1 Programmazione lineare

1.1 Modelli matematici

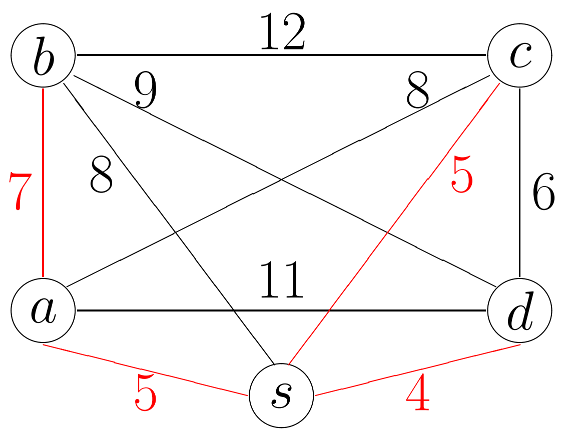
- Modelli di programmazione matematica
  - Produzione
  - Bin packing (Zaino)
  - Trasporto
  - Magazzino
  - Assegnazione
  - Commesso viaggiatore
  - Scheduling
  - Supply Chain

- Modelli su reti
  - Albero di costo minimo
  - Cammino minimo
  - Flusso massimo
  - Routing e scheduling
  - Localizzazione (*Location*)
  - PERT (*Project Evaluation and Review Techniques*)

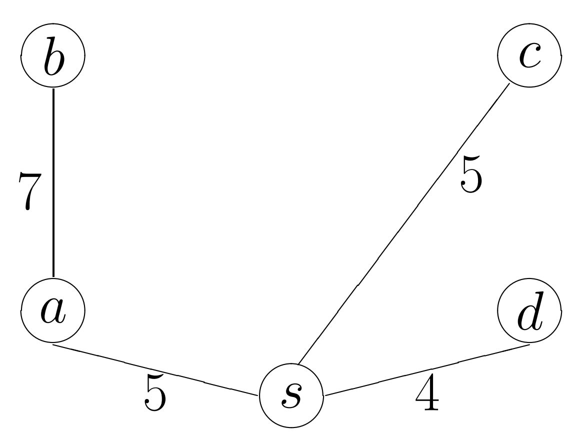
I modelli possono essere utilizzati in fase progettuale e/o gestionale

Esempio 1.1 (Connessione - Progettazione e gestione)
Un'azienda ha 4 impianti, a, b, c, d, che devono essere collegati ad una centrale elettrica, s:

PROGETTAZIONE



GESTIONE



$\diamond$

1.2 Problema di programmazione matematica

- elementi del problema da determinare: *variabili decisionali*
- relazioni (funzioni matematiche) intercorrenti tra le variabili
  - *funzione obiettivo*: valore del problema
  - *vincoli*: definiscono i valori ammissibili delle variabili

Forma generale

$$max\ \{f(x) \text{ s.t. } x \in \mathbb{R}^n, \quad g_i(x) \leq 0, \quad i = 1, \dots, m\}$$

oppure:

$$\begin{array}{ll} max & f(x) \\ s.t. & g_i(x) \leq 0, \qquad i = 1, ..., m \end{array}$$

dove $x \in \mathbb{R}^n$ sono le variabili decisionali
$f : \mathbb{R}^n \to \mathbb{R}$ è la funzione obiettivo
$g_i(x) \leq 0, g_i : \mathbb{R}^n \to \mathbb{R}, i = 1, \dots, m$ sono i vincoli

Riduzione alla forma generale

1. $min\ \{f(x)\} = -max\ \{-f(x)\}$
2. $g(x) \geq 0 \Leftrightarrow -g(x) \leq 0$
3. $g(x) = 0 \Leftrightarrow g(x) \leq 0; -g(x) \leq 0$

Soluzioni ammissibili

S_a = insieme dei valori di x per cui i vincoli sono soddisfatti
= insieme delle soluzioni ammissibili o insieme ammissibile o regione ammissibile

Può essere vuoto, limitato o illimitato

Soluzioni ottimali

S_{ott} = insieme dei valori di S_a in cui la funzione obiettivo assume il valore massimo
= insieme delle soluzioni ottimali o insieme ottimale o regione ottimale

Può essere vuoto, limitato o illimitato

Soluzione di un problema di programmazione matematica

Determinare sia una soluzione ottimale (punto di massimo) che il valore della funzione obiettivo (valore massimo)

- ricerca una strategia ottimale $\rightarrow$ punto di massimo
- confronto tra differenti strategie $\rightarrow$ valore massimo

Se l'insieme ammissibile non è chiuso può non esistere il massimo, ma solo l'estremo superiore

1.3 Problema di programmazione lineare

E' un problema di programmazione matematica in cui la funzione obiettivo e i vincoli sono lineari, o più esattamente affini

Forma canonica

$$\begin{aligned} max \;\; & z = c^T x \\ s.t. \;\; & Ax \leq b \\ & x \geq 0 \end{aligned}$$

dove c (vettore dei costi = gradiente della funzione z) e $x \in \mathbb{R}^n$, A (matrice dei vincoli) è una matrice $m \times n$ e b (vettore dei termini noti) $\in \mathbb{R}^m$

Se nella forma canonica $b \geq 0$ il problema ammette la forma normale

Riduzione alla forma canonica

1. $min \; z = c^T x \Leftrightarrow -max \; -z = (-c)^T x$
2. $Ax \geq b \Leftrightarrow (-A)x \leq (-b)$
3. $Ax = b \Leftrightarrow Ax \leq b; \quad (-A)x \leq (-b)$
4. $x \leq 0 \Leftrightarrow (-x) \geq 0$
5. x non vincolata $\Leftrightarrow x = x' - x''; \; x' \geq 0; \; x'' \geq 0$

Forma standard

$$\begin{aligned} min \quad & z = c^T x \\ s.t. \quad & Ax = b \\ & x \geq 0 \end{aligned}$$

Determinare come passare dalla forma canonica alla forma standard

Soluzioni ammissibili

S_a è convesso in quanto intersezione dei semispazi $(Ax)_j \leq b_j, j = 1, \ldots, m$; $x_i \geq 0, i = 1, \ldots, n$ e può essere:

- vuoto (vincoli incompatibili)
- limitato (poliedro convesso)
- illimitato (troncone convesso)

$(Ax)_j = b_j, j = 1, \ldots, m$; $x_i = 0, i = 1, \ldots, n$ si dicono iperpiani generatori

Un punto di S_a intersezione unica di almeno n iperpiani generatori si dice vertice di S_a

Un punto di S_a intersezione unica di più di n iperpiani generatori si dice vertice degenere

L'intersezione di almeno $n-1$ iperpiani generatori di cui $n-1$ linearmente indipendenti è una retta
La parte di retta appartenente ad S_a si dice spigolo di S_a
I due vertici estremi di uno spigolo limitato si dicono adiacenti

Soluzioni ottimali

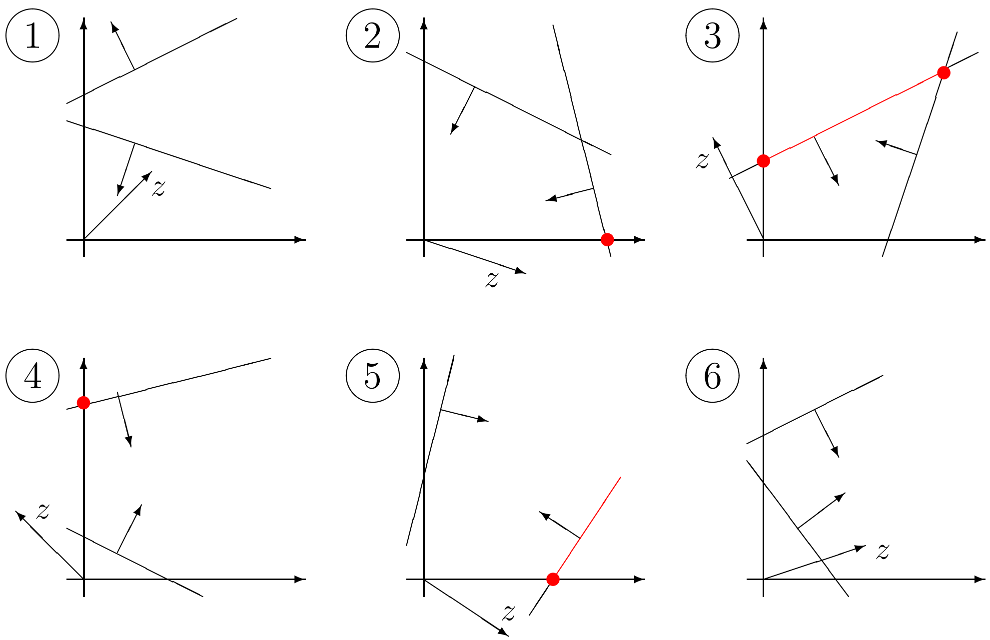
S_{ott} è insieme convesso ed è contenuto nella frontiera di S_a, se z non è costante, e può essere:

- vuoto
- costituito da un unico punto
- costituito da infiniti punti
  - limitato (poliedro convesso)
  - illimitato (troncone convesso)

Se S_a è vuoto non esistono soluzioni ottimali (1)

Se S_a è un poliedro convesso esistono sempre soluzioni ottimali (teorema di Weierstrass) $(2, 3)$

Se S_a è un troncone convesso o esistono soluzioni ottimali $(4, 5)$ o la funzione obiettivo è superiormente illimitata (6)



1.4 Teorema fondamentale

Teorema 1.1 (Teorema fondamentale della programmazione lineare)
Se un problema lineare ha soluzioni ottimali almeno una di esse è un vertice di S_a

Dimostrazione
Si possono distinguere due casi:

1. S_a costituito da un poliedro convesso K
2. S_a costituito da un troncone convesso T

Caso 1 Siano $x_1, \ldots, x_p$ i vertici di K

Sia x^* una soluzione ottimale del problema lineare scritta come combinazione convessa dei vertici di K:

$$x^* = \sum_{i=1,...,p} \lambda_i x_i \quad \text{con} \quad \sum_{i=1,...,p} \lambda_i = 1; \qquad \lambda_i \geq 0, i = 1, ..., p$$

Posto $z_i = z(x_i)$ e con $z^* = z(x^*)$, per la linearità di z si ha:

$$z^* = \sum_{i=1,...,p} \lambda_i z_i$$

Supponendo per assurdo che nessun vertice di K sia ottimale si ha $z_i < z^*, i = 1, \ldots, p$

e quindi:

$$z^* = \sum_{i=1,...,p} \lambda_i z_i < \sum_{i=1,...,p} \lambda_i z^* = z^* \sum_{i=1,...,p} \lambda_i = z^*$$

Caso 2 T può essere scritto come somma di un poliedro convesso K_1, avente gli stessi vertici di T, e un cono poliedrico K_0, avente gli spigoli paralleli agli spigoli illimitati di T:

$$T = K_1 + K_0$$

Siano $u_1, \ldots, u_p$ i vertici di K_1 e $v_1, \ldots, v_q$ i generatori di K_0
Sia x^* una soluzione ottimale del problema lineare che può essere scritta come:

$$x^* = u + v \text{ con } u \in K_1 \text{ e } v \in K_0$$

A loro volta u e v possono essere scritti come:

$$\begin{aligned} u &= \textstyle\sum_{i=1,...,p} \lambda_i u_i \ \text{ con } \textstyle\sum_{i=1,...,p} \lambda_i = 1; \qquad \lambda_i \geq 0, i = 1, ..., p \\ v &= \textstyle\sum_{j=1,...,q} \mu_j v_j \ \text{ con } \mu_j \geq 0, j = 1, ..., q \end{aligned}$$

quindi:

$$x^* = \sum_{i=1,...,p} \lambda_i u_i + \sum_{j=1,...,q} \mu_j v_j$$

Posto $z_i = z(u_i)$, $z'_j = z(v_j)$ e $z^* = z(x^*)$ per la linearità di z si ha:

$$z^* = \sum_{i=1,\dots,p} \lambda_i z_i + \sum_{j=1,\dots,q} \mu_j z'_j$$

I valori z'_j sono tutti non positivi, altrimenti al crescere di μ_j anche z^* crescerebbe, quindi:

$$\sum_{j=1,\dots,q} \mu_j z'_j \leq 0$$

Inoltre supponendo per assurdo che nessun vertice di T sia ottimale si ha $z_i < z^*, i = 1, \dots, p$ e quindi:

$$z^* = \sum_{i=1,\dots,p} \lambda_i z_i + \sum_{j=1,\dots,q} \mu_j z'_j < \sum_{i=1,\dots,p} \lambda_i z^* = z^* \sum_{i=1,\dots,p} \lambda_i = z^*$$

♣

- Il teorema precedente riduce la ricerca della soluzione ottimale ai soli vertici di S_a e costituisce il fondamento dell'algoritmo del simplesso
- L'enunciato riportato sottintende che il problema sia in forma canonica
  Si consideri il seguente problema:

$$\begin{array}{ll} max & z = x_1 + x_2 \\ s.t. & x_1 + x_2 \leq 1 \end{array}$$

  $z^* = 1$, $S_{ott} = \{(x_1, x_2) \in \mathbb{R}^2 | x_1 + x_2 = 1\}$ ma non esistono vertici
- Esiste un teorema, analogo ma meno importante, relativo alle soluzioni ammissibili:
  Se un problema lineare ha soluzioni ammissibili almeno una di esse è un vertice di S_a

Un problema lineare presenta uno dei seguenti casi mutualmente esclusivi:

1. S_a è vuoto
2. esistono soluzioni ottimali
3. la funzione z è superiormente illimitata

1 e 3 si dicono casi di impossibilità

1.5 Procedimento del cardine

Siano dati n vettori $v_1, \ldots, v_n$, non necessariamente linearmente indipendenti e m vettori $a_1, \ldots, a_m$ appartenenti allo spazio generato da $v_1, \ldots, v_n$
$a_1, \ldots, a_m$ possono essere espressi come combinazione lineare di $v_1, \ldots, v_n$:

$$a_h = \lambda_{h1} v_1 + \ldots + \lambda_{hn} v_n \qquad h = 1, \ldots, m$$

Se $\lambda_{ij} \neq 0$ è possibile scambiare il vettore a_i e il vettore v_j:

$$v_j = -\frac{\lambda_{i1}}{\lambda_{ij}} v_1 - \ldots - \frac{\lambda_{i,j-1}}{\lambda_{ij}} v_{j-1} + \frac{1}{\lambda_{ij}} a_i - \frac{\lambda_{i,j+1}}{\lambda_{ij}} v_{j+1} - \ldots - \frac{\lambda_{in}}{\lambda_{ij}} v_n$$

e sostituendo si ha:

$$\begin{aligned} a_h \ = \ & \left(\lambda_{h1} - \lambda_{i1} \frac{\lambda_{hj}}{\lambda_{ij}}\right) v_1 + \ldots + \left(\lambda_{h,j-1} - \lambda_{i,j-1} \frac{\lambda_{hj}}{\lambda_{ij}}\right) v_{j-1} + \\ & + \ \frac{\lambda_{hj}}{\lambda_{ij}} a_i + \left(\lambda_{h,j+1} - \lambda_{i,j+1} \frac{\lambda_{hj}}{\lambda_{ij}}\right) v_{j+1} + \ldots + \left(\lambda_{hn} - \lambda_{in} \frac{\lambda_{hj}}{\lambda_{ij}}\right) v_n \qquad h \neq i \end{aligned}$$

Lo scambio di a_i e v_j è detto *procedimento del cardine* e l'elemento λ_{ij} è detto $cardine$ o $pivot$
Se i vettori $v_1, \ldots, v_n$ costituiscono una base anche i vettori a_i e v_k, $k \neq j$ costituiscono una base per lo stesso spazio

Rappresentazione tabellare:

$$
\begin{array}{c|ccccc|}
 & v_1 & \dots & v_j & \dots & v_n \\
\hline
a_1 & \lambda_{11} & \dots & \lambda_{1j} & \dots & \lambda_{1n} \\
\dots & \dots & \dots & \dots & \dots & \dots \\
a_i & \lambda_{i1} & \dots & \lambda_{ij} & \dots & \lambda_{in} \\
\dots & \dots & \dots & \dots & \dots & \dots \\
a_m & \lambda_{m1} & \dots & \lambda_{mj} & \dots & \lambda_{mn} \\
\hline
\end{array}
$$

Facendo cardine su λ_{ij} si perviene alla tabella:

$$
\begin{array}{c|ccccc|}
 & v_1 & \dots & a_i & \dots & v_n \\
\hline
a_1 & \lambda_{11} - \lambda_{i1}\frac{\lambda_{1j}}{\lambda_{ij}} & \dots & \frac{\lambda_{1j}}{\lambda_{ij}} & \dots & \lambda_{1n} - \lambda_{in}\frac{\lambda_{1j}}{\lambda_{ij}} \\
\dots & \dots & \dots & \dots & \dots & \dots \\
v_j & -\frac{\lambda_{i1}}{\lambda_{ij}} & \dots & \frac{1}{\lambda_{ij}} & \dots & -\frac{\lambda_{in}}{\lambda_{ij}} \\
\dots & \dots & \dots & \dots & \dots & \dots \\
a_m & \lambda_{m1} - \lambda_{i1}\frac{\lambda_{mj}}{\lambda_{ij}} & \dots & \frac{\lambda_{mj}}{\lambda_{ij}} & \dots & \lambda_{mn} - \lambda_{in}\frac{\lambda_{mj}}{\lambda_{ij}} \\
\hline
\end{array}
$$

1.6 Algoritmo del simplesso - Dantzig (1947)

Dato un problema lineare in forma canonica:

$$
\begin{array}{lll}
max & z = c_1x_1 + \ldots + c_nx_n + c_0 & \\
s.t. & a_{11}x_1 + \ldots + a_{1n}x_n \leq b_1 & \\
 & \ldots\ldots\ldots\ldots\ldots & \\
 & a_{m1}x_1 + \ldots + a_{mn}x_n \leq b_m & \\
 & x_i \geq 0 & i = 1, \ldots, n
\end{array}
$$

Introducendo le variabili $u_j, j = 1, \ldots, m$ si ottiene:

$$
\begin{array}{lll}
max & z = c_1x_1 + \ldots + c_nx_n + c_0 & \\
s.t. & u_1 = -a_{11}x_1 - \ldots - a_{1n}x_n + b_1 & \\
 & \ldots\ldots\ldots\ldots\ldots & \\
 & u_m = -a_{m1}x_1 - \ldots - a_{mn}x_n + b_m & \\
 & x_i \geq 0 & i = 1, \ldots, n \\
 & u_j \geq 0 & j = 1, \ldots, m
\end{array}
$$

Rappresentazione tabellare:

$$
\begin{array}{c|ccc|c}
 & x_1 & \dots & x_n & \\
\hline
u_1 & -a_{11} & \dots & -a_{1n} & b_1 \\
\dots & \dots & \dots & \dots & \dots \\
u_m & -a_{m1} & \dots & -a_{mn} & b_m \\
\hline
z & c_1 & \dots & c_n & c_0
\end{array}
$$

L'intersezione dei semispazi $x_i \geq 0, i = 1, \dots, n, u_j \geq 0, j = 1, \dots, m$ definisce la regione ammissibile

Le variabili x_i formano una base di $\mathbb{R}^n$ in quanto associate ai versori di $\mathbb{R}^n$

Per definizione si associa alla tabella il vertice intersezione degli iperpiani ottenuti annullando le variabili della base corrente

L'ultima colonna rappresenta il valore delle variabili non in base corrente e il coefficiente c_0 rappresenta il valore di z

Se i coefficienti dell'ultima colonna, escluso c_0, sono tutti non negativi il punto risulta ammissibile

1.6.1 Ricerca della soluzione ottimale

Si cerca una soluzione ottimale spostandosi da un vertice ammissibile ad un altro vertice ammissibile adiacente, in modo che nel nuovo vertice z abbia un valore non peggiore
Si opera uno scambio di variabili col procedimento del cardine tra una variabile in base detta *variabile uscente* e una variabile fuori base detta *variabile entrante*
L'algoritmo del simplesso assegna un criterio per determinare le variabili da scambiare
Sia data la seguente tabella di un problema lineare ad una generica iterazione dell'algoritmo del simplesso, relativa ad un vertice ammissibile:

$$\begin{array}{c|ccccc|c}
 & y_1 & \dots & y_i & \dots & y_n & \\
\hline
y_{n+1} & \alpha_{11} & \dots & \alpha_{1i} & \dots & \alpha_{1n} & \beta_1 \\
\dots & \dots & \dots & \dots & \dots & \dots & \\
y_{n+j} & \alpha_{j1} & \dots & \alpha_{ji} & \dots & \alpha_{jn} & \beta_j \\
\dots & \dots & \dots & \dots & \dots & \dots & \\
y_{n+m} & \alpha_{m1} & \dots & \alpha_{mi} & \dots & \alpha_{mn} & \beta_m \\
\hline
z & \gamma_1 & \dots & \gamma_i & \dots & \gamma_n & \gamma_0
\end{array}$$

Scelta della variabile uscente

Facendo cardine su α_{ji} il valore di z risulta $\gamma_0 - \gamma_i \frac{\beta_j}{\alpha_{ji}}$

$y'_i = -\frac{\beta_j}{\alpha_{ji}}$ deve essere non negativo, quindi per ottenere un incremento della funzione obiettivo γ_i deve essere positivo

Scelta della variabile entrante

Scegliendo y_{n+j} come variabile entrante essendo y_i la variabile uscente si ha:

$$\begin{aligned} y'_i &= -\frac{\beta_j}{\alpha_{ji}} \\ y'_{n+k} &= \beta_k - \beta_j \frac{\alpha_{ki}}{\alpha_{ji}} \qquad k \neq j \end{aligned}$$

Per l'ammissibilità:

$$\begin{aligned} -\frac{\beta_j}{\alpha_{ji}} &\geq 0 \\ \beta_k - \beta_j \frac{\alpha_{ki}}{\alpha_{ji}} &\geq 0 \qquad k \neq j \end{aligned}$$

La prima è vera solo se $\alpha_{ji} < 0$

La seconda è vera se $\alpha_{ki} \geq 0$, mentre per $\alpha_{ki} < 0$ equivale a:

$$\frac{\beta_j}{\alpha_{ji}} \geq \frac{\beta_k}{\alpha_{ki}}$$

che è vera se:

$$j \in argmax \ \left\{ \frac{\beta_k}{\alpha_{ki}}, \alpha_{ki} < 0 \right\} = argmin \ \left\{ \left| \frac{\beta_k}{\alpha_{ki}}, \alpha_{ki} < 0 \right| \right\}$$

- Il criterio di scelta della variabile uscente non è rigoroso
  In fase di implementazione è necessario quindi precisare il criterio (il massimo dei γ_i, il primo da sinistra, ecc.)

- Il criterio di scelta della variabile entrante è rigoroso tranne nei casi in cui esistano più indici per cui si ottiene il minimo in valore assoluto; in questo caso in fase di implementazione è necessario precisare il criterio (il primo dall'alto, ecc.)

