

ACHILLE
CANNAVALE

APPUNTI
RICERCA OPERATIVA
2023

CIAO! QUESTI APPUNTI SONO
FRUTTO DEL MIO STUDIO E
DELLA MIA INTERPRETAZIONE,
QUINDI POTREBBERO
CONTENERE ERRORI, SVISTE O
COSE MIGLIORABILI. BUONO
STUDIO 

IMPORTANTE

*Ciao! Questi appunti sono frutto
del mio studio e della mia
interpretazione, quindi potrebbero
contenere errori, sviste o cose
migliorabili. Buono studio* 

Ricerca Operativa

Riassunto da

Achille Cannavale

Indice

1 Programmazione Lineare	3
1 Funzioni Lineari	3
2 Disuguaglianze Lineari	3
3 Programmazione Lineare	4
4 Regione Ammissibile	4
5 Risolvere un problema LP Graficamente	4
5.1 Esempio	5
6 Binding Constraint	5
7 Combinazione Convessa	6
8 Insieme Convesso	6
9 Funzioni Convesse	7
10 Punto Estremale	7
2 Metodo del Simplex (06)	8
1 LP in Forma Standard (04A)	8
2 BFS (04B)	9
2.1 Esempio	9
3 BFS Adiacenti (04BI)	11
3.1 Esempio	11
4 Metodo Del Simplex (Grafico) (04C)	12
5 Metodo del Tableau (04D)	12
6 Metodo della Grande M (04E)	17
7 Unrestricted-in-Sign (04F)	19
8 Problema Duale (05B)	20
3 Problema del Trasporto (07)	22
1 Formulazione del Problema del Trasporto (06A)	22
2 Metodo di Vogel (06D)	23
2.1 Esempio	24
3 Transporting Loop and Pivoting (07A)	26
3.1 Esempio	27
4 Metodo del Simplex nel Problema del Trasporto (07B)	28
4.1 Esempio	28
5 Transshipment Problem (07C)	32
5.1 Esempio	33
4 Problema di Assegnazione e del Commesso Viaggiatore (09)	35
1 Problema di Assegnazione e Hungarian Method (07D)	35
1.1 Esempio	35
2 Problema del Commesso Viaggiatore (09E)	39
3 Problema del Commesso Viaggiatore con l'Hungarian Method	40
3.1 Esempio	40

5 Grafi (08)	45
1 Shortes Path in un Grafo Orientato con Dijkstra (08C)	45
2 Minimum Spanning Tree (08E)	45
3 Maximum Flow Problem(08F)	46
4 Metodo di Ford-Fulkerson (08G)	47
4.1 Esempio	47
6 Catene di Markov	49
1 Esempio preso dalla traccia del 12/01/2023	49
1.1 Casi Particolari	51

Capitolo 1

Programmazione Lineare

1 Funzioni Lineari

Definition: Funzioni Lineari

La funzione $f(x_1, x_2, \dots, x_n)$ si dice **funzione lineare** se e solo se essa può essere scritta come:

$$f(x_1, x_2, \dots, x_n) = c_1x_1 + c_2x_2 + \dots + c_nx_n$$

con c_1, c_2, \dots, c_n sono **costanti**.

2 Disuguaglianze Lineari

Definition: Disuguaglianze Lineari

Definiamo **disuguaglianze lineari** per una **funzione lineare** $f(x_1, x_2, \dots, x_n)$ e un qualsiasi numero **costante** b :

$$\begin{cases} f(x_1, x_2, \dots, x_n) \geq b \\ f(x_1, x_2, \dots, x_n) \leq b \end{cases}$$

3 Programmazione Lineare

Definition: Programmazione Lineare

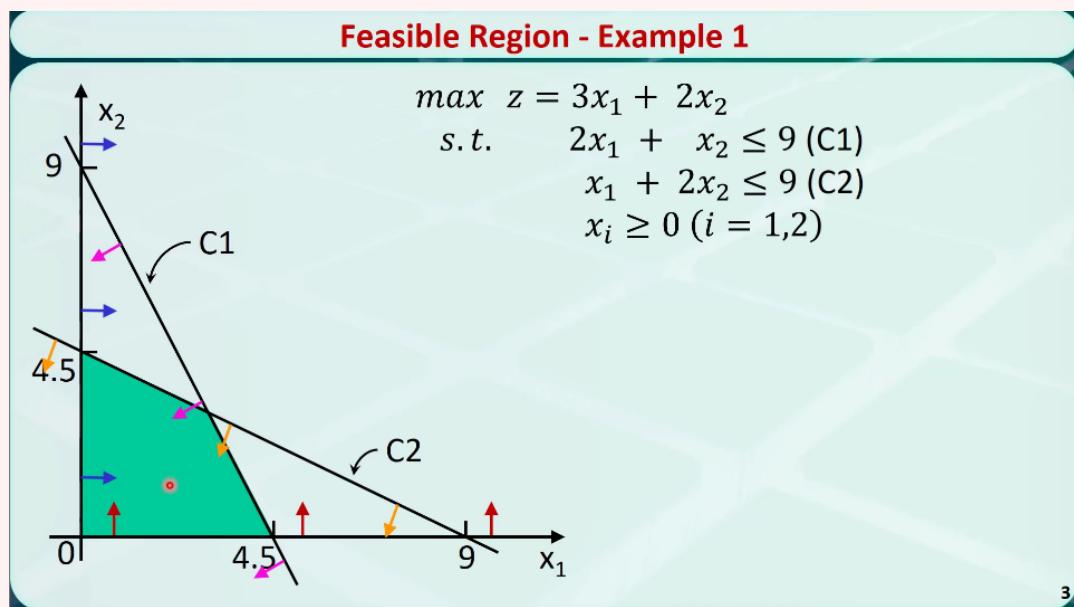
Un **problema di programmazione lineare** è un problema di **ottimizzazione** che:

- ha una **funzione obiettivo lineare**
- ogni **vincolo** deve essere un'**equazione** o una **disegualanza lineare**
- ogni **variabile** ha un **vincolo sul segno**

4 Regione Ammissibile

Definition: Regione Ammissibile

Definiamo **Regione Ammissibile** la regione che soddisfa i vincoli lineari.



5 Risolvere un problema LP Graficamente

Il metodo grafico per la risoluzione di un problema di programmazione lineare è molto semplice e si divide nei seguenti steps:

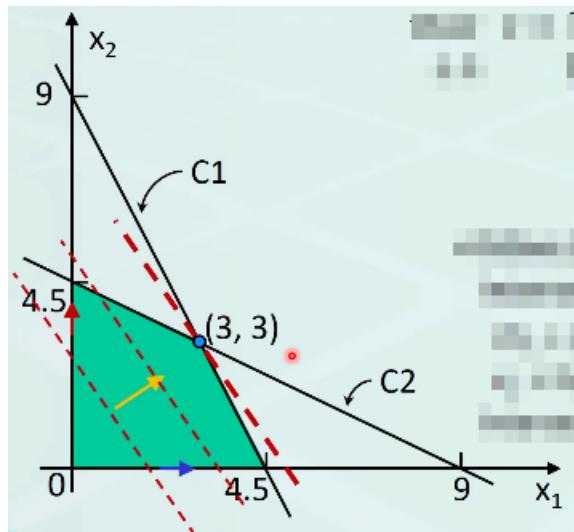
- Trovare la regione ammissibile
- Trovare la retta tangente s della funzione obiettivo

- Disegnare una linea la retta s in modo tale che intersechi la regione ammissibile
- Muovere la retta parallelamente alla direzione in cui la funzione obiettivo aumenta¹
- L'ultimo punto di intersezione, prima di uscire dalla regione ammissibile sarà la/le soluzione/i ottimale/i

5.1 Esempio

Facciamo un esempio con il seguente problema:

$$\begin{cases} \max z = 3x_1 + 2x_2 \\ 2x_1 + x_2 \leq 9 \\ x_1 + 2x_2 \leq 9 \\ x_i \geq 0 \forall i \end{cases}$$



Nel caso in cui la retta s uscendo dalla regione ammissibile, non intersechi un solo punto, ma un'intera retta, vorrà dire che tutti i punti di quella retta sono soluzione al problema.

6 Binding Constraint

Definition: Binding Constraint

Un **vincolo** si dice **binding** se, sostituendo la soluzione ottimale nel vincolo, esso diventa un'**uguaglianza**, altrimenti vengono detti **non binding**.

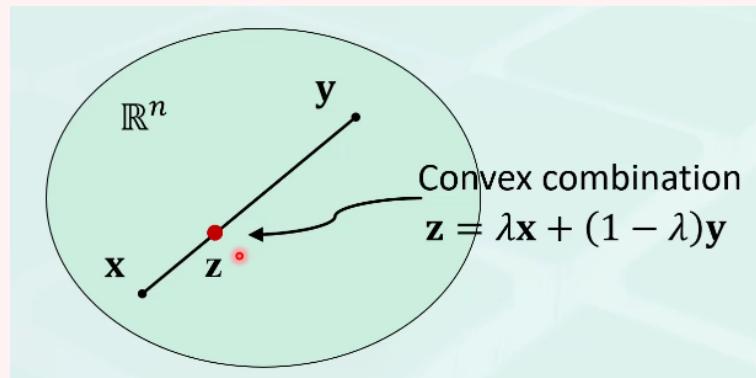
¹Per i problemi di minimo sarà il contrario

7 Combinazione Convessa

Definition: Combinazione Convessa

Dati due punti $x, y \in \mathbf{R}^n$, una loro **combinazione convessa** è un qualsiasi punto della retta:

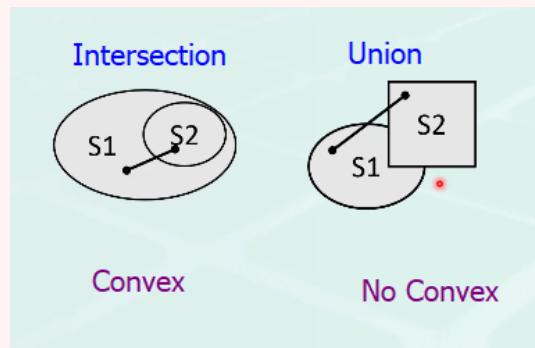
$$z = \lambda x + (1 - \lambda)y, \lambda \in [0, 1]$$



8 Insieme Convesso

Definition: Insieme Convesso

Un insieme $S \subseteq \mathbf{R}^n$ si dice **insieme convesso** se contiene tutte le **combinazioni convesse** di qualsiasi due punti in se stesso.



