

Pianificazione Automatica e sistemi di Supporto delle Decisioni

Matteo Aprile

Professore: Giampaolo Ghiani, Enamuele Manni

INDICE		V	Optimization models review	8
I	Definizioni - 22/23.09.22	1	V-A Scheduling	8
I-A	Business Analytics	1	V-B Project scheduling	8
I-B	Decisioni	1		
I-C	Business Intelligence (BI)	2		
I-D	Data visualization	2		
I-E	Decision Support Systems (DSS)	2		
I-F	Operations Research (OR)	2		
I-G	Agents	2		
I-H	Artificial Intelligence (AI)	2		
I-I	Machine Learning (ML)	2		
I-J	Deep Learning	2		
I-K	Data Mining (DM)	2		
II	Software solutions and languages for AP and DSS - 29.09.22	3		
II-A	Decisioni operative/strutturate	3		
II-B	Decisioni non strutturate o destrutturate	3		
II-C	Decisioni semistrutturate	3		
III	Introduzione all'ottimizzazione matematica - 30.09.22	4		
III-A	Introduzione	4		
III-B	Ingredienti principali	4		
III-C	Descrizione del problema	4		
III-D	Dati del problema	4		
III-E	Descrizione del problema con un modello matematico	4		
III-F	Risolvere il modello matematico	5		
III-G	Terminologia	5		
III-H	Implementazione in Python	5		
III-I	Esercitazione	5		
IV	Formulazioni equivalenti di un problema di programmazione lineare	7		
IV-A	Problema in FORMA GENERALE	7		
IV-B	Problema in FORMA CANONICA	7		
IV-C	Problema in FORMA STANDARD	7		
IV-D	Terminologia	7		
IV-E	Trasformazioni per ricondursi alla forma standard	7		
IV-F	Esempio 1	7		
IV-G	Esempio 2	8		

I. DEFINIZIONI - 22/23.09.22

Ci occuperemo di 2 tipi di scenari:

- usare **algoritmi a supporto delle decisioni**
- usare algoritmi che **sostituiscono completamente l'uomo**

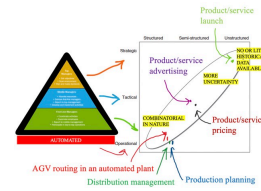


Figura 2. Diagonale decisionale

A. Business Analytics

Disciplina che utilizza dati, statistiche, modelli matematici per **aiutare a prendere delle decisioni in base a dei dati**.

Possiamo racchiudere i suoi **passaggi** in:

- 1) **descriptive analytics**: capire **cosa sia successo nel passato** tramite i dati disponibili
- 2) **predictive analytics**: cercare di **fare delle previsioni** in base ai dati già disponibili
- 3) **prescriptive analytics**: **creare un piano di azione** per poter massimizzare il KPI (Key Performance Indicator)

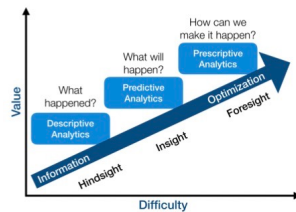


Figura 1. Fasi della business analytics

B. Decisioni

Rappresenta la **scelta di un elemento tra più soluzioni** dopo aver ponderato le opzioni.

Possiamo avere più casi d'uso:

- **simplest case**: abbiamo **poche alternative** quindi una semplice scelta
- **multiple criteria**: abbiamo **più metri di paragone** delle performance, quindi si dovranno tenere in conto:
 - **soluzioni migliori** di altre (dette di Pareto)
 - **vincoli** dovuti dai clienti o da casi logistici da gestire (es: spedizioni)
 - ottimizzazioni matematiche
 - **conflitti tra i vincoli**
- **incertezze e rischi**:
 - **decisioni operative**: di **breve periodo reversibili e limitate** a "n" persone del team
 - **decisioni tattiche**: **coinvolge una parte dell'organizzazione** per un medio periodo
 - **decisioni strategiche**: di **lungo periodo non reversibili e coinvolgono denaro**
 - **decisioni strutturate**: hanno una **procedura di risoluzione specifica**
 - **decisioni non strutturate**: richiedono **creatività ed esperienza** in un dato settore

C. Business Intelligence (BI)

Usato per indicare un **sistema dedicato alla raccolta di dati e alla loro elaborazione** al fine di un reporting, infatti per "Intelligence" si intende investigazione. Venivano **usati su dati atomici** per avere delle conoscenze approfondite in un determinato business.

