente minore rispetto alla durata dell'orizzonte temporale

le formulazioni indicizzate sui time period si rivelano le più performanti.

Tabella 4.2 Tempi d'esecuzione (in secondi) per la risoluzione delle istanze 1-30

Istanza	Algoritmo esatto		Algoritmo esatto	
	Formulazione 1	Formulazione 2	Formulazione 3	Formulazione 4
1	0.80	0.65	3.72	2.58
2	3.59	2.90	46.13	24.15
3	1.72	1.09	3389.94	152.55
4	5.22	2.97	731.04	223.10
5	2.05	1.07	10.97	6.87
6	6.93	2.84	3599.94	285.27
7	2.56	1.38	26.19	11.26
8	8.28	3.41	3606.71	3604.82
9	79.72	22.62	3601.11	3030.35
10	257.36	50.10	3601.28	3602.18
11	17.03	14.08	124.66	51.58
12	58.30	46.25	3288.09	938.48
13	26.17	15.77	3600.36	3599.93
14	73.48	23.69	3580.56	655.21
15	31.29	8.95	403.85	137.61
16	110.10	29.04	367.03	196.10
17	38.39	11.06	3602.68	1502.20
18	134.89	26.61	3636.58	3600.17
19	48.15	9.89	3600.44	2892.11
20	135.63	23.39	881.87	328.88
21	10.49	8.89	47.53	19.64
22	60.96	43.78	223.19	118.59
23	28.02	9.13	88.33	47.44
24	110.98	49.50	554.32	269.64
25	39.26	9.49	162.03	87.73
26	139.45	52.05	2751.34	790.00
27	55.41	11.77	2335.85	506.57
28	147.34	60.34	3600.29	1704.47
29	74.33	15.13	759.54	211.89
30	166.08	59.94	3631.13	3600.71

Tabella 4.3 Tempi d'esecuzione (in secondi) per la risoluzione delle istanze 31-60

Istanza	Algoritmo esatto		Algoritmo esatto	
	Formulazione 1	Formulazione 2	Formulazione 3	Formulazione 4
31	358.45	94.13	27.52	24.70
32	767.02	140.03	63.73	49.11
33	302.24	73.40	65.16	54.92
34	944.80	124.14	89.93	63.01
35	733.67	93.52	93.04	75.15
36	890.56	106.51	176.98	155.36
37	2387.98	156.94	101.99	85.15
38	2240.94	162.02	213.51	136.95
39	1511.15	118.00	277.01	195.62
40	3599.37	212.13	246.42	146.11
41	2204.80	116.35	184.71	174.96
42	3599.03	138.88	3601.48	455.15
43	511.07	132.46	48.25	29.22
44	529.89	126.98	121.37	65.50
45	505.39	115.56	154.27	86.49
46	1060.88	139.87	96.62	59.42
47	1122.98	124.26	208.00	118.03
48	1134.63	136.30	302.89	179.79
49	2346.13	176.46	207.14	130.80
50	1817.06	123.23	286.70	158.81
51	1916.49	126.87	440.34	228.85
52	2444.72	121.10	155.35	94.34
53	3598.38	3609.91	737.29	468.05
54	3603.26	309.44	923.53	564.49
55	1041.94	344.90	164.93	87.65
56	1009.59	331.06	270.69	143.74
57	1000.15	300.89	347.07	216.42
58	1900.97	332.41	223.56	161.46
59	2370.38	387.27	641.31	326.06
60	2200.98	355.87	1062.49	510.10

Per la completezza del confronto tra le formulazioni si riportano il numero di variabili e vincoli per ogni istanza nella tabella 4.4 generalizzato in base al numero di job N, numero di macchine M, numero di time slot K e numero di time period D.