Una volta determinate le variabili da scambiare si applica il procedimento del cardine

1.6.2 Interpretazione geometrica

Scelta dell'iperpiano generatore uscente

I coefficienti della funzione obiettivo rappresentano le componenti del gradiente secondo la base corrente
z è crescente lungo gli spigoli con y_i crescente se γ_i è positivo

Scelta dell'iperpiano generatore entrante

Scegliendo $y_i = 0$ come iperpiano generatore uscente di base lo spostamento dal vertice corrente al vertice ammissibile adiacente lungo lo spigolo $y_h = 0, h \neq i$ si ottiene facendo entrare in base il primo iperpiano generatore non in base
Il punto di intersezione dell'iperpiano generatore $y_{n+k} = 0, k = 1, \ldots, m$ con lo spigolo $y_h = 0, h \neq i$ ha coordinate, nella base corrente:

$$\begin{aligned} y_h &= 0 & h \neq i \\ y_i &= -\frac{\beta_k}{\alpha_{ki}} (\Leftarrow y_{n+k} = \alpha_{ki} y_i + \beta_k = 0) \end{aligned}$$

Se $y_i < 0$ lo spostamento è avvenuto nella direzione decrescente per y_i e quindi non è ammissibile altrimenti l'iperpiano generatore che viene incontrato per primo è quello per cui il valore di y_i è minimo

1.7 Terminazione

1.7.1 Ottimalità

Dall'ultima riga si ricava:

$$z = \gamma_1 y_1 + \cdots + \gamma_i y_i + \ldots + \gamma_n y_n + \gamma_0$$

Se $\gamma_i \leq 0, i = 1, \ldots, n$ allora in tutta la regione ammissibile il valore di z è non migliore di quello corrente

Si definisce *tabella ottimale* una tabella che ha l'ultima riga, tranne al più l'ultimo elemento, tutta non positiva e l'ultima colonna, tranne al più l'ultimo elemento, tutta non negativa

1.7.2 Funzione obiettivo superiormente illimitata

Dalla tabella si ricava:

$$y_{n+k} = \alpha_{k1}y_1 + \ldots + \alpha_{ki}y_i + \ldots + \alpha_{kn}y_n + \beta_k \qquad k = 1, \ldots, m$$

per cui se in corrispondenza di un coefficiente $\gamma_i > 0$ si ha $\alpha_{ki} \geq 0, k = 1, \ldots, m$ allora le variabili y_{n+k} rimangono ammissibili al crescere di y_i, per cui $\gamma_i y_i$ può crescere indefinitamente e z risulta superiormente illimitata

- Regione ammissibile illimitata
  Indipendentemente dal valore di γ_i la condizione $\alpha_{ki} \geq 0, k = 1, \ldots, m$ vuol dire che la regione ammissibile ammette uno spigolo illimitato, cioè risulta essere un troncone

Esempio 1.2 *Risolvere con l'algoritmo del simplesso il seguente problema di programmazione lineare:*

$$\begin{array}{ll} min & z = -x_1 - 4x_2 \\ s.t. & x_2 \leq 2 \\ & x_1 + 2x_2 \leq 5 \\ & x_1, x_2 \geq 0 \end{array} \quad \xrightarrow[canonica]{forma} \quad \begin{array}{ll} -max & -z = x_1 + 4x_2 \\ s.t. & x_2 \leq 2 \\ & x_1 + 2x_2 \leq 5 \\ & x_1, x_2 \geq 0 \end{array}$$

La tabella iniziale è data da:

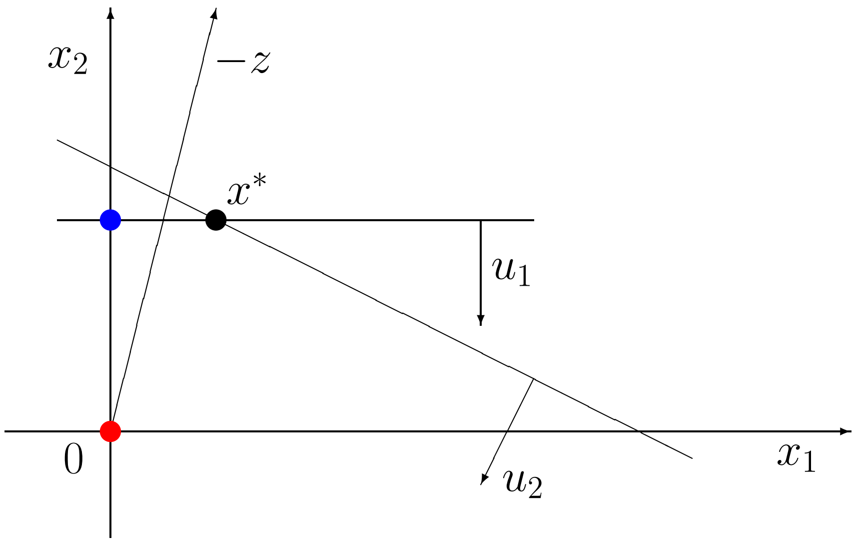
	x_1	x_2	
u_1	0	$\boxed{-1}$	2
u_2	-1	-2	5
$-z$	1	4	0

$x = (0, 0)$, $z = 0$

	x_1	u_1	
x_2	0	-1	2
u_2	$\boxed{-1}$	2	1
$-z$	1	-4	8

$x = (0, 2)$, $z = -8$

	u_2	u_1	
x_2	0	-1	3
x_1	-1	2	1
$-z$	-1	-2	9



La tabella è ottimale e la soluzione è $x^* = (1, 2)$, $z^* = -9$ ◇

1.8 Convergenza

L'algoritmo del simplesso converge in quanto i vertici sono in numero finito e, tranne nei casi degeneri, ad ogni iterazione si determina un vertice in cui la funzione obiettivo è non peggiore dei precedenti, per cui non si ritorna su uno stesso vertice
Quindi in un numero finito di passi si determina un vertice ottimale, se esiste, oppure si determina un vertice che è origine di uno spigolo illimitato su cui la funzione obiettivo è superiormente illimitata

1.9 Ricerca di una tabella ammissibile

Se il problema lineare non ammette la forma normale la tabella iniziale associata all'origine non è ammissibile, pertanto è necessario determinare una tabella ammissibile, se la regione ammissibile è non vuota

In questo caso si applica un diverso criterio per determinare le variabili da scambiare
Per la non ammissibilità esiste almeno un $\beta_j < 0$
Si cerca un $\alpha_{ji} > 0$ e si determina così la variabile uscente y_i
Scegliendo come cardine $\alpha_{ki} < 0$ con $\beta_k > 0$ la variabile entrante è quella per cui si ha il minimo di $\left|\frac{\beta_k}{\alpha_{ki}}\right|$, in modo che tutti i termini noti non negativi restino non negativi
Facendo cardine $\alpha_{ki} > 0$ con $\beta_k < 0$ la variabile entrante è quella per cui si ha il massimo di $\left|\frac{\beta_k}{\alpha_{ki}}\right|$, in modo che tutti i termini noti negativi corrispondenti a coefficienti positivi nella colonna della variabile uscente diventino non negativi

Il criterio converge poichè ad ogni passo i termini negativi diminuiscono di numero o diminuisce il valore assoluto del più negativo

- Il criterio di scelta della variabile uscente non è rigoroso
  In fase di implementazione è necessario quindi precisare il criterio (il massimo degli α_{ji}, il primo da sinistra, ecc.)

- Il criterio di scelta della variabile entrante è anche meno rigoroso
  Si noti che il caso $\alpha_{ji} > 0$ e $\beta_j < 0$ esiste sempre
  In fase di implementazione è necessario precisare il criterio

Esempio 1.3 *Risolvere con l'algoritmo del simplesso il seguente problema di programmazione lineare:*

$$\begin{array}{ll} max & z = 2x_1 + x_2 \\ s.t. & x_1 + x_2 \geq 1 \\ & x_1 \leq 4 \\ & x_1, x_2 \geq 0 \end{array} \quad \xrightarrow[canonica]{forma} \quad \begin{array}{ll} max & z = 2x_1 + x_2 \\ s.t. & -x_1 - x_2 \leq -1 \\ & x_1 \leq 4 \\ & x_1, x_2 \geq 0 \end{array}$$

La tabella iniziale è data da:

	x_1	x_2	
u_1	$\boxed{1}$	1	-1
u_2	-1	0	4
z	2	1	0

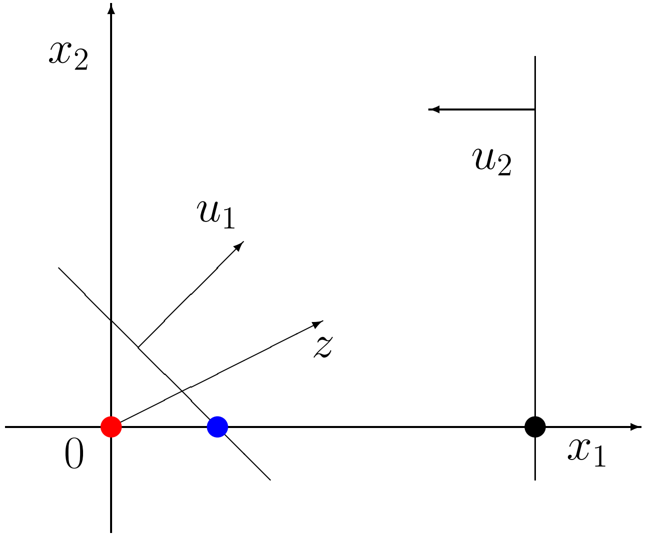
$x = (0, 0), z = 0$

	u_1	x_2	
x_1	1	-1	1
u_2	$\boxed{-1}$	1	3
z	2	-1	2

$x = (1, 0), z = 2$

	u_2	x_2	
x_1	-1	0	4
u_1	-1	1	3
z	-2	1	8

$x = (4, 0), z = 8$



La colonna di x_2 indica che la funzione obiettivo è superiormente illimitata $\diamondsuit$

Regione ammissibile vuota

Se a $\beta_j < 0$ corrispondono $\alpha_{ji} \leq 0, i = 1, \ldots, n$ allora non esistono soluzioni ammissibili perchè la condizione:

$$y_{n+j} = \alpha_{j1}y_1 + \ldots + \alpha_{ji}y_i + \ldots + \alpha_{jn}y_n + \beta_j \geq 0$$

non può essere soddisfatta

- Regione ammissibile limitata
  Se ad un termine $\beta_j \geq 0$ corrispondono coefficienti $\alpha_{ji} < 0, i = 1, \ldots, n$ vuol dire che la regione ammissibile è limitata in quanto da:

$$\alpha_{j1}y_1 + \ldots + \alpha_{ji}y_i + \ldots + \alpha_{jn}y_n + \beta_j \geq 0$$

  si ricava:

$$y_i \leq -\frac{\beta_j}{\alpha_{ji}} \quad i = 1, \ldots, n$$

- Se $\beta_j = 0$ e $\alpha_{ji} < 0, i = 1, \ldots, n$ la regione ammissibile si riduce ad un punto

1.10 Casi particolari

1.10.1 Soluzione degenere e ciclaggio

Se in una tabella esiste $\beta_j = 0$, il corrispondente iperpiano $y_{n+j} = 0$ pur non formando la base, passa ugualmente per il vertice rappresentato dalla tabella, che risulta degenere
Se y_{n+j} viene scelta come variabile entrante i valori delle variabili restano invariati e la nuova tabella rappresenta lo stesso vertice

Oltre a rimanere nello stesso vertice, possono presentarsi ciclicamente le stesse basi
Questo fenomeno detto ciclaggio può essere evitato utilizzando diversi metodi, ad esempio la *regola di Bland* e il *metodo delle perturbazioni di Wolfe*

1.10.2 Infinite soluzioni ottimali

Se $\gamma_i = 0$ incrementando la variabile y_i la funzione obiettivo non varia; pertanto se la tabella è ottimale scegliendo y_i come variabile uscente è possibile determinare un nuovo vertice ottimale (salvo nel caso di vertice degenere)
Per la linearità del problema vuol dire che tutti i punti dello spigolo sono ottimali
Se tutti gli $\alpha_{ji}, j = 1, \ldots, m$ sono non negativi allora esiste uno spigolo illimitato di punti ottimali

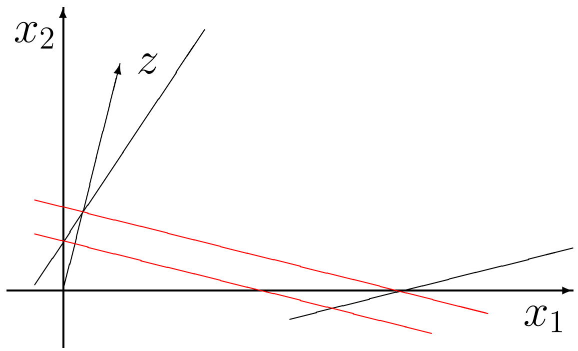
1.10.3 Problemi con vincoli di uguaglianza

Sono possibili differenti metodologie:

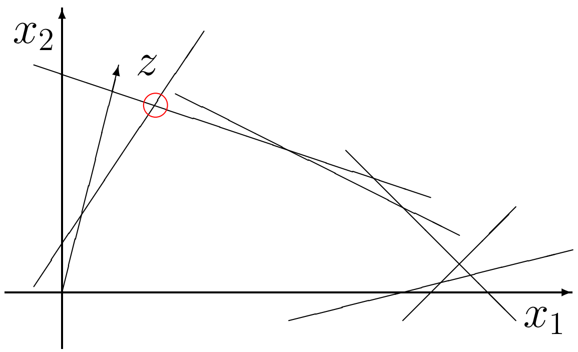
- ricondurlo alla forma canonica
- definire un problema associato in cui tutti i vincoli di uguaglianza $Ax = b$ sono scritti in forma di disuguaglianza $Ax \leq b$ e la funzione obiettivo è data dal valore dei vincoli $1^T(Ax - b)$
  se esiste una soluzione ottimale di valore nullo per il problema associato si ritorna al problema dato
  le condizioni di ottimalità escludono le u relative ai vincoli di uguaglianza
- forzare i vincoli di uguaglianza ad entrare in base (poichè devono avere valore nullo), verificando che quelli eventualmente rimasti fuori base abbiano anch'essi valore nullo
  le condizioni di ottimalità escludono le u relative ai vincoli di uguaglianza

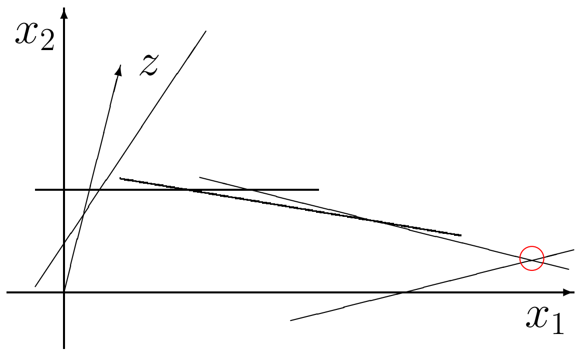
1.10.4 Scelta della variabile uscente

La scelta dell'elemento su cui "fare cardine" riveste un ruolo molto importante, ma non è possibile fissare regole generali, visto che la tabella rappresenta una situazione "locale"



Componente maggiore del gradiente o incremento maggiore della funzione obiettivo?
Dipende ...





2 Dualità

2.1 Problema duale

Dato il problema lineare (primale):

$$\begin{aligned} max \quad & z = c^T x \\ s.t. \quad & Ax \leq b \\ & x \geq 0 \end{aligned}$$

È sempre possibile definire un problema ad esso associato (duale):

$$\begin{aligned} min \quad & w = b^T u \\ s.t. \quad & A^T u \geq c \\ & u \geq 0 \end{aligned}$$

- A matrice $m \times n \Rightarrow x \in \mathbb{R}^n, u \in \mathbb{R}^m$

2.2 Proprietà di un problema e del suo duale

1. Il duale del problema duale coincide con il problema dato

   <u>Dimostrazione</u>

   Riportando il problema duale in forma canonica si ha:

$$\begin{array}{ll} -max & -w = -b^T u \\ s.t. & -A^T u \leq -c \\ & u \geq 0 \end{array}$$

   il cui duale è:

$$\begin{array}{ll} -min & -t = -c^T y \\ s.t. & -(A^T)^T y \geq -b \\ & y \geq 0 \end{array}$$

   che può essere riscritto come:

$$\begin{array}{ll} max & t = c^T y \\ s.t. & Ay \leq b \\ & y \geq 0 \end{array}$$

   che è equivalente al problema dato ♣

2. Se un problema lineare e il suo duale hanno rispettivamente soluzioni ammissibili x^0 e u^0, allora vale:

$$c^T x^0 \leq b^T u^0$$

e inoltre entrambi hanno soluzioni ottimali

Dimostrazione

Per l'ammissibilità di x^0 e u^0 si ha:

$$\begin{aligned} Ax^0 \leq b,\ u^0 \geq 0 \ &\Rightarrow\ (Ax^0)^T u^0 \leq b^T u^0 \\ A^T u^0 \geq c,\ x^0 \geq 0 \ &\Rightarrow\ (A^T u^0)^T x^0 \geq c^T x^0 \end{aligned}$$

Osservando che $(Ax^0)^T u^0 = (A^T u^0)^T x^0$ si ha la tesi ♣

3. Se un problema lineare e il suo duale hanno rispettivamente soluzioni ammissibili x^* e u^* per cui vale $c^T x^* = b^T u^*$, allora x^* e u^* sono soluzioni ottimali rispettivamente del primale e del duale

Dimostrazione

Dalla proprietà 2 si ha:

$$\begin{aligned} c^T x \leq b^T u^* = c^T x^* \ &\Rightarrow\ x^* \text{ è ottimale} \\ b^T u \geq c^T x^* = b^T u^* \ &\Rightarrow\ u^* \text{ è ottimale} \end{aligned}$$

♣

4. I teorema della dualità

Uno dei due problemi ha soluzioni ottimali se e solo se ne ha anche l'altro

In questo caso $max\ z = min\ w$

Dimostrazione

Dati un problema e il suo duale in forma canonica:

$$\begin{array}{llll} max & z = c^T x & -max & -w = (-b)^T u \\ s.t. & Ax \leq b & s.t. & (-A)^T u \leq (-c) \\ & x \geq 0 & & u \geq 0 \end{array}$$

le corrispondenti tabelle iniziali sono:

$$\begin{array}{c|c|c} & x^T & \\ \hline u & -A & b \\ \hline z & c^T & 0 \end{array} \qquad \begin{array}{c|c|c} & u^T & \\ \hline x & A^T & -c \\ \hline -w & -b^T & 0 \end{array}$$

Le due tabelle sono una la trasposta cambiata di segno dell'altra; applicando il metodo del simplesso alla prima tabella, si supponga di far cardine su $-(A)_{ji}$, allora nella seconda tabella si fa cardine su $(A^T)_{ij}$, così le tabelle successive sono ancora una la trasposta cambiata di segno dell'altra, fino alla tabella ottimale di uno dei due problemi

Allora anche la sua trasposta cambiata di segno è ottimale

Dalla tabella ottimale del problema primale si ha:

$$max \quad z = \gamma_0$$

e dalla tabella ottimale del problema duale si ha:

$$-max \quad -w = -\gamma_0$$

cioè:

$$min \quad w = \gamma_0$$

per cui $max \quad z = min \quad w$ ♣

5. Se uno dei due problemi ha soluzione illimitata allora l'altro non ha soluzioni ammissibili
<u>Dimostrazione</u>
Se l'altro problema avesse una soluzione ammissibile il valore della funzione obiettivo in quel punto sarebbe un limitante per la funzione obiettivo del primo problema ♣

6. Se uno dei due problemi non ha soluzioni ammissibili allora l'altro o ha soluzione illimitata o non ha soluzioni ammissibili
<u>Dimostrazione</u>
Se l'altro problema avesse soluzioni ottimali le avrebbe anche il primo problema ♣

7. Il teorema della dualità

Due soluzioni ammissibili x^* e u^* sono soluzioni ottimali se e solo se valgono le relazioni:

$$\begin{aligned}(b - Ax^*)^T u^* &= 0 \\ (A^T u^* - c)^T x^* &= 0\end{aligned}$$

Dimostrazione

Se valgono le relazioni:

$$\begin{gathered}(b - Ax^*)^T u^* = 0 \\ (A^T u^* - c)^T x^* = 0\end{gathered}$$

sommando membro a membro si ottiene:

$$b^T u^* - (Ax^*)^T u^* + (A^T u^*)^T x^* - c^T x^* = 0$$

e ricordando che $(Ax^*)^T u^* = (A^T u^*)^T x^*$ si ha:

$$b^T u^* = c^T x^*$$

che per la proprietà 3 fornisce l'ottimalità di x^* e u^*

Viceversa se x^* e u^* sono ottimali, per il I teorema della dualità, si ha:

$$b^T u^* = c^T x^*$$

Aggiungendo e togliendo la quantità $u^{*T} A x^*$ e riordinando si ha:

$$(b - Ax^*)^T u^* + (A^T u^* - c)^T x^* = 0$$

e per i vincoli dei due problemi si ricavano le relazioni cercate ♣

- Le relazioni del II teorema della dualità vengono chiamate anche condizioni di complementarietà
- Un vincolo di uno dei due problemi può essere soddisfatto come uguaglianza e la corrispondente variabile dell'altro problema può essere nulla
- La tabella ottimale di un problema fornisce la soluzione del problema duale:

$$
\begin{aligned}
w^* &= z^* \\
u_i &= 0 \qquad \text{se la variabile primale } u_i \text{ è fuori base} \\
u_i &= -\gamma_i \quad \text{se la variabile primale } u_i \text{ è in base}
\end{aligned}
$$

2.3 Interpretazione economica del problema duale

Problema di produzione

$$\begin{aligned} max \quad & z = c^T x \\ s.t. \quad & Ax \leq b \\ & x \geq 0 \end{aligned}$$

c	vettore dei valori unitari dei beni prodotti
x	vettore dei beni prodotti
A	matrice tecnologica (unità di risorsa per unità di bene prodotto)
b	vettore delle risorse
z	valore dei beni prodotti
vincoli	non si possono utilizzare più risorse di quelle disponibili
	quantità di beni prodotti non negative

$$[A] = \frac{unità\ di\ risorsa}{unità\ di\ bene}$$
$$[c] = \frac{denaro}{unità\ di\ bene}$$

e dalla relazione $[A][u] = [c]$ si ricava:

$$[u] = \frac{denaro}{unità\ di\ risorsa}$$

<u>Problema duale</u>

u	costo-ombra delle risorse
	(valore di una ulteriore unità di risorsa in quel processo produttivo)
w	costo-ombra globale delle risorse o valore-ombra del processo produttivo
vincoli	costi-ombra unitari non inferiori ai valori dei beni prodotti
	costi-ombra non negativi

Le variabili duali non rappresentano il valore unitario delle risorse perchè è un dato e non una variabile, inoltre alcune risorse risultebbero avere valore nullo

Il termine costo-ombra è dato dal fatto che non è legato al valore effettivo della risorsa, ma a quello che ha in quello specifico processo produttivo (matrice tecnologica e risorse)

Se una risorsa non viene completamente utilizzata il costo-ombra è nullo, altrimenti può essere positivo o nullo
Un bene non viene prodotto se il costo-ombra è superiore al suo valore, altrimenti può essere prodotto o meno

Se una risorsa non è completamente utilizzata non vi è nessun vantaggio ad averne una maggiore disponibilità, altrimenti il valore dei beni prodotti può essere incrementato disponendo di una ulteriore unità di risorsa (variabile duale non nulla) o può restare invariato (variabile duale nulla)
Se il costo-ombra delle risorse necessarie a produrre un bene è superiore al suo valore è svantaggioso produrlo, altrimenti può essere svantaggioso produrlo (variabile primale nulla) o può non esserlo (variabile primale non nulla)

Ulteriore interpretazione

Una ditta vuole acquistare tutte le risorse e deve determinare il valore ottimale dei prezzi u da offrire per ogni unità di risorsa

La ditta cerca di minimizzare la spesa di acquisto

$$min \quad b^T u$$

Il valore delle risorse necessarie a produrre una unità di bene deve essere non inferiore al suo valore unitario

$$A^T u \geq c$$

I prezzi devono essere non negativi

$$u \geq 0$$

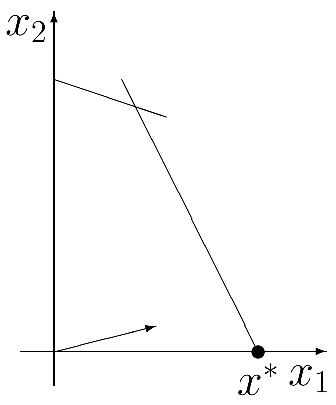
Esempio 2.1 (Modello di produzione lineare)

Due processi produttivi utilizzano due risorse, R_1 e R_2, per produrre due beni, B_1 e B_2; una unità di B_1 richiede 1 unità di R_1 e 2 unità di R_2, mentre una unità di B_2 richiede 3 unità di R_1 e 1 unità di R_2. Si supponga di disporre di 12 unità di R_1 e 6 unità di R_2 e che una unità di B_1 abbia valore 4 e una unità di B_2 abbia valore 1

Problema primale

$$\begin{array}{ll} max & z = 4x_1 + x_2 \\ s.t. & x_1 + 3x_2 \leq 12 \\ & 2x_1 + x_2 \leq 6 \\ & x_1, x_2 \geq 0 \end{array}$$

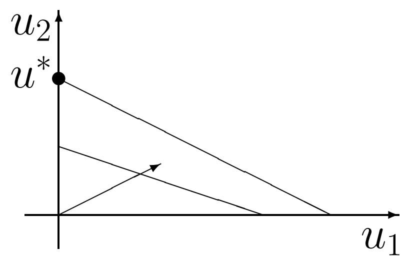
$x^* = (3, 0)$; $z^* = 12$



Problema duale

$$\begin{array}{ll} min & w = 12u_1 + 6u_2 \\ s.t. & u_1 + 2u_2 \geq 4 \\ & 3u_1 + u_2 \geq 1 \\ & u_1, u_2 \geq 0 \end{array}$$

$u^* = (0, 2)$; $w^* = 12$



$u_1^* = 0$ *in accordo col fatto che la risorsa* R_1 *non è completamente utilizzata*
$u = (1, 0)$ *porterebbe allo stesso valore di* w*, ma in questo caso si potrebbero vendere solo 9 unità di* R_1*, ottenendo 9, e utilizzare le risorse trattenute per produrre tre unità di* B_1 *con un valore 12, migliorando il risultato* ♢