D. Data visualization

Consiste nel **prendere dati e plottare un grafico**, ma in realtà ora si ha una trattazione più metodologica, cioè se visualizzare in modo statico o meno i dati.

E. Decision Support Systems (DSS)

Si indicava un **sistema computerizzato dotato di un sistema di "data management"** per creare un modello di ottimizzazione, fornendo un feedback tramite un'interfaccia. Ora indica una varietà di sistemi per visualizzare i dati in larga misura o meno.

F. Operations Research (OR)

Attività organizzative per portare avanti un sistema logistico. Per "research" si indica la ricerca delle operation per conseguire dei risultati, avremo come sottocategorie:

- **ottimizzazione matematica**
- **queueing theory**: **studio matematico delle linee in attesa** il limite è che funzionano solo con sistemi semplici e con richieste di servizio in ordine stocastico
- **simulazione**: per usarle è **necessario generare dei numeri randomici quindi inconveniente** (bisogna fare un'analisi statistica dei risultati dalle quali si farà una **stima**)
- **game theory**: decisioni con **più players**

G. Agents

È un **sistema che si muove in un environment** (ambiente), ha dei **sensori** tramite i quali percepisce alcuni aspetti del mondo che lo circonda quindi si crea una **rappresentazione del mondo circostante** che può vedere. È capace di **influenzare l'ambiente tramite degli attuatori** come ruote o braccia (intendiamo anche agenti software).

Possiamo classificarli come:

- **agenti autonomi**: se è concepito in modo tale che **tramite un'istruzione sintetica raggiunge un goal sviluppando le azioni per raggiungerlo** In realtà può anche non essere una sequenza di azioni dato che **potrebbero esserci degli imprevisti**
- **agenti intelligenti**: se

- **impara dall'esperienza**
- crea una **rappresentazione dell'ambiente** che lo circonda e **ci ragiona sopra** per un possibile risultato delle proprie azioni
- **si adatta ad un ambiente mutevole**

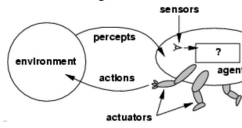


Figura 3. Schematizzazione di un agente e sue caratteristiche

H. Artificial Intelligence (AI)

Comprende tante sottodiscipline:

- **automated reasoning**: legato alla **rappresentazione del mondo** e **come ragionare su di essa** ma anche calcolandone le probabilità
- **automated planning**: usato in ambienti industriali
- **automated learning**
- **natural language processing**: **sviluppare agenti software** per fare sintesi di testi, scrivere automaticamente articoli, chat bot, ecc
- **perception**: visione artificiale
- **manipulation**: avere un **agente che può modificare l'agente** circostante

I. Machine Learning (ML)

Consiste nell'**apprendimento automatico** e quindi lo sviluppo degli **agenti che apprendo tramite la loro esperienza pregressa**. Ci sarà allora una fase di **training**. Una delle possibili **architetture che permette di farlo** sono le **Neural Networks** prima avevano solo 2/3 neuroni, ora ne hanno vari strati il che fornisce delle prestazioni impressionanti

J. Deep Learning

Si basa sull'**apprendimento automatico** con reti neurali tramite un gran numero di strati di neuroni.

K. Data Mining (DM)

Usare metodi di Machine Learning per **estrarre manualmente dei pattern dai dati**, cioè una **regolarità o un trend**. È quindi la parte nobile del knowledge discovery in db, dato che i dati sono in genere disponibili su db o da altre piattaforme.

La **sequenza** nella quale interviene è:

- 1) **prendere** i dati
- 2) trovare i vari **target**
- 3) **preprocessare** i dati
- 4) trasformare i dati tramite il **data mining**
- 5) trovare dei **patterns** (dopo il data mining)

II. SOFTWARE SOLUTIONS AND LANGUAGES FOR AP AND DSS - 29.09.22

A. Decisioni operative/strutturate

Sono una classe importante, **si possono prendere tramite una procedura standard** che può seguire un manuale o delle normative, automatizzata o no. Queste decisioni di breve periodo si collocano in basso a destra in figura 2.