Tabella 4.4 Numero di variabili e vincoli per i modelli presentati e citati rappresentati tramite i valori N, M, K, D

Modello	Tipologia	Numero di:	
		Variabili	Vincoli
1	Time slot	NMK	$MK + 2N$
2	Time slot	$\|\mathcal{P}_h\|MK + NM$	$\|\mathcal{P}_h\|(M \sum_{d \in \mathcal{P}_h}(K - d + 1) + M) + N + MK + 1$
3	Time period	$NM(2D + 1)$	$MD(3N + 1) + +2N(1 - 2M)$
4	Time period	$NM(D + 1) + ND$	$MD(N + 1) + +5ND - N(3 + M)$

Nelle tabelle 4.5 e 4.6 sono riportati i numeri effettivi di vincoli e variabili di ogni formulazione in base all'istanza.

Tabella 4.5 Tabella contenente il numero di variabili e vincoli per ogni formulazione in base alle istanze 1-30

Istanza	Formulazione 1		Formulazione 2		Formulazione 3		Formulazione 4	
	Variabili	Vincoli	Variabili	Vincoli	Variabili	Vincoli	Variabili	Vincoli
1	750	160	615	768	1125	1726	755	851
2	1500	310	1215	1518	2145	3358	1435	1633
3	1500	170	630	773	2550	3806	1710	1596
4	3000	320	1230	1523	4770	7247	3190	3002
5	2250	180	645	778	3015	4404	2025	1749
6	4500	330	1245	1528	6975	10476	4665	4081
7	3000	190	660	783	3780	5473	2540	2108
8	6000	340	1260	1533	8820	13159	5900	4964
9	3750	200	675	788	5775	8414	3875	3154
10	7500	350	1275	1538	12225	18218	8175	6723
11	1000	210	820	1022	1540	2362	970	1107
12	2000	410	1620	2022	2940	4602	1845	2122
13	2000	220	840	1027	3160	4696	1990	1921
14	4000	420	1640	2027	6360	9656	3990	3881
15	3000	230	860	1032	4500	6598	2835	2568
16	6000	430	1660	2032	8340	12486	5235	4776
17	4000	240	880	1037	6320	9236	3980	3491
18	8000	440	1680	2037	9680	14360	6080	5360
19	5000	250	900	1042	8100	11810	5100	4385
20	10000	450	1700	2042	13900	20626	8725	7546
21	1500	310	1230	1530	2250	3442	1325	1532
22	3000	610	2430	3030	5130	8050	3005	3500
23	3000	320	1260	1535	4260	6290	2510	2515
24	6000	620	2460	3035	9300	14102	5450	5497
25	4500	330	1290	1540	7290	10710	4290	4085
26	9000	630	2490	3040	12690	18990	7440	7115
27	6000	340	1320	1545	9240	13468	5440	5038
28	12000	640	2520	3045	17160	25546	10060	9361
29	7500	350	1350	1550	9750	14042	5750	5227
30	15000	650	2550	3050	23550	35018	13800	12633

Tabella 4.6 Tabella contenente il numero di variabili e vincoli per ogni formulazione in base alle istanze 31-60

Istanza	Formulazione 1		Formulazione 2		Formulazione 3		Formulazione 4	
	Variabili	Vincoli	Variabili	Vincoli	Variabili	Vincoli	Variabili	Vincoli
31	75000	1600	12250	13591	2750	2875	1750	1400
32	75000	1600	12250	13591	5250	6650	3250	2700
33	75000	1600	12250	13591	7750	10425	4750	4000
34	150000	1700	12500	13641	5500	5725	3500	2750
35	150000	1700	12500	13641	10500	13250	6500	5300
36	150000	1700	12500	13641	15500	20775	9500	7850
37	225000	1800	12750	13691	8250	8575	5250	4100
38	225000	1800	12750	13691	15750	19850	9750	7900
39	225000	1800	12750	13691	23250	31125	14250	11700
40	300000	1900	13000	13741	11000	11425	7000	5450
41	300000	1900	13000	13741	21000	26450	13000	10500
42	300000	1900	13000	13741	31000	41475	19000	15550
43	150000	3100	24500	27131	5500	5650	3250	2925
44	150000	3100	24500	27131	10500	13200	6000	5500
45	150000	3100	24500	27131	15500	20750	8750	8075
46	300000	3200	25000	27181	11000	11250	6500	5775
47	300000	3200	25000	27181	21000	26300	12000	10850
48	300000	3200	25000	27181	31000	41350	17500	15925
49	450000	3300	25500	27231	16500	16850	9750	8625
50	450000	3300	25500	27231	31500	39400	18000	16200
51	450000	3300	25500	27231	46500	61950	26250	23775
52	600000	3400	26000	27281	22000	22450	13000	11475
53	600000	3400	26000	27281	42000	52500	24000	21550
54	600000	3400	26000	27281	62000	82550	35000	31625
55	300000	6100	49000	54211	11000	11200	6250	5975
56	300000	6100	49000	54211	21000	26300	11500	11100
57	300000	6100	49000	54211	31000	41400	16750	16225
58	600000	6200	50000	54261	22000	22300	12500	11825
59	600000	6200	50000	54261	42000	52400	23000	21950
60	600000	6200	50000	54261	62000	82500	33500	32075