9 Funzioni Convesse

Definition: Funzioni Convesse

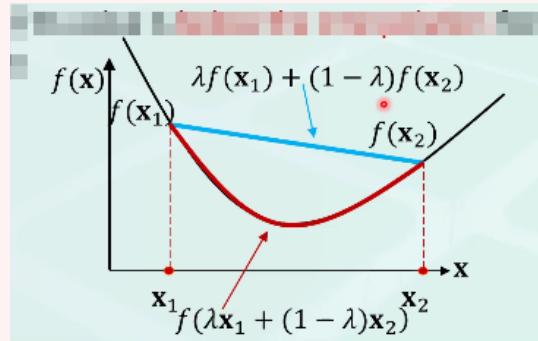
Sia S un **insieme convesso**. La funzione:

$$f(x) : S \longrightarrow \mathbf{R}$$

è una funzione convessa se per ogni due punti x_1 e x_2 in S abbiamo che:

$$f(\lambda x_1 + (1 - \lambda)x_2) \leq \lambda f(x_1) + (1 - \lambda)f(x_2), \quad \lambda \in [0, 1]$$

Graficamente possiamo dire che $f(x)$ è convessa se il suo valore si trova sotto la retta che congiunge qualsiasi coppia di punti:^a



^aOvviamente per le funzioni non convesse la funzione starà sopra

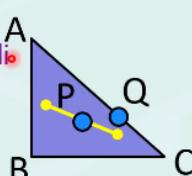
10 Punto Estremale

Definition: Punto Estremale

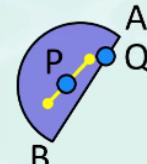
Un punto P è detto **estremale** se e solo se:

- P si trova in un **insieme convesso** S
- Per ogni segmento che contiene P ed è contenuto interamente in S , P deve essere un **punto finale**

Qui i punti estremali sono A, B e C



Qui i punti estremali sono A, B e tutti i punti sul semicerchio



Capitolo 2

Metodo del Simplex (06)

1 LP in Forma Standard (04A)

L'algoritmo del simplex può essere usato per risolvere un **problema di programmazione lineare**.

Per fare ciò, esso deve essere convertito in una **forma standard**:

- Tutti i **vincoli** devono essere **equazioni**
- Tutte le **variabili decisionali** devono essere **non-negative**

$\begin{array}{ll} \max & z = 3x_1 + 2x_2 \\ \text{s.t.} & 2x_1 + x_2 \leq 9 \\ & x_1 + 2x_2 \leq 9 \\ & x_i \geq 0 \ (i=1,2) \end{array}$	$\begin{array}{ll} \max & z = 3x_1 + 2x_2 \\ \text{s.t.} & 2x_1 + x_2 \geq 9 \\ & x_1 + 2x_2 \geq 9 \\ & x_i \geq 0 \ (i=1,2) \end{array}$
$\begin{array}{ll} \max & z = 3x_1 + 2x_2 \\ \text{s.t.} & 2x_1 + x_2 + s_1 = 9 \\ & x_1 + 2x_2 + s_2 = 9 \\ & x_i \geq 0 \ (i=1,2); s_i \geq 0 \ (i=1,2) \end{array}$ <p style="text-align: center;">Slack Variables</p>	$\begin{array}{ll} \max & z = 3x_1 + 2x_2 \\ \text{s.t.} & 2x_1 + x_2 - e_1 = 9 \\ & x_1 + 2x_2 - e_2 = 9 \\ & x_i \geq 0 \ (i=1,2); e_i \geq 0 \ (i=1,2) \end{array}$ <p style="text-align: center;">Excess Variables</p>
$\begin{array}{ll} \max & z = 3x_1 + 2x_2 \\ \text{s.t.} & 2x_1 + x_2 \leq 9 \\ & x_1 + 2x_2 \geq 9 \\ & x_i \geq 0 \ (i=1,2) \end{array}$	$\begin{array}{ll} \max & z = 3x_1 + 2x_2 \\ \text{s.t.} & 2x_1 + x_2 + s_1 = 9 \\ & x_1 + 2x_2 - e_2 = 9 \\ & x_i \geq 0 \ (i=1,2); s_1 \geq 0; e_2 \geq 0 \end{array}$ <p style="text-align: center;">Slack and Excess</p>

Quindi, possiamo generalizzare il tutto scrivendo un LP in forma standard come:

$$\left\{ \begin{array}{l} \max \quad z = c_1x_1 + c_2x_2 + \dots + c_nx_n \\ a_{11}x_1 + a_{12}x_2 + \dots + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + \dots + a_{2n}x_n = b_2 \\ \dots \\ a_{m1}x_1 + a_{m2}x_2 + \dots + a_{mn}x_n = b_m \\ x_j \geq 0, \ j = 1, \dots, n \end{array} \right.$$

Questa forma standard può essere riscritta sotto forma di matrice:

$$\left\{ \begin{array}{l} \max z = \mathbf{c}^T \mathbf{x} \\ \mathbf{Ax} = \mathbf{b} \\ \mathbf{x} \geq \mathbf{0} \end{array} \right.$$

2 BFS (04B)

Consideriamo un **problema LP** nella forma **standard**, con:

- ***n* variabili**
- ***m* vincoli**

Una **soluzione basica** si può ottenere:

- ponendo $n - m$ variabili uguali a 0 (**NBV**)
- risolvendo le rimanenti m variabili (**BV**)

Il numero totale delle **possibili soluzioni** lo si calcola nel seguente modo:

$$\binom{n}{m} = \binom{n}{n-m} = \frac{n!}{m!(n-m)!}$$

2.1 Esempio

$$\left\{ \begin{array}{l} \max z = 3x_1 + 2x_2 \\ 2x_1 + x_2 \leq 9 \\ x_1 + 2x_2 \leq 9 \\ x_i \geq 0 \forall i \end{array} \right.$$

Trasformiamola in **forma standard**:

$$\left\{ \begin{array}{l} \max z = 3x_1 + 2x_2 \\ 2x_1 + x_2 + s_1 = 9 \\ x_1 + 2x_2 + s_2 = 9 \\ x_i \geq 0 \forall i \\ s_i \geq 0 \forall i \end{array} \right.$$

In questo esempio abbiamo:

$$\left\{ \begin{array}{l} n = 4 \\ m = 2 \\ n - m = 2 NBVs \\ m = 2 BVs \end{array} \right.$$

Quindi utilizzando la formula vista prima, otteniamo:

$$\binom{n}{m} = \binom{n}{n-m} = \frac{n!}{m!(n-m)!} = \frac{24}{4} = 6$$

Ovvero 6 possibili soluzioni, che possiamo suddividere in **BVs** (negative) e **NBVs** (Non-Negative):

Basic Solution		BFS?
NBV	BV	
$x_1=x_2=0$	$s_1=9, s_2=9$	Y
$x_1=s_1=0$	$x_2=9, s_2=-9$	N
$x_1=s_2=0$	$x_2=4.5, s_1=4.5$	Y
$x_2=s_1=0$	$x_1=4.5, s_2=4.5$	Y
$x_2=s_2=0$	$x_1=9, s_1=-9$	N
$s_1=s_2=0$	$x_1=3, x_2=3$	Y

Theorem 1

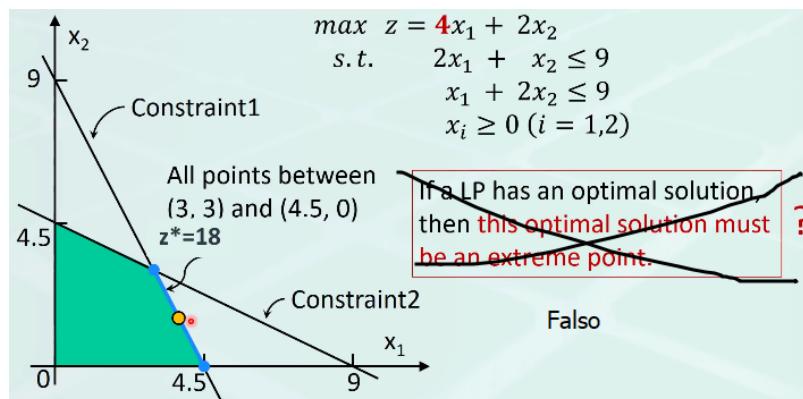
- La **regione ammissibile** di un qualsiasi LP è un **insieme convesso**^a
- La **BFS** è un **Punto Estremale**
- Qualsiasi LP ha un **numero finito** di BFSs
- Se un LP ha una **soluzione ottimale**, ci sarà un **punto estremale ottimo**

^aDato che ogni vincolo lineare divide lo spazio in due, e ogni spazio generato sarà convesso, e l'intersezione di insiemi spazi convessi è anch'esso convesso

Ragioniamo sull'ultimo enunciato, esso dice che **se un LP ha una soluzione ottimale, deve esistere un punto estremale ottimo**.

Ma questo **non** vuol dire che la soluzione ottimale deve coincidere per forza con il **punto estremale!!**

Ecco qui un esempio che mostra un caso in cui esistono **infinite soluzioni ottimali**, tra cui anche **punti non estremali**:



3 BFS Adiacenti (04BI)

Definition: BFS Adiacenti

Per qualsiasi LP con m vincoli, due BFSs si dicono **adiacenti** se hanno $m - 1$ **variabili basiche in comune**.

