UNIVERSITÀ DEGLI STUDI DI PADOVA

FACOLTÀ DI INGEGNERIA

DIPARTIMENTO DI TECNICA E GESTIONE DEI SISTEMI INDUSTRIALI

Corso di laurea triennale in INGEGNERIA GESTIONALE

TECNICHE MATEMATICHE PER IL PROJECT MANAGEMENT CON RISORSE MULTIPLE LIMITATE

Relatore: Ch.mo Prof. Romanin Jacur Giorgio

Laureando: GARZON ROBERTO
Matricola n. 541850

Anno Accademico 2012/2013

Ringrazio tutta la mia Famiglia per essermi sempre stata vicino,

per avermi incoraggiato, sostenuto e motivano nell'affrontare ogni sfida.

Ringrazio gli Amici di sempre che, a proprio modo,

mi hanno accompagnato lungo questo percorso.

Ringrazio tutte le persone con cui ho avuto il piacere di condividere tanti progetti

ed in particolar modo con i Colleghi che mi hanno sempre aiutato.

Voglio inoltre ringraziare tutte le persone speciali che ho avuto la fortuna di conoscere.

INDICE

CAPITOLO 3 – PROJECT SCHEDULING

INTRODUZIONE

In questa esposizione vengono trattati i concetti e le metodiche utilizzate per l'ottimizzazione e la schedulazione di attività all'interno di un progetto avendo come vincolo aggiuntivo la presenza di risorse limitate. Verrà sviluppato un programma in java, che mediante l'uso della tecnica euristica parallela, permetterà la risoluzione del problema C.P.M.- risorse. Verrà introdotto il concetto di Project Management, di come si è evoluto durante il corso della storia, di quali sono i concetti fondamentali che ruotano attorno ad esso e di cosa significa "Stima di un progetto".

Si introdurranno le due principali tecniche reticolari, le rappresentazioni grafiche che associano ad ogni progetto un grafo orientato (o reticolo). Verranno dunque forniti degli esempi relativamente alla rappresentazione americana ed a quella europea.

La trattazione poi parlerà delle principali tecniche reticolari, approfondendo il metodo C.P.M. – Critical Path Method. Verrà discusso della sua storia, della tecnica di base su cui si basa ed i principali vantaggi e svantaggi che compaiono nella sua adozione.

Successivamente si introdurrà il concetto di Project Scheduling, inteso come l'insieme di operazioni per determinare l'instante d'inizio di ciascuna attività in modo da minimizzare la durata complessiva di un progetto. Si accennerà brevemente ai metodi risolutivi per i problemi dove le risorse vengono considerate illimitate chiarendone gli aspetti principali, per poi approfondire, nell'ultima parte della trattazione, il metodo risolutivo per i problemi dove le risorse vengono considerate limitate utilizzando appunto il CPM con la tecnica euristica parallela. Al tal proposito verrà sviluppato un programma in java che permetterà, tramite l'adozione del metodo CPM, su un progetto formato da un insieme di attività, di calcolare il tempo di completamento, le attività appartenenti al percorso critico per poi passare alla schedulazione delle attività utilizzando 2 risorse limitate.

CAPITOLO 1 – INTRODUZIONE AL PROJECT MANAGEMENT

1.1. GENERALITA'

Con l'espressione inglese **Project Management**, si intende l'insieme di attività, tra gruppi diversi di persone, volte alla realizzazione degli scopi/obiettivi di un progetto complesso, di cui è responsabile il Project Manager. La definizione che viene utilizzata dalla maggior parte degli esperti in materia è che per progetto si intenda "uno sforzo complesso comportante compiti interrelati eseguiti da varie organizzazioni, con obiettivi, schedulazioni e budget ben definiti" (Archibald 1994).

Un **progetto** si può considerare come un insieme di attività finalizzate al raggiungimento di un risultato definito in modo univoco, attraverso l'utilizzo risorse di varia natura quali umane, materiali, finanziarie nel rispetto dei vincoli prefissati di tempo, costo e qualità.

Nella maggior parte dei casi l'obiettivo del coordinamento e della gestione di queste attività consiste nel completamento in **un tempo minimo** dell'intero progetto, rispettando vincoli tra loro diversi che interessano la durata delle attività, le precedenze fra attività e l'uso di risorse diverse. In altri casi, una volta noti i legami fra costo e durata di ogni attività, l'obiettivo può essere la **minimizzazione del costo totale** del progetto stesso. In altri casi ancora possono essere considerati molteplici obiettivi, ad esempio il tempo di completamento e un indice di qualità legato all'organizzazione del lavoro.

L'obiettivo principale del **project management** è quello di raggiungere gli obiettivi del progetto restando all'interno dei vincoli determinati dal contesto del committente, quali il costo, il tempo e lo scopo (nel senso anche della qualità). L'obiettivo secondario è quello di ottimizzare per quanto possibile l'allocazione delle risorse e integrare gli input necessari a raggiungere gli obiettivi definiti.

1.2. STORIA

Nel passato alcuni fondamenti della cultura del project management si sono sviluppati in civiltà geograficamente distanti e tra loro diverse. Ne sono esempi ben visibili la cultura Egizia con le Piramidi, il Colosseo o i grandi acquedotti romani, che rimangono ancora oggi testimonianze concrete di come queste grandi costruzioni non potrebbero mai essere state create senza alcuna base nel campo del project management.

Per testimoniare come già in tempi antichi, l'idea della gestione di risorse complesse in ottica progettuale fosse diffusa e in qualche modo strutturata, è utile citare il "De bello Gallico": in alcuni paragrafi del libro IV, Giulio Cesare descrive i dettagli tecnici ed organizzativi, quali tempi, obiettivi, materiali utilizzati, e manodopera nella costruzione di un ponte sul Reno nel corso della V campagna di Gallia. Già con queste righe, si possono capire come elementi fondamentali di un moderno di progetto, si trovassero in epoca romana.

Durante poi il corso dei secoli e soprattutto a causa della scomparsa dell'impero romano, vennero perdute queste capacità ingegneristiche di realizzazione di grandi opere, presenti soprattutto nel genio militare delle legioni romane. Non fu solamente il disfacimento del glorioso impero romano l'unica causa, infatti anche la scomparsa dello schiavismo fu un notevole fattore di rallentamento legato al difficile reperimento di manodopera, che grazie agli schiavi, era gratuita ed in forma pressochè illimitata. L'esempio più significativo è sicuramente dato dalle costruzione della piramide di Cheope dove lavorarono 100.000 schiavi per più di 20 anni.

In epoca moderna il project management si è sviluppato a partire da diversi campi di applicazione partendo in primo luogo dal dipartimento della Difesa Americano (che controllava logistica e organico militare) per poi estendersi nel settore delle costruzioni, nell'ingegneria industriale, ed in tempi più recenti con l'avvento dei computer, nella realizzazione dei programmi software.

L'ingegnere statunitense ***Henry Gantt*** (1861-1919), che introdusse nei primi anni del XX secolo una tecnica di pianificazione che ancora oggi porta il suo nome – il Diagramma di Gantt - fu uno dei maggiori contributori alla teorie del project management. Creò uno strumento grafico per la rappresentazione sull'asse temporale delle attività che concorrono al completamento di un progetto, permettendone così la programmazione ed il controllo dell'avanzamento. Henry Gantt fu un collaboratore di Taylor e sviluppò questa tecnica per contribuire all'introduzione delle sue teorie. Questa tecnica, diventata fondamentale nelle fasi di pianificazione, portò successivamente alla nascita di fondamentali concetti ampiamente usati nelle prassi di project management, come quello di allocazione delle risorse e della Work Breakdown Structure (WBS), utilizzato per rappresentare la struttura delle attività di un progetto.

Durante la seconda guerra mondiale e nel periodo successivo iniziaro a prendere vita i primi veri e propri progetti strutturati secondo una concezione moderna del project management; ne fu un esempio Il Progetto Manhattan, creato dagli Stati Uniti con l'obiettivo di anticipare gli sforzi in corso da parte del governo nazista nella realizzazione di armi nucleari. Il fisico Robert Oppenheimer che fu nominato direttore del progetto nel 1942, sfruttando la sua abilità organizzativa e la sua profonda conoscenza teorica, riuscì a dare una forma organizzativa e una profonda motivazione a tutto il team di progetto.

Negli anni successivi al 1950 vennero sviluppate altre importanti tecniche: il **PERT** (Program Evaluation and Review Technique) sviluppato dalla società Booz Allen Hamilton per il progetto di sviluppo del missile Polaris da parte della Marina statunitense ed il **CPM** (Critical Path Method) sviluppato congiuntamente da DuPont Corporation and Remington Rand Corporation per gestire i progetti di manutenzione degli impianti industriali.

Nei primi anni 60, si incominciarono a vedere i primi sistemi di gestione e pianificazione dei progetti così come sono oggi, che vennero applicati tra i primi dalla Nasa al Programma Apollo, che culminò nel luglio del 1969 con lo sbarco sulla Luna dell' Apollo 11. Questi progetti presentavano una pianificazione eseguita partendo dall'alto per giungere al dettaglio (top-down), mentre i controlli vengono effettuati in senso inverso partendo dal basso per giungere in cima (bottom-up). Con queste metodologie sì ottenne anche la metrica per comparare l'avanzamento del progetto "Actual Cost of Work Performed" sia con quanto programmato "Budgeted Cost of Work Scheduled", sia con quanto effettivamente valorizzato a costi di progetto "Earned Value", permettendo un'effettiva comprensione dello stato del progetto in relazione con l'avanzamento temporale a quello economico.

Negli anni '70 il Project Management si consolida definitivamente nei settori dell'impiantistica e dell'edilizia, ed emerge l'importanza dei cosiddetti "stake-holder", il comitato di controllo, esterno al progetto, con cui è necessario confrontarsi. Il progetto Concorde, l'aereo supersonico di produzione francese, ne è un esempio e fu fortemente penalizzato dal fatto che gli Stati Uniti non ne autorizzarono l'atterraggio fino al 1976.

Nei primi anni '80 il Project Management si diffonde definitivamente anche in altre industrie, tra cui l'industria IT di produzione di Hardware e Software ed in generale dei sistemi informativi aziendali che in quegli anni fanno la loro prima comparsa. Inoltre, la rapida evoluzione del settore, soprattutto in seguito all'introduzione del personal computer, favorisce la diffusione di sistemi di gestione dei progetti, di schedulazione, di controllo, che funzionano sul computer sfruttandone la potenza di calcolo.

La figura dell'ingegnere gestionale si evolse e, grazie al lavoro di Hans Lang e altri, vennero sviluppate tecnologie per la stima dei costi di progetto e per il controllo. Nel 1956 venne fondata la "American Association of Cost Engineers" (ora AACE International - "Association for the Advancement of Cost Engineering") da parte dei primi cultori del project management che continuò la sua attività di sviluppo degli standard tecnologici e nel 2006 ha rilasciato il "Total Cost Management Framework" sviluppato da John Hollman.

Nel 1969 viene fondato il **Project Management Institute** (PMI) con l'obiettivo di diffondere e rafforzare le regole di project management attraverso l'affermazione di uno standard a livello internazionale, sulla base della convinzione che i diversi campi di applicazione del project management avessero una larga base comune nelle tecnologie e nelle metodologie di gestione dei progetti. Nel 1981 il Comitato Direttivo autorizzò lo sviluppo della Guida al "Project Management Body of Knowledge" (altrimenti noto come **PMBOK**), contenente una guida completa e sintetica degli standard e delle linee guida indispensabili per il project manager. L'International Project Management Association (IPMA), fondata in Svizzera nel 1965, riconosciata in più di 50 nazioni, ha intrapreso una direzione simile istituendo l'IPMA Competence Baseline (ICB).

1.3. CONCETTI E TECNICHE FONDAMENTALI

Premesso che sia a livello internazionale che in Italia circola una quantità notevole di metodologie e di tecniche correlate alla teoria del project management, esistono dei concetti e delle tecniche fondamentali ricorrenti nella maggior parte degli approcci esistenti.

CARATTERISTICHE E CICLIO DI VITA DI UN PROGETTO

Le caratteristiche principali di un progetto si possono riassumere in quanto segue:

- Obiettivi: ogni attività deve avere una finalizzazione specifica il cui risultato concorre alla realizzazione dell'obiettivo del progetto

- Unicità: un progetto, anche se può essere simile ad altri progetti per tecniche utilizzate, è unico in quanto a risorse impiegate ed ambiente in cui viene sviluppato

- Temporaneità: in ogni progetto esiste una data di inizio ed una di fine

- Multidisciplinarità: nella pratica un progetto richiede competenze diversificate in vari ambiti che devono essere definiti in modo rigoroso permettendo una convergenza comune verso l'obiettivo del progetto

- Risorse limitate: qualsiasi progetto necessita di una certa disponibilità di risorse per le varie attività che compongono il progetto stesso. Per questo è necessario gestire una pianificazione attenta per ottenere una migliore allocazione possibile e risolvere eventuali conflitti generati dalla loro scarsità

Poiché un progetto implica un certo livello d'incertezza e quindi di rischio, è necessario definire delle fasi che fanno parte del ciclio di vita di un progetto "Project Life Cycle". Ogni fase si conclude con la realizzazione di un prodotto tangibile e valutabile che ne permette la valutazione ed eventualmente l'introduzione di azioni correttive. Rimanendo ad un livello di astrazione più alto possibile le fasi di un progetto possono essere identificate come le seguenti:

- Concettualizzazione

  Consiste nella valutazione dell'idea principale e comporta un'analisi preliminare che raggruppa varie attività tra cui lo studio di fattibilità, l'analisi economica e la valutazione dei rischi rispetto a tempi e costi.