Capitolo 5

Conclusioni e sviluppi futuri

In questa tesi si è approfondito il problema di pianificazione di lavori su macchine non correlate in presenza di tariffe orarie, basate sulla politica "Time-of-Use", con gli obiettivi di minimizzare il costo energetico totale e il tempo totale di completamento. A questo scopo sono state presentate quattro formulazioni matematiche, che si propongono come l'evoluzione di modelli già presenti in letteratura. Oltre a questo è presente una parte sperimentale dove sono state confrontate le tempistiche dei modelli per trovare gli insiemi delle soluzioni non dominate, ovvero la frontiera di Pareto, esponendo così i punti di forza e di debolezza di ognuna. I risultati ottenuti rispettano le aspettative, difatti le formulazioni 2 e 4 si rivelano come le migliori; nel particolare la seconda è più performante con le istanze aventi periodi molto corti (istanze 1-30), mentre la quarta con le istanze aventi periodi con durate maggiori (istanze 31-60). L'esito ottenuto ha aperto ad una nuova possibilità, difatti come sviluppo futuro sarà possibile presentare ed analizzare nuove formulazioni basate sulle idee della seconda e quarta formulazione. Dalla seconda si utilizzerebbe l'idea per distinguere i job sulla base della loro durata d'esecuzione che comporterebbe ad ottenere uno spazio di delle soluzioni più ristretto, mentre dalla quarta soluzione si userebbe il concetto di time period, che ha una rilevanza applicativa maggiore rispetto a quella dei time

slot. Oltre a questo possibile sviluppo è possibile espandere il problema sotto diversi aspetti, difatti essendo un problema di scheduling sono presenti diverse caratteristiche che possono esser applicate. Ad esempio si possono prendere in considerazione: gli istanti di rilascio dei job (*release time*), le date di scadenza dei job (*due date*), i tempi di allestimento (*set-up time*) delle macchine, che possono anche esser dipendenti dalla sequenza dei lavori schedulati. Per avvicinarsi ulteriormente ad applicazioni reali si possono introdurre proprietà come la produzione per lotti, che risulta un metodo impiegato frequentemente nel mondo manifatturiero, oppure la possibilità che la macchine presentino dei guasti, non permettendo così l'elaborazione in quel periodo. In aggiunta oltre ai due obiettivi, presi in considerazione in questa tesi, se ne possono considerare altri regolari o meno, rendendo il modello più vicino a realtà produttive.

Bibliografia

[1] A. Giret, D. Trentesaux e V. Prabhu. «Sustainability in manufacturing operations scheduling: A state of the art review». In: *Journal of Manufacturing Systems* 37 (2015), pp. 126–140. DOI: https://doi.org/10.1016/j.jmsy.2015.08.002.

[2] M. Zarte, A. Pechmann e I. Nunes. «Decision support systems for sustainable manufacturing surrounding the product and production life cycle – A literature review». In: *Journal of Cleaner Production* 219 (2019), pp. 336–349. DOI: https://doi.org/10.1016/j.jclepro.2019.02.092.