Non essendo decisioni dove possiamo solo supportare, allora si possono andare a **codificare in un linguaggio di programmazione procedurale** come C++, Java, ecc...

Potremo avere un **approccio**:

- **procedurale**: dove devo far **generare delle azioni** in seguito di un obiettivo
- **dichiarativo**: si divide in:
 - 1) **modellazione** del problema
 - 2) **descrivere tramite un linguaggio di modellazione** (modelling language) che è un linguaggio di programmazione matematico come AMPL, oppure in linguaggi come python con Amply e Pulp
 - 3) **solver of the shelf**, che ci darà delle istruzioni per il nostro contesto

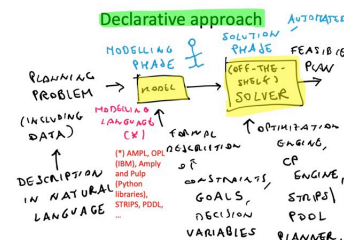


Figura 4. procedura implementata

Il che è utile dato che **per agire su un problema basterà cambiare il modello** senza cambiare solver, dovrò solo cambiare il modello. È la soluzione più **economica e flessibile** ma è **meno performante** se in ambienti realtime devo prendere soluzioni in tempi molto stretti. Quindi in questi casi servono approcci procedurali.

Per sistemi che devono prendere soluzioni nel breve, si usa C, C++, C#.

B. Decisioni non strutturate o destrutturate

In questo caso **non possiamo automatizzare**, quindi:

- 1) tiro fuori i **dati aggregati**
- 2) si creano statistiche con **modelli di ottimizzazione**

Si usano degli **spreadsheet** che però non riescono a gestire big data e tendono a generare errori.

Il linguaggio più usato è **Python** ma non è la soluzione più efficiente per tutte quelle applicazioni dove il tempo di calcolo è importante.

C. Decisioni semistrutturate

Vogliamo solo **valutare le prestazioni di un sistema**. Un esempio sono i sistemi che **presentano un comportamento random** per motivi:

- 1) i **server hanno un tempo di risposta** che possiamo modellare
- 2) le richieste del sistema **arrivano in maniera stocastica**

Si usano, in questo caso, **metodi simulativi** tramite dei Visual Interactive Modelling System, **per simulare la rete** per la quale passano le informazioni e i server ognuno con diverse proprietà di ciascun linker.

III. INTRODUZIONE ALL'OTTIMIZZAZIONE MATEMATICA - 30.09.22

A. Introduzione

Partiamo da un **insieme di formule ed equazioni che model-leranno il problema**. Con questo modello proviamo a trovare una **soluzione** al nostro problema **attraverso algoritmi o ri-solutori**. L'output è una soluzione per il nostro modello da implementare nel mondo reale.

B. Ingredienti principali

Gli ingredienti principali sanno:

- **dati** del problema
- **variabili**: dette anche var decisionali: scelte da fare in merito al problema. rappresentano quindi le scelte, quello su cui il decisore può intervenire
- **vincoli**: equazioni che definiscono i valori che le variabili possono assumere
- **funzione obiettivo**: sarà una formula che rappresenta una misura di tipo quantitativo per capire quando è buona la soluzione che abbiamo ottenuto. quindi dovremo ottimizzare questo valore in base al contesto

Parleremo di **programmazione lineare con modelli matematici** o relazioni lineari, dato che **molti problemi reali si rifanno a modelli lineari**, per quanto essi possano essere complessi.

C. Descrizione del problema

Proviamo a risolvere un problema di mix di produzione, cioè un sistema con un impianto con 2 stabilimenti in cui:

- 1) nel primo: diamo le materie prime e vengono realizzati i componenti in uscita
- 2) nel secondo: diamo i componenti realizzati che vengono assemblati per creare il prodotto finito

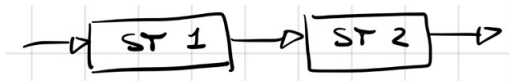


Figura 5. Catena tra i due stabilimenti

Supponendo di voler realizzare 2 prodotti A, B con un differente profitto. Determinare il **mix di produzione**, cioè quante unità di A e B produrre la prossima settimana. Saranno presenti dei **vincoli** creati dalle risorse come i macchinari o gli addetti che potranno lavorare un numero di ore finito.