Problema della dieta

Siano dati un insieme di alimenti e un insieme di fattori nutritivi; note le unità di fattori nutritivi per unità di alimento, i fabbisogni minimi dei fattori nutritivi, i costi unitari degli alimenti determinare la dieta di costo minimo che garantisce i fabbisogni minimi

Determinare il problema di programmazione lineare associato, il suo duale e darne un'interpretazione economica

Esempio 2.2 (Significato delle variabili duali)

Sia $u_i^* \neq 0$, *cioè* $\sum_{j=1,\dots,n} a_{ij}x_j^* = b_i$
Se $b_i^0 = b_i + 1$ *si ha:*

$$w^{0*} = \sum_{h \neq j} b_h u_h^* + b_i^0 u_i^* = w^* + u_i^*$$

e quindi $z^{0*} = z^* + u_i^*$ $\diamondsuit$

Esempio 2.3 (Significato dei vincoli duali)

Sia $\sum_{j=1,\dots,n} a_{ij}x_j^* = b_i$, $u_i^* \neq 0$ *e* $u_k^* = 0, k \neq i$
I vincoli duali si riducono a $a_{ij}u_i^* \geq c_j$ *da cui si ricava:*

$$u_i^* \geq \frac{c_j}{a_{ij}}, \quad j = 1, \dots, n$$

dove $\frac{c_j}{a_{ij}}$ *rappresenta il valore unitario fornito dalla risorsa* i *se è utilizzata per produrre il bene* j
La soluzione è $u_i^* = max_j \left\{ \frac{c_j}{a_{ij}} \right\}$ *cioè il costo-ombra della risorsa i è uguale al massimo valore unitario che può essere fornito dalla risorsa* $\diamondsuit$

Esempio 2.4 (Paradosso dell'anellino) *Un artigiano stravagante fabbrica anellini di plastica con diamanti autentici*
Dispone di un anellino del costo di 1 dollaro e di due diamanti del costo di 900 dollari
Un anellino con diamante viene venduto per 1000 dollari

$$\begin{array}{lll} max & z = 99x & \textit{profitto} \\ s.t. & x \leq 1 & \textit{vincolo sugli anellini} \\ & x \leq 2 & \textit{vincolo sui diamanti} \\ & x \geq 0 & \end{array}$$

$$\begin{array}{c|c|c} & x & \\ \hline u_a & -1^* & 1 \\ u_d & -1 & 2 \\ \hline z & 99 & 0 \end{array} \qquad \begin{array}{c|c|c} & u_a & \\ \hline x & -1 & 1 \\ u_d & 1 & 1 \\ \hline z & -99 & 99 \end{array}$$

$x^* = 1, z^* = 99$
si produce un anellino con un profitto di 99 dollari

$u_a^* = 99, u_d^* = 0, w^* = 99$
l'artigiano è disposto a pagare fino a 99 dollari in più per un altro anellino, ma non è disposto a pagare alcunchè per un altro diamante $\diamondsuit$

APPENDICE - Formulazione del problema duale col problema primale in forma non canonica

Problema primale di massimo	Problema duale di minimo
coefficienti della funzione obiettivo	termini noti dei vincoli
termini noti dei vincoli	coefficienti della funzione obiettivo
colonna j dei coefficienti	riga j dei coefficienti
riga i dei coefficienti	colonna i dei coefficienti
$x_j \geq 0$	$\sum_{i=1,\ldots,m} a_{ij}u_i \geq c_j$
$x_j \leq 0$	$\sum_{i=1,\ldots,m} a_{ij}u_i \leq c_j$
x_j libera	$\sum_{i=1,\ldots,m} a_{ij}u_i = c_j$
$\sum_{j=1,\ldots,n} a_{ij}x_j \leq b_i$	$u_i \geq 0$
$\sum_{j=1,\ldots,n} a_{ij}x_j \geq b_i$	$u_i \leq 0$
$\sum_{j=1,\ldots,n} a_{ij}x_j = b_i$	u_i libera

Problema primale di minimo	Problema duale di massimo
coefficienti della funzione obiettivo	termini noti dei vincoli
termini noti dei vincoli	coefficienti della funzione obiettivo
colonna j dei coefficienti	riga j dei coefficienti
riga i dei coefficienti	colonna i dei coefficienti
$x_j \geq 0$	$\sum_{i=1,\dots,m} a_{ij}u_i \leq c_j$
$x_j \leq 0$	$\sum_{i=1,\dots,m} a_{ij}u_i \geq c_j$
x_j libera	$\sum_{i=1,\dots,m} a_{ij}u_i = c_j$
$\sum_{j=1,\dots,n} a_{ij}x_j \leq b_i$	$u_i \leq 0$
$\sum_{j=1,\dots,n} a_{ij}x_j \geq b_i$	$u_i \geq 0$
$\sum_{j=1,\dots,n} a_{ij}x_j = b_i$	u_i libera

3 Programmazione lineare a numeri interi

3.1 Problemi lineari interi

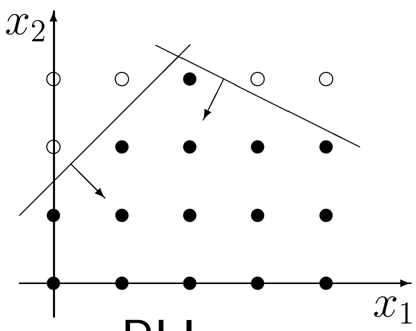
Dato il problema lineare ordinario (PLO):

$$\begin{array}{ll} max & z = c^T x \\ s.t. & Ax \leq b \\ & x \geq 0 \end{array}$$

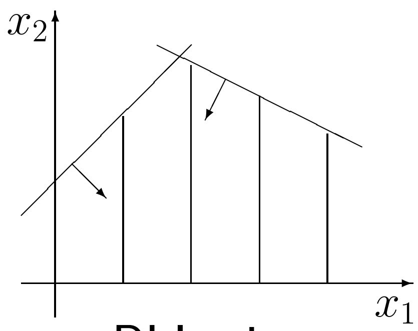
aggiungendo la condizione di integrità:

$$x_i \in N \qquad i \in I$$

si ottiene un problema lineare a numeri interi (PLI) ad esso associato



PLI puro

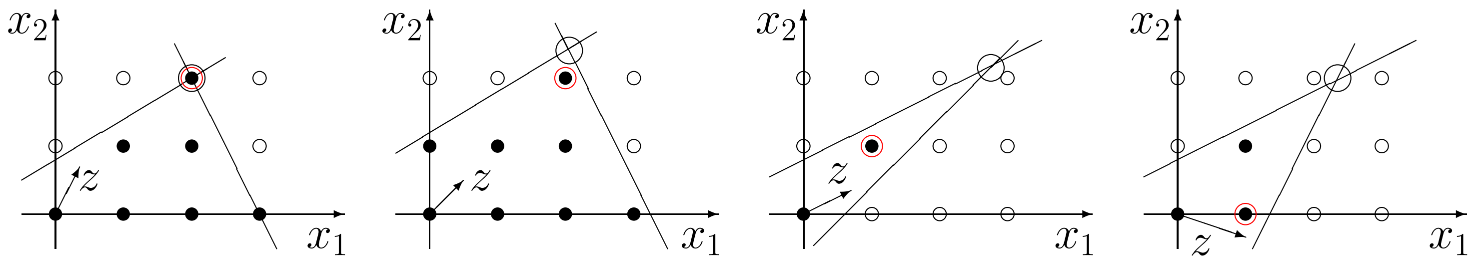


PLI misto

Proprietà generali

- $S'_a \subseteq S_a$ per cui
$$S_a = \emptyset \Rightarrow S'_a = \emptyset$$
-
$$z'(x) = z(x), x \in S'_a \Rightarrow sup\ z' \leq sup\ z$$

La soluzione di un PLI può risultare complessa poichè non esiste alcuna relazione con l'insieme delle soluzioni ottimali del PLO



Classi di metodi risolutivi:

<u>Metodi approssimati</u>

- <u>Arrotondamento o troncamento della soluzione del PLO</u>
  Non garantisce nè l'ottimailità, nè l'ammissibilità
- <u>Punto ammissibilie più vicino alla soluzione del PLO</u>
  Può essere complesso determinarlo e non garantisce l'ottimalità

<u>Metodi esatti</u>

- <u>Metodi branch and bound (separazione e valutazione)</u>
- <u>Metodi cutting plane (piani di taglio o iperpiani secanti)</u>

3.2 Metodi branch and bound

Si suddivide il problema assegnato in sottoproblemi più semplici da risolvere e si utilizzano le soluzioni di questi per ricavare la soluzione del problema dato

I concetti fondamentali sono rilassamento, separazione e eliminazione

Notazioni

Dato un problema P qualsiasi siano:

$S_a(P)$	l'insieme delle soluzioni ammissibili
$z(P)$	la funzione obiettivo da massimizzare
$x^*(P)$	la soluzione ottimale
$z^*(P)$	il valore ottimale
$z_s(P)$	il valore approssimato (stima o limitazione)

3.2.1 Rilassamento

Un problema P' si dice rilassamento di un problema P se:

$$S_a(P) \subset S_a(P')$$
$$z(P')(x) \geq z(P)(x) \qquad x \in S_a(P)$$

Proprietà del rilassamento

- $S_a(P') = \emptyset \Rightarrow S_a(P) = \emptyset$
- $z^*(P') \geq z^*(P)$
- $x^*(P') \in S_a(P), z(P')(x^*(P')) = z(P)(x^*(P')) \Rightarrow (x^*(P'), z^*(P')) = (x^*(P), z^*(P))$

3.2.2 Separazione

Un problema P si dice separato nei sottoproblemi $P_1, \ldots, P_n$ se:

- $S_a(P) = \bigcup_{i=1,\ldots,n} S_a(P_i)$
  $S_a(P_i) \cap S_a(P_j) = \emptyset \qquad i \neq j$
- $z(P_i) = z(P) \qquad i = 1, \ldots, n$

- È opportuno non avere una partizione troppo fine per non dover risolvere troppi sottoproblemi

Proprietà della separazione

- $z^*(P) = \max\limits_{i=1,\ldots,n} z^*(P_i) = z^*(P_{i^*}) \Rightarrow x^*(P) = x^*(P_{i^*})$

3.2.3 Eliminazione

Un problema P si dice eliminato se un suo rilassamento P' verifica una delle tre condizioni seguenti:

- $E1 \quad S_a(P') = \emptyset$
- $E2 \quad z^*(P') \leq z^*$ dove z^* è il valore della soluzione ottimale corrente
- $E3 \quad x^*(P') \in S_a(P)$

- Se si utilizza una stima $z_s(P') \geq z^*(P')$ non è possibile verificare la condizione $E2$ in quanto da $z_s(P') \leq z_s^*$, dove z_s^* è la migliore stima corrente, non discende $z^*(P') \leq z^*$, poichè $z_s^* \geq z^*$. D'altra parte $z_s(P') \leq z^*$ è una condizione di eliminazione

- È conveniente che il valore della stima $z_s(P')$ sia il più vicino possibile al valore $z^*(P')$, purchè questo non comporti una eccessiva difficoltà e/o troppo tempo

- È importante che le condizioni di eliminazione siano assegnate per il rilassamento P' invece che per P, in quanto nel corso dell'algoritmo si risolvono i problemi P'

Proprietà dell'eliminazione

- $E1 \Rightarrow S_a(P) = \emptyset$

- $E2 \Rightarrow z^*(P) \leq z^*$

- $E3$ e $z(P')(x^*(P')) = z(P)(x^*(P')) \Rightarrow x^*(P') = x^*(P)$

3.2.4 Algoritmo branch and bound (Land e Doig, 1960 - Little, 1963)

a) inizializzare la lista dei problemi da risolvere col problema assegnato;

b) se nella lista c'è un problema P da risolvere estrarlo, considerare un rilassamento P' e andare a c); altrimenti andare ad h);

c) risolvere il problema P' (oppure determinare $z_s(P')$);

d) se $S_a(P') = \emptyset$ allora $S_a(P) = \emptyset$, eliminare il problema P per il criterio $E1$ e tornare a b);

e) se $z^*(P') \leq z^*$ (oppure $z_s(P') \leq z^*$) allora $z^*(P) \leq z^*$, eliminare il problema P per il criterio $E2$ e tornare a b);

f) se $x^*(P') \in S_a(P)$ e $z(P')(x^*(P')) = z(P)(x^*(P'))$ allora $x^*(P') = x^*(P)$, eliminare P per il criterio $E3$ e tornare a b);
se $z^*(P) > z^*$ allora porre $(x^*(P), z^*(P))$ nuova soluzione del problema, porre $z^* = z^*(P)$ e tornare a b);

g) se P non è stato eliminato decidere se abbandonare il problema;
se si abbandona separare P, aggiornare la lista dei problemi da risolvere e tornare a b);
altrimenti considerare un nuovo rilassamento P' e tornare a c);

h) se è stata trovata una soluzione $(x^*(P), z^*(P))$ questa è la soluzione ottimale;
altrimenti il problema non ammette soluzioni;

- Quello presentato è l'algoritmo base; di volta in volta è necessario precisare i criteri di rilassamento, di valutazione e di separazione

Gestione della lista

- Esistono diverse strategie di gestione della lista dei problemi da risolvere
- Per semplificare la gestione della memoria si può utilizzare la strategia depth-first o LIFO
- Le limitazioni $z_s(P')$ permettono di utilizzare la strategia highest-first; per il criterio E2, si possono eliminare alcuni dei problemi presenti nella lista
- La lista può essere gestita in modo casuale, o pseudocasuale nei casi di informazioni particolarmente imprecise
- Si può utilizzare un albero in cui ogni foglia rappresenta un sottoproblema della lista con associata la sua limitazione

4 Modelli lineari a numeri interi

4.1 Problema dello zaino (monodimensionale)

n oggetti di peso p_i e valore c_i
Si ha a disposizione uno zaino di capienza assegnata P
Massimizzare il valore trasportato

$$\begin{array}{ll} max & \sum_{i=1,\ldots,n} c_i x_i \\ s.t. & \sum_{i=1,\ldots,n} p_i x_i \leq P \\ & x_i \in \{0,1\} \qquad\qquad i = 1,\ldots,n \end{array}$$

Possibili estensioni:

- multidimensionale
- multivincolo
- multizaino

4.2 Problema del trasporto

n centri di produzione con capacità produttiva di p_i unità e m centri di distribuzione con richiesta di r_j unità

c_{ij} è il costo unitario di trasporto dal centro di produzione i al centro di distribuzione j

Minimizzare il costo di trasporto

$$
\begin{array}{lll}
min & \sum_{i=1,\ldots,n} \sum_{j=1,\ldots,m} c_{ij} x_{ij} & \\
s.t. & \sum_{i=1,\ldots,n} x_{ij} = r_j & j = 1, \ldots, m \\
& \sum_{j=1,\ldots,m} x_{ij} \le p_i & i = 1, \ldots, n \\
& x_{ij} \in N & i = 1, \ldots, n, j = 1, \ldots, m
\end{array}
$$

Condizione di capacità produttiva

$$\sum_{j=1,\ldots,m} r_j \le \sum_{i=1,\ldots,n} p_i$$

4.3 Problema del magazzino

n periodi consecutivi per i quali sono noti la domanda d_i, la capacità produttiva massima C_i e il costo di produzione unitario c_i

Si può utilizzare un magazzino di capacità H_i, dove H_0 è la disponibilità iniziale e il costo di magazzinaggio unitario è h_i

Minimizzare il costo di produzione

$$\begin{array}{lll} min & \sum_{i=1,\ldots,n} (c_i x_i + h_i y_i) & \\ s.t. & x_i + y_{i-1} = d_i + y_i & i = 1, \ldots, n \\ & 0 \leq x_i \leq C_i & i = 1, \ldots, n \\ & 0 \leq y_i \leq H_i & i = 1, \ldots, n \end{array}$$

dove $y_0 = H_0$

Condizione di disponibilità

$$\sum_{j=1,\ldots,i} d_j \leq H_0 + \sum_{j=1,\ldots,i} C_j \qquad i = 1, ..., n$$

$$d_i \leq C_i + H_{i-1} \qquad i = 1, ..., n$$

4.4 Problema dell'assegnazione

n macchine a cui assegnare un operaio tra n

c_{ij} è il costo di assegnazione dell'operaio i alla macchina j

Minimizzare i costi di assegnazione

$$\begin{array}{lll} min & \sum\limits_{i=1,\ldots,n} \sum\limits_{j=1,\ldots,n} c_{ij}x_{ij} & \\ s.t. & \sum\limits_{i=1,\ldots,n} x_{ij} = 1 & j = 1, \ldots, n \\ & \sum\limits_{j=1,\ldots,n} x_{ij} = 1 & i = 1, \ldots, n \\ & x_{ij} \in \{0, 1\} & i, j = 1, \ldots, n \end{array}$$

4.5 Problema del commesso viaggiatore

n città da visitare senza mai ripassare da alcuna città

c_{ij} è il costo di un tragitto, non necessariamente simmetrico

Minimizzare il costo (circuito hamiltoniano di costo minimo)

$$
\begin{array}{lll}
min & \sum_{i=1,\ldots,n} \sum_{j=1,\ldots,n} c_{ij} x_{ij} & \\
s.t. & \sum_{i=1,\ldots,n} x_{ij} = 1 & j = 1, \ldots, n \\
 & \sum_{j=1,\ldots,n} x_{ij} = 1 & i = 1, \ldots, n \\
 & y_i - y_j + (n+1) x_{ij} \leq n & i, j = 1, \ldots, n \\
 & x_{ij} \in \{0, 1\} & i, j = 1, \ldots, n
\end{array}
$$

5 Scheduling

Ordinamento di una sequenza di lavori
Miglioramento della pianificazione della produzione o dell'utilizzo di una risorsa (macchina)

Formalizzazione

$N = \{1, ..., n\}$	lavori
$t_1, ..., t_n$	tempi di esecuzione
$\alpha_1, ..., \alpha_n$	costi per unità di tempo
σ_0	ordine iniziale dei lavori

5.1 Modello di base

- Una sola sequenza di lavori
- non è possibile iniziare, interrompere e riprendere un lavoro
- il costo dipende linearmente dal tempo (disponibilità immediata)

Dato un ordinamento il costo è:

$$C_\sigma = \sum_{i \in N} \alpha_i \sum_{j \in P(\sigma,i)} t_j$$

dove $P(\sigma, i)$ sono i lavori che precedono i secondo σ

Esempio 5.1 (Scheduling) *L'officina di un'azienda deve riparare 4 macchinari:*

$macchinario$	A	B	C	D
$tempo\ di\ riparazione$	1	3	2	4
$costo\ di\ fermo$	2	1	3	9

Se i macchinari vengono riparati nell'ordine in cui sono stati consegnati all'officina il costo è:

$$1 * 2 + (1 + 3) * 1 + (1 + 3 + 2) * 3 + (1 + 3 + 2 + 4) * 9 = 2 + 4 + 18 + 90 = 114$$

Riparando i macchinari per tempi di riparazione crescenti si ha l'ordine A - C - B - D e il costo:

$$1 * 2 + (1 + 2) * 3 + (1 + 2 + 3) * 1 + (1 + 2 + 3 + 4) * 9 = 2 + 9 + 6 + 90 = 107$$

Ordinando i macchinari per costi di fermo decrescenti $(D - C - A - B)$ *il costo si riduce a:*

$$4 * 9 + (4 + 2) * 3 + (4 + 2 + 1) * 2 + (4 + 2 + 1 + 3) * 1 = 36 + 18 + 14 + 10 = 78 \quad \diamondsuit$$

Ordinamento per indici di urgenza $u_i = \dfrac{\alpha_i}{t_i}, i \in N$ decrescenti (Smith, 1956)

Esempio 5.2 (Scheduling - Seconda parte) *Nel caso precedente gli indici di urgenza sono rispettivamente* $2, \frac{1}{3}, \frac{3}{2}, \frac{9}{4}$ *e l'ordine per indici decrescenti è* $D - A - C - B$ *a cui corrisponde il costo:*

$$4 * 9 + (4 + 1) * 2 + (4 + 1 + 2) * 3 + (4 + 1 + 2 + 3) * 1 = 36 + 10 + 21 + 10 = 77$$

Soluzione ottimale $\diamondsuit$

Dati due ordinamenti σ^1 e σ^2 che differiscono per due lavori consecutivi i e j (in σ^1 i precede j e in σ^2 i segue j):

$$\begin{aligned}
C_{\sigma^1} - C_{\sigma^2} &= \alpha_i \sum_{k \in P(\sigma^1, i)} t_k + \alpha_j \sum_{k \in P(\sigma^1, j)} t_k - \alpha_j \sum_{k \in P(\sigma^2, j)} t_k - \alpha_i \sum_{k \in P(\sigma^2, i)} t_k \\
&= \alpha_j t_i - \alpha_i t_j = t_i t_j \left(\frac{\alpha_j}{t_j} - \frac{\alpha_i}{t_i} \right) \\
&= t_i t_j (u_j - u_i)
\end{aligned}$$

σ^2 è più conveniente se $u_j > u_i$

5.2 Modelli evoluti

- *Sequenza*
  - Più macchine (identiche o differenti)
  - Più macchine per ogni lavoro con ordine prestabilito o meno
- *Lavori*
  - Eseguibili solo dopo un certo istante (*release time*)
  - Da completare inderogabilmente prima di un certo instante (*deadline*)
  - Penale sui ritardi (*due date*)
  - Precedenze assegnate
- *Costi*
  - Massimo ritardo sui lavori
  - Somma dei ritardi
  - Costo per anticipo (*earliness*)

6 Teoria delle reti

6.1 Grafi

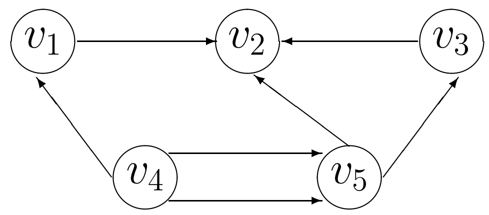
Intuitivamente un grafo è un insieme finito di punti (*nodi* o *vertici*) ed un insieme di frecce (*archi*) che uniscono coppie di punti
Il verso della freccia assegna un *orientamento* all'arco

Se non si tiene conto dell'orientamento degli archi il grafo è detto *multigrafo* o *non orientato* e gli archi sono detti *spigoli*

- p numero massimo di archi orientati nello stesso verso presenti tra due nodi (*p-grafo*)
- n numero di nodi (*ordine*)
- m numero di archi

Definizione 6.1 *Un grafo è una coppia* $G(N, A)$*, dove* $N = \{v_1, \ldots, v_n\}$ *è un insieme di elementi detti nodi o vertici e* $A = \{a_{ij} = (v_i, v_j) | v_i, v_j \in N\}$ *è una famiglia di elementi del prodotto cartesiano* $N \times N$ *detti archi.*

Esempio 6.1 (Elementi di un grafo)



$N = \{v_1, v_2, v_3, v_4, v_5\}$
$A = \{a_{12}, a_{32}, a_{41}, a_{45}, a_{45}, a_{52}, a_{53}\}$
$p = 2; n = 5; m = 7$

$\diamondsuit$

Definizione 6.2

- *Dato un arco* $a_{ij} = (v_i, v_j)$ *il nodo* v_i *è detto* nodo iniziale, primo nodo *o* predecessore del nodo v_j*; il nodo* v_j *è detto* nodo finale, secondo nodo o successore del nodo v_i*. I nodi* v_i *e* v_j *sono detti* adiacenti*. L'arco* a_{ij} *è detto* uscente da v_i *ed* entrante in v_j
- *Un arco* $a_{ii} = (v_i, v_i)$*, cioè in cui il nodo iniziale e finale coincidono è detto* cappio
- *Due archi che hanno un nodo in comune sono detti* adiacenti
- *L'insieme dei secondi nodi degli archi uscenti da un nodo* v_i *è detto* insieme dei successori di v_i *e si indica con* $\omega^+(i)$ *o semplicemente* $\omega(i)$
- *L'insieme dei primi nodi degli archi entranti in un nodo* v_i *è detto* insieme dei predecessori di v_i *e si indica con* $\omega^-(i)$*.*