3.1 Esempio

	Basic Solution		BFS? (x_1, x_2)
	NBV	BV	
1	$x_1 = x_2 = 0$	$s_1 = 9, s_2 = 9$	Y $(0, 0)$
2	$x_1 = s_1 = 0$	$x_2 = 9, s_2 = -9$	N $(0, 9)$
3	$x_1 = s_2 = 0$	$x_2 = 4.5, s_1 = 4.5$	Y $(0, 4.5)$
4	$x_2 = s_1 = 0$	$x_1 = 4.5, s_2 = 4.5$	Y $(4.5, 0)$
5	$x_2 = s_2 = 0$	$x_1 = 9, s_1 = -9$	N $(9, 0)$
6	$s_1 = s_2 = 0$	$x_1 = 3, x_2 = 3$	Y $(3, 3)$

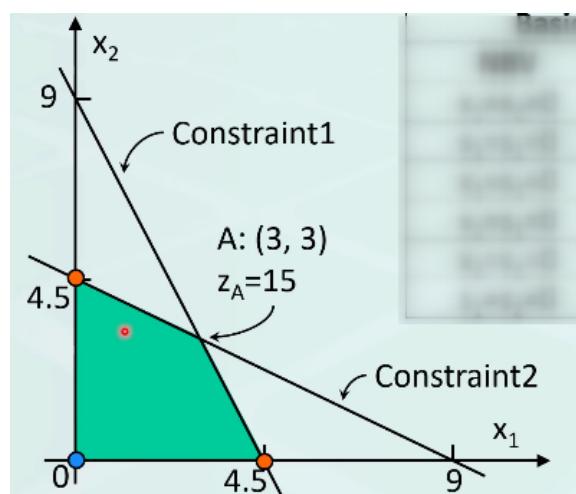
In questo esempio abbiamo:

$$\begin{cases} m = 2 \text{ eqs} \\ m - 1 = 1 \text{ common BV} \end{cases}$$

Cerchiamo le BFSs adiacenti alla prima BFS:

- La **seconda** non è una BFS, quindi la scartiamo
- La **terza** ha s_1 in comune, quindi è adiacente
- La **quarta** ha s_2 in comune, quindi è adiacente
- La **quinta** non è una BFS, quindi la scartiamo
- La **sesta** non ha nessuna BV in comune, quindi la scartiamo

L'**adiacenza** è molto semplice da vedere anche **graficamente**:



4 Metodo Del Simplesso (Grafico) (04C)

La risoluzione di un **LP** con il metodo del **simplesso** può essere fatto seguendo i seguenti steps:

Recipe

1. Disegna la **regione ammissibile**
2. Trova un **punto estremale (BFS)** e calcola il valore della sua z
3. Verifica i valori di s dei **punti adiacenti** al corrente **punto estremale**
4. Se nessun punto adiacente ha un valore di z **superiore** al corrente, allora il **punto estremale corrente** è una **soluzione ottimale**. Altrimenti, spostati sul **punto adiacente** con il valore di z **più grande** e ritorna allo **step 3**

5 Metodo del Tableau (04D)

Recipe

1. Converti il problema **LP** in forma **canonica** e mettilo in un **Tableau**
2. Scegli una **BFS** iniziale dal **tableau**
3. Determina se la **BFS** corrente è **ottimale**, ovvero se tutte le **NBVs** hanno coefficienti **non-negativi** nella riga **R0**
4. Se non accade, determina una **variabile entrante**, ovvero quella con il **coefficiente più negativo** in **R0** e, utilizzando il **Ratio Test**, determina su quale riga fare il **pivot**
5. Usa le **manipolazione** per fare il **pivoting** e ricavare una nuova **BFS**, e riparti dal **punto 2**.

Proviamo a risolvere il seguente problema utilizzando il metodo del **tableau**:

$$\left\{ \begin{array}{ll} \max & z = 3x_1 + 2x_2 \\ & 2x_1 + x_2 \leq 9 \\ & x_1 + 2x_2 \leq 9 \\ & x_i \geq 0 \forall i \end{array} \right.$$

Trasformiamola nella **forma standard** aggiungendo, in questo caso, due variabili di **slack**:

$$\left\{ \begin{array}{ll} \max & z = 3x_1 + 2x_2 \\ & 2x_1 + x_2 + s_1 = 9 \\ & x_1 + 2x_2 + s_2 = 9 \\ & x_i \geq 0 \forall i \\ & s_i \geq 0 \forall i \end{array} \right.$$

Ora riscriviamo la **funzione obiettivo** nel seguente modo:

$$z - 3x_1 - 2x_2 = 0$$

Così facendo, ora possiamo costruire il **tableau** inserendo tutti i **coefficienti** delle variabili:

	z	x_1	x_2	s_1	s_2	RHS
R0	1	-3	-2			0
R1		2	1	1		9
R2		1	2		1	9

Sfruttando questa visualizzazione possiamo ridefinire:

- **BV**: è una variabile che ha un **coefficiente unitario** in una **singola equazione** e **zero** in tutte le **altre**.
- **NBV**: sono tutte le altre variabili.

Nel caso in questione possiamo subito notare che le **BV** sono s_1 e s_2 . Per la precisione, una variabile **slack**, può essere usata come **BV** solo se il suo **RHS** (membro a destra) è **non negativo**, altrimenti violerebbe il vincolo per il quale $s_i \geq 0$. In questo caso abbiamo ad entrambi 9 come **RHS**, quindi vanno bene.

A questo punto dobbiamo domandarci se la funzione obiettivo **z può essere aumentata incrementando una NBV partendo**

da 0 come valore iniziale, lasciando le altre NBVs a 0.

Per rispondere a questa domanda ci basta studiare la funzione obiettivo:

$$z = 3x_1 + 2x_2$$

Noteremo così che se incrementassimo la **NBV** x_1 , lasciando x_2 ferma a 0, la funzione obiettivo **incrementerebbe maggiormente** rispetto al caso in cui incrementassimo la **NBV** x_2 lasciando x_1 ferma a zero, grazie al coefficiente maggiore che moltiplica x_1 .

Tutto questo può essere anche visto dal **tableau**, dove si andrà a scegliere la **NBV** con il coefficiente **più negativo**.

Questa **NBV** scelta prende il nome di **entering variable**.

Tuttavia dobbiamo mantenere tutte le **BVs non-negative!** Perchè abbiamo il **vincolo**:

$$s_i \geq 0 \forall i$$

Quindi dobbiamo determinare il **valore più grande** al quale possiamo spingere x_1 e questo può essere fatto effettuando il **ratio testo**:¹

$$\frac{\text{rhs della riga vincolante}}{\text{coefficiente della entering var nella riga vincolante}}$$

Il vincolo con il **ratio più piccolo** è detto **vincitore del ratio test**.

	z	x_1	x_2	s_1	s_2	RHS	Ratio
R0	1	-3	-2			0	
R1		2	1	1		9	9/2
R2		1	2		1	9	9/1

Ma matematicamente che significa?

$$R1: 2x_1 + s_1 = 9$$

Esplicitiamo s_1 e ricordiamo che deve essere positivo:

$$s_1 = 9 - 2x_1 \geq 0$$

Esplicitiamo ora x_1 :

$$x_1 \leq \frac{9}{2}$$

¹Per righe vincolanti intendo R1 e R2

Quindi questo significa che x_1 deve rimanere più piccolo di $\frac{9}{2}$ per far rimanere s_1 una **BV**.

Possiamo fare lo stesso procedimento con il secondo vincolo ottenendo:

$$x_1 \leq 9$$

Tra le due dobbiamo prendere quella più **stringente**, ovvero:

$$x_1 \leq 9/2$$

A questo punto dobbiamo far diventare la entering variable una BV nella riga che ha vinto il ratio test.

E questo può essere fatto con alcune **manipolazioni**. Questa procedura è chiamata **pivoting** e **R1** è detta **riga di pivot**.

Ricordiamo che il nostro obiettivo è trasformare x_1 in una **BV**, quindi deve avere un **coefficiente unitario** sulla riga che ha vinto il **ratio test** e nulla su tutte le altre righe.

Per prima cosa poniamo $R1' = R1/2$:

	Z	x_1	x_2	s_1	s_2	RHS
R0	1	-3	-2			0
R1'		1	1/2	1/2		9/2
R2		1	2		1	9

Ora sommiamo $R0' = R0 + 3R1'$:

	Z	x_1	x_2	s_1	s_2	RHS
R0'	1		-1/2	3/2		27/2
R1'		1	1/2	1/2		9/2
R2		1	2		1	9

E infine poniamo $R2' = R2 - R1'$:

	Z	x_1	x_2	s_1	s_2	RHS
R0'	1		-1/2	3/2		27/2
R1'		1	1/2	1/2		9/2
R2'			3/2	-1/2	1	9/2

Una volta finito notiamo che x_1 è diventata una **BV**, mentre s_2 è diventata una **NBV**, e viene detta **leaving variable**.

A questo punto ricontrolliamo la **funzione obiettivo**:

$$z = \frac{27}{2} + x_2 \frac{1}{2} - s_1 \frac{3}{2}$$

Domandiamoci nuovamente: possiamo aumentare la **funzione obiettivo** z incrementando una **NBV** lasciando l'altra a zero?

In questo caso ancora sì, possiamo usare x_2 come **entering variable**, quindi procediamo come già fatto in precedenza:

Svolgiamo il **ratio test**:

	Z	x_1	x_2	s_1	s_2	RHS	Ratio
R0	1		-1/2	3/2		27/2	
R1		1	1/2	1/2		9/2	9
R2			3/2	-1/2	1	9/2	3

Quindi il **ratio test** lo vince:

$$x_2 \leq 3$$

Facciamo ora il **pivoting** sulla riga R2:

$$R2' = 2R2/3$$

	Z	x_1	x_2	s_1	s_2	RHS
R0	1		-1/2	3/2		27/2
R1		1	1/2	1/2		9/2
R2'			1	-1/3	2/3	3

$$R0' = R0 + R2'/2:$$

	Z	x_1	x_2	s_1	s_2	RHS
R0'	1			4/3	1/3	15
R1		1	1/2	1/2		9/2
R2'			1	-1/3	2/3	3

$$R1' = R1 - R2'/2:$$

	Z	x_1	x_2	s_1	s_2	RHS
R0'	1			4/3	1/3	15
R1'		1		2/3	-1/3	3
R2'			1	-1/3	2/3	3

Una volta arrivati a questo punto controlliamo se la **funzione obiettivo** possa aumentare:

$$z = 15 - s_1 \frac{4}{3} - s_2 \frac{1}{3}$$

Non possiamo più aumentare la **funzione obiettivo**, questo si può vedere anche dal **tableau**, che si dice **ottimo** se ogni **NBV** ha un **coefficiente non-negativo** nella riga **R0**.