- Pianificazione

  Vengono identificate tutte le attività che compongono il progetto e le risorse che verranno impiegate per realizzarlo

- Test

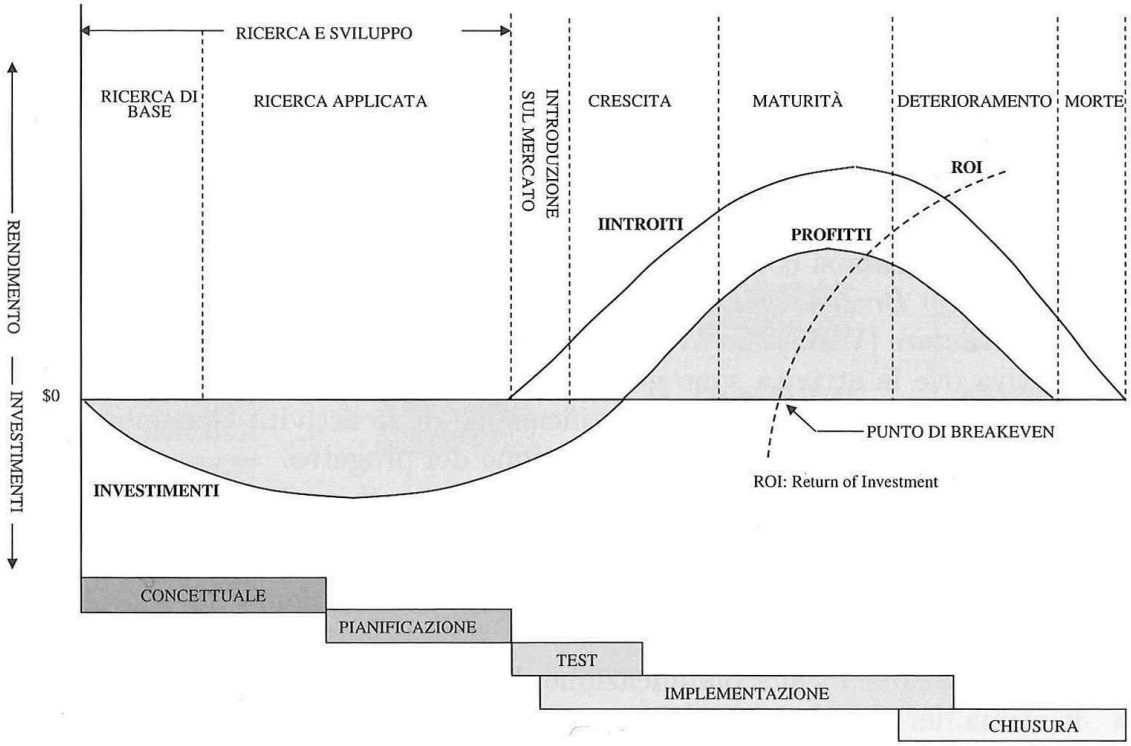
  Viene effettuata una fase di verifica e di standardizzazione finale che comprende la creazione di tutta la documentazione

- Implementazione

  Viene eseguito il progetto che porterà alla creazione di un bene o servizio in accordo con gli obiettivi definiti nella fase di concettualizzazione

- Chiusura

  Rappresenta la chiusura del progetto ed il rilascio delle risorse che saranno disponibili per l'avvio di altri progetti


RICERCA E SVILUPPO
RICERCA DI BASE
RICERCA APPLICATA
INTRODUZIONE SUL MERCATO
CRESCITA
MATURITÀ
DETERIORAMENTO
MORTE
RENDIMENTO
INVESTIMENTI
$0
IINTROITI
ROI
PROFITTI
PUNTO DI BREAKEVEN
ROI: Return of Investment
INVESTIMENTI
CONCETTUALE
PIANIFICAZIONE
TEST
IMPLEMENTAZIONE
CHIUSURA

1.4. STIMA DI UN PROGETTO

La stima dimensionale di un progetto è una delle prime attività cruciali da cui dipende il successo del progetto stesso. I passi comuni alla maggior parte delle tecniche di pianificazione prevedono le seguenti fasi:

- identificare le **attività elementari** - **Task** che comprendono tutte quelle operazioni avanti carattere di omogeneità e continuità necessarie a produrre dei risultati tangibili e misurabili – **Deliverable -** associati a ciascun elemento della **Work Breakdown Structure – WBS**

- rappresentare la scomposizione dei task in un diagramma di Gantt, mettendo in evidenza le interrelazioni ed i vincoli di precedenza tra i diversi elementi del progetto (macro-attività o work packages) in una scala temporale

- valorizzare la quantità di lavoro necessaria (il cosiddetto effort) a completare ciascun task, determinando la tipologia di risorse (umane e non) necessarie alla loro realizzazione

- calcolare i tempi di realizzazione di ciascun task in base al numero di risorse a loro assegnate

- determinare i costi del personale per la realizzazione di ciascun task moltiplicando la quantità di lavoro (effort) stimato per i costi medi della tipologie di risorse individuate; aggiungere i costi degli altri materiali e/o servizi necessari

- determinare il percorso critico in base alle dipendenze esistenti dentro la WBS

- calcolare il **tempo totale** (il cosiddetto elapsed) sommando i tempi di tutti i task che si trovano all'interno del percorso critico

- determinare il **costo totale** sommando i costi (personale + materiali + servizi) di tutti i task

Esistono, come si può immaginare, molteplici tecniche per quantificare i tempi e i costi necessari a realizzare un progetto o, se si vuole, la sua durata. La durata del progetto naturalmente dipende dalla struttura della pianificazione adottata, ed in particolare dal grado di parallelismo tra le attività che compongono il progetto, parallelismo a sua volta dipendente dal numero di risorse impiegate.

CAPITOLO 2 – MODELLI E TECNICHE RETICOLARI

2.1. RAPPRESENTAZIONI GRAFICHE

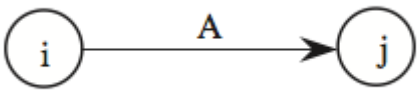
MODELLI RETICOLARI

Alla base dei metodi reticolari di programmazione c'è la costruzione del diagramma reticolare, che rappresenta la successione temporale e la reciproca dipendenza delle varie attività che concorrono all'esecuzione del progetto. Il primo passo nella costruzione del diagramma reticolare consiste nell'individuazione e nell'elencazione di tutte le attività coinvolte nell'esecuzione del progetto.

I **Modelli Reticolari** sono delle rappresentazioni grafiche che permettono di rappresentare l'insieme di attività che costituiscono un progetto ed i vincoli di precedenza esistenti sulle varie attività. Questa rappresentazione, chiamata *project network*, è composta da un *grafo orientato aciclico* o *reticolo* **R** che rappresenta l'insieme dei nodi **N** e dei archi orientati **A**:

R = (N,A)

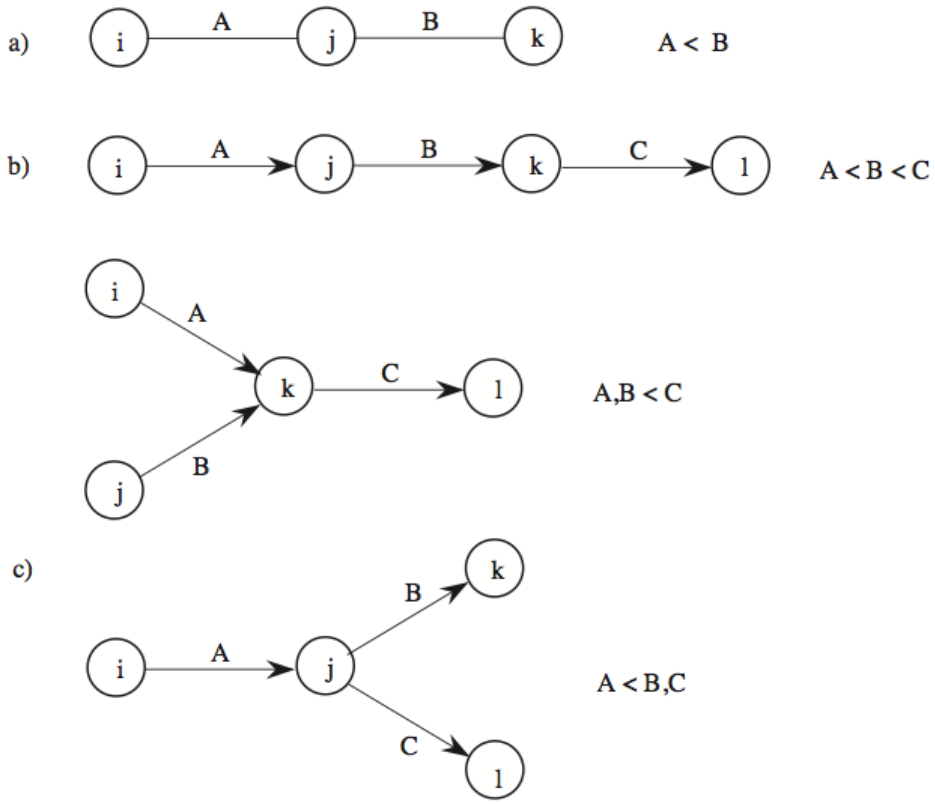
I nodi rappresentano gli eventi di inizio/fine di una attività, mentre gli archi rappresentano le attività con la propria durata. Pertanto nei diagrammi reticolari un'attivita A é rappresentata come in figura ove i nodi i e j rappresentano rispettivamente l'inizio e il termine dell'attività.



Nel disegnare il diagramma reticolare si utilizzano le seguenti regole fondamentali:

- Le attività sono rappresentate dai rami del grafo.
- L'inizio di un'attività è subordinato al completamento di tutte quelle che la precedono: in termini di diagramma reticolare ciò significa che rami diretti verso un nodo rappresentano attività da completare prima che abbiano inizio le attività rappresentate da rami aventi origine nel nodo stesso
- La lunghezza dei rami o la loro forma non hanno significato
- Due nodi non possono essere collegati da più di un ramo

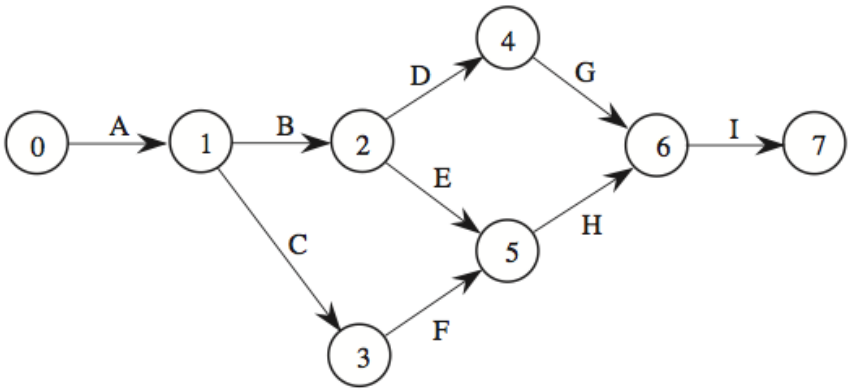
- L’inizio del progetto è rappresentato da un nodo contrassegnato con zero.
- Tutti i nodi sono numerati in modo che, se esiste un ramo diretto dal nodo *i* al nodo *j*, risulta i < j
- Il grafo può avere un solo nodo iniziale e un solo nodo finale.



Attualmente esistono 2 metodologie per la rappresentazione di un reticolo: la rappresentazione Americana e quella Europea.

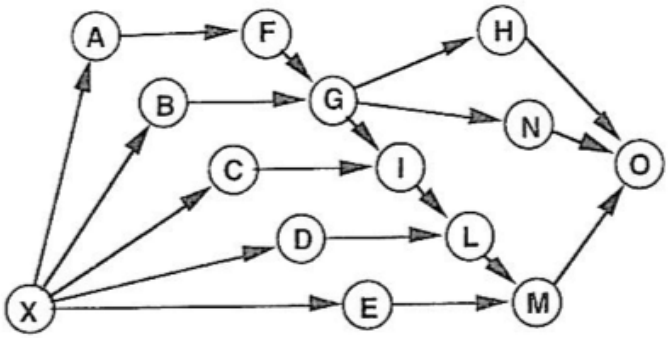
2.1.1. Rappresentazione Americana

Nella metodologia America di rappresentazione gli archi rappresentano le attività (activity on arrows – AOA) ed i nodi rappresentano gli eventi. Ad ogni attività è associato un arco orientato la cui valutazione corrisponde alla durata; ogni nodo rappresenta un evento, ossia l'istante di tempo che separa la fine di tutte le attività associate agli archi entranti nel nodo stesso dall'inizio di tutte le attività associate agli archi uscenti. Ogni evento ha durata nulla e nessuna attività uscente da un evento X può iniziare prima che siano terminate tutte le attività entranti in X.



2.1.2. Rappresentazione Europea

Nella metodologia Europea di rappresentazione con reticolo ad attività sui nodi (activity on nodes – AON) ad ogni attività è associato un nodo, mentre un arco orientato (i,j) rappresenta la precedenza dell'attività i sulla j.