- *Una sequenza di archi aventi ciascuno un nodo in comune con l'arco precedente e l'altro nodo in comune con l'arco seguente è detto* cammino
- *Un cammino in cui nessun arco viene percorso più di una volta è detto* semplice*; se nessun nodo viene incontrato più di una volta è detto* elementare
- *Un cammino semplice in cui il primo e l'ultimo nodo coincidono è detto* ciclo*; se il cammino è elementare il ciclo è detto* elementare*.*
- *Un cammino o un ciclo in cui tutti gli archi sono percorsi secondo il proprio orientamento è detto* orientato*, altrimenti è detto* non orientato
- *Se esiste un cammino tra i nodi v_i e v_j, v_j è detto* accessibile da v_i *e viceversa*
- *Un grafo $G(N, A)$ privo di cappi e in cui esiste al più un arco tra ogni coppia di nodi è detto* semplice
- *Un grafo $G(N, A)$ è detto* connesso *se ogni nodo è accessibile dagli altri*
- *Un multigrafo $G(N, A)$ connesso e privo di cicli elementari è detto* albero

- *Dato un grafo $G(N, A)$ e un sottoinsieme $A' \subset A$, il grafo $G(N, A')$ ottenuto eliminando dal grafo G gli archi dell'insieme $A \setminus A'$ è detto* grafo parziale generato da *A'.*

- *Dato un grafo connesso $G(N, A)$ un grafo parziale $G(N, A')$ connesso e privo di cicli elementari è detto* spanning tree *o* albero ricoprente *o* albero parziale*.*

- *Dato un grafo $G(N, A)$ e una bipartizione N' e N'' dell'insieme N, l'insieme degli archi aventi un estremo in N' e l'altro in N'' è detto* taglio*.*

6.2 Reti

Definizione 6.3 *Una rete è un grafo $G(N, A)$ nel quale ad ogni arco $a_{ij} \in A$ si associano tre valori interi l_{ij}, u_{ij}, c_{ij} detti rispettivamente* capacità minima, capacità massima *e* costo unitario *dell'arco e ad ogni nodo v_i si associa un valore intero b_i; se $b_i > 0$ il nodo v_i è detto* sorgente *e b_i è detta* disponibilità, *se $b_i < 0$ il nodo v_i è detto* pozzo *e b_i è detta* domanda, *se $b_i = 0$ il nodo v_i è detto* di attraversamento

- c_{ij} può indicare un costo, un peso, una lunghezza o qualsiasi altra cosa
- La condizione di integrità ha soprattutto motivazioni storiche

6.2.1 Flusso su una rete

Data una rete $G(N, A)$ si definisce *flusso sulla rete* una funzione

$$x : A \rightarrow \mathbb{Z}$$

dove x_{ij} è detto *flusso sull'arco* e sono soddisfatti i *vincoli di capacità degli archi* e i *vincoli di bilanciamento nei nodi*. Al flusso x_{ij} si associa il costo $c_{ij}x_{ij}$; la somma dei costi di tutti gli archi è detta *costo associato al flusso* x

$$\begin{array}{lll} vincoli\ di\ capacità\ degli\ archi: & l_{ij} \leq x_{ij} \leq u_{ij} & a_{ij} \in A \\ vincoli\ di\ bilanciamento\ nei\ nodi: & \sum\limits_{j \in \omega^+(i)} x_{ij} - \sum\limits_{j \in \omega^-(i)} x_{ji} = b_i & v_i \in N \\ costo\ associato\ al\ flusso: & \sum\limits_{v_i \in N} \sum\limits_{j \in \omega^+(i)} c_{ij}x_{ij} & \end{array}$$

- Il vincolo di bilanciamento nei nodi è noto anche come legge di Kirchoff

6.2.2 Problema del flusso di costo minimo

$$
\begin{aligned}
min \quad & \sum_{v_i \in N} \sum_{j \in \omega^+(i)} c_{ij} x_{ij} & \\
s.t. \quad & \sum_{j \in \omega^+(i)} x_{ij} - \sum_{j \in \omega^-(i)} x_{ji} = b_i & v_i \in N \\
& l_{ij} \le x_{ij} \le u_{ij} & a_{ij} \in A
\end{aligned}
$$

In questa forma possono essere rappresentati tutti i problemi di reti

6.3 Esempi di reti

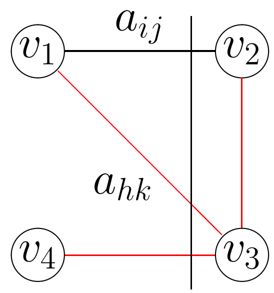
- Reti fisiche
  * Strade urbane
  * Reti elettriche
  * Circuiti VLSI (*planarità*)
- Reti di trasporto
  * Problema del trasporto
- Reti spazio-temporali
  * Problema di produzione e magazzino
  * Problema dell'orario degli equipaggi aerei
- Reti di pianificazione
  * PERT
  * Problema della miniera a cielo aperto

6.4 Minimo Spanning Tree

Trovare uno spanning tree di costo minimo
Se gli archi hanno tutti lo stesso costo la soluzione è un qualsiasi spanning tree

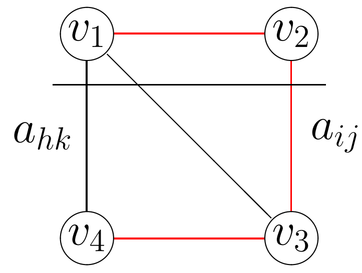
Teorema 6.1 *Dato un grafo $G(N, A)$ sia $N' \subseteq N$ un sottinsieme di vertici e sia $a_{ij} = (v_i, v_j)$ un arco di peso minimo appartenente al taglio generato da N', cioè tale che $v_i \in N'$ e $v_j \in N \setminus N'$ oppure $v_i \in N \setminus N'$ e $v_j \in N'$; allora esiste un albero di peso minimo contenente a_{ij}*

<u>Dimostrazione</u>
Per assurdo sia T' un albero non contenente a_{ij} e di peso strettamente minore di ogni albero contenente a_{ij} e sia a_{hk} l'arco di T' appartenente al taglio generato da N' e facente parte con a_{ij} di un ciclo. Sostituendo a_{ij} ad a_{hk} si ottiene un albero di peso non superiore a T'
Assurdo ♣



Teorema 6.2 *Dato un grafo $G(N, A)$ sia C un ciclo del grafo G e sia $a_{ij} = (v_i, v_j)$ un arco di peso massimo appartenente al ciclo C, allora esiste un albero di peso minimo non contenente a_{ij}*

<u>Dimostrazione</u>



Per assurdo sia T' un albero contenente a_{ij} e di peso strettamente minore di ogni albero non contenente a_{ij}, siano $N', N \setminus N' \subset N$ i sottoinsiemi del taglio di T' a cui appartiene a_{ij} e sia a_{hk} l'altro arco di C appartenente al taglio generato da N'. Sostituendo a_{hk} ad a_{ij} si ottiene un albero di peso non superiore a T'

Assurdo ♣

Algoritmo di Prim (1957)

Costruisce uno spanning tree parziale $T'(N', A')$ aggiungendo ad ogni iterazione un arco ad A' e i relativi estremi a N', fino a che $N' = N$

Algoritmo

a) $A' = \emptyset; N' = \{v_1\}$;

b) determinare l'arco $a_{ij} = (v_i, v_j)$ di peso minimo tale che $v_i \in N'$ e $v_j \in N/N'$ o $v_i \in N/N'$ e $v_j \in N'$;

c) $A' = A' \cup \{a_{ij}\}; N' = N' \cup \{v_i\} \cup \{v_j\}$;

d) se $N' \neq N$ tornare a b);
altrimenti STOP;

L'algoritmo è corretto in quanto ad ogni iterazione aggiunge l'arco di peso minimo appartenente al taglio generato da N', in accordo col Teorema 6.1

Algoritmo di Kruskal (1956)

Costruisce uno spanning tree $T(N, A')$ aggiungendo ad ogni iterazione l'arco di peso minimo che non genera cicli ad A', fino a che $card(A') = card(N) - 1$

Algoritmo

a) $A' = \emptyset$;

b) determinare l'arco $a_{ij} = (v_i, v_j)$ di peso minimo tale che $a_{ij} \in A/A'$ e non forma cicli;

c) $A' = A' \cup \{a_{ij}\}$;

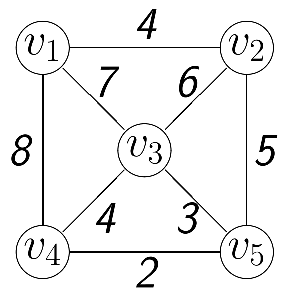
d) se card$(A') < card(N) - 1$ tornare a b);
altrimenti STOP

L'algoritmo è corretto in quanto ciascun arco di G che non compare in T genererebbe un ciclo rispetto al quale sarebbe l'arco di peso massimo, in accordo col Teorema 6.2

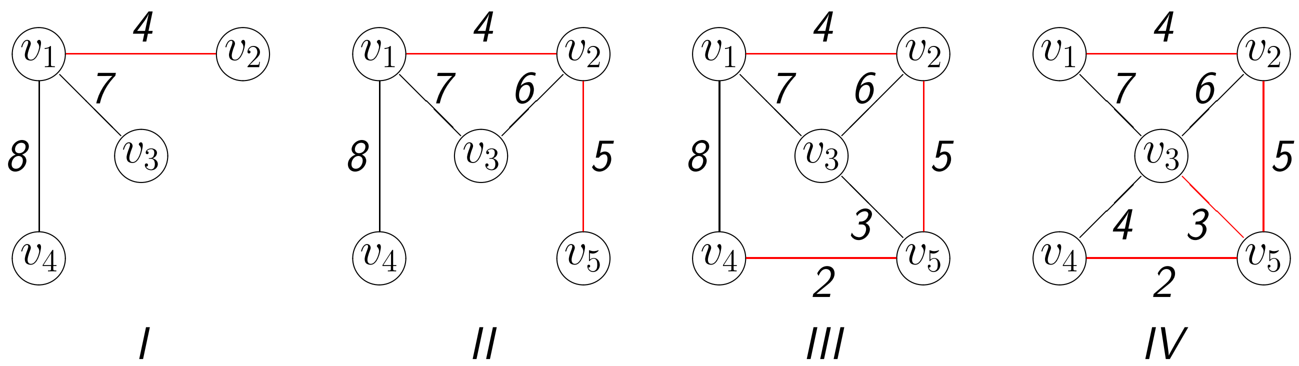
- Esiste una variante dell'algoritmo di Kruskal che elimina l'arco di peso massimo che non fa perdere la connessione

Esempio 6.2 (Minimo Spanning Tree)

Determinare uno spanning tree di peso minimo per il grafo a lato, esaminando gli archi e i vertici in ordine di indice crescente.



Algoritmo di Prim



0 $A' = \emptyset; N' = \{v_1\}$

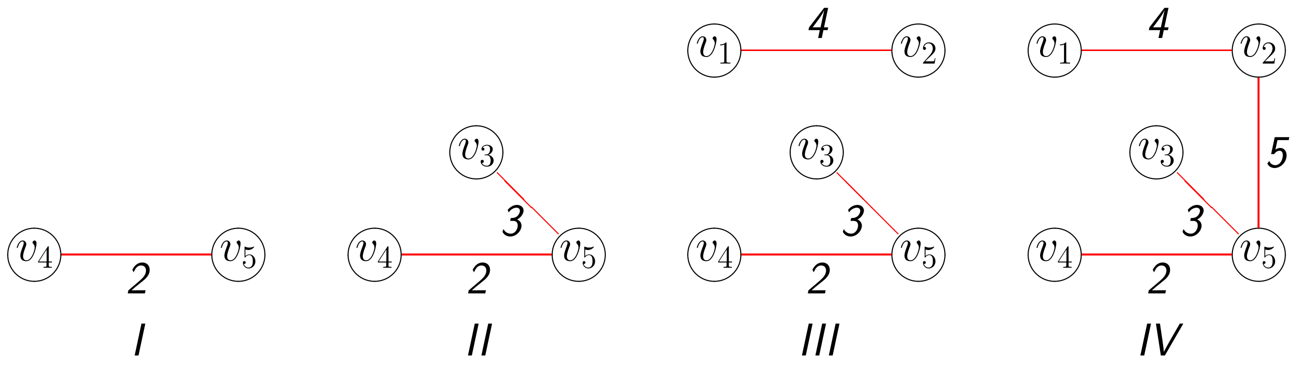
I $A' = \{a_{12}\}; N' = \{v_1, v_2\}$

II $A' = \{a_{12}, a_{25}\}; N' = \{v_1, v_2, v_5\}$

III $A' = \{a_{12}, a_{25}, a_{45}\}; N' = \{v_1, v_2, v_5, v_4\}$

IV $A' = \{a_{12}, a_{25}, a_{45}, a_{35}\}; N' = \{v_1, v_2, v_5, v_4, v_3\}$; STOP ($N' = N$)

Algoritmo di Kruskal



0 $A' = \emptyset$

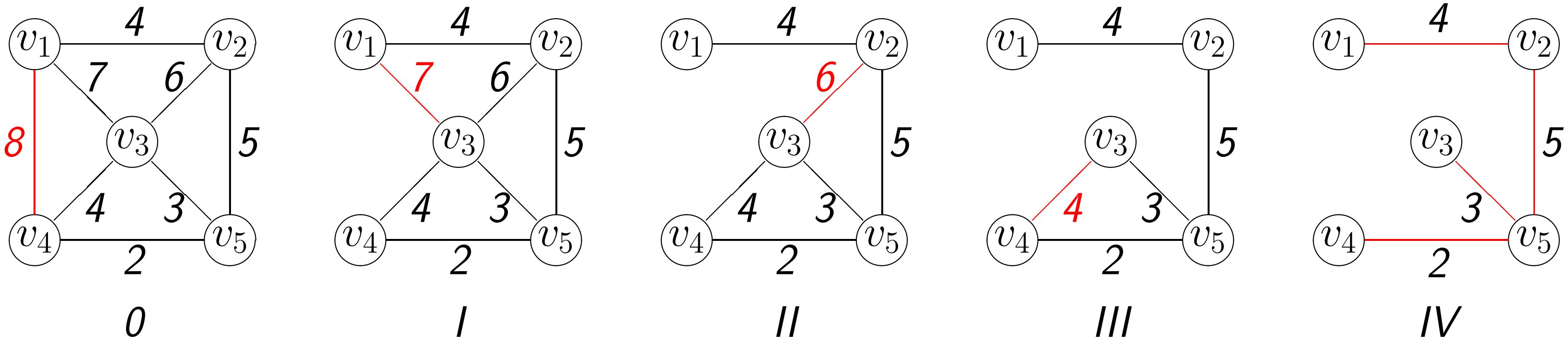
I $A' = \{a_{45}\}$

II $A' = \{a_{45}, a_{35}\}$

III $A' = \{a_{45}, a_{35}, a_{12}\}$; l'arco a_{34} di peso 4 non viene inserito perchè formerebbe il ciclo $v_3 - v_4 - v_5 - v_3$

IV $A' = \{a_{45}, a_{35}, a_{12}, a_{25}\}$; STOP ($card(A') = card(N) - 1$) $\diamondsuit$

Algoritmo di Kruskal modificato



0 Grafo iniziale

I Si elimina a_{14} di peso 8

II Si elimina a_{13} di peso 7

III Si elimina a_{23} di peso 6

IV Non si eliminano a_{23} di peso 5 e a_{12} di peso 4 perchè si perderebbe la connessione; si elimina a_{34} di peso 4; STOP (sono stati eliminati $card(A) - card(N) + 1$ archi) $\diamondsuit$

6.5 Cammino minimo

Trovare il cammino orientato di costo minimo tra due nodi assegnati
Se gli archi hanno tutti lo stesso costo la soluzione è il cammino con il minimo numero di archi, altrimenti è possibile che un cammino con più archi risulti migliore

E' un problema semplice ma servono algoritmi efficienti perchè è un sottoproblema di molti altri problemi (reti di trasporto, reti di comunicazione, percorsi veicolari, progettazione di reti, ecc.)

Per memorizzare i cammini di costo minimo si utilizza un puntatore che per ogni nodo indica l'unico predecessore

6.5.1 SPP da un nodo v_s a tutti gli altri nodi

Ad ogni nodo v_i si assegnano un valore o etichetta $d(i)$ che indica il costo del cammino corrente da v_s a v_i e un puntatore $pred(i)$ che indica il predecessore di v_i nel cammino corrente
Si parte da un valore temporaneo per $d(i)$ che può essere modificato ad ogni iterazione fino ad ottenere il valore esatto

Esistono due classi di algoritmi:

- algoritmi label setting o di assegnazione
  modificano le etichette in modo tale che ad ogni iterazione almeno una di esse diventi esatta; il più noto è l'algoritmo di Dijkstra
- algoritmi label correcting o di correzione
  modificano le etichette in modo tale che solo all'ultima iterazione si è certi dell'esattezza di tutte le etichette; i più noti algoritmi sono quello di Ford e la variante di Bellman

Gli algoritmi di assegnazione sono più efficienti nei casi peggiori, invece gli algoritmi di correzione non necessitano di archi a costo non negativo

Algoritmo di Dijkstra (1959)

Algoritmo

a) $d(s) = 0$;

b) per $i = 1, ..., n$ $d(i) = c_{si}$ e $pred(i) = s$;

c) sia $d(h) = min\{d(i)|d(i)$ non è esatta$\}$ ($d(h)$ diventa esatta);

d) se $v_i \in \omega(h)$ e $d(i)$ non è esatta $d(i) = min\{d(i), d(h) + c_{hi}\}$ eventualmente, se $d(i)$ è stata modificata, $pred(i) = h$;

e) se tutti i valori $d(i)$ sono esatti STOP ($n - 1$ iterazioni);
altrimenti tornare a c);

L'algoritmo è corretto in quanto ad ogni iterazione le etichette temporanee (che rappresentano il costo del cammino minimo che utilizza solo nodi con etichetta esatta) sono tutte non minori di quelle esatte; per il nodo v_h con etichetta temporanea minima ogni altro cammino deve comprendere almeno un nodo v_k con etichetta non esatta, per cui il costo è non minore di quello da v_s a v_k più il costo c_{kh}. Se qualche arco ha costo negativo la prova di correttezza non sussiste

Algoritmo di Ford (1956)

Algoritmo

a) $d(s) = 0$;

b) per $i = 1, ..., n$ $d(i) = M$ e $pred(i) = s$ (M maggiorante delle etichette esatte);

c) se esiste a_{hi} con $d(i) > d(h) + c_{hi}$ porre $d(i) = d(h) + c_{hi}$ e $pred(i) = h$ e ripetere c); altrimenti STOP (le etichette sono tutte esatte);

L'algoritmo è corretto in quanto dopo ogni aggiornamento l'etichetta $d(i)$ è ancora un maggiorante del cammino minimo da v_s a v_i, poichè lo è $d(h)$. D'altra parte al termine dell'algoritmo preso un qualsiasi cammino C da v_s a v_i si ha che $d(i) \leq \sum_{(v_k,v_l)\in C} c_{kl}$ per cui il valore finale di $d(i)$ è un minorante del cammino minimo da v_s a v_i e quindi $d(i)$ è esatta

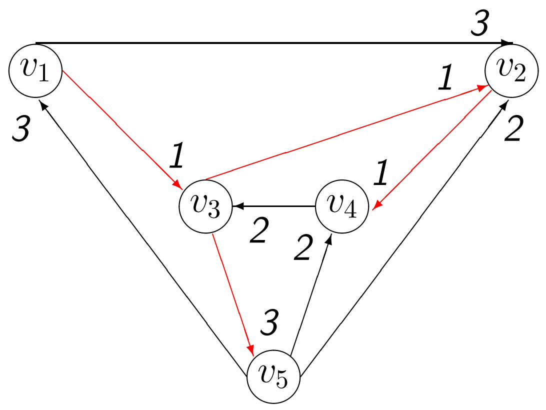
Variante di Bellman (1958)

Si esaminano gli archi secondo un ordinamento prefissato

L'algoritmo termina dopo aver esaminato gli archi al più $n - 1$ volte

Determina l'esistenza di cicli di costo negativo, eseguendo una ulteriore iterazione (cammino composto da almeno n archi)

Esempio 6.3 (Grafo con archi di peso non negativo)



Algoritmo di Dijkstra

d	0	99	99	99	99	**pred**	1	1	1	1	1	**h** = 1
d	0	3	1	99	99	**pred**	1	1	1	1	1	**h** = 3
d	0	2	1	99	4	**pred**	1	3	1	1	3	**h** = 2
d	0	2	1	3	4	**pred**	1	3	1	2	3	**h** = 4
d	0	2	1	3	4	**pred**	1	3	1	2	3	**h** = 5
d	0	2	1	3	4	**pred**	1	3	1	2	3	

STOP

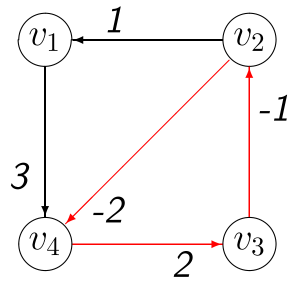
Algoritmo di Bellmann

Ordinamento per indici crescenti

d	*0*	*99*	*99*	*99*	*99*	**pred**	*1*	*1*	*1*	*1*	*1*	a_{12}
d	*0*	*3*	*99*	*99*	*99*	**pred**	*1*	*1*	*1*	*1*	*1*	a_{13}
d	*0*	*3*	*1*	*99*	*99*	**pred**	*1*	*1*	*1*	*1*	*1*	a_{24}
d	*0*	*3*	*1*	*4*	*99*	**pred**	*1*	*1*	*1*	*2*	*1*	a_{32}
d	*0*	*2*	*1*	*4*	*99*	**pred**	*1*	*3*	*1*	*2*	*1*	a_{35}
d	*0*	*2*	*1*	*4*	*4*	**pred**	*1*	*3*	*1*	*2*	*3*	$a_{43}, a_{51}, a_{52}, a_{54}$ *(1)* a_{12}, a_{13}, a_{24},
d	*0*	*2*	*1*	*3*	*4*	**pred**	*1*	*3*	*1*	*2*	*3*	$a_{32}, a_{35}, a_{43}, a_{51}, a_{52}, a_{54}$ *(2)* $a_{12}, a_{13}, a_{24}, a_{32}$ *[*$a_{35}, a_{43}, a_{51}, a_{52}, a_{54}$ *(3)]*
d	*0*	*2*	*1*	*3*	*4*	**pred**	*1*	*3*	*1*	*2*	*3*	

STOP ◊

Esempio 6.4 (Grafo con ciclo negativo)



Algoritmo di Dijkstra

d	0	99	99	99	**pred**	1	1	1	1	**h** = 1
d	0	99	99	3	**pred**	1	1	1	1	**h** = 4
d	0	99	5	3	**pred**	1	1	4	1	**h** = 3
d	0	4	5	3	**pred**	1	3	4	1	**h** = 2
d	0	4	5	3	**pred**	1	3	4	1	

STOP (non identificato ciclo negativo)

Algoritmo di Bellmann

Ordinamento per indici crescenti

d	*0*	*99*	*99*	*99*	**pred**	*1*	*1*	*1*	*1*	a_{14}
d	*0*	*99*	*99*	*3*	**pred**	*1*	*1*	*1*	*1*	a_{21}, a_{24}, a_{32}
d	*0*	*98*	*99*	*3*	**pred**	*1*	*3*	*1*	*1*	a_{43}
d	*0*	*98*	*5*	*3*	**pred**	*1*	*3*	*4*	*1*	*(1)* $a_{14}, a_{21}, a_{24}, a_{32}$
d	*0*	*4*	*5*	*3*	**pred**	*1*	*3*	*4*	*1*	a_{43} *(2)* a_{14}, a_{21}, a_{24}
d	*0*	*4*	*5*	*2*	**pred**	*1*	*3*	*4*	*2*	a_{32}, a_{43}
d	*0*	*4*	*4*	*2*	**pred**	*1*	*3*	*4*	*2*	*(3)* $a_{14}, a_{21}, a_{24}, a_{32}$

STOP (identificato ciclo negativo) $\diamondsuit$

6.5.2 SPP tra qualsiasi coppia di nodi

Si può applicare uno degli algoritmi precedenti assumendo ogni volta un nodo iniziale diverso, oppure si possono usare algoritmi specifici, ad esempio l'algoritmo di Floyd (1962)

6.6 Flusso massimo

Data una rete di trasporto $G(N, A)$, cioè:

- senza cappi
- con una unica sorgente v_s priva di archi entranti e con un unico pozzo v_t privo di archi uscenti, per i quali non sono definite la disponibilità e la domanda
- con gli altri nodi di attraversamento (cioè con domanda nulla)

determinare un flusso x che soddisfa i vincoli e per cui sia massimo il *valore del flusso da* v_s *a* v_t, indicato con $F(v_s, v_t)$:

$$\sum_{i \neq s,t} \left(\sum_{j \in \omega^+(i)} x_{ij} - \sum_{j \in \omega^-(i)} x_{ji} \right) + \sum_{j \in \omega^+(s)} x_{sj} - \sum_{j \in \omega^-(t)} x_{jt} = 0$$

per la legge di Kirchoff la prima sommatoria è nulla, per cui si ha:

$$\sum_{j \in \omega^+(s)} x_{sj} = \sum_{j \in \omega^-(t)} x_{jt} = F(v_s, v_t)$$