D. Dati del problema

- ore di lavoro:

Stab	A	B	Addetti
1	4 ore	2 ore	10
2	2 ore	4 ore	10

Tabella I
TABELLA DELLE ORE DI LAVORO

- ogni addetto lavora 40 ore/settimana
- profitto /pallet:
- richiesta del prodotto nella prossima settimana:

A	B
15k	10k

Tabella II

TABELLA DEL PROFITTO /PALLET

A	B
40	120

Tabella III

TABELLA DEL PROFITTO EURO/PALLET

E. Descrizione del problema con un modello matematico

Per effettuare una modellazione faremo:

- 1) **identificare le variabili decisionali:**
 - x_A : # di pallet di prodotto A da realizzare
 - x_B : # di pallet di prodotto B da realizzare
- 2) **definire la funzione obiettivo (FO)**, per massimizzare il profitto
- 3) **definire i vincoli espressi come uguaglianza o disuguaglianza**
 - vincolo 1: capacità produttiva dello stab 1 $4x_A + 2x_B$ che non può superare $40 \cdot 10$ cioè ore disponibili ogni settimana per un addetto * numero di addetti:

$$4x_A + 2x_B \leq 400$$

- vincolo 2: capacità produttiva dello stabilimento 2 $2x_A + 4x_B$ che non può superare $40 \cdot 10$ cioè ore disponibili ogni settimana per un addetto * numero di addetti:

$$2x_A + 4x_B \leq 400$$

- vincolo 3: vincolo sulla richiesta di A:

$$x_A \leq 40$$

- vincolo 4: vincolo sulla richiesta di B:

$$x_B \leq 120$$

- vincolo 5: vincolo di non-negatività:

$$x_A, x_B \geq 0$$

Nella forma completa il **modello complessivo** è:

$$MAX = z = 15x_A + 10x_B$$

sottoposto ai vincoli (sv):

- $4x_A + 2x_B \leq 400$
- $2x_A + 4x_B \leq 400$
- $x_A \leq 40$
- $x_B \leq 120$
- $x_A, x_B \geq 0$

F. Risolvere il modello matematico

Rappresentiamo sul piano cartesiano tutte le soluzioni ammissibili cercando quella che massimizza il nostro risultato

Impostiamo delle rette per ogni vincolo:

- presa $4x_A + 2x_B \leq 400$ poniamo $= 0$, a turno, x_A e x_B : (200, 0)
- presa $2x_A + 4x_B \leq 400$ poniamo $= 0$, a turno, x_A e x_B : (0, 200)
- presa $x_A \leq 40$: (40, 0)
- presa $x_B \leq 120$: (0, 120)

Avremo allora una **regione ammissibile** dove valgono tutti i vincoli e nella quale dovrebbe essere presente la nostra soluzione ammissibile. Per trovare il punto che rende massima la funzione z usiamo il **metodo del gradiente**:

$$\nabla z = \begin{bmatrix} \frac{dz}{dx_A} \\ \frac{dz}{dx_B} \end{bmatrix} = \begin{bmatrix} 15 \\ 10 \end{bmatrix}$$

dove ∇z sarà la massima crescita che viene rappresentata tramite (15, 10).

Tracciando una **retta perpendicolare (curve di livello)** alla retta del gradiente avremo valori sempre buoni ma più bassi di quelli sul gradiente, a patto che siano validi. Troveremo in fine il punto massimo che consente di massimizzare, cioè il più estremo alla regione ammissibile sarà il nostro punto.

Seguendo la retta del gradiente troviamo che la **soluzione ottimale** si trova nell'intersezione tra le rette del vincolo 2 con il 3: $x_A = 40$ $2 \cdot 40 + 4x_B = 400$ quindi $x_B = 80$.

La soluzione ottimale sarà:

$$\begin{cases} x_A = 40 \\ 2x_A + 4x_B = 400 \end{cases} \quad \begin{cases} x_A = 40 \\ x_B = 80 \end{cases}$$

Si nota che lo stabilimento 2 viene saturato e quello 1 no, dal fatto che la soluzione giace sulla retta del vincolo per il quale si satura.