6 Metodo della Grande M (04E)

Come abbiamo visto, il metodo del **Simplex** ha bisogno di una **BFS iniziale**, che abbiamo trovato nei precedenti problemi usando le variabili di **slack** come **BV**.

Ma questo non funziona se un **LP** ha dei vincoli con $= 0 \geq$, e da qui nasce l'esigenza di utilizzare il Metodo della Grande M.

Recipe

1. Modifica gli **RHS** in modo tale che **non siano negativi**
2. Converti il **LP** nella **forma Standard**
3. Aggiungi, per ogni $= e \geq$ una **variabile artificiale** con **segno positivo**
4. Definisci M come un **valore positivo grandissimo** e per ogni **variabile artificiale**, aggiungi Ma_i nella **R0**
5. **Elimina** tutte le **variabili artificiali** da **R0** usando le manipolazioni e risolvi il problema con il **metodo del simplex**
6. Se alla fine tutte le **variabili artificiali** sono **nulle**, la **soluzione** sarà **ottima**, altrimenti il **problema non è risolvibile**

Vediamo ora questo metodo applicato ad un esempio per comprenderlo appieno:

$$\left\{ \begin{array}{l} \max z = 3x_1 + 2x_2 \\ 2x_1 + x_2 \leq 9 \\ x_1 + 2x_2 \geq 9 \\ x_i \geq 0 \forall i \end{array} \right.$$

Il primo passo di questo metodo sta nel **modificare i vincoli** in modo tale che ogni **RHS** sia **non negativo**. In questo caso lo sono di già.

Il secondo passo è quello di convertire il **LP** nella forma **standard**:

$$\left\{ \begin{array}{l} R0 : z - 3x_1 - 2x_2 = 0 \\ R1 : 2x_1 + x_2 + s_1 = 9 \\ R2 : x_1 + 2x_2 - e_2 = 9 \\ x_i \geq 0 s_i \geq 0 e_i \geq 0 \forall i \end{array} \right.$$

Il terzo passo è quello di aggiungere, per ogni vincolo con $= 0 \geq$, una **variabile artificiale** con **segno positivo**:

$$\left\{ \begin{array}{l} R0 : z - 3x_1 - 2x_2 = 0 \\ R1 : 2x_1 + x_2 + s_1 = 9 \\ R2 : x_1 + 2x_2 - e_2 + a_2 = 9 \\ x_i \geq 0 s_i \geq 0 e_i \geq 0 a_i \geq 0 \forall i \end{array} \right.$$

Il quarto passo consiste nel definire con M un numero **positivo grandissimo** e per ogni **variabile artificiale**, aggiungere Ma_i nella **R0**:

$$\left\{ \begin{array}{l} R0 : z - 3x_1 - 2x_2 + Ma_2 = 0 \\ R1 : 2x_1 + x_2 + s_1 = 9 \\ R2 : x_1 + 2x_2 - e_2 + a_2 = 9 \\ x_i \geq 0 s_i \geq 0 e_i \geq 0 a_i \geq 0 \forall i \end{array} \right.$$

Infine il **quinto passo** consiste nell'**eliminare tutte le variabili artificiali** da **R0** usando le **manipolazioni** e risolvere il problema infine con il metodo del **simplesso**.

Alla fine se tutte le **variabili artificiali** nella **soluzione finale** saranno uguali a zero, la soluzione in questione sarà **ottima**.

Mentre se qualsiasi **variabile artificiale** risulti **positiva**, il problema **non è risolvibile**.

Questo è il **tableau** del nostro problema:

	z	x_1	x_2	s_1	e_2	a_2	RHS
R0	1	-3	-2			M	0
R1		2	1	1			9
R2		1	2		-1	1	9

Eliminiamo la variabile artificiale da R0 con la manipolazione $R0' = R0 - MR2$:

	z	x_1	x_2	s_1	e_2	a_2	RHS
R0'	1	-3-M	-2-2M		M		-9M
R1		2	1	1			9
R2		1	2		-1	1	9

E da questo punto in poi utilizzeremo il metodo del **simplesso**, quindi **calcolo del ratio, vincitore del ratio, variabile entrante, variabile uscente**, etc.

Arriveremo alla fine al seguente **tableau**:

	z	x_1	x_2	s_1	e_2	a_2	RHS
R0	1	1		2		M	18
R1		3		2	1	-1	9
R2		2	1	1		0	9

In questo caso la soluzione esiste dato che a_2 non è una **BV**, quindi va messa a 0.

La soluzione **ottima** sarà:

$$\begin{cases} x_1 = 0 \\ x_2 = 9 \\ s_1 = 0 \\ e_2 = 9 \\ a_2 = 0 \\ z^* = 18 \end{cases}$$

7 Unrestricted-in-Sign (04F)

Nel caso in cui, una variabile fosse **URS**, dovremmo comportarci in una maniera diversa, come possiamo vedere nel seguente esempio:

$$\left\{ \begin{array}{l} \max z = 3x_1 + 2x_2 \\ 2x_1 + x_2 \leq 9 \\ x_1 + 2x_2 \leq 9 \\ x_1 \geq 0, x_2 \text{ URS} \end{array} \right.$$

$$\left\{ \begin{array}{l} R0 : z - 3x_1 - 2x_2 = 0 \\ R1 : 2x_1 + x_2 + s_1 = 9 \\ R2 : x_1 + 2x_2 + s_2 = 9 \\ x_1, s_1, s_2 \geq 0, x_2 \text{ URS} \end{array} \right.$$

A questo punto, **splittiamo** la variabile x_2 in $x_2^+ - x_2^-$:

$$\left\{ \begin{array}{l} R0 : z - 3x_1 - 2x_2^+ + 2x_2^- = 0 \\ R1 : 2x_1 + x_2^+ - x_2^- + s_1 = 9 \\ R2 : x_1 + 2x_2^+ - 2x_2^- + s_2 = 9 \\ x_1, s_1, s_2 \geq 0, x_2^+, x_2^- \text{ URS} \end{array} \right.$$

E da questo punto in avanti si può continuare attraverso il metodo del **simplesso**.

8 Problema Duale (05B)

Per ogni **Linear Programming Problem "Primale"**, ne esiste uno che si chiama **duale**.

Per ricavare il problema duale esaminiamo il seguente esempio:

$$\left\{ \begin{array}{l} \max z = 30x_1 + 100x_2 \\ x_1 + x_2 \leq 7 \\ 4x_1 + 10x_2 \leq 40 \\ x_1 \geq 3 \\ x_1 \geq 0 \\ x_2 \geq 0 \end{array} \right.$$

Cerchiamo adesso di calcolare un **lower-bound** e un **upper-bound**.

Per quanto riguarda un **lower-bound**, esso può essere trovato scegliendo una qualsiasi **BFS**:

$$EX : (x_1, x_2) = (5, 2) \longrightarrow z = 350 \leq z^*$$

Per quanto riguarda il calcolo di un **upper-bound**, il procedimento è più complesso:

1. Moltiplica il vincolo i-esimo di un fattore u_i scegliendo il suo segno in modo tale da far diventare \leq tutte le **disuguaglianze**
2. Fa sì che i **coefficienti** dei vincoli risultanti siano gli stessi della **funzione obiettivo**. Così facendo, gli **RHS** dei **vincoli** risultanti saranno un **upper-bound** di z^*

$$\left\{ \begin{array}{l} \textcolor{red}{u_1}x_1 + \textcolor{red}{u_1}x_2 \leq 7\textcolor{red}{u_1} \\ 4\textcolor{red}{u_2}x_1 + 10\textcolor{red}{u_2}x_2 \leq 40\textcolor{red}{u_2} \\ -\textcolor{red}{u_3}x_1 \leq -3\textcolor{red}{u_3} \\ -\textcolor{red}{u_4}x_1 \leq -0 \\ -\textcolor{red}{u_5}x_2 \leq -0 \end{array} \right.$$

Adesso, leggendo in **colonna** otteniamo:

$$(u_1 + 4u_2 - u_3 - u_4)x_1 + (u_1 + 10u_2 - u_5)x_2 \leq 7u_1 + 40u_2 - 3u_3$$

A questo punto dobbiamo sostituire tutte le u_i in modo tale da ugualare i coefficienti della **funzione obiettivo**, quindi, il problema **duale** assume la seguente forma:

$$\left\{ \begin{array}{l} \min w = 7u_1 + 40u_2 - 3u_3 \\ u_1 + 4u_2 - u_3 - u_4 = 30 \\ u_1 + 10u_2 - u_5 = 100 \\ u_1, u_2, u_3, u_4, u_5 \geq 0 \end{array} \right.$$

Capitolo 3

Problema del Trasporto (07)

1 Formulazione del Problema del Trasporto (06A)

Un tipo di problema di **programmazione lineare** che è molto più simile a ciò che può accadere nella realtà è detto **Problema del Trasporto**.

Per risolvere questo tipo di problema è possibile utilizzare il **metodo del simplex**, ma in una nuova veste, più specifica per il problema del **trasporto**.

L'obiettivo di questo problema è quello di trovare la **combinazione migliore** di quantità di merci da trasportare, per **minimizzare** i costi.

Ecco un esempio di **Tableau di Trasporto**, dove sulle intestazioni delle righe ho le **Factories**, mentre sulle intestazioni delle colonne ho i **Clienti**:

	C1	C2	C3	
F1	8	6	10	
F2	x_{11}	x_{12}	x_{13}	
	9	12	13	
	x_{21}	x_{22}	x_{23}	
$d_1 = 30$	$d_2 = 30$	$d_3 = 30$		
				$s_1 = 40$
				$s_2 = 50$

Nei piccoli quadratini ci sono i costi che comportano trasferire un'unità di **supply** dalla **Factory** a quel determinato cliente. (es. Il numero nella celletta [1,1] indica che trasportare un'unità di merce del fornitore 1 fino al cliente 1 costa 8).