E' utile anche per valutare l'affidabilità di una rete (indisponibilità di qualche arco)

Problema di programmazione lineare a numeri interi:

$$
\begin{array}{lll}
max & z = F & \\
s.t. & \sum_{j \in \omega^+(s)} x_{sj} - F = 0 & \\
 & \sum_{j \in \omega^+(i)} x_{ij} - \sum_{j \in \omega^-(i)} x_{ji} = 0 & v_i \in N \setminus \{v_s, v_t\} \\
 & - \sum_{j \in \omega^-(t)} x_{jt} + F = 0 & \\
 & l_{ij} \leq x_{ij} \leq u_{ij}; x_{ij} \; intero & a_{ij} \in A
\end{array}
$$

Algoritmi di programmazione lineare o algoritmi specifici

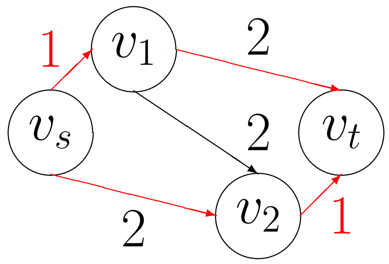
Algoritmo del contrassegno (Ford e Fulkerson, 1956): costruisce una sequenza di cammini dalla sorgente v_s al pozzo v_t sui quali incrementare il flusso fino a raggiungere il massimo

Algoritmo del contrassegno

a) inizializzare x con un flusso ammissibile (se le capacità minime sono tutte nulle il flusso nullo è ammissibile);

b) $V = \{v_s\}$ (V è l'insieme dei nodi contrassegnati);

c) se esistono due nodi adiacenti $v_i \in V$ e $v_j \notin V$ contrassegnare v_j
con $+i$ se $j \in \omega^+(i)$ e $x_{ij} < u_{ij}$ oppure
con $-i$ se $j \in \omega^-(i)$ e $x_{ji} > l_{ji}$ e andare a d);
altrimenti STOP (flusso massimo);



d) se $v_t \in V$ esiste un cammino non orientato C da v_s a v_t sul quale si può variare il flusso di Δ andare a e);
altrimenti tornare a c);

e) costruire un nuovo flusso x ponendo:

$$x_{ij} = \begin{cases} x_{ij} & \text{se } a_{ij} \notin C \\ x_{ij} + \Delta & \text{se } a_{ij} \in C \text{ secondo l'orientamento} \\ x_{ij} - \Delta & \text{se } a_{ij} \in C \text{ non secondo l'orientamento} \end{cases}$$

tornare a b);

L'algoritmo è corretto in quanto preso $U \subset N$ qualsiasi con $v_s \in U$, $v_t \notin U$ si ha:

$$F(v_s, v_t) = \sum_{j \in \omega^+(s)} x_{sj} + \sum_{v_i \in U, i \neq s} \left\{ \sum_{j \in \omega^+(i)} x_{ij} - \sum_{j \in \omega^-(i)} x_{ji} \right\}$$

Eliminando i termini di segno opposto si ha:

$$F(v_s, v_t) = \sum_{v_i \in U, v_j \notin U} x_{ij} - \sum_{v_i \in U, v_j \notin U} x_{ji} \leq \sum_{v_i \in U, v_j \notin U} u_{ij} - \sum_{v_i \in U, v_j \notin U} l_{ji}$$

Preso $V = \{$nodi contrassegnati al termine dell'algoritmo$\}$ si ha:

$$F(v_s, v_t) = \sum_{v_i \in V, v_j \notin V} x_{ij} - \sum_{v_i \in V, v_j \notin V} x_{ji} = \sum_{v_i \in V, v_j \notin V} u_{ij} - \sum_{v_i \in V, v_j \notin V} l_{ji}$$

Definizione 6.4 *Data una rete di trasporto* $G(N, A)$ *e il taglio generato da un insieme* $U \subset N$*, con* $v_s \in U$ *e* $v_t \notin U$ *si chiama capacità del taglio la quantità* $\sum_{v_i \in U, v_j \notin U} u_{ij} - \sum_{v_i \in U, v_j \notin U} l_{ji}$

Teorema 6.3 (Teorema di Ford e Fulkerson (max flow - min cut, 1956)) *Data una rete di trasporto* $G(N, A)$ *si ha:*

$$max\ F(v_s, v_t) = min \left\{ \sum_{v_i \in U, v_j \notin U} u_{ij} - \sum_{v_i \in U, v_j \notin U} l_{ji} | U \subset N, v_s \in U, v_t \notin U \right\}$$

Altri algoritmi di cammino aumentante

- Algoritmo del minimo cammino aumentante (Edmonds e Karp, 1972)
- Algoritmo delle reti stratificate (Dinic, 1970 - Ahuja e Orlin, 1991)

Algoritmi di preflusso

Sono algoritmi più flessibili che consentono di aumentare il flusso non su un cammino ma su un singolo arco con una operazione di invio (*push*)

- Algoritmo di preflusso (Karzanov, 1974 - Goldberg e Tarjan, 1986)
- Variante (Goldberg e Tarjan, 1986)
- Algoritmo di scaling dell'eccesso (Ahuja e Orlin, 1991)

Problema del flusso compatibile

Definizione 6.5 *Si definisce distanza di un flusso x dalle capacità l e u la quantità:*

$$d(x) = \sum_{a_{ij} \in A} d(x_{ij})$$

dove $d(x_{ij}) = \begin{cases} l_{ij} - x_{ij} & \textit{se } x_{ij} < l_{ij} \\ \textit{0} & \textit{se } l_{ij} \leq x_{ij} \leq u_{ij} \\ x_{ij} - u_{ij} & \textit{se } x_{ij} > u_{ij} \end{cases}$

Si noti che $d(x) \geq 0$ e in particolare $d(x) = 0 \Leftrightarrow x$ è ammissibile

Algoritmo di Herz (1967)

a) inizializzare x in modo che siano soddisfatti i vincoli di conservazione del flusso ma non (necessariamente) quelli di capacità (flusso nullo);

b) se esiste $x_{hk} < l_{hk}$ contrassegnare v_k con $+h$ e andare a c);

b') se esiste $x_{kh} > u_{kh}$ contrassegnare v_k con $-h$;

c) se si contrassegna v_t si contrassegna anche v_s con $+t$;

c') se si contrassegna v_s si contrassegna anche v_t con $-s$;

d) se esistono due nodi adiacenti v_i contrassegnato e v_j non contrassegnato, si contrassegna v_j con $+i$ se $j \in \omega^+(i)$ e $x_{ij} < u_{ij}$ oppure con $-i$ se $j \in \omega-(i)$ e $x_{ji} > l_{ji}$ e andare a e);
altrimenti STOP (non esiste un flusso ammissibile);

e) se si contrassegna v_h esiste un ciclo non orientato C sul quale si può variare il flusso di Δ e andare a f);
altrimenti tornare a c);

f) costruire un nuovo flusso x ponendo:

$$x_{ij} = \begin{cases} x_{ij} & \text{se } a_{ij} \notin C \\ x_{ij} + \Delta & \text{se } a_{ij} \in C \text{ secondo l'orientamento} \\ x_{ij} - \Delta & \text{se } a_{ij} \in C \text{ non secondo l'orientamento} \end{cases}$$

g) se $d(x) = 0$ STOP (il flusso x è ammissibile)
altrimenti tornare a b);

L'algoritmo converge poichè ad ogni iterazione la distanza $d(x)$ si riduce

Teorema 6.4 (Teorema di Hoffman, 1960)
Data una rete di trasporto $G(N, A)$ esiste un flusso ammissibile x se e solo se preso comunque $U \subset N$,con $v_s, v_t \in U$ oppure $v_s, v_t \notin U$ si ha:

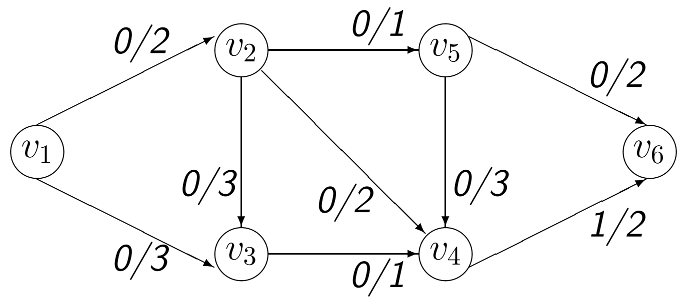
$$\sum_{v_i \in U} \sum_{j \in \omega^+(i)} l_{ij} \leq \sum_{v_i \in U} \sum_{j \in \omega^-(i)} u_{ji}$$

Esempio 6.5 *Si considerino le seguenti due situazioni di flusso impossibile:*

a) $v_s \xrightarrow{1/2} v_1 \xrightarrow{3/5} v_t$ *b)* $v_s \xrightarrow{2/4} v_1 \xrightarrow{0/1} v_t$

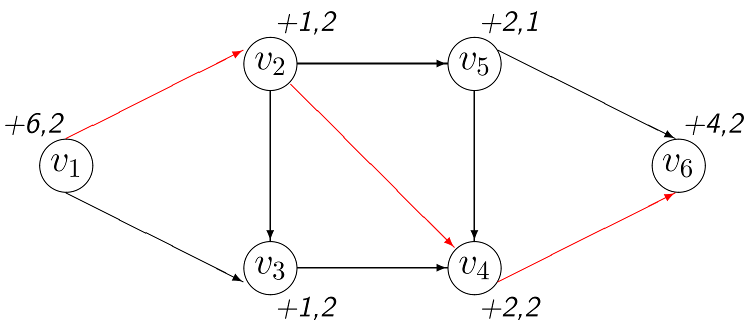
Nel caso a) l'insieme $U = \{v_1\}$ e nel caso b) l'insieme $U = \{v_s, v_t\}$ non verificano le ipotesi del teorema $\diamondsuit$

Esempio 6.6 (Flusso massimo)



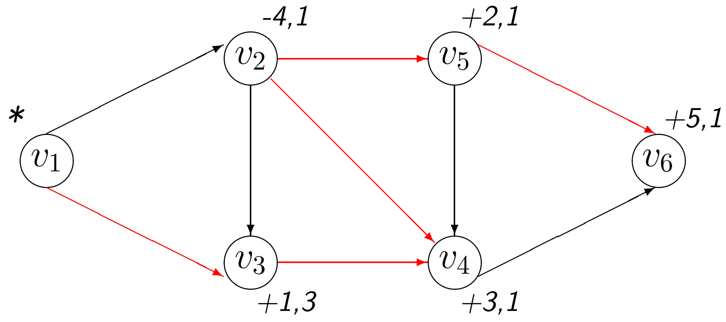
- *si esaminano nodi e archi secondo l'ordine crescente degli indici*
- *si contrassegnano tutti i nodi possibili*
- *al contrassegno si aggiunge il massimo incremento corrente*

Algoritmo di Herz dall'arco a_{46}

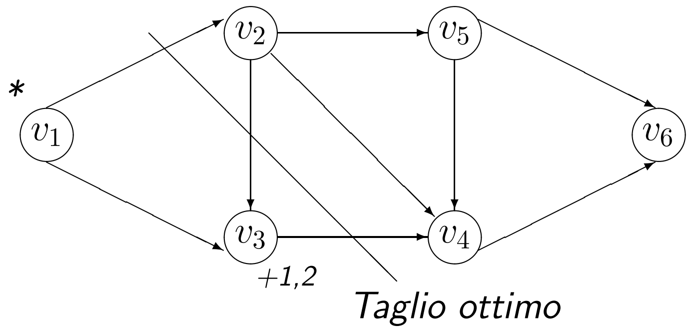


Ciclo: $v_4 - v_6 - v_1 - v_2 - v_4$; $\Delta = 2$
Flusso ammissibile

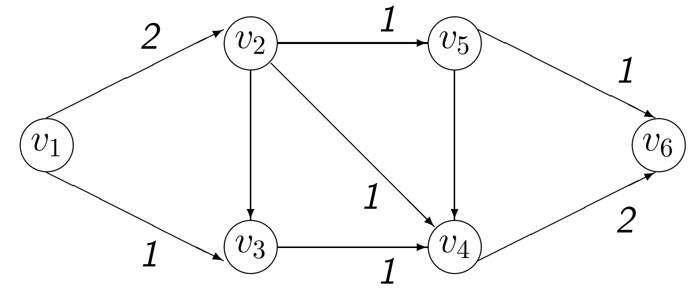
Algoritmo di Ford e Fulkerson



Cammino aumentante: $v_1 - v_3 - v_4 - v_2 - v_5 - v_6$; $\Delta = 1$



Flusso massimo



Taglio minimo $(v_1, v_2), (v_2, v_3), (v_3, v_4)$; *capacità 3* $\diamondsuit$

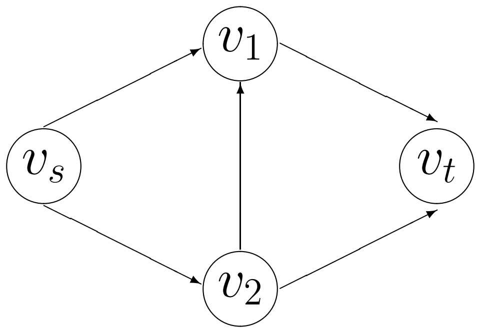
Relazione di dualità tra flusso massimo e taglio minimo

Trascurando il vincolo di integrità del flusso il duale è:

$$
\begin{array}{lll}
min & w = \sum_{a_{ij}\in A} u_{ij}\alpha_{ij} + \sum_{a_{ij}\in A} l_{ij}\beta_{ij} & \\
s.t. & y_i - y_j + \alpha_{ij} + \beta_{ij} = 0 & a_{ij} \in A \\
 & y_t - y_s = 1 & \\
 & \alpha_{ij} \geq 0 & a_{ij} \in A \\
 & \beta_{ij} \leq 0 & a_{ij} \in A
\end{array}
$$

- y_i sono libere e possono essere considerate variabili 0-1 col significato $y_i = 0$ se v_i appartiene all'insieme U che genera il taglio e $y_i = 1$ altrimenti
- $y_t - y_s = 1$ assicura $y_t = 1$ e $y_s = 0$, cioè $v_s \in U, v_t \notin U$

Esempio 6.7 (Max flow - min cut)



	x_{s1}	x_{s2}	x_{1t}	x_{21}	x_{2t}	F		min
y_s	1	1				-1	$=$	0
y_1	-1		1	-1			$=$	0
y_2		-1		1	1		$=$	0
y_t			-1		-1	1	$=$	0
α_{s1}	1						$\leq$	u_{s1}
α_{s2}		1					$\leq$	u_{s2}
α_{1t}			1				$\leq$	u_{1t}
α_{21}				1			$\leq$	u_{21}
α_{2t}					1		$\leq$	u_{2t}
β_{s1}	1						$\geq$	l_{s1}
β_{s2}		1					$\geq$	l_{s2}
β_{1t}			1				$\geq$	l_{1t}
β_{21}				1			$\geq$	l_{21}
β_{2t}					1		$\geq$	l_{2t}
	$=$	$=$	$=$	$=$	$=$	$=$		
max	0	0	0	0	0	1		

Dato un taglio generato da un insieme $U \subset N$, per un arco $a_{ij} \in A$ sono possibili i seguenti casi:

1. $v_i \in U, v_j \notin U \Rightarrow y_i = 0, y_j = 1$; *allora per l'ammissibilità si ha* $\alpha_{ij} + \beta_{ij} = 1$ *e per l'ottimalità si ha* $\alpha_{ij} = 1, \beta_{ij} = 0$
2. $v_i \notin U, v_j \in U \Rightarrow y_i = 1, y_j = 0$; *allora per l'ammissibilità si ha* $\alpha_{ij} + \beta_{ij} = -1$ *e per l'ottimalità si ha* $\alpha_{ij} = 0, \beta_{ij} = -1$
3. $v_i, v_j \in U$ *oppure* $v_i, v_j \notin U \Rightarrow y_i = y_j$; *allora per l'ammissibilità si ha* $\alpha_{ij} + \beta_{ij} = 0$ *e per l'ottimalità si ha* $\alpha_{ij} = 0, \beta_{ij} = 0$

Quindi il valore della funzione obiettivo è la capacità del taglio e la soluzione ottimale è un taglio di capacità minima $\diamondsuit$

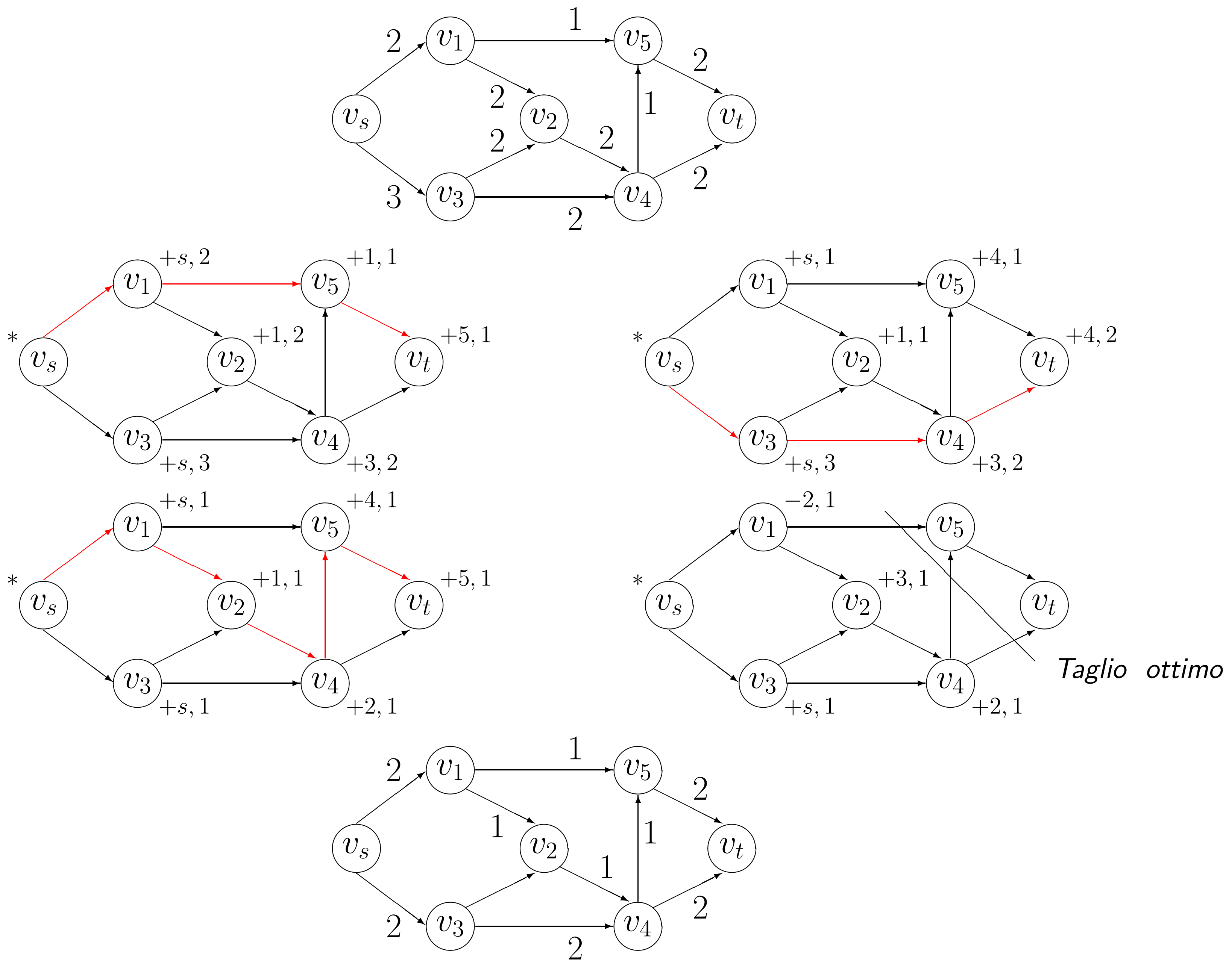
Il Teorema di Ford e Fulkerson è un caso particolare del I Teorema della dualità

6.7 Algoritmo del minimo cammino aumentante (Edmonds e Karp, 1972)

Si analizzano i nodi etichettati in ampiezza (ordine di etichettatura)
E' sufficiente modificare il passo c) nel modo seguente:

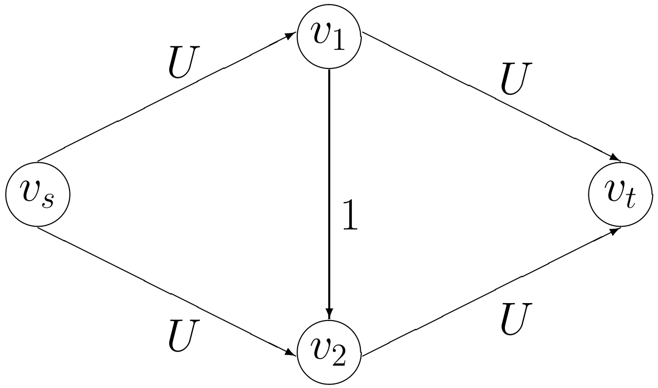
c') se $V \neq \emptyset$ detto v_i il primo nodo di V secondo l'ordine di etichettatura, contrassegnare tutti i nodi adiacenti $v_j \notin V$ con $+$i se $j \in w^+(i)$ e $x_{ij} < u_{ij}$ oppure con -i se $j \in w^-(i)$ e $x_{ij} > l_{ij}$; eliminare v_i da V e andare a d);

Esempio 6.8 (Algoritmo di Edmonds e Karp)



$\diamondsuit$

Esempio 6.9 (Confronto di complessità)



L'algoritmo di Edmonds e Karp determina i cammini aumentanti $v_s - v_1 - v_t$ *e* $v_s - v_2 - v_t$*, entrambi con incremento* U

L'algoritmo di Ford e Fulkerson può determinare sequenzialmente i cammini aumentanti $v_s - v_1 - v_2 - v_t$ *e* $v_s - v_2 - v_1 - v_t$ *con incremento unitario, richiedendo complessivamente* $2U$ *cammini* $\diamondsuit$

6.8 Complessità computazionale degli algoritmi di flusso massimo

Si consideri una rete con n nodi e m archi:

Algoritmo di Ford e Fulkerson

- Analisi degli archi per ogni aggiornamento $=$ O(m)
- Aggiornamento del flusso $=$ O(n) - Ogni cammino contiene al più $n-1$ archi
- Numero degli aggiornamenti $=$ O(nU) - U è il massimo delle capacità massime degli archi; il flusso viene aumentato di almeno un'unità e il taglio generato da v_s e $N \setminus \{v_s\}$ contiene al più n archi

Complessivamente si ha O($(n+m)nU$) e quindi l'algoritmo ha complessità O(nmU)

Algoritmo di Edmonds e Karp

- Analisi degli archi per ogni aggiornamento $=$ O(m)
- Aggiornamento del flusso $=$ O(n) - Ogni cammino contiene al più $n-1$ archi
- Numero dei cammini minimi $=$ O(nm) - Per ogni lunghezza ci sono al più O(m) cammini; infatti un arco a_{ij} viene saturato due volte con cammini di lunghezza k se esiste un cammino di lunghezza k contenente l'arco a_{ij} nel verso opposto all'orientamento, ma allora esisterebbe un cammino di lunghezza minore di k.