G. Terminologia

Possiamo avere altre forme di modelli di PL:

- fo da minimizzare
- vincoli di uguaglianza
- vincoli \geq
- variabili negative
- variabili non vincolate

Terminologie da sapere:

- **soluzione**: quella di output
- **soluzione ammissibile**: soluzione, se esiste, che **soddisfa tutti i vincoli**
- **soluzione inammissibile**: se **viola almeno un vincolo**
- **regione ammissibile**: tutti i punti che rispettano i vincoli
- **prob inammissibile**: **regione ammissibile vuota**
- **prob ammissibile**:
 - soluzione ottima singola
 - soluzioni multiple
 - fo illimitata

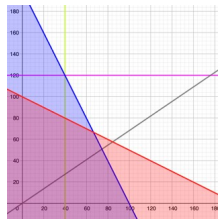


Figura 6. Rappresentazione grafica esempio

H. Implementazione in Python

```
1 import pulp as p
2
3 # 1. creazione del modello
4 model = p.LpProblem("ProductMix", p.LpMaximize)
5
6 # 2. definisco le variabili decisionali
7 x_A = p.LpVariable("x_A", cat="Continuous", lowBound=0)
8 x_B = p.LpVariable("x_B", cat="LpContinuous", lowBound=0)
9
10 # 3. definisco la funzione obiettivo in funzione
    delle variabili decisionali
11 model += 15 * x_A + 10 * x_B
12
13 # 4. definire i vincoli
14 model += 4 * x_A + 2 * x_B <= 400
15 model += 2 * x_A + 4 * x_B <= 400
16 model += x_A <= 40
17 model += x_B <= 120
18
19 # 5. risolvere il problema
20 model.solve()
21
22 # print della soluzione
23 print("next week produce {} pallets of A".format(x_A
    .varValue))
24 print("next week produce {} pallets of B".format(x_B
    .varValue))
```

I. Esercitazione

1) Massimizzare la f.o. $z = 8x_1 + 6x_2$, con i vincoli:

- $x_1 \leq 5$
- $x_2 \leq 7$
- $4x_1 + 3x_2 \leq 29$
- $x_1, x_2 \geq 0$

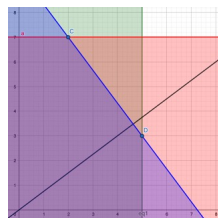


Figura 7. Rappresentazione grafica esempio 1

$$\nabla z = \begin{bmatrix} \frac{dz}{dx_A} \\ \frac{dz}{dx_B} \end{bmatrix} = \begin{bmatrix} 8 \\ 6 \end{bmatrix}$$

Abbiamo che esiste una curva di livello coincidente con lo spigolo CD, quindi abbiamo delle soluzioni ottime multiple:

- vertice C, prendiamo allora vincolo 2 e 3:

$$x_2 = 7 \rightarrow x_1 = 2$$

- vertice D, prendiamo allora vincolo 1 e 3:

$$x_1 = 5 \rightarrow x_2 = 3$$

- punti del segmento CD

Quindi $z = 58$

2) Minimizziamo la f.o. $z = 25x_1 + 22x_2$, con i vincoli:

- $x_1 + x_2 \geq 5$
- $3x_1 + 2x_2 \geq 12$
- $3x_1 + 6x_2 \geq 18$
- $x_1, x_2 \geq 0$

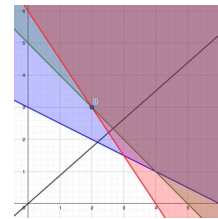


Figura 8. Rappresentazione grafica esempio 2

$$\nabla z = \begin{bmatrix} \frac{dz}{dx_A} \\ \frac{dz}{dx_B} \end{bmatrix} = \begin{bmatrix} 25 \\ 22 \end{bmatrix}$$

Per poter minimizzare, tracciando la curva di livello, trovando che la soluzione ottima si troverà dal punto B dato dall'intersezione dei vincoli 1 e 2:

$$x_1 = 2, x_2 = 3$$

Quindi $z = 116$

3) Massimizziamo la f.o. $z = 2x_1 + x_2$, con i vincoli:

- $x_1 - x_2 \leq 1$
- $2x_1 + x_2 \geq 6$
- $x_2 \geq 6$
- $x_1, x_2 \geq 0$

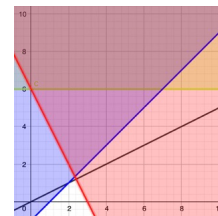


Figura 9. Rappresentazione grafica esempio 3

$$\nabla z = \begin{bmatrix} \frac{dz}{dx_A} \\ \frac{dz}{dx_B} \end{bmatrix} = \begin{bmatrix} 2 \\ 1 \end{bmatrix}$$

Non raggiungeremo la regione ammissibile, quindi il problema non ammette una soluzione ottima.

4) Minimizziamo la f.o. $z = -2x_1 + 3x_2$, con i vincoli:

- $x_1 - 2x_2 \geq -2$
- $2x_1 - x_2 \leq 3$
- $x_2 \geq 4$
- $x_1, x_2 \geq 0$

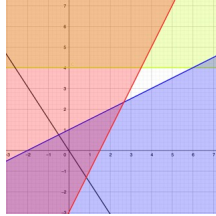


Figura 10. Rappresentazione grafica esempio 4

$$\nabla z = \begin{bmatrix} \frac{dz}{dx_A} \\ \frac{dz}{dx_B} \end{bmatrix} = \begin{bmatrix} -2 \\ 3 \end{bmatrix}$$

La regione ammissibile è vuota e per tanto il problema è inammissibile, quindi non esiste un punto che soddisfa contemporaneamente tutti i vincoli.

5) L'azienda vuole decidere oltre al piano di produzione anche la giusta riallocazione degli addetti (10-10) tra i due reparti. Le variabili decisionali sono:

- x_A, x_B : i prodotti
- n_p : # addetti allocati al reparto produzione
- n_a : #addetti allocati al reparto assemblaggio

Quindi andremo ad aggiungere il vincolo per cui $n_p + n_a = 20$.

Perciò avremo: $z = 15x_A + 10x_B$, con vincoli:

- $4x_A + 2x_B \leq 40n_p$
- $2x_A + 4x_B \leq 40n_a$
- $x_A \leq 40$
- $x_B \leq 120$
- $n_a + n_p = 20$
- $x_A, x_B, n_a, n_p \geq 0$

IV. FORMULAZIONI EQUIVALENTI DI UN PROBLEMA DI PROGRAMMAZIONE LINEARE

A. Problema in FORMA GENERALE

In generale nella zona ammissibile diciamo:

$$X = x \in R^n : A_x \geq b, D_x = l, x_j \geq 0 \quad (*)$$

$$\forall j \in J \subseteq \{1, 2, \dots, n\}$$

dove definiamo la possibilità di vincoli di $\geq, =$ e variabili ≥ 0 .

La funzione obiettivo è definita da:

$$z = c_x \quad t.c. \quad z = \min z, x \in X$$

che rappresenta il problema espresso in forma generale.

B. Problema in FORMA CANONICA

Se in (*) abbiamo:

- $D = 0$
- $J = \{1, 2, \dots, n\}$

allora il problema si dice in forma canonica:

$$\min_x \{z = c_x : A_x \geq b, x \geq 0\}$$

C. Problema in FORMA STANDARD

Se in (*) abbiamo:

- $A = 0$
- $J = 1, 2, \dots, n$

allora il problema è:

$$\min_x \{z = c_x : D_x = l, x \geq 0\}$$

Possiamo sempre ricondurci tramite trasformazioni alla forma standard.