2.1.3. ESEMPIO DI MODELLO RETICOLARE CON ATTIVITA' SUGLI ARCHI

A titolo di esempio consideriamo il seguente progetto: l'obiettivo di questo progetto, all'interno di un impianto di riscaldamento, è la sostituzione della vecchia caldaia a gasolio con una nuova caldaia a gas. Il progetto è stato suddiviso in attività fra le quali esistono vincoli di precedenza del tipo *Finish to Start*: un'attività vincolata può iniziare solo se tutte le attività vincolanti (precedenti) sono state completate. Le attività sono elencate nella tabella seguente:

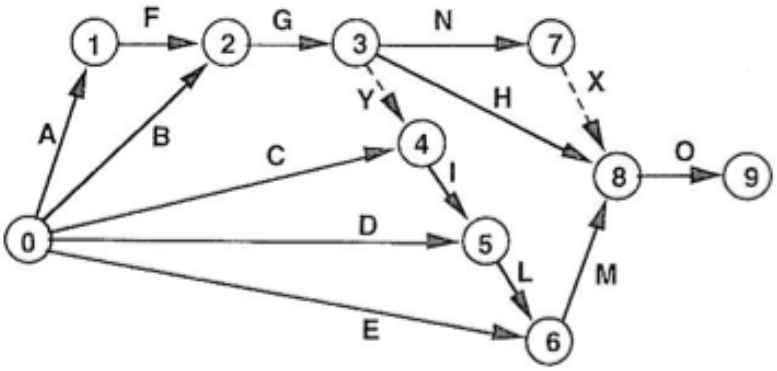
Attività	Attività precedenti	Durata
A	-	4
B	-	1
C	-	40
D	-	10
E	-	30
F	A	2
G	B,F	8
H	G	2
I	C,G	3
L	D,I	3
M	E,L	1
N	G	11
O	H,M,N	1

Supponendo di voler tracciare il reticolo di questo progetto utilizzando la rappresentazione con attività sugli archi (AOA), potrebbero sorgere alcune difficoltà tra cui:

- le attività che non sono precedute da nessuna altra attività vengono comunemente chiamate attività iniziali ed a tutte queste gli viene attribuito un unico evento chiamato di *inizio progetto*. Allo stesso modo tutte le attività che non hanno attività successive gli viene associato un evento chiamato di *fine progetto*;

- due diversi nodi non possono essere collegati da più di un ramo; per risolvere questa problematica che annullerebbe la corrispondenza biunivoca fra archi e coppie ordinate di nodi e non costituirebbe un grafo, viene inserita un attività *fittizia (dummy activity)* che presenta durata nulla;

- l'introduzione di una attività fittizia può anche essere dovuta per poter rappresentare correttamente tutte le relazioni di precedenza



La figura riporta il grafico aciclico o reticolo del progetto riguardante la sostituzione della caldaia, indicando con il nodo 0 l'evento di inizio e con il nodo 9 l'evento di fine del progetto.

VINCOLI DI PRECEDENZA

- *Fine-Inizio*
  La relazione Fine-Inizio è la più comune delle relazioni; nell'esempio seguente l'attività 2 non può iniziare fino a che l'attività 1 è finita.

- *Inizio-Inizio*
  La relazione Inizio-Inizio rappresenta la relazione di successione fra gli inizi delle attività; nell'esempio seguente l'attività 2 può iniziare solo dopo che l'attività 1 è iniziata. Questo tipo di relazione viene impostato per ridurre la durata globale del processo che si vuole programmare in quanto consente la sovrapposizione (o parallelizzazione) delle attività.

- *Fine-Fine*
  La relazione Fine-Fine rappresenta la relazione di successione fra la fine delle attività; nell'esempio seguente l'attività 2 può terminare solo dopo che l'attività 1 è terminata. Anche questo tipo di relazione viene impostato per ridurre la durata globale del processo che si vuole programmare in quanto consente la sovrapposizione (o parallelizzazione) delle attività.

- *Inizio-Fine*

  La relazione Inizio-Fine rappresenta la relazione di successione fra l'inizio della prima attività e la fine della seconda; nell'esempio seguente l'attività 2 può terminare solo dopo un determinato tempo che l'attività 1 è iniziata. La relazione deve pertanto essere specificata imponendo un ritardo fra l'inizio di una attività e la fine della successiva, in questa relazione la fine della prima attività e l'inizio della seconda attività non sono vincolati, fatte salve le relazioni causali o logico-tecniche fra le attività.

2.2. TECNICHE RETICOLARI

Le tecniche reticolari più note ed impiegate per la soluzione del problema di pianificazione e controllo sono le seguenti:

- C.P.M.
    - C.P.M. – RISORSE
    - C.P.M. – COSTI
    - C.P.M. – BUDGET
- P.E.R.T.
- M.P.M.
- G.E.R.T.

2.2.1. C.P.M. – CRITICAL PATH METHOD

Il **CRITICAL PATH METHOD** (CPM) ovvero **Metodo del Percorso Critico** è un algoritmo matematico per la pianificazione di una serie di attività di un progetto. In pratica questa tecnica analizza tutta la struttura del progetto per determinare il percorso più lungo composto dalla successione delle attività presenti all'interno del progetto (Newbold, 1998). Concentrandosi sulle attività più critiche si può garantire che il progetto tenga il passo con la pianificazione definita.

Il **Metodo del Percorso Critico** è costituito da tre fasi: Pianificazione - Analisi e Programmazione - Controllo.

Storia

Il (CPM) è una tecnica di modellazione dei progetti sviluppata alla fine degli anni 50 da parte di Morgan R. Walker dell'azienda DuPont e James E. Kelley, Jr. della Remington Rand.

I concetti di quello che è venuto a conoscersi come *critical path* sono stati sviluppati e messi in pratica da DuPont tra il 1940 e il 1943 e hanno contribuito al successo del Progetto Manhattan.

Tecnica di base

L’obiettivo del CPM è la minimizzazione del tempo di completamento dell’intero progetto. La tecnica fondamentale per l'utilizzo del CPM è quello di costruire un modello del progetto che include quanto segue:

- Un elenco di tutte le attività necessarie per completare il progetto
- Il tempo o durata di ciascuna attività
- Le dipendenze tra le attività con vincoli di precendenza di tipo *Finish To Start:* se l’attività i precede l’attività j, ciò significata che l’attività j non può iniziare finché i non é completata
- I *time lag* tra le attività sono tutti pari a zero
- Nessun limite nella disponibilità delle risorse

Con l’utilizzo di questi valori il CPM calcola il percorso più lungo di attività previste per la fine del progetto, e la data “al più presto” e “al più tardi” in cui ogni attività può iniziare e finire senza aumentare la data di conclusione del progetto.

Nella gestione dei progetti, un percorso critico è la sequenza delle attività concatenate che si sommano dando luogo alla più lunga durata complessiva. Questo percorso determina quali sono le attività "critiche" (sul percorso più lungo) che producendo un ritardo investirebbero direttamente la prevista data di completamento del progetto e quelle invece che permettono un possibile slittamento ("total float"), che possono essere cioè ritardate senza aumentare la data di fine del progetto. Un progetto può avere diversi percorsi critici. Un percorso parallelo aggiuntivo attraverso la rete con una durata totale inferiore al percorso critico è chiamato percorso sub-critico o non critico.

Sviluppo

Il metodo CPM, che in origine considerava solamente le dipendenze logiche tra gli elementi terminali, è stato ampliato per consentire l'inserimento di risorse relative a ciascuna attività, attraverso processi che prendono il nome di assegnazioni di risorse per attività e livellamento. Un calendario delle risorse livellato può visualizzare i ritardi dovuti alla carenza di risorse e può causare un percorso subito più breve per diventare poi il più lungo o richiedere più "risorse critiche". Un concetto correlato è chiamato “catena critica”, che cerca di tutelare l'attività e la durata del progetto dai ritardi imprevisti a causa di risorse limitate.

Vantaggi

Di seguito un elenco di vantaggi che nella letteratura sono stati individuati per quanto concerne il metodo CPM:

• il CPM incoraggia i manager ed i membri del progetto a disegnare graficamente ed identificare le varie attività che devono essere eseguite per il completamento del progetto. Questo passaggio incoraggia tutti i membri del team di progetto ad identificare i requisiti del progetto in modo critico e logico. Attività che precedono e seguono altre attività richiedono molto spesso una propria valutazione e analisi. Questo fattore diventa molto importante se le attività sono svolte in luoghi e tempi diversi e l'elemento di costo è sottoposto a variabili esterne che possono avere un impatto decisivo sul tempo totale di progetto.

• Individuazione del percorso critico per il progetto è la fase successiva all'analisi del diagramma di rete. In questo modo il management del progetto ha una ragionevole stima dei potenziali problemi che potrebbero verificarsi, ed in molti casi, il percorso critico determina anche l'allocazione delle risorse. L'interpretazione del diagramma di rete assicura anche che la stessa risorsa non sia allocabile durante determinati periodi di tempo.

• il metodo CPM favorisce anche un approccio disciplinato e logico di pianificazione, programmazione e gestione di un progetto con un tempo di svolgimento lungo. Spesso, la causa principale dello slittamento della data pianificata di completamento del progetto è l'incapacità di identificare i fattori che sono potenzialmente critici durante il progetto. Forzando i project manager ad identificare le attività, si migliora l'attenzione ai dettagli, ed a sua volta, questo porta ad una rappresentazione veritiera e molto più accurata dei processi, dei tempi e costi che devono essere considerati nel progetto.

• L'analisi dei punti di forza, punti di debolezza, opportunità, e minacce è un compito importante da svolgere all'interno della propria organizzazione. Effettuare un'analisi di questo tipo rivelerà i cambiamenti che possono essere realizzati, oltre a quelle modifiche che sembrano essere le soluzioni ottimali inizialmente ma non sarebbe altrettanto efficaci nel lungo periodo. Devono essere valutate anche le aree di miglioramento, i problemi precedentemente affrontati, e le scelte sbagliate che potevano essere evitate.

• Utilizzando il CPM è possibile l'ottimizzazione del rapporto tempo-costi nella gestione di progetti, ed i manager possono identificare visivamente le attività che rappresentano un problema se non gestiti e monitorati per un periodo di tempo in modo efficace. In molte situazioni la strutturazione dei costi nelle organizzazioni si basa ancora su regole funzionali, ed il compito di individuare in modo accurato il costo del progetto non è facile e non è universale per tutti i progetti o tutte le società. Lo sviluppo di relazioni tempo-costo dei progetti richiede che i project manager siano in grado di identificare la causa principale che stanno avendo un impatto negativo su tempi e costi.

• Sulla base delle variabili tempo-costo, il progetto può essere modificato per soddisfare al meglio le finalità e gli obiettivi dell'organizzazione. Ad esempio, se il team del progetto è in grado di identificare che serve più tempo per restare all'interno di un certo budget o questo fatto può essere chiaro fin dall'inizio del progetto. Mentre è presuntuoso affermare che ogni fattore che influenza le attività può essere identificato nelle fasi iniziali, una gran parte delle variabili e dei rischi e incertezze associate possono essere comunque valutati prima dell'inizio del progetto.

• Utilizzando il metodo CPM i manager possono identificare le aree in cui focalizzare l'attenzione. I percorsi critici non rimangono statici per la durata del progetto, ma piuttosto vi è una buona probabilità che potrebbero cambiare a causa di fattori interni ed esterni che influenzano l'organizzazione, questioni sindacali e l'insufficienza improvvisa di attrezzature potrebbero fanno parte proprio di questi fattori.

• La programmazione delle attività con il metodo CPM permette di identificarne l'intero percorso di svolgimento. Spesso, durante le fasi iniziali del progetto, il numero di attività e dei requisiti di costo potrebbe essere alto, ma nell'avanzare del progetto questo numero si riduce potendo raggruppare in fasi gruppi di attività. I responsabili di progetto, invece di concentrarsi sull'intero progetto, possono focalizzare la loro attenzione su questi gruppi di attività che sono in corso di svolgimento e hanno dunque la capacità di influenzare i successivi step di progetto.

• Il CPM individua i tempi di float nel progetto, cioè di quelle attività che possono essere ritardate senza aumentare il tempo di fine del progetto stesso, così i project manager possono identificare quando le risorse possono essere riassegnate ad attività diverse e come effettuare lo spostamento di queste attività per ottimizzare al meglio l'utilizzo delle risorse.

• In molti progetti di grandi dimensioni, possono esserci più di un percorso critico nel diagramma di rete. Quando una tale situazione si verifica, il metodo CPM può aiutare i project manager ad identificare un idoneo piano di azioni per gestire questi molteplici percorsi critici.

• il metodo CPM è stato ampiamente utilizzato con grande successo in una varietà di organizzazioni ed in quasi tutti i settori. Il metodo CMP può aiutare a stimare la durata del progetto e queste informazioni possono essere utilizzate per ridurre al minimo la somma dei costi diretti e indiretti coinvolti nella pianificazione e programmazione del progetto.

Il metodo CPM offre all'organizzazione una documentazione strutturata che si può riutilizzare per progetti simili da intraprendere in futuro. Documentare le varie attività e le cause che hanno generato i problemi può aiutare il futuro project manager ad evitare trappole simili. Inoltre, la documentazione può fornire dati importanti per la stima dei requisiti di tempo e fattori di costo, al contrario di dati basati su stime e supposizioni.

"L’analisi del percorso critico individua formalmente le attività che devono essere completate in tempo per rispettare la data di completamento dell'intero progetto, e identifica anche quali attività possono essere ritardate per un po' se è necessario riallocare una risorsa per recuperare il ritardo su altre attività" (MindTools , 2004). Il CPM è in grado di identificare i percorsi che possono essere adottati per accelerare un progetto completandolo prima della sua data di scadenza o di identificare il più breve tempo possibile, o il minor costo possibile che è necessario per completare un'attività.

Il Metodo CPM è basato su modelli deterministici e la stima della durata delle attività si basa su dati storici mantenuti all'interno dell'organizzazione o di dati provenienti da fonti esterne.

Svantaggi

I principali svantaggi del metodo del percorso critico sono elencati di seguito. Molti svantaggi sono il risultato di fattori tecnici e concettuali presenti nell’Analisi del Percorso Critico - CPA.

• Il processo di CPA può diventare complicato se l'ambito e la portata del progetto aumentano. Troppe attività interconnesse possono rendere il diagramma di rete molto complicato. Il rischio di fare un errore nel calcolo della catena critica diventa molto alto al crescere del numero di attività.