Complessivamente si ha O($(n+m)nm$) e quindi l'algoritmo ha complessità O(nm^2)

Reti sparse ($m = O(n)$) e dense ($m = O(n^2)$):

algoritmo	rete sparsa	rete densa
Ford e Fulkerson	$O(n^2U)$	$O(n^3U)$
Edmonds e Karp	$O(n^3)$	$O(n^5)$

- Se $U =$ O(n) i due algoritmi sono equivalenti per le reti sparse
- Teorema della decomposizione del flusso: *Dati due flussi ammissibili, relativi alla stessa rete, è sempre possibile passare dall'uno all'altro decomponendo il flusso in al più $n + m$ cammini e cicli di cui al più m cicli* fornisce un limite inferiore per il numero di iterazioni di un algoritmo nel caso peggiore. Il Teorema assicura l'esistenza, ma dal punto di vista pratico non da indicazioni sul come determinare i cammini e i cicli

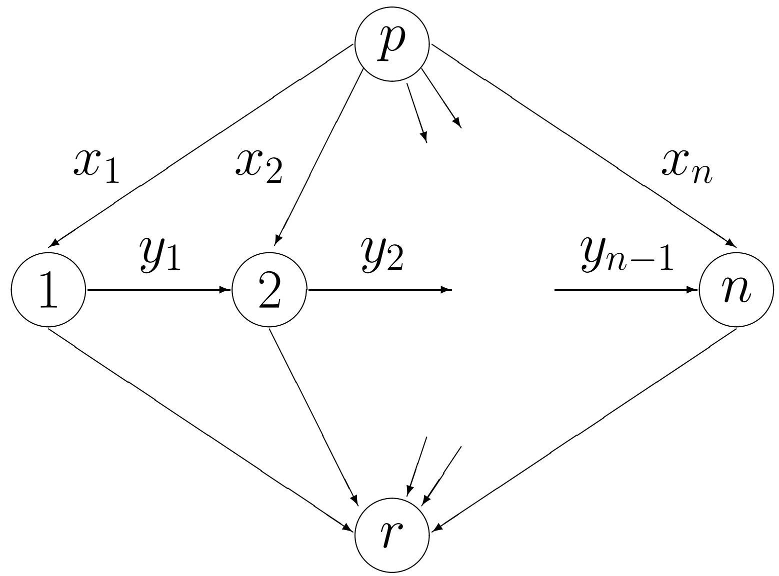
6.9 Problema di produzione e magazzino

Problema di produzione e magazzino

Una azienda produce per n periodi consecutivi; per ciascun periodo $i = 1, ..., n$ sono noti:

- d_i domanda
- c_i costo di produzione unitario
- h_i costo di immagazzinamento unitario
- C_i capacità produttiva massima
- H_i capacità del magazzino

$$
\begin{array}{ll}
c_{pi} = c_i & i = 1, ..., n \\
c_{ir} = 0 & i = 1, ..., n \\
c_{i,i+1} = h_i & i = 1, ..., n-1 \\
l_{ij} = 0 & a_{ij} \in A \\
u_{pi} = C_i & i = 1, ..., n \\
u_{ir} = d_i & i = 1, ..., n \\
u_{i,i+1} = H_i & i = 1, ..., n-1 \\
b_p = \sum_{i=1,...,n} d_i & \\
b_r = -\sum_{i=1,...,n} d_i & \\
b_i = 0 & i = 1, ..., n
\end{array}
$$



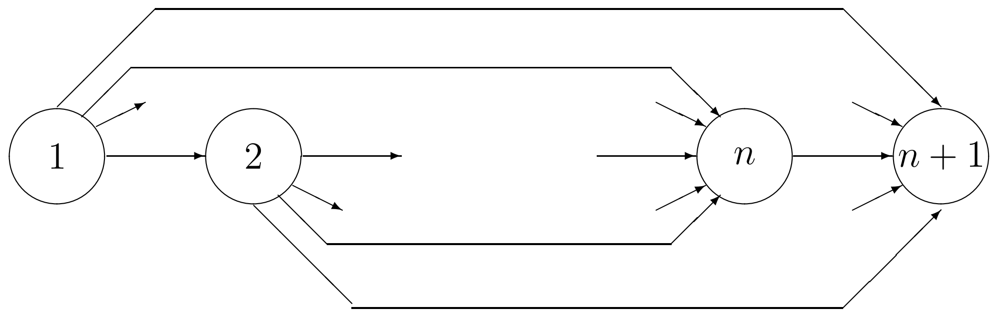
I vincoli di bilanciamento nel nodo i sono i vincoli dinamici del problema del magazzino:

$$x_i + y_{i-1} = d_i + y_i \qquad i = 1, ..., n$$

Problema di flusso di costo minimo da p a r

Se la capacità produttiva e la capacità del magazzino non sono limitate si ottiene un problema di cammino minimo da p a ciascun i: per ciascun periodo si attinge al magazzino o alla produzione

Se per ogni periodo $i = 1, ..., n$ esiste un costo fisso di produzione F_i si costruisce la seguente rete:



$$c_{ij} = F_i + \sum_{k=i,...,j-1} c_i d_k + \sum_{h=i,...,j-2} \sum_{k=h+1,...,j-1} h_h d_k \qquad \begin{array}{l} i = 1, ..., n \\ j = i+1, ..., n+1 \end{array}$$

Problema di cammino minimo da 1 a $n + 1$

Gli archi facenti parte della soluzione indicano in quali periodi produrre e per quanti periodi

6.10 Problema di routing e scheduling

Variante del problema del trasporto con itinerari reali e operazioni di carico e scarico
Si suppone un'unica origine o deposito (*depot*) ma più destinazioni
E' possibile che un unico viaggio non consenta di soddisfare tutte le richieste
Se le operazioni presso le destinazioni devono rispettare vincoli temporali (*finestre*) si ha un problema di routing e scheduling
La complessità del problema può diventare molto elevata

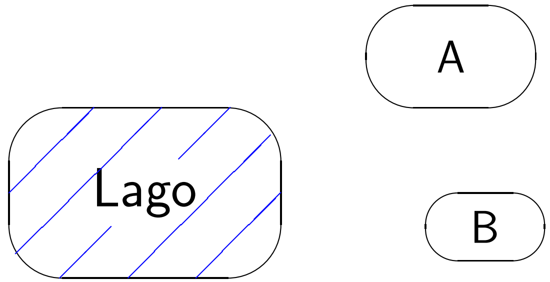
7 Teoria dei giochi e utilità

7.1 Esempio preliminare (da Young)

Due paesi A e B, aventi rispettivamente 3.600 e 1.200 abitanti, vogliono costruire un acquedotto attingendo allo stesso lago

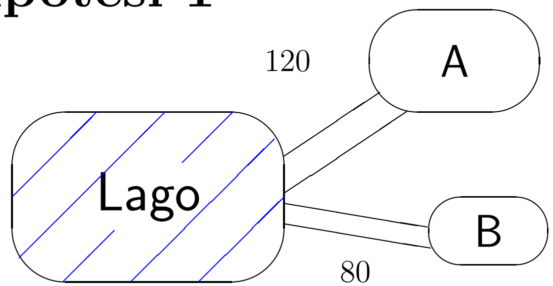
<u>Modello di programmazione matematica</u>

min *Spesa di costruzione*
s.t. *Collegare A al lago*
Rispettare le specifiche di A
Collegare B al lago
Rispettare le specifiche di B

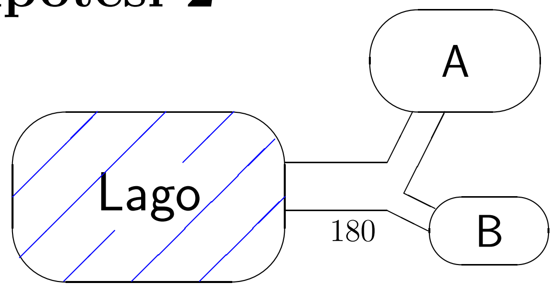


Le soluzioni ammissibili possono essere raggruppate in due sottoinsiemi che corrispondono alle ipotesi 1 e 2

Ipotesi 1



Ipotesi 2



La soluzione ottimale è:

x^* = *ipotesi 2*

z^* = *180*

La soluzione è attuabile se i paesi sono disponibili ad accordarsi per una realizzazione in comune, ma è necessario stabilire come ripartire la spesa

	Criterio	A	B
1	*Uguale divisione dei costi tra i paesi*	*90*	*90*
2	*Uguale divisione dei costi tra gli abitanti*	*135*	*45*
3	*Uguale divisione del risparmio tra i paesi*	*110*	*70*
4	*Uguale divisione del risparmio tra gli abitanti*	*105*	*75*
5	*Divisione dei costi (e del risparmio) in proporzione all'acquedotto singolo*	*108*	*72*

Il criterio 1 è il più vantaggioso per il paese A ma è rifiutato dal paese B

Il criterio 2 è il più vantaggioso per il paese B ma è rifiutato dal paese A

Gli altri criteri risultano più o meno vantaggiosi per i due paesi ma nessuno dei due può rifiutarne a priori qualcuno (A preferisce il criterio 4 e B preferisce il criterio 3)

La programmazione matematica non fornisce una metodologia di scelta; la Teoria dei Giochi non fornisce "la" soluzione, ma propone una soluzione (*solution concept*)

7.2 Introduzione

La Teoria dei Giochi tratta le situazioni in cui il risultato dipende dalle scelte fatte da più persone, dette *giocatori*, che operano perseguendo obiettivi che possono risultare comuni, ma non identici, differenti ed eventualmente contrastanti; possono essere presenti anche aspetti aleatori

Il nome deriva da *Theory of Games and Economic Behavior* di von Neumann e Morgenstern (1944)

Esempio 7.1 (Dilemma del prigioniero)

I/II	C	NC
C	$-5, -5$	$-1, -6$
NC	$-6, -1$	$-2, -2$

$\diamondsuit$

Esempio 7.2 (Battaglia dei sessi)

I/II	T	P
T	$2, 1$	$0, 0$
P	$0, 0$	$1, 2$

$\diamondsuit$

Esempio 7.3 (Puro coordinamento)

I/II	T	P
T	$1,1$	$0,0$
P	$0,0$	$1,1$

$\diamondsuit$

- Nell'Esempio 7.2 (e soprattutto nel 7.3) una telefonata, un accordo al 50 per cento o una strategia correlata possono risolvere facilmente il problema
- Nell'Esempio 7.1 la possibilità di comunicare renderebbe probabile un accordo per la strategia NC, ma al momento della decisione sia I che II risceglierebbero C, poichè $-1 > -2$

Classificazione di Harsanyi (1966):

Giochi non cooperativi	Non sono possibili accordi vincolanti tra i giocatori
Giochi cooperativi	Sono possibili accordi vincolanti tra i giocatori

- Attualmente si preferisce assumere, più restrittivamente, che in un gioco non cooperativo i giocatori non possano nemmeno comunicare in quanto ciò potrebbe alterare il risultato
- I giochi cooperativi sono divisi in due sottoclassi: giochi a utilità non trasferibile (NTU) o senza pagamenti laterali, e giochi a utilità trasferibile (TU) o a pagamenti laterali, che costituiscono un caso particolare dei giochi NTU

7.3 Rappresentazione di un gioco

- forma estesa - von Neumann (1928) e Kuhn (1953)
- forma strategica - Shubik (1982); forma normale - von Neumann e Morgenstern (1944)
- forma caratteristica - von Neumann e Morgenstern (1944); per i giochi cooperativi

Definizione 7.1

- *Si chiama funzione dei pagamenti (payoff) una funzione f che assegna ad ogni giocatore la sua vincita per ogni possibile terminazione del gioco*
- *Si chiama strategia del giocatore i una funzione σ_i che assegna al giocatore i una mossa per ogni possibile situazione del gioco*

La strategia è un "piano di azione" che individua in ogni situazione del gioco una "azione" tra le tante possibili

7.4 Forma estesa

Descrizione puntuale del gioco, delle mosse e delle relative probabilità, della situazione dopo ogni mossa, delle strategie, degli insiemi di informazione (insiemi di nodi che globalmente rappresentano la situazione di un giocatore), ecc.; risulta molto ricca ma poco maneggevole

Si utilizza una rappresentazione ad albero in cui:

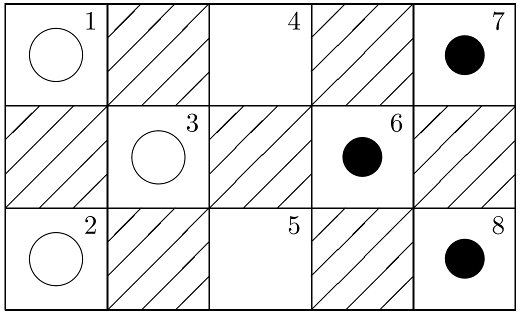
nodi	possibili situazioni del gioco
archi uscenti da un nodo	possibili mosse del giocatore chiamato a muovere
nodi terminali	valori delle vincite (payoff) di ciascun giocatore

Esempio 7.4 (Dama semplificata)

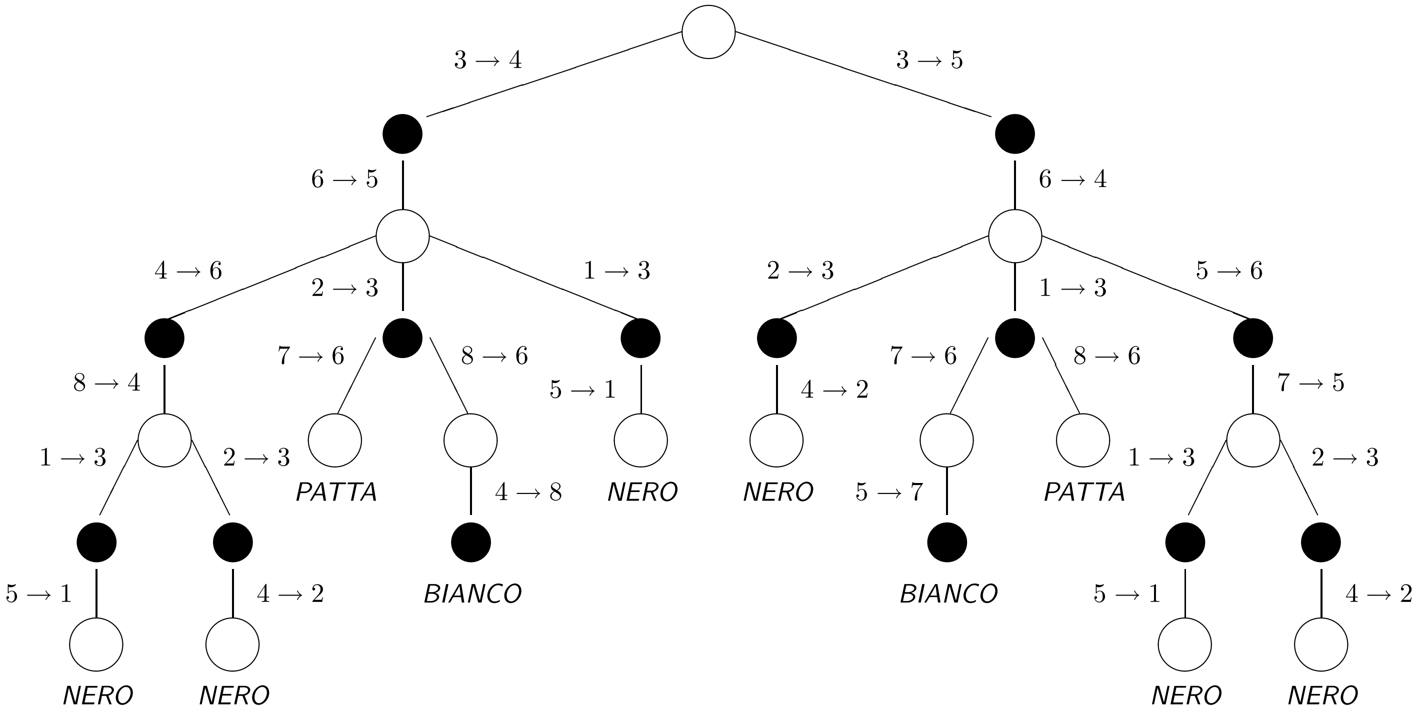
E' obbligatorio "mangiare"

Vince chi riesce a portare per primo una sua pedina sull'ultima colonna

Parità se un giocatore non può muovere



Rappresentazione ad albero della forma estesa:



◊

7.5 Forma strategica

$2n$-upla $(\sum_1, \sum_2, \ldots, \sum_n, f_1, f_2, \ldots, f_n)$ dove:

$\sum_1, \sum_2, \ldots, \sum_n$ insiemi non vuoti delle possibili strategie di ogni giocatore

$f_1, f_2, \ldots, f_n$ funzioni reali $f_i : \prod_{k=1,\ldots,n} \Sigma_k \to \mathbb{R} \qquad i = 1, .., n$

- Tutti i giocatori scelgono contemporaneamente la loro strategia e la f_i dice quale è il guadagno del giocatore i determinato dalle scelte fatte
- E' possibile passare dalla forma estesa a quella strategica (il passaggio inverso è più complesso)
- Gli elementi della forma strategica possono essere riassunti in una tabella come negli esempi precedenti
- Se il gioco è a due giocatori si parla di *gioco a matrice doppia* o *bimatrice*

7.6 Forma caratteristica

Può essere usata solo per i giochi cooperativi

Definizione 7.2

- *Detto N l'insieme dei giocatori, ogni sottoinsieme S di N è detto coalizione. Se $S = N$ si ha la grande coalizione*
- *Si dice funzione caratteristica di un gioco ad n giocatori una funzione indicata con v (se il gioco è senza pagamenti laterali si usa V ed è più complessa) per cui si ha:*

$$v : \wp(N) \to \mathbb{R} \text{ con } v(\emptyset) = 0$$

- *Se per ogni coppia di coalizioni disgiunte S e T si ha $v(S \cup T) = v(S) + v(T)$ la funzione v è detta additiva; se si ha $v(S \cup T) \geq v(S) + v(T)$ la funzione v è detta superadditiva; se si ha $v(S \cup T) \leq v(s) + v(T)$ la funzione v è detta subadditiva*
- v assegna ad S la massima vincita possibile indipendentemente dal comportamento degli altri giocatori
- La funzione caratteristica e il gioco possono essere identificati

Un gioco descritto tramite la funzione caratteristica è detto in *forma caratteristica* o *coalizionale*
Se la funzione caratteristica è additiva o superadditiva o subadditiva anche il gioco è detto *additivo* o *superadditivo* o *subadditivo*
Se per ogni coalizione S si ha $v(S) + v(N \setminus S) = v(N)$ il gioco è detto *a somma costante*

Esempio 7.5 (Maggioranza semplice)
Tre giocatori vogliono conseguire un risultato; se almeno due di essi si uniscono raggiungono il loro obiettivo. Questa situazione può essere rappresentata dal seguente gioco:

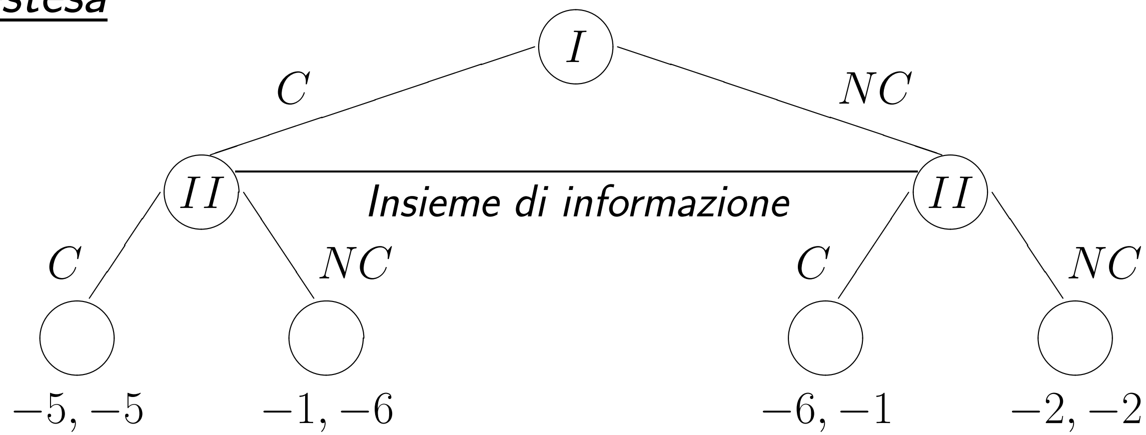
$N = \{1, 2, 3\}$

$v(\emptyset) = v(1) = v(2) = v(3) = 0; v(1, 2) = v(1, 3) = v(2, 3) = v(1, 2, 3) = 1$ $\diamondsuit$

La descrizione del gioco è molto "povera", in quanto non permette di definire la vincita di ogni singolo giocatore della coalizione, ma solo la vincita complessiva

Esempio 7.6 (Rappresentazioni del dilemma del prigioniero)

Forma estesa



Forma strategica

$$\Sigma_I = \{C, NC\}; \Sigma_{II} = \{C, NC\}$$
$$f_I(C, C) = -5; f_I(C, NC) = -1; f_I(NC, C) = -6; f_I(NC, NC) = -2$$
$$f_{II}(C, C) = -5; f_{II}(C, NC) = -6; f_{II}(NC, C) = -1; f_{II}(NC, NC) = -2$$

Forma caratteristica

$$N = \{I, II\}$$
$$v(\emptyset) = 0; v(I) = v(II) = -5; v(I, II) = -4$$

$\diamondsuit$

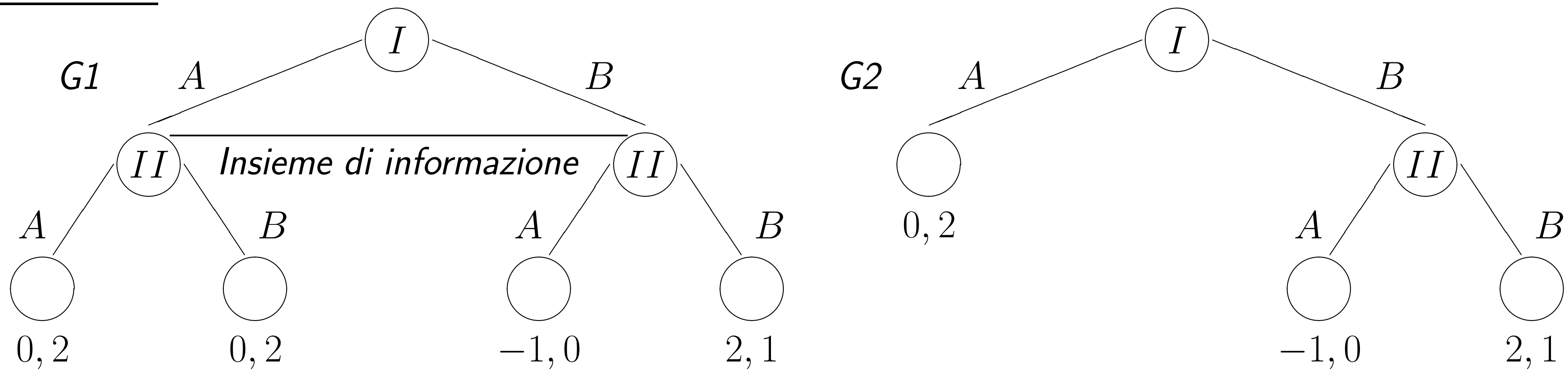
La forma estesa contiene più informazione sul gioco rispetto alla forma strategica, che risulta comunque sufficiente a rappresentare un gioco

Esempio 7.7 (Rappresentazioni in forma estesa e in forma strategica)

G1 I *e* II *scelgono contemporaneamente tra* A *e* B*; se giocano* (A, A) *oppure* (A, B) *i payoff sono* $(0, 2)$*, se giocano* (B, A) *i payoff sono* $(-1, 0)$*, se giocano* (B, B) *i payoff sono* $(2, 1)$

G2 I *e* II *scelgono successivamente tra* A *e* B*; se* I *gioca* A *il gioco termina con payoff* $(0, 2)$*, se gioca* B *il turno passa a* II*; se* II *gioca* A *il gioco termina con payoff* $(-1, 0)$*, se gioca* B *il gioco termina con payoff* $(2, 1)$

Forma estesa



Forma strategica

G1 - G2

I/II	A	B
A	$0, 2$	$0, 2$
B	$-1, 0$	$2, 1$

La forma strategica è unica, ma è sufficiente a descrivere i giochi ◊

7.7 Teoria dell'utilità

I concetti di *preferenza* e di *utilità di von Neumann-Morgenstern* permettono di assegnare e interpretare i valori numerici utilizzati nelle rappresentazioni dei giochi

I giocatori cercano di massimizzare la loro utilità, ma è necessario prendere in considerazione valori differenti: economico, sentimentale, sociale, ecc.
Se un giocatore deve decidere se donare una somma di denaro senza ricevere nulla in cambio, considerando solo i valori monetari la decisione sarebbe sempre non donare

Definizione 7.3

- *Dati due eventi A e B si dice che A è preferibile a B per un giocatore se egli cerca di conseguire A invece di B e si indica con $A \succ B$*
- *Dati due eventi A e B si dice che A è indifferente a B per un giocatore se nessuno è preferibile all'altro e si indica con $A \equiv B$*

Assiomi

A1 *Dati due eventi* A *e* B *allora* $A \succ B$ *oppure* $B \succ A$ *oppure* $A \equiv B$

A2 $A \equiv A$

A3 $A \equiv B \Rightarrow B \equiv A$

A4 $A \equiv B, B \equiv C \Rightarrow A \equiv C$

A5 $A \succ B, B \succ C \Rightarrow A \succ C$

A6 $A \succ B, B \equiv C \Rightarrow A \succ C$

A7 $A \equiv B, B \succ C \Rightarrow A \succ C$

- **A1**: *completezza delle preferenze* o *legge di tricotomia*; **A2**, **A3**, **A4**: *riflessività dell'indifferenza*, *simmetria dell'indifferenza*, *transitività dell'indifferenza* $\Rightarrow$ l'indifferenza è una *relazione di equivalenza*; **A5**: *transitività della preferenza*; **A6**, **A7**: *transitività della preferenza sull'indifferenza* (cfr. Owen, 1994)
- La relazione di preferenza è solo qualitativa
- Nessun bene soddisfa l'ipotesi di linearità, tranne al più in brevi intervalli

Gli eventi possono essere certi oppure incerti secondo una probabilità nota

Definizione 7.4 *Dati due eventi* A *e* B *si chiama lotteria l'evento* $rA+(1-r)B, 0 \leq r \leq 1$, *in cui* A *si verifica con probabilità* r *e l'evento* B *con probabilità* $1-r$

- La lotteria non è una combinazione lineare di eventi, ma permette di valutare l'evento "esce A o esce B"

Proprietà

P1 $A \equiv C \Rightarrow \{rA+(1-r)B\} \equiv \{rC+(1-r)B\} \qquad 0 \leq r \leq 1,$ per ogni B
P2 $A \succ C \Rightarrow \{rA+(1-r)B\} \succ \{rC+(1-r)B\} \qquad 0 < r \leq 1,$ per ogni B
P3 $A \succ C \succ B \Rightarrow \exists!\ r,\ 0 < r < 1\ t.c.\ \{rA+(1-r)B\} \equiv C$

- Se un decisore soddisfa gli assiomi **A1** - **A7** e le proprietà **P1** - **P3** viene considerato "razionale".