D. Terminologia

Nella forma standard abbiamo che:

- **z**: la funzione obiettivo per la quale trovare il valore minimo
- **A**: matrice dei vincoli
- **D**: matrice dei coefficienti, matrice di dimensione $m \times n$
- **l**: vettore dei termini noti vettore colonna
- **c**: detto vettore dei coefficienti di costo, vettore di riga

E. Trasformazioni per ricondursi alla forma standard

1) variabili non vincolate di segno:

$$x_j t.c. j \notin J$$

per trasformarla possiamo sostituire a x_j la somma algebrica di 2 variabili non negative:

$$x_j = x_j^+ - x_j^-$$

$$\forall x_j^+ \geq 0, x_j^- \geq 0$$

Se abbiamo k ($\leq n$) variabili non vincolate in segno, possiamo evitare di introdurre k coppie di variabili non negative. È possibile considerare una variabile $x_0 \geq 0$ e sostituire la generica variabile non vincolata di segno con $x_j = x_j^+ - x_0$. **introduciamo "solo" $k + 1$ variabili.**

- 2) **vincoli non espressi in forma di uguaglianza (\leq)**

$$\sum_{j=i}^n a_{ij}x_j \leq b_i$$

presa la **variabile di Slack**: $S_i \geq 0$, avremo:

$$\sum_{j=i}^n a_{ij}x_j + S_i = b_i$$

questa variabile misura lo Slack che esiste per far sì che **il vincolo sia rispettato per uguaglianza** o no (vincolo > 0)

- 3) **vincoli non espressi in forma di uguaglianza ($>$)**

$$\sum_{j=i}^n a_{ij}x_j > b_i$$

presa una **variabile di Surplus**: $S_i \geq 0$, avremo:

$$\sum_{j=i}^n a_{ij}x_j - S_i = b_i$$

- 4) **trasformazione di vincoli da uguaglianza in disuguaglianza sostituendo:**

$$\sum_{j=i}^n a_{ij}x_j = b_i$$

sostituendolo a:

$$\sum_{j=i}^n a_{ij}x_j \geq b_i$$

$$\sum_{j=i}^n a_{ij}x_j \leq b_i$$

- 5) **funzione obiettivo:**

Se la f.o. è $\max z = c_x$, si può trasformare:

$$\max z = -\min -z$$

F. Esempio 1

- 1) DATI

f.o.: $\min z = x_1 + 2x_2$

vincoli:

- $6x_1 + 4x_2 \leq 24$
- $4x_1 + 8x_2 \leq 32$
- $x_2 \geq 3$
- $x_1, x_2 \geq 0$

- 2) TRASFORMAZIONE IN FORMA STANDARD

Per il primo vincolo **del tipo \leq , aggiungiamo una variabile non negativa (slack):**

$$6x_1 + 4x_2 + x_3 = 24$$

per il secondo vincolo operiamo in maniera analoga :

$$4x_1 + 8x_2 + x_4 = 32$$

per il terzo vincolo **del tipo \geq , aggiungiamo una variabile ausiliaria negativa:**

$$x_2 - x_5 = 3$$

vincolo sulle variabili:

$$x_1, x_2, x_3, x_4, x_5 \geq 0$$

quindi nella formulazione standard avremo $j = 1, 2, 3, 4, 5$ quando in precedenza avevamo: $j = 1, 2$

G. Esempio 2

- 1) DATI

f.o.: $\max z = z_1 + z_2$

vincoli:

- $8x_1 + 6x_2 \geq 48$
- $5x_1 + 10x_2 \geq 50$
- $13x_1 + 10x_2 \leq 130$
- $x_1 \geq 0$

- 2) TRASFORMAZIONE IN FORMA STANDARD

Dato che non abbiamo vincoli su x_2 poniamo:

$$x_2 = x_2^+ - x_2^-$$

la f.o. sarà:

$$\max z = -\min -z = -x_1 - x_2 = -x_1 - x_2^+ + x_2^-$$

invece i vincoli:

- $8x_1 + 6x_2^+ - 6x_2^- - x_3 = 48$
- $5x_1 + 10x_2^+ - 10x_2^- - x_4 = 50$
- $13x_1 + 10x_2^+ - 10x_2^- + x_5 = 130$
- $x_1, x_2^+, x_2^-, x_3, x_4, x_5 \geq 0$

V. OPTIMIZATION MODELS REVIEW

A. Scheduling

Nell'ambito dei problemi dello scheduling abbiamo degli elementi ben specificati:

- task/job già assegnati
- n macchine/processori
- potremmo attrezzare le macchine con dei tools

Intendiamo allocare i tasks alla macchine in "overtime" quindi capire anche la fascia temporale nella quale eseguire il task. Potrebbe esserci un unico tempo di esecuzione oppure un task può avere dei tempi di esecuzione differenti su macchine differenti.