• Il CPA dipende dal concetto fondamentale che i dirigenti e il personale coinvolti nel team di progetto siano a conoscenza della natura delle varie attività. “Purtroppo l'esperienza pratica ha dimostrato che l'ipotesi principale alla base di tecniche di CPM, cioè la capacità del team di progetto di prevedere ragionevolmente il campo di applicazione, il calendario, e il costo di ogni progetto, è spesso sopravvalutata”. (Knoke e Garza, 2003)

• Diventa molto più complicato comprendere le esigenze del percorso critico quando c'è più di un percorso critico nel progetto. In molte situazioni, questi percorsi potrebbero essere paralleli ed alimentare un nodo comune nel diagramma di rete. Diventa difficile in queste situazioni identificare la migliore utilizzazione delle tecnologie e delle risorse all’interno dei percorsi critici.

• In molti casi, con l’avanzare del progetto, i percorsi critici potrebbero cambiare ed evolversi, rendendo quelli del passato a non essere più validi. Questo implica che il project manager del progetto deve rivedere costantemente il diagramma di rete inizialmente creato e identificare le variazioni del percorso critico nel tempo.

• Come la programmazione dei percorsi critici cambia durante l'avanzamento del progetto anche la disponibilità del personale può cambiare. La riallocazione del personale è spesso molto difficile in quanto la singola risorsa potrebbe lavorare su più di un progetto alla volta e se le attività appartiene a più di un percorso critico l'identificazione e la distribuzione del tempo di lavoro può causare un sovraccarico del personale stesso.

• Molto spesso, i percorsi critici non sono facili da individuare, soprattutto se il progetto è unico e non è mai stata intrapreso dall'organizzazione in passato. La capacità di fornire stime di tempo e costo per ogni attività in un processo CPM tradizione dipende da dati storici mantenuti dalla società. In assenza di questi dati, i project managers sono costretti a stimare il tempo ed i costi richiesti per questi tipi di progetto.

• Tradizionalmente, ogni CPA richiede la risoluzione degli algoritmi "fase in avanti" e "fase all'indietro" per determinare i ritardi ed il total float di ciascuna attività. Tuttavia, i project manager spesso suddividono il progetto in parti durante la fase di pianificazione per determinare il costo massimo che sarebbe necessario per completare un progetto e durante la stima, usano un valore di costo intermedio per il progetto. Questa mentalità può costare la sovrastima spesso di tempo e di costo ed incoraggia le risorse a posticipare la data di inizio di qualsiasi attività sul diagramma di rete. Di conseguenza eventuali gravi scostamenti provocano lo slittamento della data di completamento del progetto aumentando così il costo totale del progetto stesso.

• Il CPA ed i diagrammi di rete sono altamente dipendenti delle tecnologie software. Il costo della realizzazione di questi sistemi software può avere una elevato costo iniziale; la manutenzione ed il monitoraggio del software richiede competenze che possono rapidamente diventare molto costose se l'organizzazione non dispone di figure professionali al proprio interno.

• Le aziende stanno diventando sempre più globali. L'Instabilità sociale ed economica in una regione del mondo può influire seriamente sulla produzione in un altro. Se le organizzazioni dipendono da attività che sono localizzate in tutto il mondo il compito di coordinare la pianificazione e la programmazione delle attività in varie location si complica ulteriormente.

• Al fine di migliorare i profitti, è necessario per le aziende razionalizzare le loro operazioni per mantenere la loro posizione in un mercato in continua evoluzione. Per fare questo, le aziende sono costrette a migliorare i loro processi di produzione e ridurre il costo delle operazioni. I manager ad ogni livello sono costretti a valutare i loro processi lungo tutta la catena dai fornitori all'utente finale e parte delle analisi viene estesa anche alla catena di fornitura della società e dei singoli fornitori. Le aziende si stanno spostando da una moltitudine di fornitori ai pochi più attendibili e affidabili nel tentativo di monitorare la qualità e contenere i costi.

• Anche se il metodo CPM offre molti dettagli può non essere adatto se si modifica il sistema costantemente, soprattutto se si tratta di riallocazione delle risorse e di tempo.

• Nonostante l'uso diffuso del metodo CPM, l'utilizzo può differire in modo significativo nelle varie organizzazioni. Società che hanno una forte cultura al completamento delle attività nel rispetto delle tempistiche avranno un approccio alla metodologia in modo più appropriato rispetto alle aziende che utilizzano il metodo CPM solo in parte per la pianificazione e la programmazione delle attività.

• La gestione della conoscenza aziendale è importante. Definire la conoscenza non è mai facile, in quanto conoscenze e informazioni sono differenti anche se spesso si presume essere la stessa cosa. I dati sono i dati grezzi raccolti con l'osservazione o il monitoraggio. Quando i dati sono filtrati per identificare particolari tendenze si convertono in informazioni e quando queste informazioni vengono utilizzate per il funzionamento, la pianificazione e la strategia si sono convertiti in conoscenza. (Yahya e Goh, 2002) Informazioni e conoscenze vengono trasmesse all'interno di un'organizzazione attraverso i canali di comunicazione. Il metodo CPM dipende proprio dall'efficienza di queste reti

Il C.P.M. Risorse

Il C.P.M. Risorse tiene conto di tutti i vincoli derivanti dalla disponibilità limitata delle risorse da impiegare (ad esempio macchinari, personale, ecc…)

Il C.P.M. Costi

Il C.P.M. Costi è caratterizzato da attività la cui durata non è determinata a priori, ma può essere modificata con variazione conseguente dei costi diretti (l'allungamento della durata fa diminuire il costo diretto secondo una legge data); l'obiettivo consiste nella minimizzazione del costo totale, che include i costi diretti di tutte le attività e le spese generali relative al progetto. Nei casi in cui il progetto deve essere completato entro una data prefissata ciò può essere imposto come ulteriore vincolo al problema.

Il C.P.M. Budget

Il C.P.M. Budget determina gli instanti di inizio e fine di ogni attività con l'obiettivo di minimizzare il tempo di completamento dell'intero progetto, ma fornisce in più l'ammontare delle spesa totale prevista in corrispondenza ad ogni instante ed ad ogni stato di avanzamento del progetto.

2.2.2. P.E.R.T

PERT, acronimo dalla lingua inglese che sta per **Program Evaluation** and **Review Technique**, è una tecnica (formalismo grafico) di project management sviluppata nel 1958 dalla Booz, Allen & Hamilton, Inc. una ditta di consulenza ingegneristica, per l'ufficio Progetti Speciali della Marina degli Stati Uniti. L'obiettivo era quello di ridurre i tempi ed i costi per la progettazione e la costruzione dei sottomarini nucleari armati con i missili Polaris, coordinando nel contempo diverse migliaia di fornitori e di subappaltatori in aree geografiche anche completamente diverse tra loro.

Con questa tecnica si tengono sotto controllo le attività di un progetto utilizzando una rappresentazione reticolare che tiene conto dell'interdipendenza tra tutte le attività necessarie al completamento del progetto.

Si noti che l'algoritmo PERT non schedula (cioè non elabora una sequenza temporizzata delle attività), perché non tiene conto della disponibilità delle risorse; considera cioè che le risorse siano a disponibilità infinita.

Varianti del PERT:

- L'algoritmo Semplice PERT-Tempi calcola i tempi minimi e massimi per la realizzazione di ogni attività
- L'algoritmo Full PERT-Tempi è lo stesso del PERT Semplice, ma considera la durata delle attività in forma probabilistica
- PERT Costi e CPM consentono di effettuare analisi considerando anche i costi associati alle attività.

2.2.3. M.P.M.

Il Metra Potential Method ed il Precedence Diagram sono caratterizzati da vincoli di tempo fra le attività che sono complessi rispetto al C.P.M.: in particolare fra 2 attività i e j le precedenze possono essere del tipo:

a) Finish to Start (F.S.) quando è dato un confine inferiore (che può essere uguale a zero) e un confine superiore (che può essere uguale ad infinito) al tempo che deve intercorrere fra la fine di i e l'inizio di j;
b) Start to start (S.S.) quando sono dati il confine superiore ed inferiore al tempo intercorrente fra l'inizio di i e la fine di j;
c) Finish to Finish (F.F.) quando sono imposti dei confini al tempo che intercorre tra la fine di i e la fine di j;
d) Start to Finish (S.F.) quando sono imposti dei confini al tempo fra l'inizio di i e la fine di j.

2.2.4. G.E.R.T

Il G.E.R.T – **Graphical Evaluation** and **Review Technique** è caratterizzato da vincoli di precedenza più flessibili rispetto al C.P.M., in particolare mentre nel C.P.M. ogni vincolo pone in relazione una attività "precedente" con una "preceduta", nel G.E.R.T ogni vincolo pone in relazione due sottoinsiemi di attività. Si può distinguere tra vincoli di tipo AND o OR e tra vincoli di natura deterministica oppure stocastica. Quando il vincolo di precedenza è di tipo AND, allora le attività "precedute" possono iniziare solo dopo che tutte le attività "precedenti" sono state completate. Quando il vincolo è di tipo OR invece, tutte le attività "precedute" possono iniziare non appena una delle attività precedenti sia stata completata.

Quando il vincolo è di natura stocastica, solo una fra le attività "precedute" dovrà essere eseguita e ciò in accordo con probabilità assegnate. Se invece in vincolo è di natura deterministica, allora le attività "precedute" devono essere tutte eseguite.

Infine nel G.E.R.T. vi è la possibilità che alcune parti del progetto siano svolte ripetutamente.

2.2.5. DIAGRAMMA DI GANTT

Il diagramma di Gantt, chiamato in ricordo dell'ingegnere statunitense che lo inventò nel 1917 **Henry Laurence Gantt** (1861 - 1919), è costruito utilizzando un asse orizzontale che rappresenta l'arco temporale totale del progetto, suddiviso in unità temporali (ad esempio

giorni, settimane, mesi) e da un asse verticale che rappresenta l'insieme delle attività che costituiscono il progetto.

Nel diagramma vengono presentate l'insieme di tutte le attività che costituiscono lo scheletro del progetto. Per indicare ogni singola attività, le varie sequenze, la durata e l'arco temporale vengono utilizzate barre orizzontali di lunghezza variabile. Nel grafico per indicare la possibilità dello svolgimento in parallelo di alcune di queste attività le barre possono sovrapporsi durante lo stesso arco temporale.

Durante il proseguimento del progetto, delle barre secondarie colorate vengono aggiunte al diagramma per indicare il completamento, o una porzione di esso, alle attività sottostanti. Sono presenti anche linee verticali utilizzate per indicare le varie date di riferimento, quali le date di inizio e fine di ciascuna attività, o l'avanzamento del progetto rispetto alla data attuale.

Il diagramma di Gantt permette una rappresentazione grafica intuitiva di un calendario di attività, utile al fine di pianificare, coordinare e monitorare specifiche attività in un progetto dando una chiara illustrazione dell'avanzamento del progetto rappresentato.

Ad ogni attività possono essere associate una o più risorse. Contestualmente, può essere definito il calendario dei giorni lavorativi e festivi, e il numero di ore di lavoro giornaliere per permettere di calcolare il workload di ogni risorsa e la sua saturazione, impostando una certa disponibilità per ogni risorsa.

Ad ogni attività può poi essere associato un costo. Il costo può essere attribuito a una singola attività oppure si può assegnare un costo orario alle risorse, determinando il costo dell'attività in base al relativo impegno orario. Dai dati di costo si ricavano tre curve e due indicatori di avanzamento dell'intero progetto. Le tre curve riportano la cumulata del costo preventivato e/o effettivo in funzione del tempo, ossia i costi totali effettivi e/o preventivati dall'inizio:

- BCWS: Budget Cost of Work Scheduled: riporta i costi preventivati da 0 a vita intera
- ACWP: Actual Cost of Work Performed: riporta i costi sostenuti da 0 a tempo attuale
- BCWP. Budget Cost of Work Performed: riporta i costi preventivati da 0 a tempo attuale

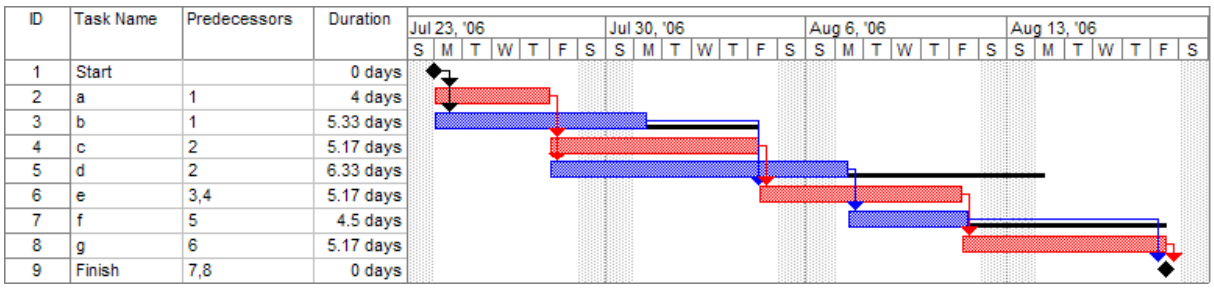


Figura. Esempio di diagramma di Gantt

CAPITOLO 3 – PROJECT SCHEDULING

3.1. PROJECT SCHEDULING SU UN RETICOLO CON ATTIVITA' SUGLI ARCHI

Il problema, nella fase di pianificazione delle attività e risorse di un progetto, consiste nel determinare l'instante d'inizio di ciascuna attività in modo che la durata complessiva del progetto sia minima. Questo processo, chiamato *project scheduling,* è presente in molti scenari in cui un numero di risorse generalmente scarse devono essere allocate nel tempo ad attività interdipendenti.