[3] Donella H Meadows et al. «The limits to growth: a report to the club of Rome (1972)». In: *Google Scholar* 91 (1972). URL: https://scholar.google.com/scholar?q=+author:Donella%5C%20H.%5C%20Meadows.

[4] Brian R. Keeble. «The Brundtland report: 'Our common future'». In: *Medicine and War* 4.1 (1988), pp. 17–25. DOI: 10/b5rsts.

[5] John Elkington e Ian H Rowlands. «Cannibals with forks: The triple bottom line of 21st century business». In: *Alternatives Journal* 25.4 (1999), p. 42. DOI: https://doi.org/10.1023/A:1006129603978.

[6] Marco Garetti e Marco Taisch. «Sustainable manufacturing: trends and research challenges». In: *Production Planning & Control* 23.2-3 (2012), pp. 83–104. DOI: 10.1080/09537287.2011.591619.

[7] IEA. *Tracking Industry 2021*. URL: https://www.iea.org/reports/tracking-industry-2021.

[8] Manish Kumar e Monto Mani. «Chapter 11 - Sustainability assessment in manufacturing: perspectives, challenges, and solutions». In: *Sustainable Manufacturing*. A cura di K. Gupta e K. Salonitis. Handbooks in Advanced Manufacturing. Elsevier, 2021, pp. 287–311. DOI: https://doi.org/10.1016/B978-0-12-818115-7.00013-4.

[9] US EPA - United States Environmental Protection Agency. *Sustainable Manufacturing*. URL: https://www.epa.gov/sustainability/sustainable-manufacturing.

[10] Mohamed Saeed Khaled et al. «An Analysis of Research Trends in the Sustainability of Production Planning». In: *Energies* 15.2 (2022). ISSN: 1996-1073. DOI: 10.3390/en15020483.

[11] EIA - US Energy Information Administration. *Annual Energy Outlook 2021 - World total primary energy consumption*. 2021. URL: https://www.eia.gov/outlooks/ieo/data/pdf/ref/A01_r.pdf (visitato il 17/03/2022).

[12] EIA - US Energy Information Administration. *Annual Energy Outlook 2021 - World total primary energy consumption*. 2021. URL: https://www.eia.gov/outlooks/ieo/data/pdf/ref/F01_r.pdf (visitato il 17/03/2022).

[13] M.F. Rajemi, P.T. Mativenga e A. Aramcharoen. «Sustainable machining: selection of optimum turning conditions based on minimum energy considerations». In: *Journal of Cleaner Production* 18.10 (2010), pp. 1059–1065. DOI: https://doi.org/10.1016/j.jclepro.2010.01.025.

[14] Alireza Zakariazadeh, Shahram Jadid e Pierluigi Siano. «Smart microgrid energy and reserve scheduling with demand response using stochastic optimization». In: *International Journal of Electrical Power & Energy Systems* 63 (2014), pp. 523–533. ISSN: 0142-0615. DOI: https://doi.org/10.1016/j.ijepes.2014.06.037.

[15] Feihu Hu, Xuan Feng e Hui Cao. «A Short-Term Decision Model for Electricity Retailers: Electricity Procurement and Time-of-Use Pricing». In: *Energies* 11.12 (2018). ISSN: 1996-1073. DOI: 10.3390/en11123258.

[16] Terna. *Total Load - Carico Totale*. 2022. URL: https://www.terna.it/en/electric-system/transparency-report/total-load (visitato il 18/03/2022).

[17] I. Horowitz e C.K. Woo. «Designing Pareto-superior demand-response rate options». In: *Energy* 31.6 (2006). Electricity Market Reform and Deregulation, pp. 1040–1051. ISSN: 0360-5442. DOI: https://doi.org/10.1016/j.energy.2005.02.013.

[18] Weiwei Cui e Biao Lu. «Energy-aware operations management for flow shops under TOU electricity tariff». In: *Computers & Industrial Engineering* 151 (2021), p. 106942. DOI: https://doi.org/10.1016/j.cie.2020.106942.