	C1	C2	C3	
F1	x ₁₁ 8	x ₁₂ 6	x ₁₃ 10	s ₁ = 40
F2	x ₂₁ 9	x ₂₂ 12	x ₂₃ 13	s ₂ = 50
Dummy F3	x ₃₁ 20	x ₃₂ 22	x ₃₃ 23	s ₃ = 10
	d ₁ = 30	d ₂ = 30	d ₃ = 30	

	C1	C2	C3	Dummy C4	
F1	x ₁₁ 8	x ₁₂ 6	x ₁₃ 10	x ₁₄ 0	s ₁ = 40
F2	x ₂₁ 9	x ₂₂ 12	x ₂₃ 13	x ₂₄ 0	s ₂ = 50
	d ₁ = 30	d ₂ = 30	d ₃ = 30	d ₄ = 10	

Mentre nelle celle più grandi ci sono le quantità che effettivamente dovranno essere trasportate.

Nel caso in cui, non riuscissimo a soddisfare tutta la domanda dei clienti, dovremo inserire un cosiddetto "**Dummy Factory**", che bilancerà la domanda e l'offerta: Viceversa, se ci trovassimo di fronte ad un'eccessiva produzione, dovremo inserire un "**Dummy Client**":

2 Metodo di Vogel (06D)

Uno tra i migliori metodi di **risoluzione dei problemi di Trasporto** è il **Metodo di Vogel** che si compone delle seguenti fasi:

Recipe

1. Per ogni **riga** (colonna), individua una **penalità** sottraendo il **costo unitario** più piccolo nella **riga** (colonna) dal secondo più piccolo nella stessa **riga** (colonna).
2. Trova la **riga** (colonna) con la **penalità** più **grande** e trova la **cella** con il **costo unitario** più piccolo in quella **riga** (colonna). Alloca tutto il possibile di quella cella e sistema i valori di **supply** e **domanda**, sottraendo la quantità allocata.
3. Elimina la **riga** (colonna) con 0 **supply** o **domanda**. Se sia una **riga** che una **colonna** hanno 0, sceglie una da **eliminare** arbitrariamente.
4. Se **una e una sola cella** è rimasta, **eliminala** e fermati, altrimenti ritorna al **primo step**.

2.1 Esempio

Prendiamo ad esempio questo **tableau**, in cui sono state calcolate le **penalità** su tutte le **righe** e su tutte le **colonne**:

				Row Penalty
				$s_1=10 \quad 7-6=1$
				$s_2=15 \quad 78-15=63$
		$d_1=15$	$d_2=5$	$d_3=5$
		Column Penalty	15-6=9	80-7=73
				78-8=70

La **penalità** più **alta** la troviamo sulla **seconda colonna**, quindi la selezioniamo e scegliamo la **cella** con il **costo più basso**, ovvero la **cella** con costo 7.

Assegniamo quindi tutto quello che possiamo, in questo caso 5 di s_1 , facendo diventare 0 d_2 , quindi possiamo **eliminare** la colonna e aggiornare i valori:

			Row Penalty		
			s ₁ =5		7-6=1
			s ₂ =15		78-15=63
	6				
	5				
	15		80		78
d ₁ =15			d ₃ =5		
Column Penalty	15-6=9		78-8=70		

Di nuovo, scegliamo la **penalità più alta**, in questo caso quella sulla **terza colonna**, e scegliamo la cella con il **costo unitario più basso**, ovvero 8.

A questo punto possiamo assegnare tutto ciò che possiamo di s_1 , in questo caso 5 unità, che manderanno a zero sia s_1 e sia d_3 , rendendo **eliminabili** sia la **riga 1** e sia la **colonna 3**:

			Row Penalty		
			s ₁ =0		
			s ₂ =15		
.	6				
	5		5		
	15		80		78
d ₁ =15					
Column Penalty	15-6=9				

Come **ultimo passaggio** non ci rimane che scegliere la **colonna 1** e la **cella con costo unitario 15**, dato che per quanto riguarda la **riga del costo unitario 6**, $s_1 = 0$.

			Row Penalty		
			s ₁ =0		
			s ₂ =15		
0	6				
	5		5		
	15		80		78
15	.				
Column Penalty					

A questo punto abbiamo risolto il problema, che avrà la seguente

soluzione:

$$\begin{cases} x_{11} = 0 \\ x_{12} = 5 \\ x_{13} = 5 \\ x_{21} = 15 \end{cases}$$

3 Transporting Loop and Pivoting (07A)

Nel **problema del trasporto**, è necessario utilizzare i **Loop** per poter trovare una **Entering Variable**, partendo dalla **BFS iniziale**.

Ma per prima cosa andiamo a definire cosa è un **Loop**:

Definition

Una sequenza ordinata di almeno 4 celle diverse è detto **Loop** se:

- Qualsiasi paio di celle consecutive si trovano sulla stessa riga o sulla stessa colonna
- Non più di 2 celle consecutive si trovano sulla stessa riga o colonna
- L'ultima cella si trova nella stessa riga o colonna della prima cella

A questo punto andiamo a vedere quali passi compongono il **Loop Pivoting**:

Recipe

1. Individua una **Entering Variable**
2. Trova l'unico **Loop** che coinvolge la **Entering Variable** e alcune **BVs**
3. Conta le celle nel **loop** (partendo da 0) e contrassegnale come "**dispari**" e "**pari**"
4. Trova la cella **dispari** con il valore **più piccolo**, che sarà la **Leaving Variable**
5. **Decrementa** ogni cella **dispari** e **incrementa** ogni cella **pari** del **loop**, della quantità della **Leaving Variable**

3.1 Esempio

Vediamo questo esempio, nel quale partiamo già con una **BFS** impostata nel **Tableau** e supponiamo di conoscere già l'**Entering Variable** dalla quale partire:

35								
10	20	20						
			10		30			

A questo punto dobbiamo trovare il **Loop**:

35								
10	20	20						
			10		30			

E ora occorre contrassegnare le celle con "dispari" e "pari":

35								
10	20	20						
			10		30			

La cella dispari con il valore più piccolo è la cella con 20, che sarà anche la **Leaving Variable**.

Quindi non ci resta che decrementare tutte le celle **dispari** di 20 e aumentare le celle **pari** di 20:

Abbiamo trovato così una **nuova BFS**.

4 Metodo del Simplex nel Problema del Trasporto (07B)

Recipe

1. **Bilancia** il problema
2. Usa il metodo di **Vogel** per trovare una **BFS** di partenza
3. Per ogni **BVs** setta $u_1 = 0$ e **aggiorna** tutte le altre variabili con $u_i + v_j = c_{ij}$
4. Per ogni **NBVs** se $w_{ij} = u_i + v_j - c_{ij} \leq 0$, allora la **BFS** attuale è **ottima**. Altrimenti, scegli la variabile con il w_{ij} più **positivo** come **Entering Variable**
5. Calcola una **nuova BFS** usando il **loop pivoting** e torna allo **step 3**

4.1 Esempio

Partiamo da questa **BFS** e introduciamo le variabili u_i e v_i :

	v_1	v_2	v_3	v_4
u_1	8	6	10	9
u_2	35			
u_3	9	12	13	7
	10	20	20	
	14	9	16	5
			10	30

Settiamo $u_1 = 0$ e utilizziamo la formula:

$$u_i + v_j = c_{ij}$$

per ricavarci tutte le altre variabili:

	$v_1=8$	$v_2=11$	$v_3=12$	$v_4=1$
$u_1=0$	8	6	10	9
$u_2=1$	35			
$u_3=4$	9	12	13	7
	10	20	20	
	14	9	16	5
			10	30

Ora, in tutte le **NBV**s, dobbiamo calcolare:

$$w_{ij} = u_i + v_j - c_{ij}$$

E inserirle nel **tableau**:

	$v_1=8$	$v_2=11$	$v_3=12$	$v_4=1$
$u_1=0$	8	6	10	9
$u_2=1$	35	=5	=2	=-8
$u_3=4$	9	12	13	7
	10	20	20	
	14	9	16	5
	=-2	=6	10	30

$w_{12} = u_1 + v_2 - c_{12} = 5$
 $w_{13} = u_1 + v_3 - c_{13} = 2$
 $w_{14} = u_1 + v_4 - c_{14} = -8$
 $w_{24} = u_2 + v_4 - c_{24} = -5$
 $w_{31} = u_3 + v_1 - c_{31} = -2$
 $w_{32} = u_3 + v_2 - c_{32} = 6$

A questo punto dobbiamo verificare se tutte le **NBV**s sono ≤ 0 , dato che se lo fossero, questa sarebbe una **soluzione ottimale**.

In questo caso abbiamo tre **NBVs positive**.

Allora identifichiamo come **Entering Variable** la **NBV più positiva**, nel nostro caso quella con il coefficiente 6 e ricaviamoci una **nuova BFS** utilizzando il **Loop Pivoting**:

	$v_1=8$	$v_2=11$	$v_3=12$	$v_4=1$	
$u_1=0$	8	6	10	9	
$u_2=1$	35				
$u_3=4$	9	12	13	7	
	10	20	20		
	14	9	16	5	
	Enter	10	30		
			•		

Otterremo così questa **nuova BFS**:

	$v_1=8$	$v_2=11$	$v_3=12$	$v_4=1$	
$u_1=0$	8	6	10	9	
$u_2=1$	35				
$u_3=4$	9	12	13	7	
	10	10	30		
	14	9	16	5	
	10		30		

A questo punto dobbiamo aggiornare i valori di u e v ottenendo il seguente **tableau** che ancora **non è ottimale**, in quanto c'è un coefficiente positivo nelle **NBV** (5):

	$v_1=8$	$v_2=11$	$v_3=12$	$v_4=7$	
$u_1=0$	8	6	10	9	
$u_2=1$	35	=5	=2	=-2	
$u_3=-2$	9	12	13	7	
	10	10	30	=1	
	14	9	16	5	
	=-8	10	=-6	30	

Allora si procederà nuovamente al **Loop Pivoting**, ottenendo una **nuova BFS**:

	$v_1=8$	$v_2=11$	$v_3=12$	$v_4=7$	
$u_1=0$	8	6	10	9	
$u_2=1$	25	10			
$u_3=-2$	9	12	13	7	
	20		30		
	14	9	16	5	
	10			30	

Che ancora una volta, dopo l'aggiornamento delle variabili u e v **non restituirà una soluzione ottimale**, in quanto è presente un **coefficiente positivo** sulle NBVs (2):

	$v_1=8$	$v_2=6$	$v_3=12$	$v_4=2$	
$u_1=0$	8	6	10	9	
$u_2=1$	25	10	=2	=-7	
$u_3=3$	9	12	13	7	
	=-5	30		=-4	
	20				
	14	9	16	5	
	=-3	10	=-1	30	

Procediamo così, un'ultima volta al **loop pivoting**, ottenendo una **nuova BFS** che ci restituirà, aggiornando le variabili v e u , una **solu-**

zione ottimale, in quanto tutte le NBVs avranno **coefficienti negativi**¹:

	$v_1=6$	$v_2=6$	$v_3=10$	$v_4=2$	
$u_1=0$	8	6	10	9	
$u_2=3$	-2	10	25	-7	
$u_3=3$	9	12	13	7	
45	=-3	5	=-2		
-5	14	9	16	5	
10		=-3	30		

La **soluzione** alla fine sarà:

$$\begin{cases} x_{12} = 10 \\ x_{13} = 25 \\ x_{21} = 45 \\ x_{23} = 5 \\ x_{32} = 10 \\ x_{34} = 30 \\ \text{All other } x_{ij} = 0 \end{cases}$$

5 Transshipment Problem (07C)

Il **Transshipment Problem** non è altro che un caso speciale del **problema del Trasporto**.