3.1.1 CLASSIFICAZIONE DEI PROBLEMI DI PROJECT SCHEDULING

La classificazione può venir fatta secondo vari parametri tra cui i principali sono:

- Ambiente risorse
- Caratteristiche attività
- Obiettivo da perseguire

Ambiente Risorse

Si caratterizza sulla base della tipologia delle risorse, della quantità e del numero di risorse, della presenza di una o più alternative per il processamento delle attività. Le risorse si possono distinguere nelle seguenti tipologie: risorse rinnovabili, risorse non rinnovabili, risorse doppiamente vincolate, risorse parzialmente rinnovabili e risorse dedicate.

Le risorse rinnovabili sono caratterizzate da una certa quantità sempre disponibile in ogni istante di tempo nell'orizzonte temporale del progetto. Si tratta di risorse per quali ci sono vincoli solo sull'uso complessivo ad ogni istante della durata del progetto: esempi sono la manodopera, i macchinari, lo spazio, la strumentazione.

Le risorse non rinnovabili sono caratterizzate da una certa disponibilità complessiva per tutto l'orizzonte temporale del progetto e si consumano via via che sono utilizzate per le varie attività.

Risorse doppiamente vincolate hanno entrambe le caratterizzazioni ossia sono sempre simultaneamente vincolate sia in termini di uso complessivo a ogni istante che in termini di consumo totale. A questa tipologia può appartenere ad esempio il cash-flow.

Risorse parzialmente rinnovabili sono limitate solo ad un sottoinsieme di periodi dell'orizzonte temporale del progetto. Per le risorse di questo tipo c'è una disponibilità fissata per tutta la durata di ogni periodo del sottoinsieme.

Le risorse dedicate sono risorse che possono essere assegnate ad una sola attività. Naturalmente queste risorse possono essere considerate come risorse rinnovabili la qui disponibilità è di una unità per periodo.

La quantità di risorse disponibili determina principalmente due importanti classi di problemi: il project scheduling con risorse illimitate, è quello con risorse limitate.

Caratteristiche attività

È possibile caratterizzare le attività sulla base: del tipo di relazione tra le attività, della natura della durata delle attività, del tipo di relazioni di precedenza, della presenza di vincoli temporali tra le attività, della presenza di vincoli temporali sulle attività, della presenza di deadline per il completamento del progetto. La durata delle attività può essere deterministica o stocastica e possono esistere differenti vincoli di precedenza. I vincoli temporali tra le attività sono del tipo minimo o massimo ritardo tra i tempi di inizio o di fine delle attività. Queste relazioni sono di solito associate ad un vincolo di precedenza tra le attività i e j. Un vincolo temporale sull'attività tipicamente è un vincolo che impone l'esecuzione, l'inizio o la fine di un'attività nel rispetto di una finestra temporale ad essa assegnata. La presenza di deadline nell'eseguire il progetto impone che l'ultima attività sia terminata entro un tempo prefissato.

Obiettivo

La funzione obiettivo può avere varie forme tra cui: la minimizzazione del tempo di completamento, la minimizzazione del valore attuale, il livellamento dell'uso delle risorse, la minimizzazione dei costi totali, la minimizzazione del massimo ritardo sulla fine delle attività rispetto al termine del progetto stesso, la minimizzazione del flow-time, cioè il tempo complessivo di permanenza nel sistema delle attività in esecuzione.

In letteratura ormai è consolidata l'esistenza di 2 classi fondamentali di problemi di Projet scheduling: con risorse illimitate e con risorse limitate. Lo scenario a risorse illimitate non corrisponde mai a realtà e viene utilizzato solo come ipotesi per ottenere una valutazione sul calcolo della durata delle attività che sia computazionalmente più semplice. Tuttavia con

l'introduzione dei vincoli sulle risorse, nel caso appunto di risorse limitate, il problema passa da una complessità polinomiale ad un problema *NP-hard*.

Il problema principale che si deve risolvere in presenza di risorse limitate riguarda la questione dell'incompatibilità che può verificarsi nell'attimo in cui due o più attività richiedono una stessa risorsa causando un problema circa la sua disponibilità. Questo problema porta ad effettuare un scelta decisionale in quanto bisogna scegliere quale delle attività processare per prima.

Le osservazioni fin qui considerate spiegano come si generi la complessità nei problemi ad insufficienza di risorse, e se pensiamo a k incompatibilità tra coppie di attività che sono in conflitto per la medesima risorsa, bisogna nel caso peggiore risolvere 2^k problemi.

Standard Single-mode RCPSP – Resource-Constrained Project Scheduling Problem

Nel caso del project scheduling con risorse limitate un generico problema è usualmente identificato la sigla "Resource-Constrained Project Scheduling Problem". Nella realtà i problemi che appartengono a questa classe si possono suddividere in 4 insiemi ed andremo a considerare il primo di questa suddivisione: lo "Standard Single-mode".

Nei problemi Single-mode il processamento di una attività avviene in unico modo ovvero con un insieme di risorse e una durata definita a priori. Assumiamo che il progetto sia rappresentato da una rete di attività, che la suddetta rete sia rappresentata da un grafo i cui nodi siano in corrispondenza biunivoca con le attività. Lo **Standard Single-mode RCPSP** è caratterizzato dalla seguente classificazione:

- funzione obiettivo del tipo minimo tempo di completamento
- tutte le relazioni di precedenza che sono presenti all'interno delle attività sono del tipo finish to start
- la disponibilità delle risorse all'interno del progetto è costante
- la richiesta di risorse per periodo è costante
- per questo tipo di problemi non c'è "preemption", cioè non è presente la possibilità di interrompere il processamento di un'attività per poi riprenderla successivamente

3.2. PROJECT SCHEDULING CON RISORSE ILLIMITATE

INTRODUZIONE

Per gestire la pianificazione di un insieme di attività supponendo di avere a disposizione risorse illimitate si utilizzano due tecniche: il CPM ed il PERT. Ci concentreremo solo sul metodo CPM utilizzando la rappresentazione reticolare AOA delle relazioni di precedenza assumendo che tali relazioni siano del tipo finish to start e che i time-lag tra le attività siano tutti uguali a zero.

ANALISI CAMMINO CRITICO RETI AOA

Per poter rispondere alle classiche domande: qual è il tempo minimo di completamento dell'intero progetto o quanto una specifica attività può essere ritardata senza causare ritardi sull'intera durata del progetto, bisogna calcolare il *"cammino critico"* avendo già definito a priori l'insieme delle attività e la loro durata, i vincoli di precedenza e quindi la loro rappresentazione tramite un reticolo con attività sugli archi.

La ricerca del cammino critico su reti AOA parte dal presupposto che in questo tipo di rappresentazione un nodo evento non può essere raggiunto se prima non si siano stati raggiunti gli eventi e siano state eseguite le attività (archi) che lo precedono.

Algoritmo di numerazione

Step 1: Numera l'evento sorgente della rete con il numero 0.

Step 2: Assegna il numero successivo ad un qualunque evento non numerato a cui i predecessori siano tutti numerati

Step 3: Ripeti lo step2 finchè tutti gli eventi non sono stati numerati

Algoritmo Fase in Avanti

Step 1: i:= 0 E(0) = 0.

Step 2: i = i+1; E(i) = massimo {E(i) + Dij} (dove il massimo è calcolato su tutte le attività comprese tra (i,j) che hanno j come nodo finale).

Algoritmo Fase all'indietro

Step 1: assegna ad L(n) il valore del tempo minimo di completamento del progetto Tf

Step 2: i = i-1; L(i) = minimo {L(j) - Dij} (dove il minimo è calcolato su tutte le attività comprese tra (i,j) che hanno i come nodo iniziale).

Utilizzando questi algoritmi possiamo calcolare i seguenti valori per ciascuna attività:

$$ES(i,j) = E(i)$$

$$EF(i,j) = E(i) + D_{ij}$$

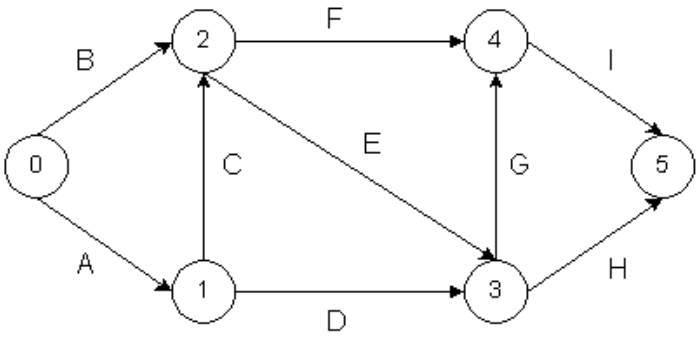
$$LF(i,j) = L(j)$$

$$LS(i,j) = L(j) - D_{ij}$$

3.3. PROJECT SCHEDULING CON RISORSE LIMITATE

Nel capitolo precedente abbiamo presentato la tecnica CPM per lo scheduling di progetti dove le risorse potevano essere considerate illimitate.

Prendiamo ora in esame il progetto rappresentato dal reticolo di figura. Le attività presenti saranno caratterizzate oltre che dalla loro durata anche da una richiesta di risorse, condivise tra più attività aggiungendo un ulteriore vincolo al problema, ed in particolare non rendendo più ammissibile che tutte le attività inizino alla rispettiva data minima.



Con l'aggiunta del nuovo vincolo legato alla disponibilità delle risorse, il problema come dimostrato nel libro di Elmaghraby è *NP-hard*, e quindi per la sua risoluzione si adottano tecniche euristiche. Il seguente problema è risolvibile impiegando la tecnica **C.P.M. Risorse**.

Da un punto di vista matematico questo problema può essere espresso in forma di programmazione lineare nel rispetto dei vincoli definiti, ma dato che il problema si riduce al calcolo del cammino massimo su un reticolo aciclico con durata delle attività non negative, è più conveniente, al posto dell'algoritmo del simplesso, ricorrere a specifici algoritmi che implementano tecniche euristiche.

Queste tecniche, tra cui le più diffuse sono la *parallela*, *la seriale* e *la strettamente seriale*, operano lo *scheduling* di tutte le attività, ovvero gli assegnano una data di inizio. Questa procedura chiamata anche *tempificazione* è composta da una serie di azioni che vengono compiute utilizzando il *decision set*, una lista dinamica di attività. Di seguito l'ordine delle operazioni che vengono compiute con il decision set:

- Inserimento delle attività nel decision set
- Ordinamento delle attività nel decision set
- Estrazione delle attività dal decision set e successiva schedulazione

Queste azioni che vengono compiute sul decision set, sono svolte in base a regole che dipendono dalla particolare tecnica euristica e da uno specifico criterio di priorità stabilito a priori in base alla natura del problema da risolvere.

Definizione del Problema

In questo contesto un progetto è costituito da un insieme di attività legate tra loro da vincoli di precedenza e vincoli sulle risorse. I dati deterministici noti per ogni attività sono: la durata, le risorse e le relazioni di precedenza, mentre per le risorse è nota la loro disponibilità.

Dato **V** = {1,…,j,…,n} l'insieme delle attività e **dj** il tempo di completamento di ogni attività. Per definire le relazioni di precedenza viene utilizzato un insieme **Pj** che rappresenta i *predecessori*, l'attività j non più iniziare prima che ogni attività i appartiene ad Pj sia svolta, e **Sj** che rappresenta i *successori,* cioè ogni attività i appartiene ad Sj non può cominciare se prima j non è stata completata. L'insieme delle risorse appartiene all'insieme **K** e per ogni risorsa k appartenente a K esiste una massima capacità di allocazione. Tutti i parametri sono non negativi e interi. Il problema consiste nello schedulare le attività rispettando i vincoli precedenti e minimizzando il tempo di completamento del progetto, il cosidetto ***makespan***.

Le tecniche euristiche costruttive partono da una schedula vuota ed aggiungono le attività ad ogni passo fino a raggiungere una schedula ammissibile, mentre quelle migliorative partono da una schedula ottenuta con la tecnica costruttiva e attraverso varie tecniche giungono ad una soluzione localmente ottima.

La tecnica parallela, assumendo che il tempo sia discreto, cioè che le durate delle attività sono intere, costruisce un diverso decision set per ogni istante di tempo. Come detto, sarà necessario stabilire un criterio di priorità, in base all'urgenza che si vuole attribuire a ciascuna attività. Supponiamo che tale criterio sia quello del cosiddetto **total float** parallelo basato sulle date massime di inizio calcolate con la tecnica C.P.M. a risorse illimitate.

Il Ritardo Totale (total float), è il massimo ritardo ammissibile che può avere il tempo di completamento di un'attività, pena il ritardo nel completamento dell'intero progetto, nell'ipotesi che le attività precedenti terminino alla loro data minima di fine e le attività successive possano iniziare alla loro data massima di inizio. Le attività a ritardo totale nullo sono critiche e ogni cammino dall'attività iniziale a quella terminale, che attraversa solo attività critiche è un cammino critico.

Ritornando alla tecnica parallela quindi per ogni attività k, al generico istante di tempo t, il total float è dato dalla seguente relazione:

$$TFt(k) = LS(k) - t$$

In tal modo si pongono all'inizio della lista le attività che potrebbero ritardare maggiormente il tempo di completamento del progetto. In caso di parità del total float fra più attività, l'ordine può avvenire in modo arbitrario o seguendo un secondo criterio di priorità diverso.