[19] Jian-Ya Ding et al. «Parallel Machine Scheduling Under Time-of-Use Electricity Prices: New Models and Optimization Approaches». In: *IEEE Transactions on Automation Science and Engineering* 13.2 (2016), pp. 1138–1154. DOI: 10.1109/TASE.2015.2495328.

[20] Michael Pinedo. *Scheduling*. 5th. Springer Publishing Company, Incorporated, gen. 2016. ISBN: 978-3-319-26578-0. DOI: 10.1007/978-3-319-26580-3.

[21] Davide Anghinolfi, Massimo Paolucci e Roberto Ronco. «A bi-objective heuristic approach for green identical parallel machine scheduling». In: *European Journal of Operational Research* 289.2 (2021), pp. 416–434. ISSN: 0377-2217. DOI: https://doi.org/10.1016/j.ejor.2020.07.020.

[22] R.L. Graham et al. «Optimization and Approximation in Deterministic Sequencing and Scheduling: a Survey». In: *Discrete Optimization II*. A cura di P.L. Hammer, E.L. Johnson e B.H. Korte. Vol. 5. Annals of Discrete Mathematics. Elsevier, 1979, pp. 287–326. DOI: https://doi.org/10.1016/S0167-5060(08)70356-X.

[23] Kaizhou Gao et al. «A review of energy-efficient scheduling in intelligent production systems». In: *Complex & Intelligent Systems* 6.2 (2020), pp. 237–249. DOI: 10.1007/s40747-019-00122-6.

[24] Alicia Troncoso Lora et al. *Application of Evolutionary Computation Techniques to the Optimal Short-Term Scheduling of the Electrical Energy Production*. DOI: 10.1007/978-3-540-25945-9_65.

[25] Yan He et al. «A bi-objective model for job-shop scheduling problem to minimize both energy consumption and makespan». In: *Journal of Central South University of Technology* 12.2 (giu. 2005), pp. 167–171. DOI: 10.1007/s11771-005-0033-x.

[26] Christian Gahm et al. «Energy-efficient scheduling in manufacturing companies: A review and research framework». In: *European Journal of Operational Research* 248.3 (2016), pp. 744–757. ISSN: 0377-2217. DOI: https://doi.org/10.1016/j.ejor.2015.07.017.

[27] Maroua Nouiri, Abdelghani Bekrar e Damien Trentesaux. «An energy-efficient scheduling and rescheduling method for production and logistics systems†». In: *International Journal of Production Research* 58.11 (2020), pp. 3263–3283. DOI: 10.1080/00207543.2019.1660826.

[28] Ada Che, Shibohua Zhang e Xueqi Wu. «Energy-conscious unrelated parallel machine scheduling under time-of-use electricity tariffs». In: *Journal of Cleaner Production* 156 (2017), pp. 688–697. ISSN: 0959-6526. DOI: https://doi.org/10.1016/j.jclepro.2017.04.018.

[29] Daniele Catanzaro, Raffaele Pesenti e Roberto Ronco. *Job Scheduling under Time-of-Use Energy Tariffs for Sustainable Manufacturing: A Survey*. LIDAM Discussion Papers CORE 2021019. Université catholique de Louvain, Center for Operations Research e Econometrics (CORE), gen. 2021.

[30] Bo Chen e Xiandong Zhang. «Scheduling with Time-of-Use Costs». In: *European Journal of Operational Research* 274 (nov. 2018), pp. 900–908. DOI: 10.1016/j.ejor.2018.11.002.

[31] Emre Celebi e J. David Fuller. «Time-of-Use Pricing in Electricity Markets Under Different Market Structures». In: *IEEE Transactions on Power Systems* 27.3 (2012), pp. 1170–1181. DOI: 10.1109/TPWRS.2011.2180935.

[32] Guohua Wan e Xiangtong Qi. «Scheduling with variable time slot costs». In: *Naval Research Logistics (NRL)* 57.2 (2010), pp. 159–171. DOI: https://doi.org/10.1002/nav.20393.

[33] M. R. Garey e D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness (Series of Books in the Mathematical Sciences)*. First Edition. W. H. Freeman, 1979. ISBN: 0716710455.