La differenza principale risiede nel fatto che in un **Trasshipment Problem** una **factory** può spedire merci per dei cosiddetti "**trasshipment points**" prima di arrivare al **cliente**.

Vediamo come un **Trasshipment Problem** può essere convertito in un **problema di trasporto**:

¹assafà

Recipe

1. Costruisci il **problema del trasporto originale**, ignorando tutti i **transshipment points**
2. Bilancia il **problema del trasporto originale**
3. Aggiungi una **riga** e una **colonna** per ogni **transshipment point**. Ogni **transshipment point** ha un valore di **supply** e un valore di **domanda**, entrambi uguali al **supply totale originale**

5.1 Esempio

Partiamo dal seguente esempio, dove abbiamo due **factories**, due **clienti** e due **transshipment points**.

Supply of Factory 1 ≤ 150, Supply of Factory 2 ≤ 200				
Demand of Customer 1 ≥ 130, Demand of Customer 2 ≥ 130				
	Transfer 1	Transfer 2	Customer 1	Customer 2
Factory 1	8	13	25	28
Factory 2	15	12	26	25
Transfer 1	0	6	16	17
Transfer 2	6	0	14	16

Costruiamo il **tableau** del **problema del trasporto** ignorando i **transshipment points**:

		Customer 1	Customer 2	
		1	2	
Factory 1		25	28	150
		26	25	
Factory 2				200
		130	130	

Dato che la somma dei **supplies** è maggiore della somma della domanda, per **bilanciare** il problema dobbiamo introdurre una **Dummy Demand**:

		Customer 1	Customer 2	Dummy	
Factory 1		25	28	0	150
		26	25	0	200
		130	130	90	

A questo punto inseriamo una **riga** e una **colonna** per ogni **trasshipment point**, ponendo sia il valore di **supply** e sia il valore di **demand** pari al totale del **supply** del problema originale:

	Transfer 1	Transfer 2	Customer 1	Customer 2	Dummy	
Factory 1	8	13	25	28	0	150
Factory 2	15	12	26	25	0	200
Transfer 1	0	6	16	17	0	350
Transfer 2	6	0	14	16	0	350
	350	350	130	130	90	

Ora si può risolvere questo problema con il **metodo del simplex** per problemi di trasporto.

Capitolo 4

Problema di Assegnazione e del Commesso Viaggiatore (09)

1 Problema di Assegnazione e Hungarian Method (07D)

Recipe

1.
 - Nella matrice $m \times m$ dei costi, trova il **costo minimo** per ogni **riga**.
 - **Sottrai** questo costo minimo ad ogni costo della **riga** in questione.
 - Nella **nuova matrice** trova il **costo minimo** su ogni **colonna**.
 - **Sottrai** ad ogni costo della colonna il minimo trovato.
2. Disegna il **numero minimo di linee** sulle **righe** o sulle **colonne**, per coprire **ogni 0**. Se il **numero delle linee** è m , una **soluzione ottimale** si trova tra gli **zeri coperti**. Se il **numero di linee** è **minore** di m vai allo **step 3**.
3.
 - Trova l'**elemento positivo** più **piccolo** che **non è coperto** dalle **linee** disegnate.
 - **Sottrai** questo numero da ogni elemento che **non è coperto** e **aggiungilo** da ogni elemento che è **coperto da 2 linee**.
 - **Ritorna** allo **step 2**.

1.1 Esempio

Partiamo dalla seguente tabella di costi e calcoliamo il **minimo** su ogni **riga**:

				Row Min
14	5	8	7	5
2	12	6	5	2
7	8	3	9	3
2	4	6	10	2

E **sottraiamo** ad ogni elemento il **minimo** di quella **riga**, ottenendo così:

9	0	3	2
0	10	4	3
4	5	0	6
0	2	4	8

A questo punto troviamo gli **elementi minimi** su ogni **colonna**:

	9	0	3	2
	0	10	4	3
	4	5	0	6
	0	2	4	8
Col Min	0	0	0	2

E sottraiamo ad ogni **elemento** il **minimo** di quella **colonna**, ottenendo così:

	9	0	3	2
	0	10	4	3
	4	5	0	6
	0	2	4	8
Col Min	0	0	0	2

Ora, **coprendo** tutti gli **zeri**, con il numero **minimo** di **linee**, notiamo che il **numero** di **linee** sarà:

$$3 \leq m = 4$$

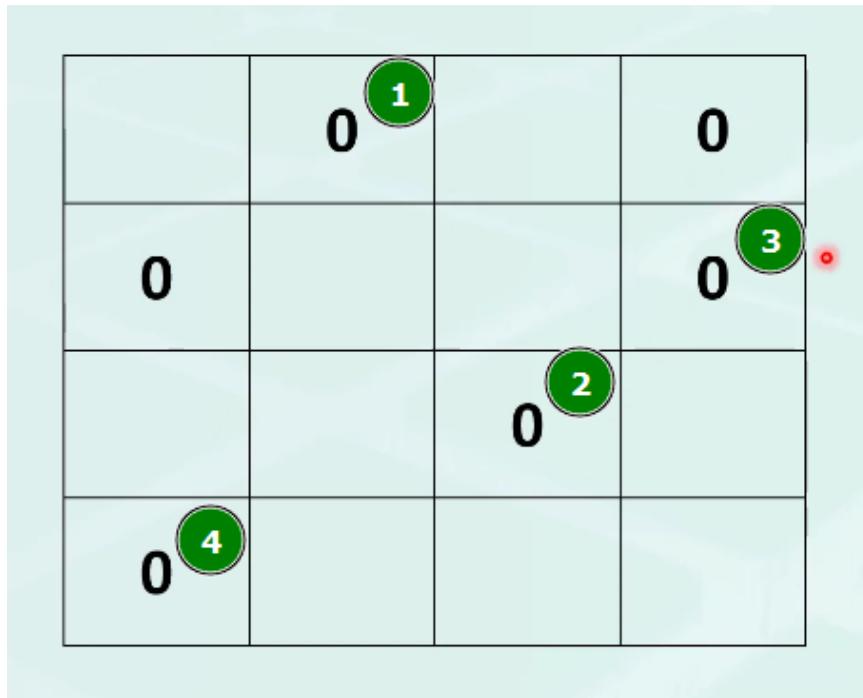
9	0	3	0
0	10	4	1
4	5	0	4
0	2	4	6

Allora dobbiamo trovare l'**elemento minimo non coperto**, che in questo caso è **1** e sottrarlo ad ogni **elemento scoperto** e sommarlo ad ogni **elemento coperto** da **2 linee**, ottenendo così una **nuova tabella**, i cui **zeri verranno coperti** da esattamente **4 linee**:

10	0	3	0
0	9	3	0
5	5	0	4
0	1	3	5

Arrivati qui, la **soluzione ottimale** sarà la **somma** dei **costi originali** nelle **celle** con **0** come elemento. Ma quali celle prendere in considerazione? Si procede **iterativamente** scegliendo una **cella** che **non abbia altri zeri** sulla stessa **riga** o sulla stessa **colonna** e la **successiva** in modo tale che non abbia un'altra cella **selezionata** sulla stessa **riga** o sulla stessa **colonna**, come il **sudoku ja**,

per capirci, cioè alla fine è pure divertente (quando esce):



Quindi nel nostro caso la **soluzione** è:

$$z^* = 5 + 5 + 2 + 3 = 15$$

2 Problema del Commesso Viaggiatore (09E)

Il **Problema del Commesso Viaggiatore** è un caso particolare del **problema del trasporto**. Cerchiamo di capire come risolverlo attraverso un piccolo esempio nella prossima sezione.

3 Problema del Commesso Viaggiatore con l'Hungarian Method

Recipe

1. Trova il minimo su ogni riga e sottrailo ad ogni elemento della sua riga.
2. Nella nuova matrice trova il minimo su ogni colonna e sottrailo come fatto prima.
3. Calcola le penalità di ogni zero sommando il minimo valore sulla riga corrente e il minimo sulla colonna corrente.
4. Scegli lo zero con la penalità più grande e includi l'arco dalla riga alla colonna al tour generale.
5. Elimina la riga e la colonna creando così una nuova matrice.
6. Fermati se non ci sono più righe o colonne, altrimenti torna al punto 1.