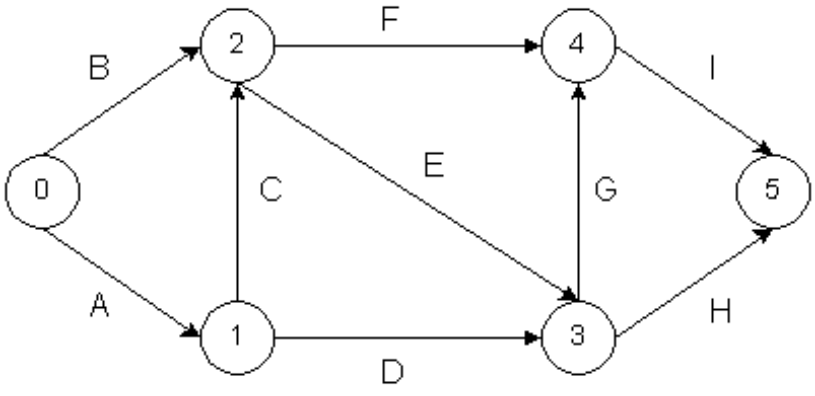
3.4. TECNICA PARALLELA PER IL PROBLEMA C.P.M. – RISORSE - Algoritmi ed implementazione

Per prima cosa dobbiamo calcolare, tramite il metodo CPM a risorse illimitate, i valori di ES, EF, LS, LF, e di conseguenza del Total Float per ciascuna delle attività che appartengono al progetto, rispettando i vincoli sulle precedenze tra le attività stesse.

Riprendiamo l'esempio del progetto composto dalle seguenti nove attività:

Attività	Predecessori	Durata (giorni)
A	-	4
B	-	3
C	A	2
D	A	7
E	B, C	9
F	B, C	12
G	D, E	2
H	D, E	5
I	F, G	6

Con la seguente rappresentazione del reticolo con attività sugli archi:



Tramite gli algoritmi per la fase in avanti e per la fase all'indietro presentati nella sezione CPM con risorse illimitate otteniamo i seguenti valori:

Step 1

E(0) = 0

Step 2

J = 1 - E(1) = Max{E(0) + D01} = 4

j = 2 - E(2) = Max{E(0) + D02; E(1) + D12} = 6

j = 3 - E(3) = Max{E(1) + D13; E(2) + D23} = 15

j = 4 - E(4) = Max{E(2) + D24; E(3) + D34} = 18

j = 5 - E(5) = Max{E(3) + D35; E(4) + D45} = 24

Il minimo tempo di completamento del progetto è 24 con appunto E(5) = 24

Step 1

L(5) = E(5) = 24

Step 2

j = 4 - L(4) = Min {L(5) - D45} = 18

j = 3 - L(3) = Min {L(5) - D35; L(4) - D34} = 16

j = 2 - L(2) = Min {L(4) - D24; L(3) - D23} = 6

j = 1 - L(1) = Min {L(3) - D13; L(2) - D12} = 4

j = 0 - L(0) = Min {L(2) - D02; L(1) - D01} = 0

Di seguito verrà inserito il codice del programma in linguaggio java che realizza gli algoritmi visti in precedenza, permettendo di calcolare il tempo di completamento del progetto con l'indicazione delle attività che appartengono al percorso critico.

```
package criticalpathmethod;


import java.util.*;
/**
 *
 * @author robertogarzon
 */
public class CriticalPathMethod {

   /**
    * @param args the command line arguments
    */
   public static int criticalPath = 0;
   public static String format = "%1$-10s %2$-10s %3$-18s %4$-5s %5$-5s %6$-5s %7$-9s %8$-14s
%9$-10s\n";

   public static void main(String[] args) {

      // ProjectNetwork rappresenta insiemi di oggetti Task
      HashSet<Task> ProjectNetwork = new HashSet<Task>();


      Task A = new Task("A", 0,1, 4,  5);
      Task B = new Task("B", 0,2, 3,  3);
      Task C = new Task("C", 1,2, 2,  7, A);
      Task D = new Task("D", 1,3, 7,  6, A);
      Task E = new Task("E", 2,3, 9,  9, B,C);
      Task F = new Task("F", 2,4, 12, 8, B,C);
      Task G = new Task("G", 3,4, 2,  5, D,E);
      Task H = new Task("H", 3,5, 5,  8, D,E);
      Task I = new Task("I", 4,5, 6,  9, F,G);


      ProjectNetwork.add(A);
      ProjectNetwork.add(B);
      ProjectNetwork.add(C);
      ProjectNetwork.add(D);
      ProjectNetwork.add(E);
      ProjectNetwork.add(F);
      ProjectNetwork.add(G);
      ProjectNetwork.add(H);
      ProjectNetwork.add(I);



      // calcolo utilizzando il metodo CPM il percorso critico
      //Task[] result = criticalPathMethod(ProjectNetwork);

      Task[] result = MycriticalPathMethod(ProjectNetwork);

      // stampo a video il risultato
      print(result);
```

```
    // stampa le attività che appartenfono al cammino critico
    printActivitiesOnCriticalPath(result);


    // calcolo la schedulazione delle attività con 2 Risorse Limitate
    calculateParallelResourceScheduling(ProjectNetwork);
}

public static class OrderByTFParallel implements Comparator<Task> {

    @Override
    public int compare(Task o1, Task o2) {
        return o1.TFParallel > o2.TFParallel ? 1 : (o1.TFParallel < o2.TFParallel ? -1 : 0);
    }
}

public static class Resource {

    // nome della risorsa
    public String name;

    // capacità massima della risorsa
    public int max_occupation;

    // capacità non allocata della risorsa
    public int cna;


    // occupazione della risorsa in questo momento t
    //public int occupation;

    // attività che sta svolgendo la risorsa
    public HashSet<Task> tasks = new HashSet<Task>();


    public Resource(String name, int max) {
        this.name = name;
        this.max_occupation = max;
        this.cna = max;
    }

}

public static class Task {

    // nome dell'attività
    public String name;

    // numero nodo ingresso attività
    public int predecessorNode;

    // numero nodo uscita attività
    public int successorNode;
```

```
    // durata dell'attività
    public int duration;

    // quantita di risorsa da utilizzare per eseguire questa attività
    public int resources_allocation;

    // utilizzo non allocato dell'attività
    public int una;

    // progressione dell'attività durante l'avanzamento del progetto
    public int progress;

    public int TFParallel;

    // the duration of the task along the critical path
    public int criticalTaskPath;


    // per la FASE in AVANTI
       // early start o date minime di inizio (d.m.i)
       public int earlyStart;

       // early finish o date minime di fine (d.m.f)
       public int earlyFinish;

    // per la FASE all'INDIETRO
       // late start o date massime di inizio (D.M.I)
       public int latestStart;

       // late finish o date massime di fine (D.M.F)
       public int latestFinish;

    // attività che dipendono da attività precedenti
    public HashSet<Task> dependencies = new HashSet<Task>();

    public Task(String name, int preNode, int sucNode, int duration, int resallocation,Task...
dependencies) {
       this.name = name;

       this.predecessorNode = preNode;
       this.successorNode = sucNode;

       this.duration = duration;

       this.resources_allocation = resallocation;

       this.progress = 0;


       for (Task t : dependencies) {
          this.dependencies.add(t);
       }
```

```
        this.earlyFinish = -1;
    }


    public void setTFParallel( int value ){
        TFParallel = value;
    }

    public void setProgress( int value ){
        progress = value;
    }

    public int getTFParallel(){
        return TFParallel;
    }

    public String[] toStringArray() {
        String criticalCond = earlyStart == latestStart ? "Sì" : "No";
        String[] toString = { name+" ("+predecessorNode+","+successorNode+")", duration+"",
resources_allocation+"", earlyStart + "", earlyFinish + "", latestStart + "", latestFinish + "",
            latestStart - earlyStart + "", criticalCond };
        return toString;
    }

    public boolean isDependent(Task t) {
        // is t a direct dependency?
        if (dependencies.contains(t)) {
            return true;
        }
        // is t an indirect dependency
        for (Task dep : dependencies) {
            if (dep.isDependent(t)) {
                return true;
            }
        }
        return false;
    }
}


public static Task[] MycriticalPathMethod(Set<Task> tasks) {

    HashSet<Task> completed = new HashSet<Task>(tasks);
    Task[] ret = completed.toArray(new Task[0]);
    int maxNumberNodes = -1;

    // calcolo il numero del nodo di fine progetto
    for(Task attivita : tasks){

        int max = Math.max(attivita.predecessorNode, attivita.successorNode);
        if (max > maxNumberNodes) maxNumberNodes = max;


    }
```

```
// Algoritmo fase in avanti per il calcolo dei tempi minimi
int[] E = new int[maxNumberNodes+1];
E[0] = 0;
System.out.println("E(0) = 0");


for (int i=1; i<= maxNumberNodes; i++) {

   System.out.print("E("+i+") ");
   HashSet<Task> TaskNodes = TaskFromSuccessorNode(tasks, i);

   int max = -1;

   for(Task singolaAttivita : TaskNodes){

      int temp = singolaAttivita.duration + E[singolaAttivita.predecessorNode];
      if (temp > max) max = temp;

   }

   E[i] = max;
   System.out.println("= "+E[i]+"");


}

// tempo minimo di completamento dell'intero progetto
int Tf = E[maxNumberNodes];

System.out.println("\nTempo minimo di completamento dell'intero progetto = "+ Tf +" \n");

// Algoritmo fase all'indietro per il calcolo dei tempi massimi
int[] L = new int[maxNumberNodes+1];
L[maxNumberNodes] = Tf;
System.out.println("L("+maxNumberNodes+") = "+Tf);

for (int i=maxNumberNodes-1; i >= 0; i--) {

   System.out.print("L("+i+") ");
   HashSet<Task> TaskNodes = TaskFromPredecessorNode(tasks, i);

   int min = 99999;

   for(Task singolaAttivita : TaskNodes){

      int temp = L[singolaAttivita.successorNode] - singolaAttivita.duration;
      if (temp < min) min = temp;

   }

   L[i] = min;
   System.out.println("= "+L[i]+"");

}
```

```
AssignValueToTasks(tasks, E,L);


    Arrays.sort(ret, new Comparator<Task>() {

        @Override
        public int compare(Task o1, Task o2) {
            return o1.name.compareTo(o2.name);
        }
    });


    return ret;

}

public static int geIndextMinValue(List<Integer> numbers){

    int minValue = numbers.get(0);
    int index = 0;

    for(int i=0;i<numbers.size();i++){
      if(numbers.get(i) < minValue){
        minValue = numbers.get(i);
        index = i;
      }
    }
    return index;
}


public static void AssignValueToTasks(Set<Task> tasks, int[] E, int[] L) {

  for (Task task : tasks) {

      task.earlyStart = E[task.predecessorNode];
      task.earlyFinish = task.earlyStart + task.duration;

      task.latestFinish = L[task.successorNode];
      task.latestStart = task.latestFinish - task.duration;

  }

}

public static HashSet<Task> TaskFromSuccessorNode(Set<Task> tasks, int suc) {

    HashSet<Task> taskFind = new HashSet<Task>();

    for (Task task : tasks) {

        if (task.successorNode == suc) {
            taskFind.add(task);
```

```
        }
      }
      return taskFind;
   }

   public static HashSet<Task> TaskFromPredecessorNode(Set<Task> tasks, int pre) {


      HashSet<Task> taskFind = new HashSet<Task>();

      for (Task task : tasks) {

         if (task.predecessorNode == pre) {
            taskFind.add(task);

         }
      }
      return taskFind;
   }


   public static void printActivitiesOnCriticalPath(Task[] tasks) {

      System.out.print("Attività nel cammino critico: ");

      for (Task t : tasks) {
         if ((t.latestStart - t.earlyStart)  == 0) {
            System.out.print(t.name+" ");
         }

      }
      System.out.print("\n\n\n");
   }

   public static void print(Task[] tasks) {
      System.out.print("\n");
      System.out.format(format, "Attività", "Durata", "Utilizzo Risorsa", "ES", "EF", "LS", "LF", "Total
Float", "Attività Critica ?");
      for (Task t : tasks)
           System.out.format(format, (Object[]) t.toStringArray());
           System.out.print("\n");
   }
}
```

Eseguendo il programma l'output che riporta è il seguente:

E(0) = 0

E(1) = 4

E(2) = 6

E(3) = 15

E(4) = 18

E(5) = 24

Tempo minimo di completamento dell'intero progetto = 24

L(5) = 24

L(4) = 18

L(3) = 16

L(2) = 6

L(1) = 4

L(0) = 0

Attività	Durata	ES	EF	LS	LF	Total Float	Attività Critica ?
A (0,1)	4	0	4	0	4	0	Sì
B (0,2)	3	0	3	3	6	3	No
C (1,2)	2	4	6	4	6	0	Sì
D (1,3)	7	4	11	9	16	5	No
E (2,3)	9	6	15	7	16	1	No
F (2,4)	12	6	18	6	18	0	Sì
G (3,4)	2	15	17	16	18	1	No
H (3,5)	5	15	20	19	24	4	No
I (4,5)	6	18	24	18	24	0	Sì

Attività nel cammino critico: A C F I

Riprendiamo l'esempio del progetto composto dalle seguenti nove attività ed aggiungiamo l'impiego delle risorse per ciascuna attività:

Attività	Predecessori	Durata	Risorsa Utilizzata
A	-	4	5
B	-	3	3
C	A	2	7
D	A	7	6
E	B, C	9	9
F	B, C	12	8
G	D, E	2	5
H	D, E	5	8
I	F, G	6	9

Le risorsa sono R1 con 8 come massima capacità di utilizzo e R2 con 10 come massima capacità di utilizzo.

Di seguito la spiegazione dell'algoritmo risolutivo, basato sulla tecnica Parallela per il problema CPM – Risorse con 2 risorse limitate:

Step 1:

inizializzazione di tutte le variabili coinvolte

Step 2:

le attività inserite nel decision set vengono ordinate in base al criterio di priorità, cioè in ordine del total float parallelo calcolo sull'attività.