Esempio 7.8 (Preferenze) *Siano date le lotterie:*

$$\begin{aligned}
E_1 &= \{0, 100 \quad con \quad \mathbb{P}(0) = 1/2, \mathbb{P}(100) = 1/2\} \\
E_2 &= \{40, 60 \quad con \quad \mathbb{P}(40) = 3/4, \mathbb{P}(60) = 1/4\} \\
E_3 &= \{0, 100, 40, 60 \quad con \quad \mathbb{P}(0) = 1/4, \mathbb{P}(100) = 1/4, \mathbb{P}(40) = 3/8, \mathbb{P}(60) = 1/8\}
\end{aligned}$$

Il guadagno atteso, $E_1 = 50$*,* $E_2 = 45$ *e* $E_3 = 47.5$*, non impone una preferenza tra i tre eventi, ma le uniche relazioni da soddisfare sono:*

$$\begin{aligned}
& E_1 \equiv E_2 \Rightarrow E_1 \equiv E_3, E_2 \equiv E_3 \\
& oppure \\
& E_1 \succ E_2 \Rightarrow E_1 \succ E_3, E_3 \succ E_2 \\
& oppure \\
& E_2 \succ E_1 \Rightarrow E_2 \succ E_3, E_3 \succ E_1
\end{aligned}$$

$\diamondsuit$

Dato un insieme di eventi E, una relazione di preferenza su E può essere rappresentata con una funzione di utilità $u : E \rightarrow \mathbb{R}$ tale che per ogni $E_1, E_2 \in E$ si ha:

$$E_1 \succ E_2 \Leftrightarrow u(E_1) > u(E_2)$$
$$u(rE_1 + (1-r)E_2) = ru(E_1) + (1-r)u(E_2)$$

- La funzione di utilità permette di quantificare le preferenze
- L'utilità di von Neumann-Morgenstern impone la linearità sulle lotterie

La funzione u è unica a meno di trasformazioni affini, cioè u è una funzione di utilità se e solo se lo è anche:

$$\hat{u} = \alpha u + \beta \quad \text{con} \quad \alpha > 0$$

Esempio 7.9 (Funzioni di utilità)

I/II	C	NC
C	$-5, -5$	$-1, -6$
NC	$-6, -1$	$-2, -2$

I/II	C	NC
C	$1, 1$	$5, 0$
NC	$0, 5$	$4, 4$

I/II	C	NC
C	$-4, 0$	$0, -10$
NC	$-5, 40$	$-1, 30$

Le tre matrici sono legate dalle relazioni affini:

$$u'_I = u_I + 6 \qquad u''_I = u_I + 1$$
$$u'_{II} = u_{II} + 6 \qquad u''_{II} = 10u_{II} + 50$$

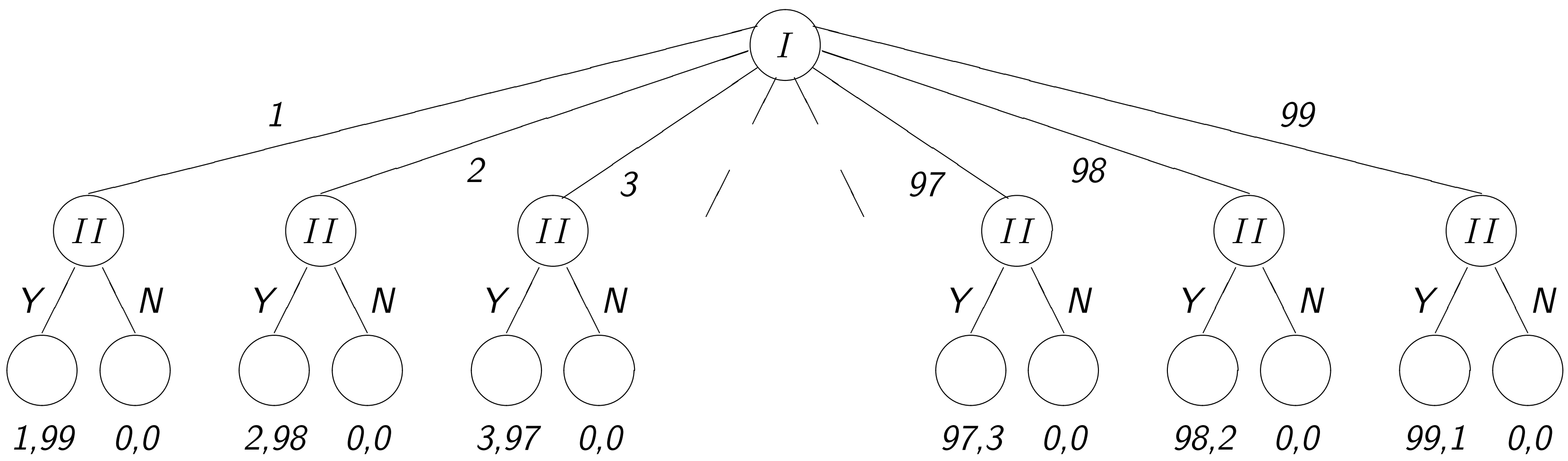
$\diamondsuit$

Esempio 7.10 (Ultimatum game)

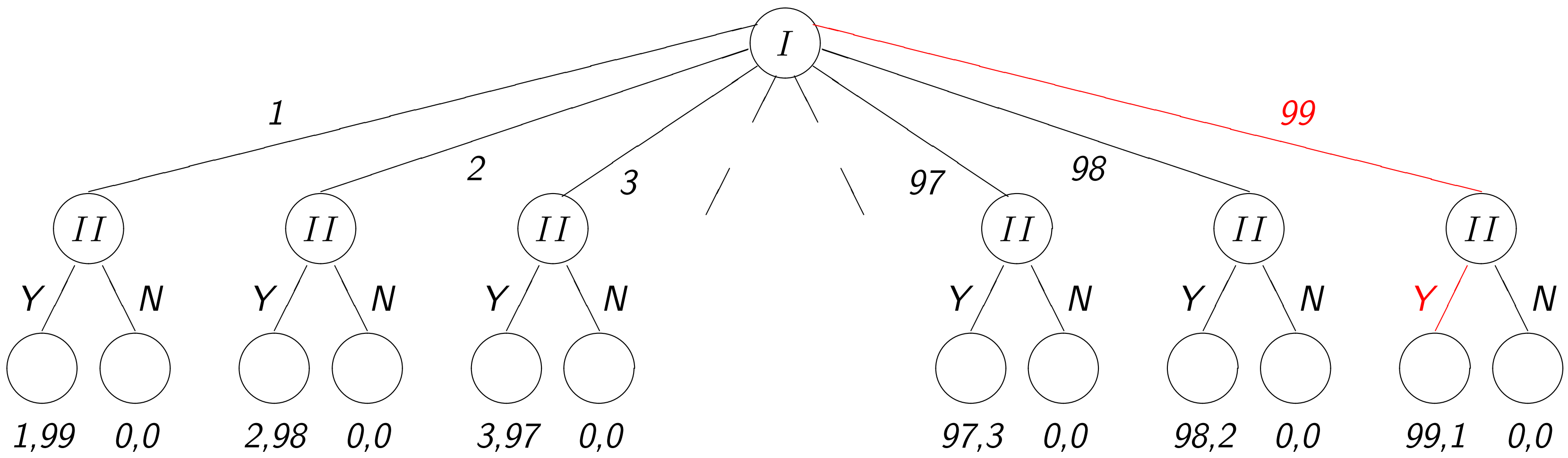
Due persone devono dividersi la cifra di 100 euro con le seguenti regole:

- I *propone una divisione (numeri interi, lasciando almeno 1 euro a ciascuno)*
- *se* II *accetta la divisione proposta, la divisione ha luogo*
- *se* II *non accetta la divisione proposta, non si assegna alcuna cifra*
- *entrambe le persone non sanno e non sapranno mai chi è l'altra*

Quale cifra conviene proporre a I*?*



La scelta ottimale del secondo giocatore è accettare sempre
In conseguenza la scelta ottimale del primo giocatore è proporre il massimo



Nelle sperimentazioni, questa soluzione non si realizza quasi mai, poichè l'utilità reale dei giocatori tiene conto di altri fattori ♢

7.8 Game Form

Permette di evidenziare esplicitamente la differenza tra le *regole del gioco* e le *preferenze* dei giocatori

Esempio 7.11 (Decisore) *Si deve decidere se uscire con o senza l'ombrello in una giornata con tempo variabile*

strategia / stato del mondo	*piove*	*non piove*
prendo l'ombrello	*non mi bagno*	*non mi bagno (ho l'ombrello)*
non prendo l'ombrello	*mi bagno*	*non mi bagno (non ho l'ombrello)*

Un decisore prudente prende sempre l'ombrello
In generale è necessario tenere conto delle preferenze del decisore ed eventualmente di ulteriori informazioni (distanza da percorrere, caratteristiche dell'ombrello, etc.) $\diamondsuit$

Sia dato un gioco a n giocatori in forma strategica $(\Sigma_1, \Sigma_2, \ldots, \Sigma_n, f_1, f_2, \ldots, f_n)$
La game form è:

$$(\Sigma_1, \Sigma_2, \ldots, \Sigma_n, E, h)$$

dove E: insieme degli eventi finali (o esiti)
$h : \prod_{i=1,\ldots,n} \Sigma_i \longrightarrow E$ individua a quale esito si perviene per ogni profilo di strategie

Per studiare il comportamento dei giocatori e quindi risolvere il gioco è necessario conoscere le preferenze dei giocatori sui possibili esiti, eventualmente rappresentate da una funzione di utilità

$$u_i : E \longrightarrow \mathbb{R} \qquad i = 1, \ldots, n$$

da cui si ottengono le funzioni di utilità indotte:

$$f_i = u_i \circ h, \;\; f_i : \prod_{k=1,\ldots,n} \Sigma_k \longrightarrow \mathbb{R} \qquad i = 1, \ldots, n$$

7.9 Soluzione di un gioco (Solution concept)

Risolvere un gioco consiste nel fornire delle indicazioni ad uno o più giocatori, eventualmente tutti, su:

- strategie da adottare se il gioco è non cooperativo o cooperativo ad utilità non trasferibile
- suddivisione della vincita se il gioco è cooperativo ad utilità trasferibile

Le indicazioni non possono essere assolute in quanto bisogna tenere conto di fattori aleatori, o legati a preferenze e sensazioni del singolo giocatore. Un "concetto di soluzione" indica quella che secondo alcuni criteri assoluti è una scelta che può risultare accettabile a tutti i giocatori secondo i loro criteri soggettivi

Nell'esempio della battaglia dei sessi contano "egoismo", "altruismo" e situazioni precedenti

Esempio 7.12 (Divisione di una torta tra due giocatori)
È uno dei problemi più significativi: molto semplice, molto comune e molto complesso
La soluzione più usuale, uno taglia e l'altro sceglie, non è equa in quanto può favorire chi sceglie se chi taglia non è preciso, o chi taglia se è a conoscenza di qualche preferenza o "punto debole" di chi sceglie ♢

8 Giochi cooperativi

8.1 Introduzione

I giocatori possono associarsi per migliorare il proprio risultato
Per realizzare la cooperazione:

- deve essere possibile stipulare accordi (ad esempio non devono esserci regole antitrust o difficoltà di comunicazione)
- deve esserci la possibilità di far rispettare tali accordi, nel senso che deve esistere una autorità sufficientemente forte e accettata da tutti i componenti

Si distinguono due sottoclassi:

- Giochi cooperativi senza pagamenti laterali (NTU-Games)
  i giocatori ricevono un payoff assegnato
- Giochi cooperativi a pagamenti laterali (TU-Games)
  i giocatori di una coalizione possono ripartirsi in qualsiasi modo la vincita

I secondi costituiscono un caso particolare dei primi

In particolare per avere un gioco TU devono essere soddisfatte tre ipotesi:

- deve essere possibile trasferire l'utilità (da un punto di vista normativo)
- deve esistere un mezzo comune di scambio, ad esempio il denaro, con cui trasferire l'utilità (da un punto di vista materiale)
- le funzioni di utilità dei giocatori devono essere equivalenti

Esempio 8.1 (Coalizione semplice) *Sono dati tre giocatori I, II, III; se due di loro si accordano, formando una coalizione, il terzo giocatore da ad ognuno di essi una moneta, altrimenti nessuno riceve nulla. I payoff sono:*

$(1, 1, -2)$	*se I e II si coalizzano*
$(1, -2, 1)$	*se I e III si coalizzano*
$(-2, 1, 1)$	*se II e III si coalizzano*
$(0, 0, 0)$	*altrimenti*

Se i payoff relativi alla coalizione $\{II, III\}$ fossero $(-2.0, 1.1, 0.9)$ la posizione del giocatore II non si rafforza in quanto il giocatore III ha più interesse a coalizzarsi con I che con II; questa situazione non sussiste nel caso in cui sia possibile per II "trasferire" parte della propria vincita al giocatore III, ritornando alla situazione precedente $\diamondsuit$

8.1.1 Funzione caratteristica per un gioco TU

Può essere costruita a partire dal gioco a due persone tra S ed $N \setminus S$:

$$v'(S) = \max_{\sigma_S \in \Sigma_S} \quad \min_{\sigma_{N\setminus S} \in \Sigma_{N\setminus S}} \left\{ \sum_{i \in S} u_i(\sigma_S, \sigma_{N\setminus S}) \right\} \quad \text{(von Neumann-Morgenstern)}$$

$$v''(S) = \min_{\sigma_{N\setminus S} \in \Sigma_{N\setminus S}} \quad \max_{\sigma_S \in \Sigma_S} \left\{ \sum_{i \in S} u_i(\sigma_S, \sigma_{N\setminus S}) \right\}$$

I due risultati possono non coincidere ($v'' \geq v'$)

Il problema è assegnare correttamente il valore di $v(S)$

8.2 Giochi cooperativi a pagamenti laterali

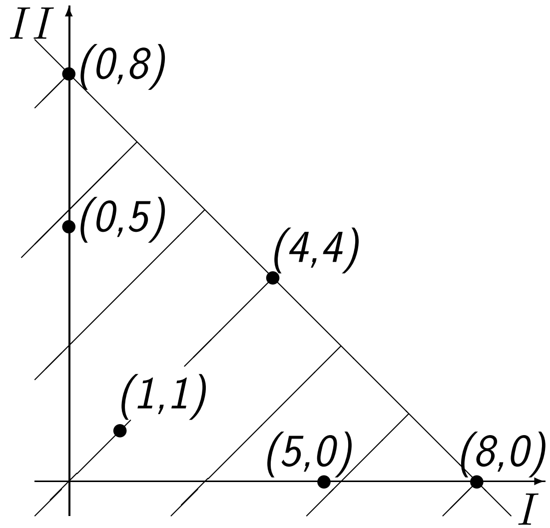
In questi giochi introdotti da Von Neumann e Morgenstern (1944) i giocatori possono stipulare accordi vincolanti e possono ripartirsi la vincita con un accordo al di fuori delle regole del gioco, la cui validità può estendersi anche oltre la fine del gioco

Come trasferire la vincita se i giocatori hanno differenti funzioni di utilità?

Definizione 8.1 *Un gioco TU è una coppia* $G = (N, v)$ *dove* N *è l'insieme dei giocatori e* v *è la funzione caratteristica, con* $v(\emptyset) = 0$

Se $v(S) \leq 0, S \subseteq N$ si ha un *gioco di costi* o cost game (N, c) in cui si pone $c = -v$

Esempio 8.2 (Dilemma del prigioniero) *In questo caso sono possibili tutte le vincite aventi somma non superiore a 8. Ad esempio è possibile la vincita (8, 0), nel caso in cui il giocatore* I *prende la sua vincita (4) e la vincita del giocatore* II *(4)*



$$v(\emptyset) = 0$$
$$v(I) = 1$$
$$v(II) = 1$$
$$v(N) = 8 = max\{f_I + f_{II}\}$$

◊

Esempio 8.3 (Gioco dei guanti) *Due insiemi di giocatori, L ed R, possiedono dei guanti; i giocatori di L possiedono solo guanti sinistri mentre i giocatori di R possiedono solo guanti destri. Il valore di una coalizione è dato dal numero di paia di guanti che riescono a formare. In generale ogni giocatore possiede un solo guanto. Se i giocatori di L sono 1 e 2 e i giocatori di R sono 3 e 4 si ha:*

$$
\begin{array}{ll}
N = \{1, 2, 3, 4\} & \\
v(S) = 0 & \textit{se } |S| \leq 1 \textit{ oppure se } S = \{12\}, S = \{34\} \\
v(S) = 1 & \textit{se } |S| = 2 \textit{ e } S \neq \{12\}, S \neq \{34\} \textit{ oppure se } |S| = 3 \\
v(N) = 2 &
\end{array}
$$

$\diamondsuit$

Scrivere la funzione caratteristica nel caso generale

Definizione 8.2 *Un gioco $G = (N, v)$ si dice monotono se $v(S) \leq v(T), S \subseteq T$. Anche per un cost game si ha $c(S) \leq c(T), S \subseteq T$*

Definizione 8.3 *Un gioco $G = (N, v)$ si dice convesso se vale una delle seguenti condizioni equivalenti:*

- $v(S) + v(T) \leq v(S \cup T) + v(S \cap T), S, T \subseteq N$
- $v(S \cup \{i\}) - v(S) \leq v(T \cup \{i\}) - v(T), S \subset T \subseteq N \setminus \{i\}, i \in N$

Definizione 8.4 *Un gioco $G = (N, v)$ si dice semplice 0-1 o semplice se le coalizioni possono assumere solo i valori 0 e 1*
Se una coalizione ha valore 1 è detta vincente, se ha valore 0 è detta perdente
Solitamente la grande coalizione è vincente

Definizione 8.5 *Un gioco $G = (N, v)$ si dice coesivo se per ogni partizione di N $\{S_1, S_2, \ldots, S_k\}$ si ha:*

$$\sum_{i=1,\ldots,k} v(S_i) \leq v(N)$$

Per un cost game deve valere $\sum_{i=1,\ldots,k} c(S_i) \geq c(N)$

- Nella definizione di monotonia non si tiene conto della cardinalità delle coalizioni
- L'equivalenza delle definizioni di convessità è oggetto di un teorema.
- I giochi semplici trovano applicazione nelle situazioni in cui una coalizione è caratterizzata dal riuscire a conseguire o meno un determinato risultato, come nei giochi di maggioranza, utilizzati in politica
- La coesività è più debole della superadditività ed esprime la "convenienza" dei giocatori a formare la grande coalizione, piuttosto che riunirsi in sottocoalizioni. L'importanza deriva dal fatto che in generale i concetti di soluzione più comuni costituiscono una ripartizione del valore della grande coalizione

Dimostrare l'equivalenza delle definizioni di convessità e che la coesività è più debole della superadditività

Le soluzioni di un gioco TU possono essere raggruppate in due famiglie:

- *soluzioni insiemistiche* che individuano un insieme di vettori payoff che ripartiscono il valore del gioco tra tutti i giocatori
- *soluzioni puntuali* che individuano una sola ripartizione e che costituiscono l'attuale tendenza in quanto più simili all'idea classica di soluzione di un problema

9 Soluzioni insiemistiche di un gioco TU

9.1 Imputazioni

Per determinare le singole vincite si potrebbe risolvere un sottogioco ristretto ai giocatori di ciascuna coalizione, oppure suddividere in parti uguali la vincita, trascurando il contributo dei singoli giocatori
Altri metodi più complessi tengono conto del ruolo svolto da ciascun giocatore

Definizione 9.1 *Dato un gioco $G = (N, v)$ si dice imputazione o ripartizione del valore del gioco o soluzione del gioco un vettore $x = (x_1, x_2, \ldots, x_n)$ tale che:*

$$\sum_{i \in N} x_i = v(N) \qquad \textit{efficienza}$$
$$x_i \geq v(i), i = 1, \ldots, n \qquad \textit{razionalità individuale}$$

Nel caso di un cost game la razionalità individuale richiede $x_i \leq c(i)$

L'insieme di tutte le imputazioni si indica con $E(v)$

Definizione 9.2 *Se per un gioco $G = (N, v)$ si ha:*

$$\sum_{i \in N} v(i) = v(N)$$

allora $E(v)$ ha come unico elemento $x = (v(1), v(2), \ldots, v(n))$; in questo caso il gioco è detto inessenziale; se $\sum_{i \in N} v(i) < v(N)$ il gioco è detto essenziale

La razionalità individuale costituisce una condizione per ogni concetto di soluzione
Se il gioco è essenziale esistono più imputazioni possibili e si ripropone il problema di scegliere la “soluzione”: se due imputazioni x e y sono distinte esiste almeno un giocatore $k \in N$ per cui $x_k > y_k$ e almeno un giocatore $h \in N$ per cui $x_h < y_h$

Definizione 9.3

- *Date* $x, y \in E(v)$ *e una coalizione* S *si dice che* x *domina* y *mediante* $S, x \succ_S y$*, se:*

  1. $x_i > y_i \qquad i \in S$
  2. $x(S) \leq v(S)$

  dove $x(S) = \sum_{i \in S} x_i$.