3.1 Esempio

Partiamo dalla seguente tabella dove abbiamo inserito le distanze tra ogni città:

	A	B	C	D	E
A	--	132	217	164	58
B	132	--	290	201	79
C	217	290	--	113	303
D	164	201	113	--	196
E	58	79	303	196	--

Calcoliamo il minimo sulle righe e sottraiamo ad ogni elemento:

	A	B	C	D	E
A	--	74	159	106	0
B	53	--	211	122	0
C	104	177	--	0	190
D	51	88	0	--	83
E	0	21	245	138	--

A questo punto calcoliamo anche il minimo sulle colonne e sottraiamo:

	A	B	C	D	E
A	--	53	159	106	0
B	53	--	211	122	0
C	104	156	--	0	190
D	51	67	0	--	83
E	0	0	245	138	--

Per ogni zero, andiamo a vedere il valore più piccolo che abbiamo sulla stessa riga e sulla stessa colonna e sommiamoli assieme:

	A	B	C	D	E
A	--	53	159	106	0(53)
B	53	--	211	122	0(53)
C	104	156	--	0(210)	190
D	51	67	0(210)	--	83
E	0(51)	0(53)	245	138	--

E adesso scegliamo uno dei due valori più alti che ci sono usciti, ovvero 210.

	A	B	C	D	E
A	--	53	159	106	0 (53)
B	53	--	211	122	0 (53)
C	104	156	--	0 (210)	190
D	51	67	0 (210)	--	83
E	0 (51)	0 (53)	245	138	--

$C \rightarrow D$
 $D \rightarrow C$ prohibited

In questo caso abbiamo scelto $C \rightarrow D$, quindi l'arco che andrà da $D \rightarrow C$ sarà "proibito".

A questo punto ricalcoliamo il minimo sulle righe e sottraiamo, ottenendo:

	A	B	C	D	E
A	--	53	159		0
B	53	--	211		0
C				$C \rightarrow D$	
D	0	16	--		32
E	0	0	245		--

E stessa cosa vale sulle colonne:

	A	B	C	D	E
A	--	53	0		0
B	53	--	52		0
C				$C \rightarrow D$	
D	0	16	--		32
E	0	0	86		--

Calcoliamo le penalità e scegliamo la più grande, in questo caso il 52 in $A \rightarrow C$, proibendo $C \rightarrow A$:

	A	B	C	D	E
A	--	53	0(52)	0(0)	
B	53	--	52		0(52)
C				C→D	
D	0(16)	16			32
E	0(0)	0(16)	86		--

A→C •
C→A prohibited

Calcoliamo i minimi sulle righe e sottraiamo agli elementi delle righe e facciamo lo stesso procedimento, ottenendo:

	A	B	C	D	E
A			A→C		
B	53	--			0 •
C				C→D	
D	0	16			32
E	0	0			--

Calcoliamo le penalità e scegliamo la più grande:

	A	B	C	D	E
A			A→C		
B	53				0(85) •
C				C→D	
D	0(16)	16			32
E	0(0)	0(16)			--

B→E
E→B prohibited

Ripetiamo lo stesso procedimento iterativamente, mi so scocciato di fa le foto, e otteniamo:

	A	B	C	D	E
A			A→C		
B					B→E
C				C→D	
D		D→B*			
E	E→A				

A→C→D→B→E→A

E sommando le distanze delle celle rimaste, otterremo la Minima Distanza da percorrere:

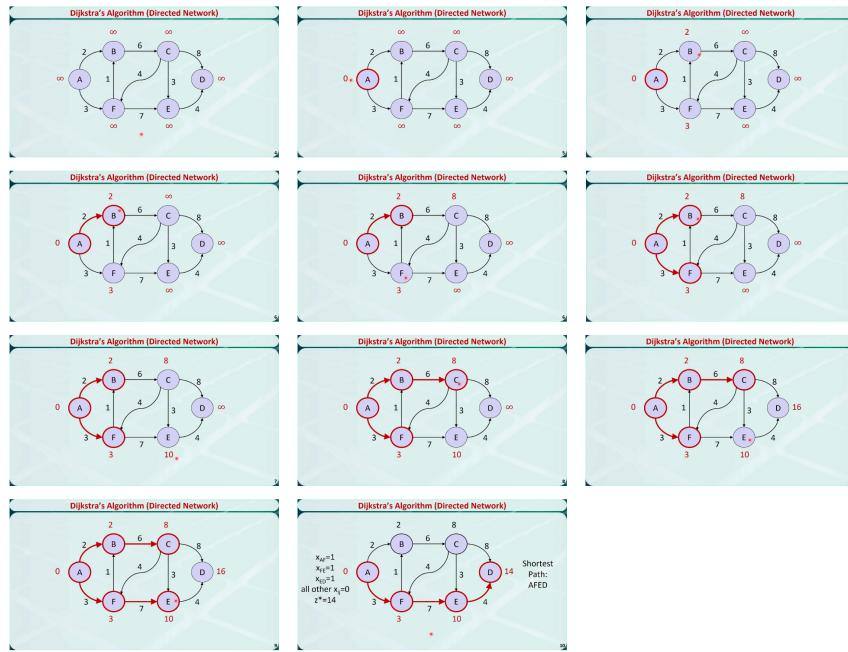
$$\text{Min distance : } 217 + 113 + 201 + 79 + 58 = 668$$

Capitolo 5

Grafi (08)

1 Shortes Path in un Grafo Orientato con Dijkstra (08C)

Supponendo che un'introduzione sui **Grafi** sia superflua¹, andiamo direttamente a descrivere la ricerca del percorso più piccolo all'interno di un **Grafo Orientato** mediante l'utilizzo dell'**Algoritmo di Dijkstra**.



2 Minimum Spanning Tree (08E)

Definition

Per un Grafo con n nodi, uno **Spanning Tree** è un insieme di $n-1$ archi che connettono ogni nodo del grafo senza contenere **cicli**.

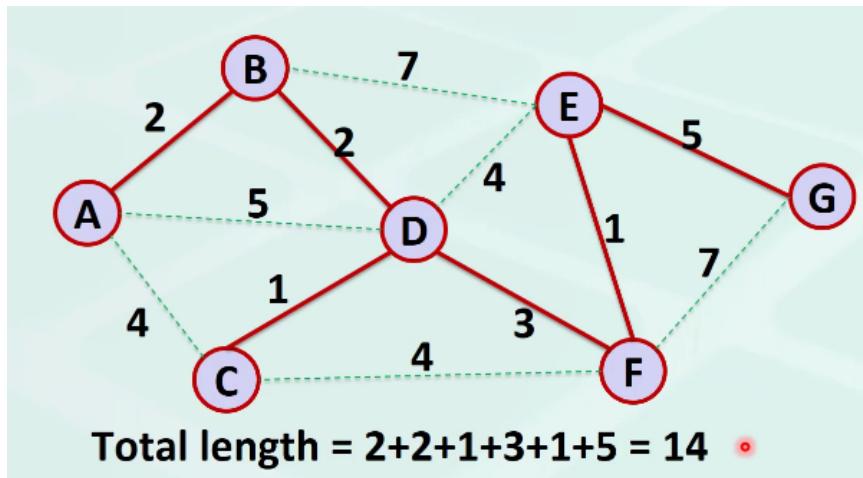
¹Vedere appunti di Bria.

Per trovare un **Minimum Spanning Tree**, dato un **grafo**, possiamo seguire la seguente procedura:

Recipe: Greedy Algorithm

1. Seleziona un qualsiasi **nodo** e connettilo con il **nodo** più vicino.
2. Identifica un **nodo non connesso** che sia il più vicino a qualsiasi **nodo connesso**, e connetti questi **due nodi**. Ripeti finché tutti i nodi non sono connessi.

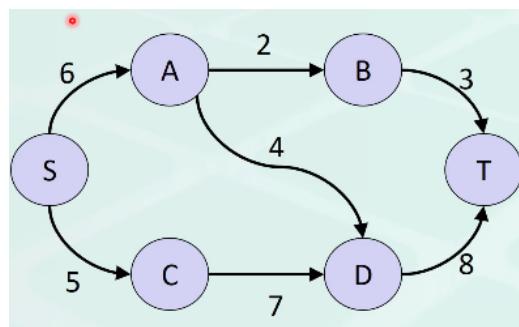
Ecco qui un esempio di **Minimum Spanning Tree**:



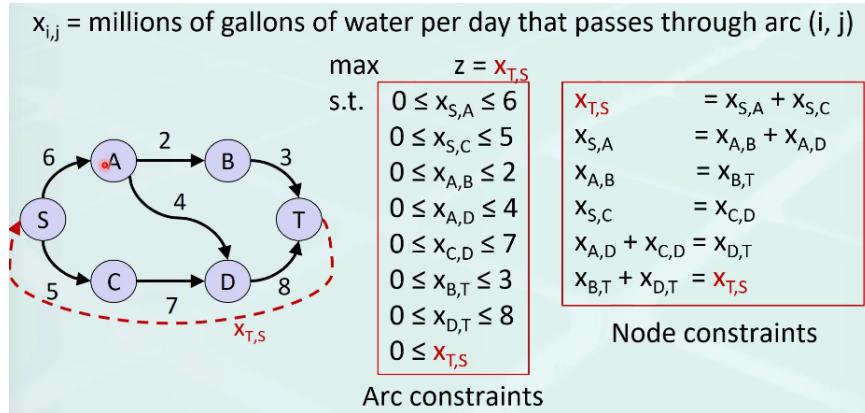
3 Maximum Flow Problem(08F)

Il valore che vediamo sugli archi in un grafo, rappresenta la capacità che limita la quantità che può passare per quell'arco. L'obiettivo è quello di trasportare la quantità massima da un punto iniziale a un punto finale.

Vediamo questo piccolo esempio per formalizzare meglio le caratteristiche di questo problema. Abbiamo questo grafo orientato con i seguenti pesi:



Per preservare il bilanciamento del grafo, inseriamo un arco artificiale con peso $X_{T,S}$ che va dal nodo T al nodo S.
A questo punto, denotando con $x_{i,j}$ il peso sull'arco che va dal nodo i al nodo j , possiamo ricavare dal problema una funzione da massimizzare e due insiemi di vincoli, quelli sugli archi e quelli sui nodi:



4 Metodo di Ford-Fulkerson (08G)

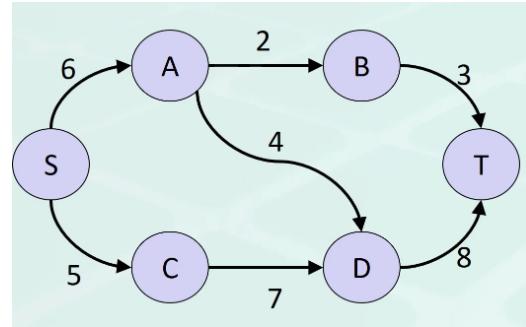
Questo metodo ha lo scopo di risolvere i Maximum Flow Problems.