Step 3:

le attività vengono estratte dalla lista ordinata DS e schedulate al tempo t, se vi è una capacità massima di risorsa sufficiente tra quelle disponibili; le attività che possono quindi essere schedulate vengono tolte dalla lista, mentre le attività rimaste restano nel DS

Step 4:

t = t+1; ogni attività le cui precedenti siano tutte terminate al tempo t, viene aggiunta a DS(t)

Step 5:

se DS(t) = 0 allora STOP, altrimenti si ritorna al Passo 2.

Funzione in java per la lo scheduling del cpm risorse – tecnica parallela con 2 risorse:

```
public static void calculateParallelResourceScheduling (Set<Task> tasks) {

    List<Task> remaining = new ArrayList<Task>(tasks);
    List<Task> decisionset = new ArrayList<Task>();
    List<Task> inprogress = new ArrayList<Task>();
    List<Task> completed = new ArrayList<Task>();

    Resource R1 = new Resource("R1", 8);
    Resource R2 = new Resource("R2", 10);

    List<Resource> resources_list = new ArrayList<Resource>();
    resources_list.add(R1);
    resources_list.add(R2);

    int R1_cdl = 0;
    int R2_cdl = 0;

    int t = 0;


    System.out.println("SCHEDULAZIONE ATTIVITA' PROGETTO CON R1 E R2 LIMITATE \n");

    System.out.println("R1 Max capacità = "+R1.max_occupation);
    System.out.println("R2 Max capacità = "+R2.max_occupation);

    // PASSO iniziale
    for (Task task : tasks) {

        if (task.dependencies.isEmpty()) {

            task.setTFParallel(task.latestStart-t);
            decisionset.add(task);
        }
    }
    Collections.sort(decisionset, new OrderByTFParallel());




do {



// ASSEGNO TASK A RISORSE

   List<Integer> indextasktodelete = new ArrayList<Integer>();

      // scrivo tutte le variabili
      System.out.println("\nt=["+t+"]");
```

```
    System.out.print("DS=[");
    for (int i =0;i < decisionset.size(); i++){ System.out.print(" "+decisionset.get(i).name);  }
    System.out.print(" ]\n");

    System.out.print("Attività da Completare = [");
    for (int i =0;i < remaining.size(); i++){ System.out.print(" "+remaining.get(i).name);  }
    System.out.print(" ]\n");

    System.out.print("Attività in Progress = [");
    for (int i =0;i < inprogress.size(); i++){ System.out.print(" "+inprogress.get(i).name);  }
    System.out.print(" ]\n");

    System.out.print("Attività Completate = [");
    for (int i =0;i < completed.size(); i++){ System.out.print(" "+completed.get(i).name);  }
    System.out.print(" ]\n");

    System.out.print("Risorse:\n");


    for (int j =0; j < resources_list.size(); j++) {

       System.out.print(resources_list.get(j).name+"=[");


       for (Iterator<Task> it = resources_list.get(j).tasks.iterator(); it.hasNext();) {
          Task task = it.next();

          System.out.print(task.name);
       }

       System.out.print("]");

       int capacity = resources_list.get(j).max_occupation - resources_list.get(j).cna;
       double workload = ((capacity/ (double) resources_list.get(j).max_occupation)*100);

       System.out.print(" - Workload(%): ");
       System.out.format("%7.2f%n", workload);
    }


 // ciclo tutte le attività inserite in DS

 for (int i =0;i < decisionset.size(); i++) {

    int totalCNA = 0;

    for (int j =0; j < resources_list.size(); j++) {

       totalCNA += resources_list.get(j).cna;
    }

    if (totalCNA >= decisionset.get(i).resources_allocation ) {
```

```
        int[] intArray = new int[3];

        intArray[0] = R1.cna - decisionset.get(i).resources_allocation;
        intArray[1] = R2.cna - decisionset.get(i).resources_allocation;
        intArray[2] = (R1.cna + R2.cna) - decisionset.get(i).resources_allocation;

        List<Integer> x = new ArrayList<Integer>();

        if (intArray[0] >= 0) x.add(intArray[0]);
        if (intArray[1] >= 0) x.add(intArray[1]);
        if (intArray[2] >= 0 && intArray[2] != intArray[0] && intArray[2] != intArray[1])
x.add(intArray[2]);

        int k = geIndextMinValue(x);

        int minValue = x.get(k);
        int numeroRisorsa = -1;

        for(int indice=intArray.length-1;indice>=0 ;indice--){
           if (intArray[indice] == minValue) numeroRisorsa = indice+1;
        }

        // se posso usare le eseguire questa attività con una sola risorsa in quanto la sua capacità
        // è maggiore della richiesta di risorsa dell'attività considerata
        if (numeroRisorsa == 1 || numeroRisorsa == 2) {


           decisionset.get(i).una = decisionset.get(i).resources_allocation -
resources_list.get(numeroRisorsa-1).cna;


           resources_list.get(numeroRisorsa-1).cna = Math.abs(decisionset.get(i).una);
           resources_list.get(numeroRisorsa-1).tasks.add(decisionset.get(i));

           // aggiungo l'attività ad inprogress
           inprogress.add(decisionset.get(i));

           // poi dovrò togliere l'attività dal decision set

        }
        // altrimenti devo "spezzare" l'attività eseguendola una parte su una risorsa e
        // una parta sull'altra partendo dalla risorsa con capacità non allocata maggiore
        else {

           // se la capacità non allocata delle risorsa1 è maggiore di quella della risorsa2
           if (resources_list.get(0).cna > resources_list.get(1).cna) {


              R1_cdl = resources_list.get(0).cna;

              decisionset.get(i).una = decisionset.get(i).resources_allocation -
resources_list.get(0).cna;
              resources_list.get(0).cna = 0;
```

```
                resources_list.get(0).tasks.add(decisionset.get(i));

                decisionset.get(i).una = decisionset.get(i).una - resources_list.get(1).cna;
                resources_list.get(1).cna = Math.abs(decisionset.get(i).una);
                R2_cdl = Math.abs(decisionset.get(i).una);

                resources_list.get(1).tasks.add(decisionset.get(i));

            }
            else {


                R2_cdl = resources_list.get(1).cna;

                decisionset.get(i).una = decisionset.get(i).resources_allocation -
resources_list.get(1).cna;
                resources_list.get(1).cna = 0;
                resources_list.get(1).tasks.add(decisionset.get(i));

                decisionset.get(i).una = decisionset.get(i).una - resources_list.get(0).cna;
                resources_list.get(0).cna = Math.abs(decisionset.get(i).una);
                R1_cdl = Math.abs(decisionset.get(i).una);

                resources_list.get(0).tasks.add(decisionset.get(i));

            }

            // aggiungo l'attività ad inprogress
            inprogress.add(decisionset.get(i));

          }

        } // se


    }  // for -> ciclo tutte le attività inserite in DS

    //se ci sono attavità che sono passate inprogress devo toglierle dal DecisionSet

    for (int i =0;i < inprogress.size(); i++){

       if (decisionset.contains(inprogress.get(i)))   decisionset.remove(inprogress.get(i));

    }



    for(int i=0;i< indextasktodelete.size();i++){

       decisionset.remove(inprogress.get(i));
    }
```

```
// AVANZAMENTO TEMPORALE

t += 1;

// ciclo tutte le attività inserite in inprogress

for (int i =0;i < inprogress.size(); i++) {

   inprogress.get(i).progress += 1;

   // se ho completato l'attività
   if (inprogress.get(i).progress == inprogress.get(i).duration) {

      completed.add(inprogress.get(i));
      remaining.remove(inprogress.get(i));


      // LIBERO LE RISORSE

      // R1
      if ( resources_list.get(0).tasks.contains(inprogress.get(i)) &&
            !(resources_list.get(1).tasks.contains(inprogress.get(i))) ) {

         resources_list.get(0).tasks.remove(inprogress.get(i));
         resources_list.get(0).cna += inprogress.get(i).resources_allocation;
      }
      // R2
      else if ( !(resources_list.get(0).tasks.contains(inprogress.get(i))) &&
            resources_list.get(1).tasks.contains(inprogress.get(i)) ) {

         resources_list.get(1).tasks.remove(inprogress.get(i));
         resources_list.get(1).cna += inprogress.get(i).resources_allocation;
      }
      else {

        // ri-assegno alle risorse la quantità che precedentemente mi ero salvato

            resources_list.get(0).cna += R1_cdl;
            resources_list.get(0).tasks.remove(inprogress.get(i));

            resources_list.get(1).cna += R2_cdl;
            resources_list.get(1).tasks.remove(inprogress.get(i));

      }


   }

} // for ciclo tutte le attività inserite in inprogress

//se ci sono attvità che sono passate a completed devono essere tolte da inprogress

for (int i =0;i < completed.size(); i++){
```

```
        if (inprogress.contains(completed.get(i)))   inprogress.remove(completed.get(i));

    }


    // AGGIORNAMENTO DS

    boolean reorder = false;
        // inserisco nuove attività nel Decision Set le cui precedenze sono già state processate e che non sono in svolgimento nel Decision Set
        for (Task task : tasks) {

            if ( !(decisionset.contains(task)) & !(inprogress.contains(task)) & !(completed.contains(task)) & completed.containsAll(task.dependencies)) {

                task.setTFParallel(task.latestStart-t);
                decisionset.add(task);
                reorder = true;

            }
        }

        if (reorder) Collections.sort(decisionset, new OrderByTFParallel());


    } while (!remaining.isEmpty() || !inprogress.isEmpty());

      // scrivo tutte le variabili nella condizione finale
      System.out.println("\nt=["+t+"]");

      System.out.print("DS=[");
      for (int i =0;i < decisionset.size(); i++){ System.out.print(" "+decisionset.get(i).name);  }
      System.out.print(" ]\n");

      System.out.print("Attività da Completare = [");
      for (int i =0;i < remaining.size(); i++){ System.out.print(" "+remaining.get(i).name);  }
      System.out.print(" ]\n");

      System.out.print("Attività in Progress = [");
      for (int i =0;i < inprogress.size(); i++){ System.out.print(" "+inprogress.get(i).name);  }
      System.out.print(" ]\n");

      System.out.print("Attività Completate = [");
      for (int i =0;i < completed.size(); i++){ System.out.print(" "+completed.get(i).name);  }
      System.out.print(" ]\n");

      System.out.print("Risorse:\n");


      for (int j =0; j < resources_list.size(); j++) {

        System.out.print(resources_list.get(j).name+"=[");
```

```
            for (Iterator<Task> it = resources_list.get(j).tasks.iterator(); it.hasNext();) {
                Task task = it.next();

                System.out.print(task.name);
            }

            System.out.print("]");

            int capacity = resources_list.get(j).max_occupation - resources_list.get(j).cna;
            double workload = ((capacity/ (double) resources_list.get(j).max_occupation)*100);

            System.out.print(" - Workload(%): ");
            System.out.format("%7.2f%n", workload);
        }

}
```

L’output del programma è il seguente:

SCHEDULAZIONE ATTIVITA' PROGETTO CON R1 E R2 LIMITATE

R1 Max capacità = 8
R2 Max capacità = 10

t=[0]
DS=[A B]
Attività da Completare = [F E H A I C D G B]
Attività in Progress = [ ]
Attività Completate = [ ]
Risorse:
R1=[] - Workload(%): 0,00
R2=[] - Workload(%): 0,00

t=[1]
DS=[ ]
Attività da Completare = [F E H A I C D G B]
Attività in Progress = [A B]
Attività Completate = [ ]
Risorse:
R1=[AB] - Workload(%): 100,00
R2=[] - Workload(%): 0,00

t=[2]
DS=[ ]
Attività da Completare = [F E H A I C D G B]
Attività in Progress = [A B]
Attività Completate = [ ]
Risorse:
R1=[AB] - Workload(%): 100,00
R2=[] - Workload(%): 0,00

t=[3]
DS=[ ]
Attività da Completare = [F E H A I C D G]
Attività in Progress = [A]
Attività Completate = [B]
Risorse:
R1=[A] - Workload(%): 62,50
R2=[] - Workload(%): 0,00

t=[4]
DS=[C D]
Attività da Completare = [F E H I C D G]
Attività in Progress = [ ]
Attività Completate = [B A]
Risorse:
R1=[] - Workload(%): 0,00
R2=[] - Workload(%): 0,00

t=[5]
DS=[ ]
Attività da Completare = [F E H I C D G]
Attività in Progress = [C D]
Attività Completate = [B A]
Risorse:
R1=[C] - Workload(%): 87,50
R2=[D] - Workload(%): 60,00

t=[6]
DS=[F E]
Attività da Completare = [F E H I D G]
Attività in Progress = [D]
Attività Completate = [B A C]
Risorse:
R1=[] - Workload(%): 0,00
R2=[D] - Workload(%): 60,00

t=[7]
DS=[E]
Attività da Completare = [F E H I D G]
Attività in Progress = [D F]
Attività Completate = [B A C]
Risorse:
R1=[F] - Workload(%): 100,00
R2=[D] - Workload(%): 60,00

t=[8]
DS=[E]
Attività da Completare = [F E H I D G]
Attività in Progress = [D F]
Attività Completate = [B A C]
Risorse:
R1=[F] - Workload(%): 100,00
R2=[D] - Workload(%): 60,00

t=[9]
DS=[E]
Attività da Completare = [F E H I D G]
Attività in Progress = [D F]
Attività Completate = [B A C]
Risorse:
R1=[F] - Workload(%): 100,00
R2=[D] - Workload(%): 60,00

t=[10]
DS=[E]
Attività da Completare = [F E H I D G]
Attività in Progress = [D F]
Attività Completate = [B A C]
Risorse:
R1=[F] - Workload(%): 100,00
R2=[D] - Workload(%):   60,00