[34] Weiya Zhong e Xiaolei Liu. «A single machine scheduling problem with time slot costs». In: *Recent advances in computer science and information engineering*. Springer, 2012, pp. 677–681.

[35] Janardhan Kulkarni e Kamesh Munagala. «Algorithms for Cost-Aware Scheduling». In: *Approximation and Online Algorithms*. A cura di Thomas Erlebach e Giuseppe Persiano. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 201–214.

[36] Michal Penn e Tal Raviv. «Complexity and algorithms for min cost and max profit scheduling under time-of-use electricity tariffs». In: *Journal of Scheduling* 24 (feb. 2021), pp. 1–20. DOI: 10.1007/s10951-020-00674-3.

[37] Ada Che, Yizeng Zeng e Ke Lyu. «An efficient greedy insertion heuristic for energy-conscious single machine scheduling problem under time-of-use electricity tariffs». In: *Journal of Cleaner Production* 129 (2016), pp. 565–577. ISSN: 0959-6526. DOI: https://doi.org/10.1016/j.jclepro.2016.03.150.

[38] Ada Che et al. «Energy-efficient bi-objective single-machine scheduling with power-down mechanism». In: *Computers & Operations Research* 85 (2017), pp. 172–183. ISSN: 0305-0548. DOI: https://doi.org/10.1016/j.cor.2017.04.004.

[39] Y. Haimes, L. Lasdon e D. Wismer. «On a bicriterion formulation of the problems of integrated system identification and system optimization». In: *IEEE Transactions on Systems, Man, and Cybernetics* 1 (1971), pp. 296–297.

[40] Kan Fang et al. «Scheduling on a single machine under time-of-use electricity tariffs». In: *Annals of Operations Research* 238 (mar. 2016), pp. 199–227. DOI: 10.1007/s10479-015-2003-5.

[41] Lin Chen et al. «Optimal algorithms for scheduling under time-of-use tariffs». In: *Annals of Operations Research* 304.1 (2021), pp. 85–107. ISSN: 1572-9338. DOI: 10.1007/s10479-021-04059-3.

[42] Junheng Cheng et al. «Bi-objective optimization for single-machine batch scheduling considering energy cost». In: *2014 International Conference on Control, Decision and Information Technologies (CoDIT)*. 2014, pp. 236–241. DOI: 10.1109/CoDIT.2014.6996899.

[43] Cheng, Junheng et al. «Bi-objective optimization of single-machine batch scheduling under time-of-use electricity prices». In: *RAIRO - Operations Research* 50.4-5 (2016), pp. 715–732. DOI: 10.1051/ro/2015063.

[44] Fadi Shrouf et al. «Optimizing the production scheduling of a single machine to minimize total energy consumption costs». In: *Journal of Cleaner Production* 67 (2014), pp. 197–207. ISSN: 0959-6526. DOI: https://doi.org/10.1016/j.jclepro.2013.12.024.

[45] Mohammad Mohsen Aghelinejad, Yassine Ouazene e Alice Yalaoui. «Machine and production scheduling under electricity time varying prices». In: *2016 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)* (2016), pp. 992–996.

[46] MohammadMohsen Aghelinejad, Yassine Ouazene e Alice Yalaoui. «Production scheduling optimisation with machine state and time-dependent energy costs». In: *International Journal of Production Research* 56.16 (2018), pp. 5558–5575. DOI: 10.1080/00207543.2017.1414969.

[47] MohammadMohsen Aghelinejad, Yassine Ouazene e Alice Yalaoui. «Complexity analysis of energy-efficient single machine scheduling problems». In: *Operations Research Perspectives* 6 (2019), p. 100105. ISSN: 2214-7160. DOI: https://doi.org/10.1016/j.orp.2019.100105.

[48] Mohammad Mohsen Aghelinejad, Yassine Ouazene e Alice Yalaoui. «Preemptive Scheduling of a Single Machine with Finite States to Minimize Energy Costs». In: *Optimization and Decision Science: Methodologies and Applications*. A cura di Antonio Sforza e Claudio Sterle. Cham: Springer International Publishing, 2017, pp. 591–599. ISBN: 978-3-319-67308-0.