Recipe

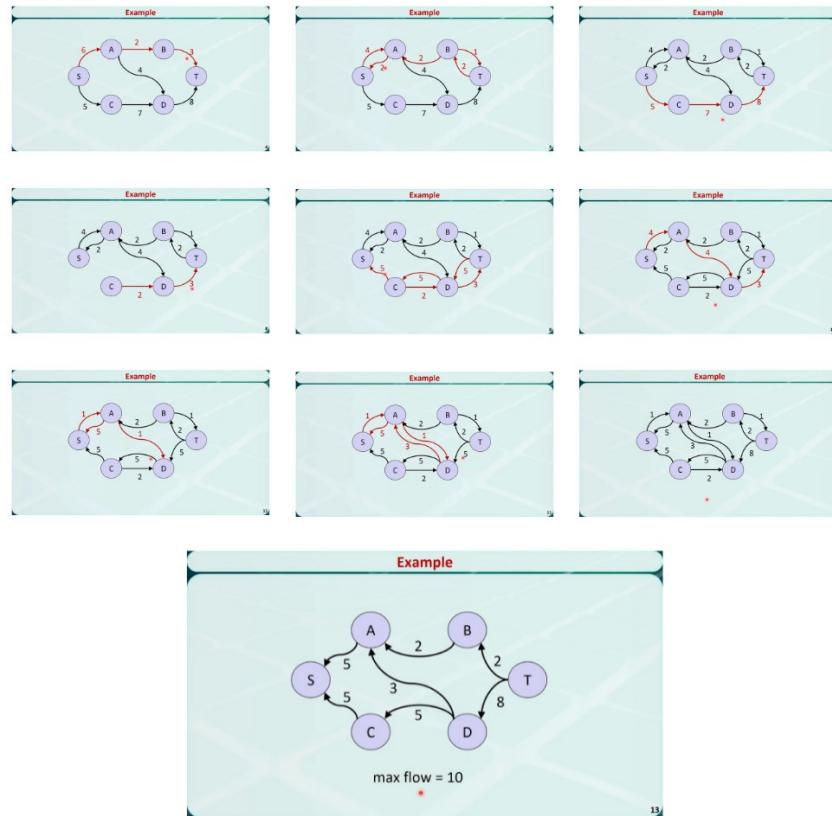
- Se esiste un percorso dalla sorgente alla destinazione, vai allo step 2, altrimenti la soluzione ottimale è determinata dall'arco precedente.
- Calcola il grafo rimanente:
 - Trova la capacità più piccola degli archi sul percorso. Indicala con c .
 - Sottrai c dalla capacità di ogni arco sul percorso.
 - Aggiungi un arco che fa il percorso opposto per ogni arco sul percorso. Aggiungi c alla capacità di ogni arco opposto.
 - Torna allo step 1.

4.1 Esempio

Abbiamo il seguente grafo, sul quale vogliamo massimizzare la quantità che va da S a T :



Questi sono gli step che ci portano alla soluzione del problema:



Capitolo 6

Catene di Markov

Definition

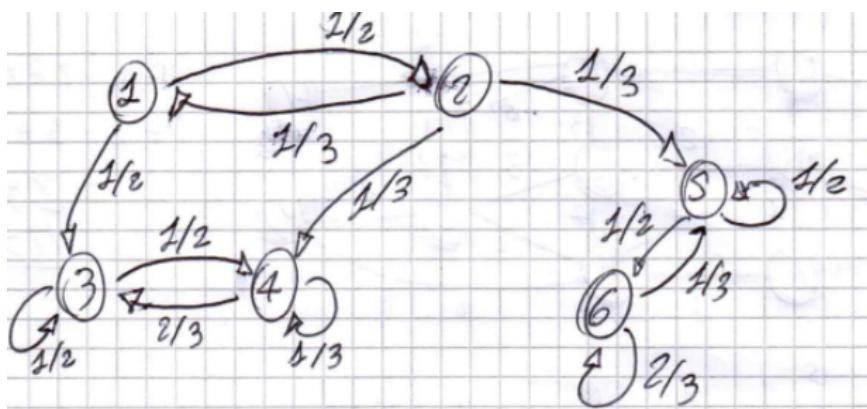
Un processo stocastico, a tempo discreto, è detto Catena di Markov se per $t = 0, 1, 2, \dots$ e per ogni stato, abbiamo che:

$$P(x_{t+1} = i_{t+1} | x_t = i_t, x_{t-1} = i_{t-1}, \dots, x_1 = i_1, x_0 = i_0) = \\ = P(x_{t+1} = i_{t+1} | x_t = i_t)$$

Questo significa che la probabilità della distribuzione di uno stato a **time t+1** dipende solo dallo stato al **tempo t** e non dipende dagli istanti precedenti.

1 Esempio preso dalla traccia del 12/01/2023

Per questo ultimo capitolo tralasciamo la maggior parte delle nozioni teoriche, affrontando direttamente un esercizio di un compito passato cercando di assimilare tutti i passi necessari al suo svolgimento con questo approccio pratico.



Il primo passo è quello di scrivere la matrice degli stati, ovvero una matrice dove, per ogni stato, sono presenti le probabilità di finire in un

altro stato:

$$P = \left[\begin{array}{cc|cccc} 0 & 1/2 & 1/2 & 0 & 0 & 0 \\ 1/3 & 0 & 0 & 1/3 & 1/3 & 0 \\ \hline 0 & 0 & 1/2 & 1/2 & 0 & 0 \\ 0 & 0 & 2/3 & 1/3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1/2 & 1/2 \\ 0 & 0 & 0 & 0 & 1/3 & 2/3 \end{array} \right]$$

Dal grafico possiamo capire che:

- Gli stati **1 e 2** sono stati **Transitori**, in quanto una volta usciti da loro, non vi è più la possibilità di tornarci.
- Gli stati **(3,4)** e **(5,6)** sono due sottocatene **Assorbenti**, in quanto una volta entrati in una di queste due sottocatene non sarà più possibile uscirne. In particolare queste sottocatene sono fatte da stati **Ricorrenti**, in quanto ogni stato può raggiungerne un altro.

Una volta aver categorizzato gli stati, calcoliamo il limite della matrice Q (in alto a sinistra) e della matrice S (in basso a destra).

Il limite di Q è semplicemente la matrice nulla, dato che stiamo parlando di stati Transitori, e per questo, a "regime" non saremo più in uno di questi stati:

$$\underset{n}{Q^n} \longrightarrow 0$$

Per quanto riguarda la matrice S , è possibile notare anche qui che abbiamo a che fare con due sottocatene assorbenti che non comunicano tra di loro:

$$\left[\begin{array}{cc|cc} 1/2 & 1/2 & 0 & 0 \\ 2/3 & 1/3 & 0 & 0 \\ \hline 0 & 0 & 1/2 & 1/2 \\ 0 & 0 & 1/3 & 2/3 \end{array} \right]$$

Quindi ci toccherà calcolare due limiti separatamente.

Il procedimento per calcolare il limite di S è il seguente:

$$(\pi_1, \pi_2) \cdot \begin{bmatrix} 1/2 & 1/2 \\ 2/3 & 1/3 \end{bmatrix} = (\pi_1, \pi_2)$$

Risolvendo il sistema otterremo:

$$\begin{cases} \pi_1 = \frac{4}{7} \\ \pi_2 = \frac{3}{7} \end{cases}$$

Quindi la matrice S_1 al limite sarà:

$$S_1^n \xrightarrow[n]{\longrightarrow} \begin{bmatrix} 4/7 & 3/7 \\ 4/7 & 3/7 \end{bmatrix} = T_1$$

Con lo stesso procedimento, calcoliamo la seconda matrice:

$$S_2^n \xrightarrow[n]{\longrightarrow} \begin{bmatrix} 2/5 & 3/5 \\ 2/5 & 3/5 \end{bmatrix} = T_2$$

A questo punto calcoliamo la seguente matrice:

$$(I - Q)^{-1} \cdot R \cdot T = X$$

Potremo così costruire finalmente la matrice risultante dal limite della matrice P in questo modo:¹

$$P^n \xrightarrow[n]{\longrightarrow} \begin{bmatrix} 0 & X \\ 0 & \begin{bmatrix} T_1 & 0 \\ 0 & T_2 \end{bmatrix} \end{bmatrix}$$

1.1 Casi Particolari

Matrice messa male in partenza

Nel caso in cui avessimo la seguente matrice:

$$P = \begin{bmatrix} 0 & 1/2 & 1/2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1/2 & 1/2 \\ 1/6 & 0 & 1/2 & 1/3 & 0 & 0 \\ 0 & 1/3 & 0 & 1/2 & 0 & 1/6 \\ 0 & 1/2 & 0 & 0 & 1/2 & 0 \\ 0 & 0 & 0 & 0 & 1/2 & 1/2 \end{bmatrix}$$

Non riusciamo a vedere bene gli stati assorbenti, quindi dobbiamo manipolare questa matrice, ad esempio scambiando lo stato 2 con lo stato 4. Ma che cambia nella matrice? Occorre switchare la riga 2 con la riga 4 e la colonna 2 con la colonna 4:²

$$\left[\begin{array}{ccc|ccc} 0 & 0 & 1/2 & 1/2 & 0 & 0 \\ 0 & 1/2 & 0 & 1/3 & 0 & 1/6 \\ 1/6 & 1/3 & 0 & 1/3 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 1/2 & 1/2 \\ 0 & 0 & 0 & 1/2 & 1/2 & 0 \\ 0 & 0 & 0 & 0 & 1/2 & 1/2 \end{array} \right]$$

¹Ovviamente questi zeri sono matrici di zeri

²Consiglio: forse meglio fare due step e riscriverti due volte la matrice

Matrice S unica

Nel caso in cui la matrice S non sia formata da più sottocatene, nel passo in cui facciamo:

$$(I - Q)^{-1} \cdot R \cdot T$$

La moltiplicazione per T è inutile, dato che uscirà la stessa matrice.