t=[11]
DS=[E]
Attività da Completare = [F E H I G]
Attività in Progress = [F]
Attività Completate = [B A C D]
Risorse:
R1=[F] - Workload(%): 100,00
R2=[] - Workload(%):   0,00

t=[12]
DS=[ ]
Attività da Completare = [F E H I G]
Attività in Progress = [F E]
Attività Completate = [B A C D]
Risorse:
R1=[F] - Workload(%): 100,00
R2=[E] - Workload(%):   90,00

t=[13]
DS=[ ]
Attività da Completare = [F E H I G]
Attività in Progress = [F E]
Attività Completate = [B A C D]
Risorse:
R1=[F] - Workload(%): 100,00
R2=[E] - Workload(%):   90,00

t=[14]
DS=[ ]
Attività da Completare = [F E H I G]
Attività in Progress = [F E]
Attività Completate = [B A C D]
Risorse:
R1=[F] - Workload(%): 100,00
R2=[E] - Workload(%):   90,00

t=[15]
DS=[ ]
Attività da Completare = [F E H I G]
Attività in Progress = [F E]
Attività Completate = [B A C D]
Risorse:
R1=[F] - Workload(%): 100,00
R2=[E] - Workload(%):   90,00

t=[16]
DS=[ ]
Attività da Completare = [F E H I G]
Attività in Progress = [F E]
Attività Completate = [B A C D]
Risorse:
R1=[F] - Workload(%): 100,00
R2=[E] - Workload(%):   90,00

t=[17]
DS=[ ]
Attività da Completare = [F E H I G]
Attività in Progress = [F E]
Attività Completate = [B A C D]
Risorse:
R1=[F] - Workload(%): 100,00
R2=[E] - Workload(%):   90,00

t=[18]
DS=[ ]
Attività da Completare = [E H I G]
Attività in Progress = [E]
Attività Completate = [B A C D F]
Risorse:
R1=[] - Workload(%):   0,00
R2=[E] - Workload(%):   90,00

t=[19]
DS=[ ]
Attività da Completare = [E H I G]
Attività in Progress = [E]
Attività Completate = [B A C D F]
Risorse:
R1=[] - Workload(%):   0,00
R2=[E] - Workload(%):   90,00

t=[20]
DS=[G H]
Attività da Completare = [H I G]
Attività in Progress = [ ]
Attività Completate = [B A C D F E]
Risorse:
R1=[] - Workload(%): 0,00
R2=[] - Workload(%): 0,00

t=[21]
DS=[ ]
Attività da Completare = [H I G]
Attività in Progress = [G H]
Attività Completate = [B A C D F E]
Risorse:
R1=[G] - Workload(%): 62,50
R2=[H] - Workload(%): 80,00

t=[22]
DS=[I]
Attività da Completare = [H I]
Attività in Progress = [H]
Attività Completate = [B A C D F E G]
Risorse:
R1=[] - Workload(%): 0,00
R2=[H] - Workload(%): 80,00

t=[23]
DS=[ ]
Attività da Completare = [H I]
Attività in Progress = [H I]
Attività Completate = [B A C D F E G]
Risorse:
R1=[I] - Workload(%): 100,00
R2=[HI] - Workload(%): 90,00

t=[24]
DS=[ ]
Attività da Completare = [H I]
Attività in Progress = [H I]
Attività Completate = [B A C D F E G]
Risorse:
R1=[I] - Workload(%): 100,00
R2=[HI] - Workload(%): 90,00

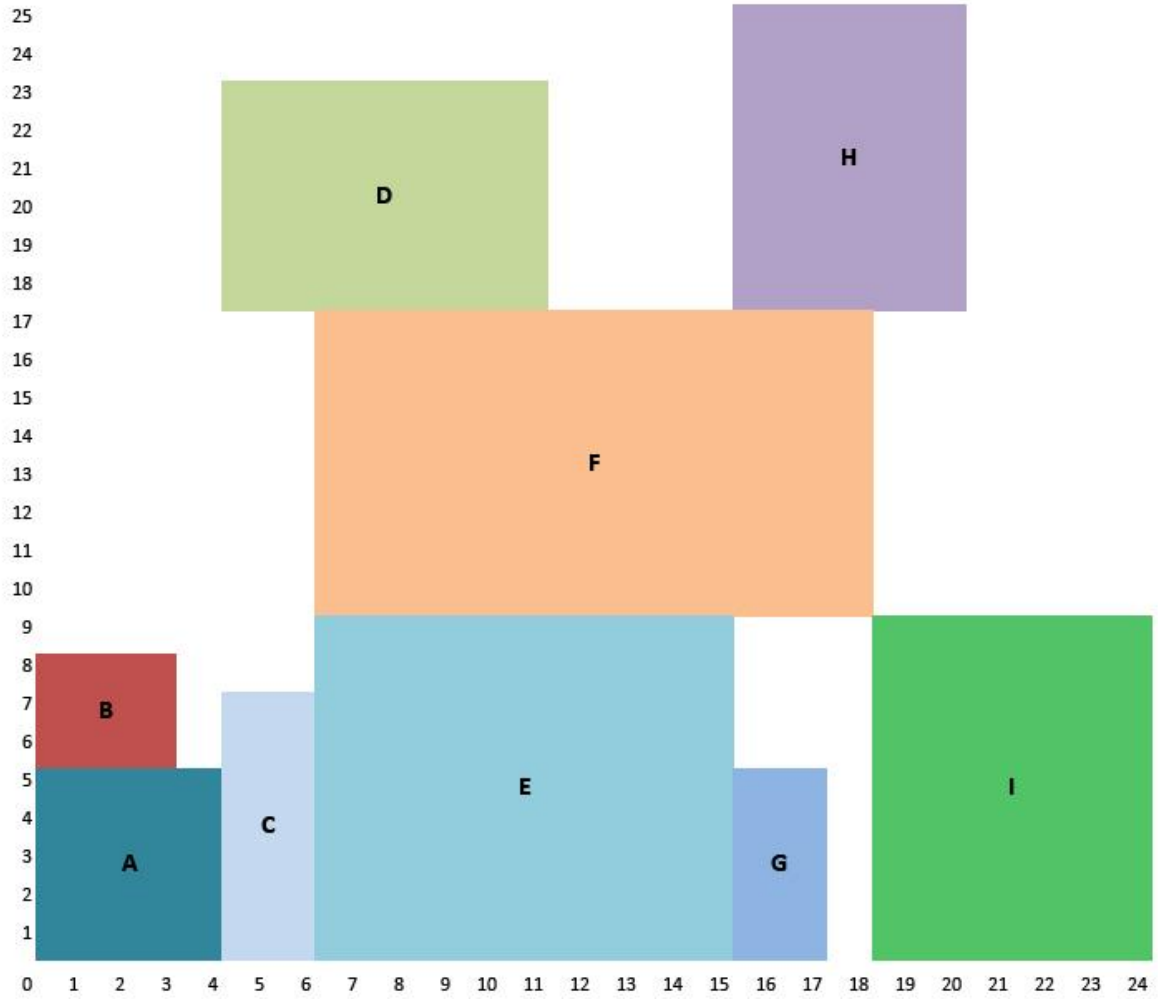
t=[25]
DS=[ ]
Attività da Completare = [I]
Attività in Progress = [I]
Attività Completate = [B A C D F E G H]
Risorse:
R1=[I] - Workload(%): 100,00
R2=[I] - Workload(%):   10,00

t=[26]
DS=[ ]
Attività da Completare = [I]
Attività in Progress = [I]
Attività Completate = [B A C D F E G H]
Risorse:
R1=[I] - Workload(%): 100,00
R2=[I] - Workload(%):   10,00

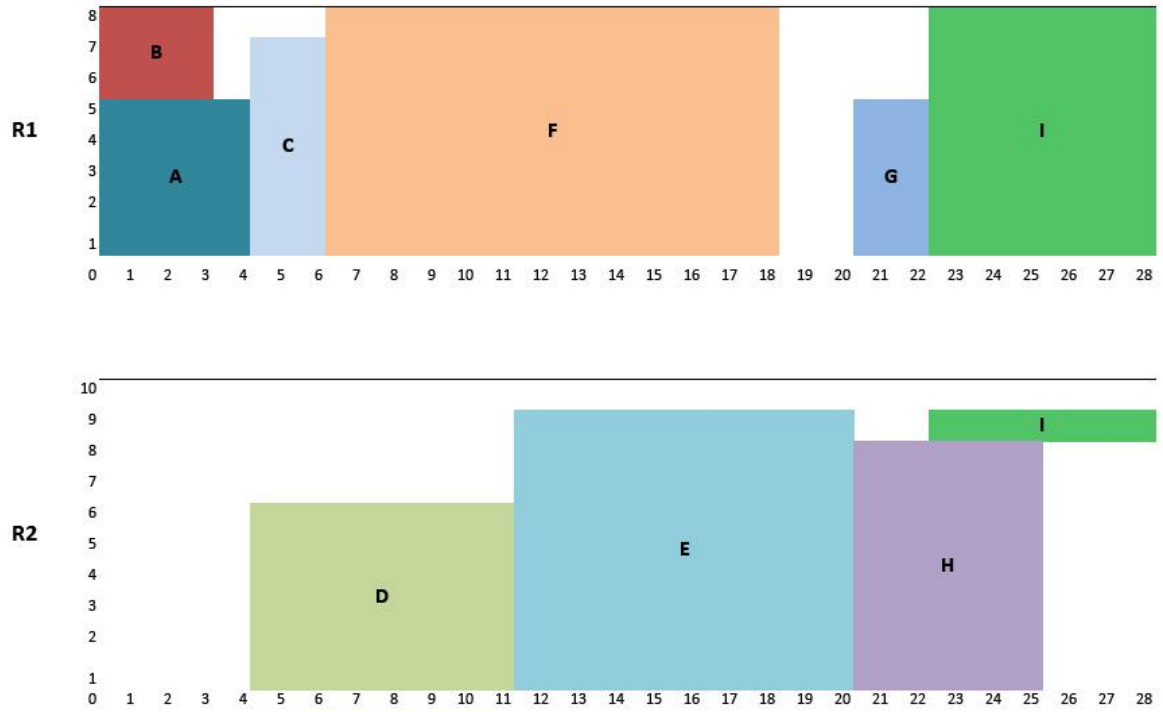
t=[27]
DS=[ ]
Attività da Completare = [I]
Attività in Progress = [I]
Attività Completate = [B A C D F E G H]
Risorse:
R1=[I] - Workload(%): 100,00
R2=[I] - Workload(%):   10,00

t=[28]
DS=[ ]
Attività da Completare = [ ]
Attività in Progress = [ ]
Attività Completate = [B A C D F E G H I]
Risorse:
R1=[] - Workload(%):   0,00
R2=[] - Workload(%):   0,00

Di seguito viene riportato il grafico del progetto formato da 9 attività con la schedulazione delle attività considerando le risorse illimitate:



Di seguito viene invece riportato il grafico del progetto formato da 9 attività con la schedulazione delle attività considerando però il vincolo di 2 risorse limitate (R1 max 8, R2 max 10):



Dal confronto emerge chiaramente come, introducendo il vincolo sul numero e sulla capacità delle risorse, la data di completamento del progetto non può che aumentare passando da 24 a 28 unità.

CONCLUSIONI

In questa esposizione è stata trattato l'approfondimento dei metodi matematici per la schedulazione delle attività appartenenti ad un progetto, nel caso di risorse illimitate e nello specifico nel caso di risorse limitate. Le tecnica principale che è stata utilizzata ed approfondita è quella del metodo C.P.M.. Nella prima parte sono stati introdotti i concetti che stanno alla base del Project Management, identificando quelle terminologie e metodiche che sono entrati a far parte di uno standard nell'ambito della gestione organizzata di un progetto.

Introducendo le tecniche reticolari principali e le rappresentazioni attraverso i modelli reticolari, sì sono poste le basi per l'approfondimento della tecnica C.P.M. e di come questa tecnica sia stata concepita dai suoi creatori come strumento principale per identificare il cammino minimo di completamento dell'intero progetto rispettando i vincoli imposti dal problema, derivanti dalla precedenza delle varie attività, ed in un secondo momento inserendo anche il vincolo sulla capacità finita delle risorse. Con il metodo C.P.M. si è introdotto anche il concetto importante di attività critica come quella attività che se fosse in ritardo rispetto a quanto pianificato, comporterebbe un ritardo nel tempo di completamento dell'intero progetto, facendone aumentare la data di fine progetto.

Con la realizzazione di un programma scritto con il linguaggio java, ed in particolare nella parte riguardante la schedulazione delle attività con risorse multiple limitate, si è cercato di dimostrare l'efficacia del metodo C.P.M. come strumento per permettere al Project Manager di individuare correttamente la schedulazione della attività evidenziandone, per ciascuna risorsa coinvolta, il carico di lavoro durante tutte le fasi di svolgimento del progetto.

BIBLIOGRAFIA

- G. Andreatta, F. Mason, G. Romanin Jacur, 1996, Ottimizzazione su reti – 2° EDIZIONE – Capitolo 10, Padova, Italia: Edizioni Libreria Progetto Padova

- L. Bianco, M. Caramia, 2006, Metodi Quantitativi per il Project Management – Pianificazione delle attività e gestione delle risorse, Italia: HOEPLI

- Cay S. Horstmann , 2005, Concetti di informatica e fondamenti di Java – Terza Edizione – Italia Apogeo

- Project Management – Wikipedia: https://it.wikipedia.org/wiki/Project_management

- P. Stelth (MSc) - Professor G. Le Roy (PhD) - Projects' Analysis through CPM (Critical Path Method)