[49] Mohammad Mohsen Aghelinejad, Yassine Ouazene e Alice Yalaoui. «Energy efficient scheduling problems under Time-Of-Use tariffs with different energy consumption of the jobs». In: *IFAC-PapersOnLine* 51.11 (2018). 16th IFAC Symposium on Information Control Problems in Manufacturing INCOM 2018, pp. 1053–1058. ISSN: 2405-8963. DOI: https://doi.org/10.1016/j.ifacol.2018.08.468.

[50] Saeed Rubaiee e Mehmet Bayram Yildirim. «An energy-aware multiobjective ant colony algorithm to minimize total completion time and energy cost on a single-machine preemptive scheduling». In: *Computers & Industrial Engineering* 127 (2019), pp. 240–252. ISSN: 0360-8352. DOI: https://doi.org/10.1016/j.cie.2018.12.020.

[51] Saeed Rubaiee, Suna Cinar e Mehmet Bayram Yildirim. «An Energy-Aware Multiobjective Optimization Framework to Minimize Total Tardiness and Energy Cost on a Single-Machine Nonpreemptive Scheduling». In: *IEEE Transactions on Engineering Management* 66.4 (2019), pp. 699–714. DOI: 10.1109/TEM.2018.2846627.

[52] Shibohua Zhang et al. «Improved mixed-integer linear programming model and heuristics for bi-objective single-machine batch scheduling with energy cost consideration». In: *Engineering Optimization* 50.8 (2018), pp. 1380–1394. DOI: 10.1080/0305215X.2017.1400026.

[53] Peng Wu, Junheng Cheng e Feng Chu. «Large-scale energy-conscious bi-objective single-machine batch scheduling under time-of-use electricity tariffs via effective iterative heuristics». In: *Annals of Operations Research* 296 (gen. 2021), pp. 471–494. DOI: 10.1007/s10479-019-03494-7. URL: https://hal.archives-ouvertes.fr/hal-02424362.

[54] Shengchao Zhou, Mingzhou Jin e Ni Du. «Energy-efficient scheduling of a single batch processing machine with dynamic job arrival times». In: *Energy* 209 (2020), p. 118420. ISSN: 0360-5442. DOI: https://

doi.org/10.1016/j.energy.2020.118420. URL: https://www.sciencedirect.com/science/article/pii/S0360544220315279.

[55] Shijin Wang et al. «Bi-objective optimization of a single machine batch scheduling problem with energy cost consideration». In: *Journal of Cleaner Production* 137 (2016), pp. 1205–1215. ISSN: 0959-6526. DOI: https://doi.org/10.1016/j.jclepro.2016.07.206.

[56] Shijin Wang et al. «Bi-objective identical parallel machine scheduling to minimize total energy consumption and makespan». In: *Journal of Cleaner Production* 193 (2018), pp. 424–440. ISSN: 0959-6526. DOI: https://doi.org/10.1016/j.jclepro.2018.05.056.

[57] YiZeng Zeng, Ada Che e Xueqi Wu. «Bi-objective scheduling on uniform parallel machines considering electricity cost». In: *Engineering Optimization* 50.1 (2018), pp. 19–36. DOI: 10.1080/0305215X.2017.1296437.

[58] Shengchao Zhou et al. «A multi-objective differential evolution algorithm for parallel batch processing machine scheduling considering electricity consumption cost». In: *Computers & Operations Research* 96 (2018), pp. 55–68. ISSN: 0305-0548. DOI: https://doi.org/10.1016/j.cor.2018.04.009.

[59] Jens Rocholl, Lars Mönch e John Fowler. «Bi-criteria parallel batch machine scheduling to minimize total weighted tardiness and electricity cost». In: *Journal of Business Economics* 90.9 (2020), pp. 1345–1381. ISSN: 1861-8928. DOI: 10.1007/s11573-020-00970-6.