L'output sarà un diagramma di Ganth.

I task possono avere degli istanti di rilascio dove non potrebbe essere rilasciato dopo un certo istante di tempo (ready time).

Possono esserci delle relazioni di precedenza tra i tasks. Quindi non posso effettuare un task se prima non ho concluso l'altro.

Il diagramma mi dice nel tempo a che macchina è associato quale task ed in quali intervalli di tempo e con quale tool.

B. Project scheduling

Per progetto intendiamo un insieme di tasks che sono realizzati al fine di raggiungere un goal. La caratteristica di un progetto è che nel complesso le attività non sono mai state eseguite in precedenza.

Le caratteristiche di un progetto sono:

- durata delle attività che nota
- ha a capo un Project Manager: responsabile del progetto e dei tempi di realizzazione, costi di produzione, ecc...

in genere in project manager usano dei metodi per tener sotto controllo il progetto (es: microsoft project) per esempio degli applicativi che suppongono i p.m. nella schedulazione e rischedulazione dei progetti

il più semplice modello di modellazione:

un progetto è rappresentato da diverse attività:
expected duration

Attività	Durata stimata d_i	Predecessori
1	10	-
2	10	-
3	10	1
4	10	1, 2

Tabella IV

TABELLA DELLE ORE DI LAVORO

questi problemi vengono rappresentati tramite diagramma aciclico:

rappresentazione activity on node (AoN):

dove abbiamo degli "archi" che rappresentano le predecessioni

(la tabella mi viene fornita dal project manager dopo aver parlato con degli esperti)

a volte si introducono dei vertici fittizi che sono lo start e l'end

start: lo colleghiamo tutte le attività che non hanno predecessori
end: ci colleghiamo tutte le attività finali

vedremo come il p.m. può usare delle risorse per accelerare delle attività

prima cosa: variabili decisionali: nel nostro caso è lo start time. ipotizziamo che il progetto inizi al tempo $t = 0$, quindi per ogni task abbiamo che $s_i \geq 0 \forall i \in TASKS$ dove s_i è lo start time del task che appartiene a $TASKS$
in più possiamo definire $T_i = 0$ che è il tempo di completamento del progetto (completion time)

in questo modello vogliamo minimizzare il completion time:
 $\min z = T$

cioè: $z = 1T + 0s_1 + 0s_2 + \dots$ su cui mi serve

per quanto riguarda le relazioni di precedenza:

$p_{ij} = (sistema) 1 \Leftrightarrow i \text{ è predecessore di } j, 0 \text{ altrimenti}$
con p_{ij} una matrice costata binaria:

$p =$

0 0 1 1 0 0 0 1 0 0 0 0 0 0 0 0

VINCOLI DI PRECEDENZA:

T è un maggiorante del tempo di completamento delle task, quindi:

$s_i + d_i \leq T$

$p_j(s_i + d_i) \leq s_j \forall i \in TASKS, j \in TASKS$ se $p_{ij} = 1$ allora i è predecessore di j quindi il tempo di inizio del task j deve essere d_i

se invece i non è il predecessore: $p_{ij} = 0$ quindi avremo che $0 \leq s_j$ allora il vincolo è ridondante dato che è un dato che già abbiamo

questo vale per un generico numero di task, mi serve però un linguaggio di modellazione per scrivere il modello in forma compatta per avere dati e modello divisi

per semplificare l'ultimo vincolo possiamo scrivere $s_i + d_i \leq s_j \forall i \in TASKS, j \in TASKS, p_{ij} \geq 0$

esempio modello espanso per problemi di istanza con i dati di sopra:

$\min z = T$

$s_1 + 10 \leq T$
 $s_2 + 10 \leq T$
 $s_3 + 10 \leq T$
 $s_4 + 10 \leq T$

$s_1 + 10 \leq s_3 (p_{13} = 1)$
 $s_1 + 10 \leq s_4 (p_{14} = 1)$
 $s_2 + 10 \leq s_4 (p_{24} = 1)$

$s_1, s_2, s_3, s_4 \geq 0$
 $T \geq 0$