[60] Zhao-hong Jia et al. «Bi-criteria ant colony optimization algorithm for minimizing makespan and energy consumption on parallel batch machines». In: *Applied Soft Computing* 55 (2017), pp. 226–237. ISSN: 1568-4946. DOI: https://doi.org/10.1016/j.asoc.2017.01.044.

[61] Si-yuan Qian, Zhao-hong Jia e Kai Li. ≪A multi-objective evolutionary algorithm based on adaptive clustering for energy-aware batch scheduling problem≫. In: *Future Generation Computer Systems* 113 (2020), pp. 441–453. ISSN: 0167-739X. DOI: https://doi.org/10.1016/j.future.2020.06.010.

[62] Mao Tan, Hua-Li Yang e Yong-Xin Su. ≪Genetic algorithms with greedy strategy for green batch scheduling on non-identical parallel machines≫. In: *Memetic Computing* 11 (2019). DOI: 10.1007/s12293-019-00296-z.

[63] Joon-Yung Moon, Kitae Shin e Jinwoo Park. ≪Optimization of production scheduling with time-dependent and machine-dependent electricity cost for industrial energy efficiency≫. In: *The International Journal of Advanced Manufacturing Technology* 68 (set. 2013). DOI: 10.1007/s00170-013-4749-8.

[64] Junheng Cheng, Peng Wu e Feng Chu. ≪Mixed-integer programming for unrelated parallel machines scheduling problem considering electricity cost and makespan penalty cost≫. In: *2019 International Conference on Industrial Engineering and Systems Management (IESM)*. 2019, pp. 1–5. DOI: 10.1109/IESM45758.2019.8948095.

[65] Bobby Kurniawan et al. ≪Triple-chromosome genetic algorithm for unrelated parallel machine scheduling under time-of-use tariffs≫. In: *IEEJ Transactions on Electrical and Electronic Engineering* 15.2 (2020), pp. 208–217. DOI: https://doi.org/10.1002/tee.23047.

[66] Junheng Cheng, Feng Chu e Mengchu Zhou. ≪An Improved Model for Parallel Machine Scheduling Under Time-of-Use Electricity Price≫. In: *IEEE Transactions on Automation Science and Engineering* 15.2 (2018), pp. 896–899. DOI: 10.1109/TASE.2016.2631491.

[67] Hossein Saberi-Aliabad, Mohammad Reisi-Nafchi e Ghasem Moslehi. «Energy-Efficient Scheduling in an Unrelated Parallel-Machine Environment Under Time-Of-Use Electricity Tariffs». In: *Journal of Cleaner Production* 249 (nov. 2019), p. 119393. DOI: 10.1016/j.jclepro.2019.119393.

[68] Zixiao Pan, Deming Lei e Qingyong Zhang. «A New Imperialist Competitive Algorithm for Multiobjective Low Carbon Parallel Machines Scheduling». In: *Mathematical Problems in Engineering* 2018 (2018), pp. 1–13. DOI: 10.1155/2018/5914360.

[69] Zhi Pei et al. «An Approximation Algorithm for Unrelated Parallel Machine Scheduling Under TOU Electricity Tariffs». In: *IEEE Transactions on Automation Science and Engineering* 18.2 (2021), pp. 743–756. DOI: 10.1109/TASE.2020.2995078.

[70] Roberto Ronco. «Exact and Heuristic Algorithms for Energy-Efficient Scheduling». Tesi di dott. 2022. DOI: 10.48550/ARXIV.2203.14070. URL: https://arxiv.org/abs/2203.14070.

Ringraziamenti

Mi è doveroso dedicare questo spazio alle persone che hanno contribuito, con il loro instancabile supporto, ad arrivare al giorno della mia laurea.
In primis i miei due relatori: dott. Roberto Ronco e prof. Massimo Paolucci per la loro immensa disponibilità ed i loro instancabili consigli.
Ringrazio la mia famiglia che mi ha sempre sostenuto ed incoraggiato.
Ringrazio i miei amici ed i miei compagni di università per essere stati sempre presenti e di sostegno.

Grazie a tutti!
Stefano Lavaggi