- *Date* $x, y \in E(v)$ *si dice che* x *domina* y*,* $x \succ y$*, se esiste* S *tale che* $x \succ_S y$

La dominanza non è riflessiva, nè antisimmetrica, nè transitiva

Esempio 9.1 (Non antisimmetria) *Sia dato il seguente gioco:*

$$\begin{array}{ll} N = \{1, 2, 3, 4\} & \\ v(S) = 0 & |S| \leq 2 \\ v(S) = 1 & |S| \geq 3 \end{array}$$

Date le seguenti imputazioni $x = \left(\frac{1}{2}, \frac{1}{2}, 0, 0\right)$ *e* $y = \left(0, 0, \frac{1}{2}, \frac{1}{2}\right)$ *si ha:*

$$x \succ_{\{1,2\}} y \ e \ y \succ_{\{3,4\}} x$$

$\diamondsuit$

9.2 Insiemi stabili

E' stato proposto da Von Neumann - Morgenstern (1944) come la "soluzione" dei giochi TU

Definizione 9.4 *Un insieme $V \subset E(v)$ si dice stabile se:*

1. *dati $x, y \in V$ si ha $x \not\succ y$ e viceversa* *stabilità interna*
2. *dato $x \notin V$, $\exists\ y \in V$ per il quale si ha $y \succ x$* *stabilità esterna*

Un insieme stabile contiene la soluzione ma la decisione dipende da altre informazioni non espresse dalla forma caratteristica

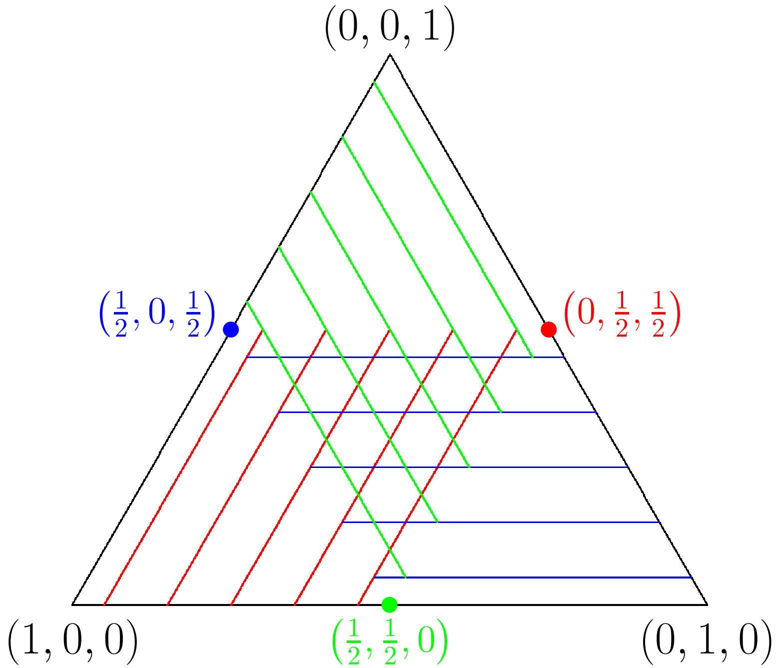
Un gioco può avere più insiemi stabili

Esempio 9.2 (Maggioranza semplice)

L'insieme stabile più comune è:

$$\left\{\left(\frac{1}{2},\frac{1}{2},0\right),\left(\frac{1}{2},0,\frac{1}{2}\right),\left(0,\frac{1}{2},\frac{1}{2}\right)\right\}$$

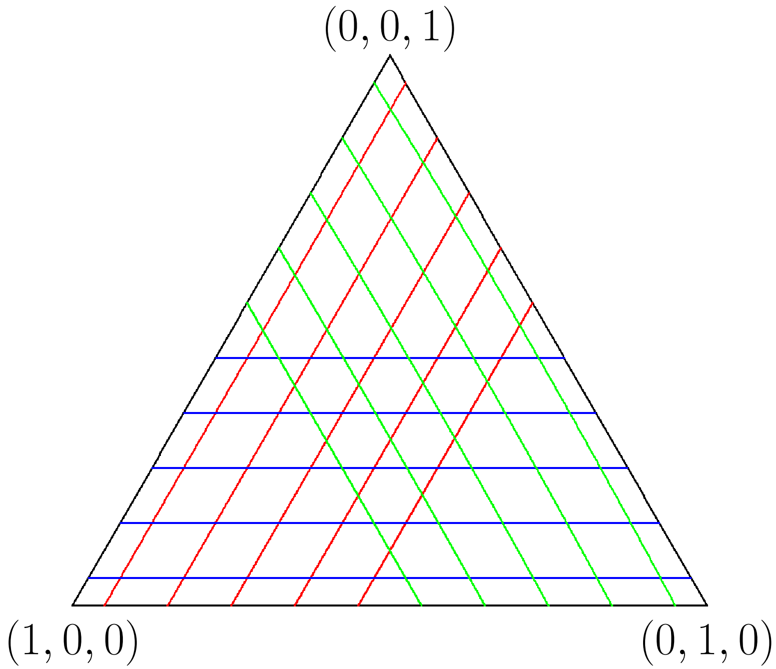
$(\frac{1}{2},\frac{1}{2},0)$ *domina mediante* $\{1,2\}$
$(\frac{1}{2},0,\frac{1}{2})$ *domina mediante* $\{1,3\}$
$(0,\frac{1}{2},\frac{1}{2})$ *domina mediante* $\{2,3\}$



Altri insiemi stabili sono:

- $V_{1,c} = \{c, t, 1 - c - t\}$
- $V_{2,c} = \{1 - c - t, c, t\}$
- $V_{3,c} = \{t, 1 - c - t, c\}$

$$\begin{array}{lll} \textit{con} \;\; c \in [0, \frac{1}{2}[& \textit{funzione delle tradizioni} \\ \phantom{\textit{con}} \;\; t \in [0, 1 - c] & \textit{funzione della forza} \end{array}$$



$\diamondsuit$

Nel 1968 Lucas ha dato un esempio di gioco senza insiemi stabili, indebolendo ulteriormente questo concetto di soluzione

9.3 Nucleo

E' probabilmente il concetto di soluzione insiemistico più interessante per numerose classi di giochi; è stato introdotto da Gillies (1953 e 1959)

$$x(S) \geq v(S), S \subset N \qquad \textit{razionalità di coalizione}$$

Definizione 9.5 *Si dice nucleo di un gioco, o core, l'insieme:*

$$C(v) = \{x \in E(v) | x(S) \geq v(S), S \subset N\}$$

Nel caso di un cost game c *la razionalità di coalizione richiede* $x(S) \leq c(S), S \subset N$

- Le imputazioni non dominate costituiscono il nucleo del gioco
- Il nucleo può essere vuoto come nel gioco di maggioranza semplice e in generale nei giochi essenziali a somma costante
- Il nucleo ha un aspetto normativo (quali soluzioni non bisogna scegliere). Se il nucleo è vuoto non si può concludere che la grande coalizione non si forma, ma solo che è instabile

Dimostrare che il nucleo è stabile

Esempio 9.3 (Nucleo del gioco dei guanti) *Riferendosi all'Esempio 8.3, il nucleo è:*

$$C(v) = \{(\alpha, \alpha, 1-\alpha, 1-\alpha) \; s.t. \; 0 \leq \alpha \leq 1\}$$

In generale se $L = \{1, ..., n_l\}$ *e* $R = \{1, ..., n_r\}$ *si ha:*
se $n_l = n_r$*:*

$$C(v) = \{(\alpha, ..., \alpha, 1-\alpha, ..., 1-\alpha) \; s.t. \; 0 \leq \alpha \leq 1\}$$

se $n_l < n_r$*:*

$$C(v) = \{(\underbrace{1, ..., 1}_{1,...,n_l}, \underbrace{0, ..., 0}_{1,...,n_r})\}$$

se $n_l > n_r$*:*

$$C(v) = \{(\underbrace{0, ..., 0}_{1,...,n_l}, \underbrace{1, ..., 1}_{1,...,n_r})\}$$

Il nucleo evidenzia il comportamento del mercato quando uno tra due beni complementari è carente $\diamondsuit$

9.3.1 Bilanciamento

Per stabilire se un gioco ha nucleo vuoto o meno, la coesività o la superadditività non danno informazioni precise; ad esempio il gioco di maggioranza semplice ha nucleo vuoto ma è superadditivo e quindi anche coesivo

Un gioco può non essere superadditivo, ma avere nucleo non vuoto

Esempio 9.4 (Gioco non superadditivo a nucleo non vuoto) *Si consideri il gioco TU:*

$$\begin{array}{ll} N = \{1, 2, 3\} & \\ v(S) = 1 & S \neq N \\ v(N) = 3 & \end{array}$$

Il gioco non è superadditivo poichè $v(1) + v(2) = 2$ *e* $v(12) = 1$ *ma ha nucleo non vuoto in quanto* $x = (1, 1, 1) \in C(v)$ $\diamondsuit$

Se un gioco non è coesivo ha nucleo vuoto, in quanto per ogni allocazione x esisterebbe una partizione $\{S_1, S_2, \ldots, S_k\}$ tale che:

$$x(N) = v(N) < \sum_{i=1,\ldots,k} v(S_i) \leq \sum_{i=1,\ldots,k} x(S_i) = x(N)$$

Le imputazioni del nucleo possono essere caratterizzate come le soluzioni del problema lineare:

$$\begin{aligned} min \;\; & z = \sum_{i \in N} x_i \\ s.t. \;\; & \sum_{i \in S} x_i \geq v(S) \qquad S \subseteq N \end{aligned}$$

per le quali $z^* = v(N)$

Il duale del problema è:

$$\begin{aligned} max \;\; & w = \sum_{S \subseteq N} y_s v(S) \\ s.t. \;\; & \sum_{S \ni i} y_S = 1 \qquad i \in N \\ & y_S \geq 0 \qquad S \subseteq N \end{aligned}$$

per le quali $w^* = v(N)$

Teorema 9.1 *Un gioco v ha nucleo non vuoto se e solo se esiste una soluzione del problema primale con $z^* = v(N)$ o equivalentemente (per il primo teorema della dualità) esiste una soluzione del problema duale con $w^* = v(N)$*

L'utilità di questo teorema è molto limitata in quanto la difficoltà di verificare una delle tre condizioni è equivalente

Definizione 9.6

- *Una collezione* $\mathcal{B} = \{S_1, S_2, \ldots, S_m\}$ *di sottoinsiemi di* N *è detta bilanciata se esistono* m *numeri non negativi* $y_1, y_2, \ldots, y_m$ *detti coefficienti di bilanciamento, tali che:*

$$\sum_{S_j \ni i} y_j = 1 \qquad i \in N$$

- *Una collezione bilanciata è detta minimale se nessuna sottocollezione è bilanciata*

- *Un gioco è detto bilanciato se per ogni collezione bilanciata minimale* $\mathcal{B} = \{S_1, S_2, \ldots, S_m\}$ *con coefficienti di bilanciamento* $y_1, y_2, \ldots, y_m$*, si ha:*

$$\sum_{j=1,\ldots,m} y_j v(S_j) \leq v(N)$$

Proprietà

- Ogni collezione bilanciata è unione di collezioni bilanciate minimali

- Una collezione bilanciata è minimale se e solo se i coefficienti di bilanciamento sono unici

- Le collezioni bilanciate non dipendono dalla funzione caratteristica, ma solo da N

Esempio 9.5 (Collezioni bilanciate I)

1. *Ogni partizione di N è una collezione bilanciata, con coefficienti unitari.*
2. *Sia $N = \{1,2,3\}$; $\mathcal{B} = \{\{1,2\},\{1,3\},\{2,3\}\}$ è una collezione bilanciata con coefficienti $\left(\frac{1}{2},\frac{1}{2},\frac{1}{2}\right)$. In generale per ogni N la collezione di $\binom{n}{s}$ sottoinsiemi distinti di s elementi è bilanciata con coefficienti $\binom{n-1}{s-1}^{-1}$.*

$\diamondsuit$

Teorema 9.2 (Bondareva, 1963 - Shapley, 1967) *Un gioco $G = (N,v)$ ha nucleo non vuoto se e solo se è bilanciato*

<u>Dimostrazione</u>

$$C(v) \neq \emptyset \iff v(N) = min \left\{\sum_{i=1,\dots,n} x_i |\; x(S) \geq v(S), S \subseteq N\right\} \iff$$

$$\iff v(N) = max \left\{\sum_{S\subseteq N} y_S v(S) | \sum_{S\ni i} y_S = 1, i \in N, y_S \geq 0, S \subseteq N\right\} \iff$$

$$\iff \sum_{S\subseteq N} y_S v(S) \leq v(N), \sum_{S\ni i} y_S = 1, i \in N, y_S \geq 0, S \subseteq N$$

$$\iff G \text{ è bilanciato (vertici della regione ammissibile)}$$

$\clubsuit$

- Il teorema di Bondareva-Shapley considera un sistema lineare generato da un sottoinsieme dei vincoli del problema duale associato al nucleo
- Per un gioco superadditivo il teorema di Bondareva-Shapley è vero per le partizioni di N, quindi è sufficiente verificarlo per le altre collezioni bilanciate minimali
- Il teorema è particolarmente utile per dimostrare che un gioco ha nucleo vuoto in quanto è sufficiente trovare una collezione bilanciata che non verifica la condizione
- Un gioco a nucleo non vuoto viene anche detto bilanciato

Esempio 9.6 (Collezioni bilanciate II)

1. *Un gioco a tre giocatori superadditivo è bilanciato se e solo se* $v(12) + v(13) + v(23) \leq 2\ v(123)$ *poichè* $\mathcal{B} = \{\{1,2\},\{1,3\},\{2,3\}\}$ *è l'unica collezione bilanciata minimale con coefficienti* $\left(\frac{1}{2}, \frac{1}{2}, \frac{1}{2}\right)$.
2. *Sia dato il gioco:*

$$\begin{array}{l} N = \{1,2,3,4\} \\ v(1) = v(2) = v(3) = v(4) = v(14) = v(24) = 0; v(23) = v(34) = 2 \\ v(12) = v(13) = v(123) = 3; v(124) = 4; v(134) = v(234) = 5; v(N) = 6 \end{array}$$

Il gioco non è bilanciato in quanto $\mathcal{B} = \{\{1,2\},\{1,3,4\},\{2,3,4\}\}$ *è una collezione bilanciata con coefficienti* $\left(\frac{1}{2}, \frac{1}{2}, \frac{1}{2}\right)$ *per la quale si ha:*

$$\frac{1}{2}\ v(12) + \frac{1}{2}\ v(134) + \frac{1}{2}\ v(234) = \frac{13}{2} > 6 = v(N)$$

$\diamondsuit$

9.4 Esempi di giochi e nucleo

9.4.1 Bankruptcy game

Allocazione di una risorsa insufficiente

$$\mathcal{B} = (N, c, E) = (E; c_1, ..., c_n)$$

dove	$N = \{1, ..., n\}$	insieme dei creditori
	$c = \{c_1, ..., c_n\}$	vettore delle richieste
	E	capitale, con $E < \sum_{i \in N} c_i = C$

Ogni ripartizione ammissibile ("razionale") del capitale, $x = \{x_1, ..., x_n\}$ deve soddisfare:

$$\sum_{i \in N} x_i = E$$
$$0 \leq x_i \leq c_i \qquad i \in N$$

Soluzioni

- $PROP$ - Le quote assegnate sono proporzionali alle richieste di ciascuno:

$$PROP_i = \frac{c_i}{C}E \qquad i \in N$$

- CEA - Le quote assegnate sono uguali per tutti, col vincolo di non superare le richieste di ciascuno:

$$CEA_i = \min(\alpha, c_i) \qquad i \in N$$

  dove α è l'unico valore reale positivo per cui $\sum_{i \in N} CEA_i = E$

- CEL - Le quote assegnate sono uguali alle richieste di ciascuno diminuite di una quantità uguale per tutti, col vincolo di non assegnare quote negative:

$$CEL_i = \max(c_i - \beta, 0) \qquad i \in N$$

  dove β è l'unico valore reale positivo per cui $\sum_{i \in N} CEL_i = E$

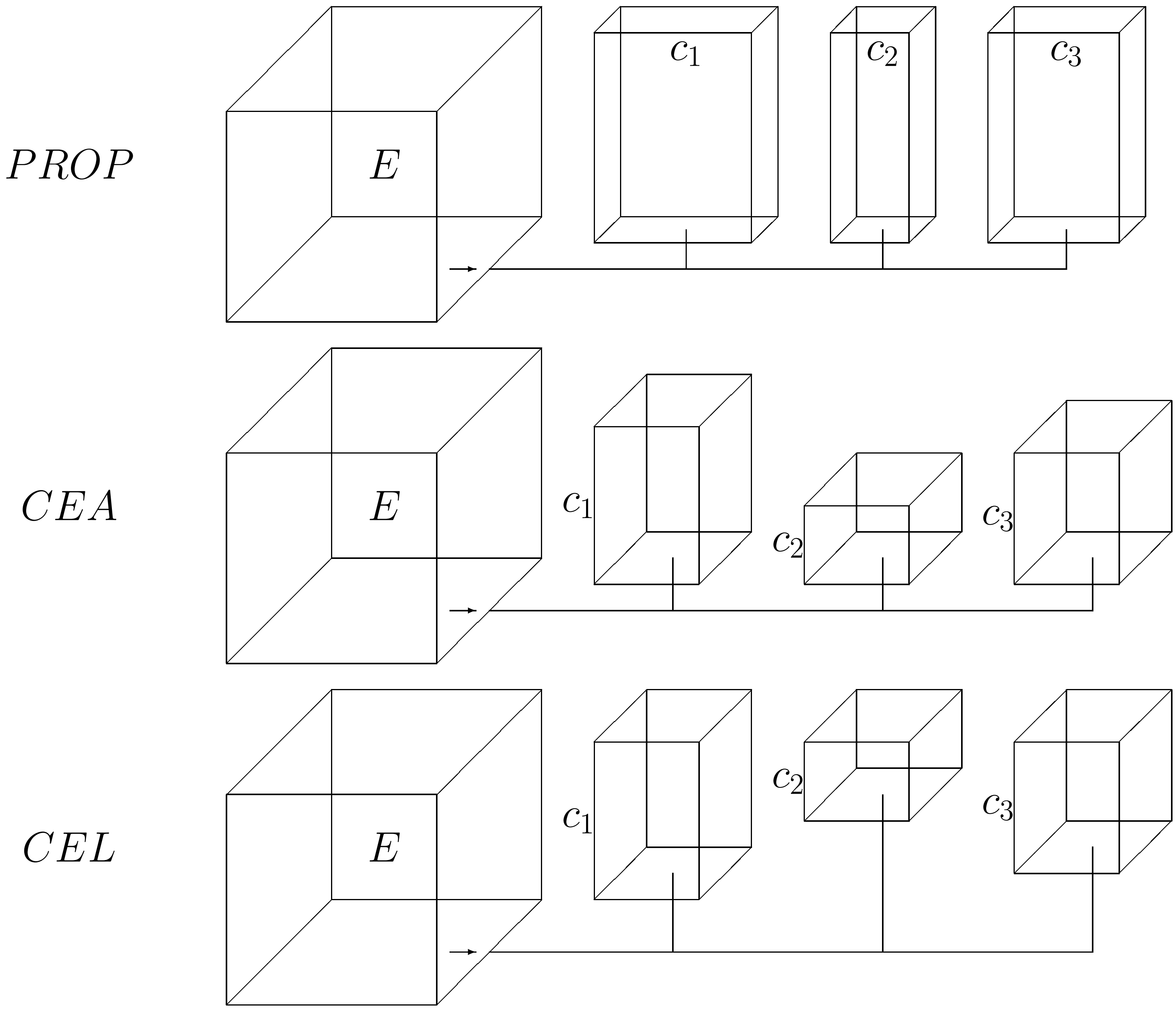
Esempio 9.7 (Soluzioni) *Si consideri il problema di bancarotta* $(15; 3, 6, 7, 14)$

$$PROP = (1.5, 3, 3.5, 7)$$
$$CEA = (3, 4, 4, 4)$$
$$CEL = (0, 2, 3, 10)$$

$\diamondsuit$

$PROP$ è la soluzione più intuitiva
CEA è quella che più protegge i piccoli creditori
CEL è quella più favorevole ai grossi creditori

Interpretazione dei vasi comunicanti



Si possono definire due giochi TU, uno pessimistico, (N, v_P), e uno ottimistico, (N, v_O), con:

$$v_P(S) = max\left(0, E - \sum_{i \in N \setminus S} c_i\right) \qquad S \subseteq N$$

$$v_O(S) = min\left(E, \sum_{i \in S} c_i\right) \qquad S \subseteq N$$

Esempio 9.8 (Inconsistenza del gioco ottimistico) *Si consideri il problema di bancarotta* $(5; 3, 4)$*. I due giochi sono definiti rispettivamente da:*

$$v_O(1) = 3; v_O(2) = 4; v_O(12) = 5$$
$$v_P(1) = 1; v_P(2) = 2; v_P(12) = 5$$

per cui il gioco ottimistico dice che i due giocatori separatamente possono ottenere rispettivamente 3 *e* 4*, mentre il capitale è solo* 5 $\diamondsuit$

Il nucleo del gioco pessimistico coincide con l'insieme delle soluzioni ammissibili del problema di bancarotta:

$$x \in core(v_P) \iff \begin{cases} \displaystyle\sum_{i \in N} x_i = E \\ 0 \leq x_i \leq c_i, \qquad i \in N \end{cases}$$

"$\Rightarrow$" La prima è la condizione di efficienza
Per la seconda condizione per ogni $i \in N$ si ha $x_i \geq v_P(i) \geq 0$ e $E - x_i = \sum_{j \in N \setminus \{i\}} x_j \geq v_P(N \setminus \{i\}) \geq E - c_i \Rightarrow x_i \leq c_i$

"$\Leftarrow$" La condizione di efficienza è ovviamente soddisfatta
Per ogni $S \subset N$ si hanno due casi:

1) se $v_P(S) = 0 \leq \sum_{i \in S} x_i$

2) se $v_P(S) = E - \sum_{i \in N \setminus S} c_i \leq E - \sum_{i \in N \setminus S} x_i = \sum_{i \in S} x_i$

9.4.2 Sequencing game

Problema di sequenziamento (ordinamento di operazioni)

$$\mathcal{S} = (N, \sigma_0, \alpha, s)$$

dove	$N = \{1, ..., n\}$	insieme degli agenti
	σ_0	ordine iniziale (permutazione)
	$\alpha = (\alpha_1, ..., \alpha_n)$	vettore dei costi per unità di tempo
	$s = (s_1, ..., s_n)$	vettore dei tempi di servizio

$$C_\sigma = \sum_{i \in N} \alpha_i \left(\sum_{j \in P(\sigma,i)} s_j + s_i \right)$$

dove $P(\sigma, i)$ è l'insieme degli agenti che precedono i nell'ordinamento σ
Smith (1956) ha dimostrato che l'ordinamento ottimale si può ottenere ordinando gli agenti secondo indici di urgenza $u_i = \dfrac{\alpha_i}{s_i}, i \in N$ debolmente decrescenti

Esempio 9.9 (Problema di sequenziamento) *Si consideri il problema di sequenziamento definito da* $N = \{1, 2, 3\}, \sigma_0 = (1, 2, 3), \alpha = (5, 9, 8), s = (5, 3, 4)$*; il costo iniziale è* $C_{\sigma_0} = 25 + 72 + 96 = 193$ *e gli indici di urgenza sono* $u = (1, 3, 2)$*, per cui* $\sigma^* = (2, 3, 1)$ *con costo* $C_{\sigma^*} = 27 + 56 + 60 = 143$ $\diamondsuit$

E' possibile associare al problema il gioco TU semplice (N, v) con v definita nel modo seguente:

- una coalizione $T \subseteq N$ è detta connessa secondo σ se per ogni $i, j \in T$ e $k \in N$ si ha $\sigma(i) < \sigma(k) < \sigma(j) \Rightarrow k \in T$
- scambiando due giocatori i, j la variazione di costo è $\alpha_j s_i - \alpha_i s_j$; la variazione è positiva se e solo se $u_i < u_j$; se la variazione è negativa non si ha lo scambio
- il guadagno di uno scambio è $g_{ij} = max\{0, \alpha_j s_i - \alpha_i s_j\}$, quindi il guadagno di una coalizione T connessa secondo σ è $v(T) = \sum_{j \in T} \sum_{i \in P(\sigma,j) \cap T} g_{ij}$
- data una coalizione $S \subseteq N$, l'ordine σ induce una partizione in componenti connesse, S/σ

$$v(S) = \sum_{T \in S/\sigma} v(T) \qquad S \subset N$$

Esempio 9.10 (Sequencing game) *Riferendosi all'Esempio 9.9 si ha:*

S	1	2	3	12	13	23	123
$v(S)$	0	0	0	30	0	0	50

$v(23) = 0$ *poichè lo scambio produrrebbe una perdita, in quanto* $u_2 > u_3$
$v(13) = 0$ *perchè la coalizione non è connessa e i giocatori* 1 *e* 3 *non possono scambiarsi anche se otterrebbero un guadagno di 20 e il giocatore* 2 *avrebbe un guadagno, poichè il tempo di servizio di* 3 *è 4 e quello di* 1 *è 5* $\diamondsuit$

$$EGS_i = \frac{1}{2} \sum_{k \in P(\sigma,i)} g_{ki} + \frac{1}{2} \sum_{j: i \in P(\sigma,j)} g_{ij} \qquad i \in N$$

Esempio 9.11 (EGS-Rule) *Riferendosi all'Esempio 9.9 i guadagni g_{ij} sono:*

ij	12	13	21	23	31	32
g_{ij}	30	20	0	0	0	12

e conseguentemente

$$EGS_1 = \frac{1}{2}(g_{12} + g_{13}) = 25$$
$$EGS_2 = \frac{1}{2}g_{12} + \frac{1}{2}g_{23} = 15$$
$$EGS_3 = \frac{1}{2}(g_{13} + g_{23}) = 10$$

$\diamondsuit$

- EGS non è simmetrica per i giocatori 1 e 2 che sono simmetrici per il gioco, cioè $v(S \cup \{1\}) = v(S \cup \{2\}), \forall\ S \subseteq N \setminus \{1,2\}$, ma non per il problema associato; infatti 1 può scambiarsi vantaggiosamente sia con 2 che con 3, mentre 2 può scambiarsi vantaggiosamente solo con 1
- g_{21}, g_{31}, g_{32} non vengono utilizzati perchè l'ordine iniziale dato non permette questi scambi
- Una variante è $\varepsilon - GS_i = \varepsilon \sum_{k \in P(\sigma,i)} g_{ki} + (1-\varepsilon) \sum_{j: i \in P(\sigma,j)} g_{ij}, i \in N, \varepsilon \in [0,1]$

9.4.3 Production game

Problema di produzione

$$\mathcal{P} = (N, A, (b^i)_{i \in N}, c)$$

dove	$N = \{1, ..., n\}$	insieme degli agenti
	A	matrice tecnologica del processo produttivo
	b^i	vettore delle risorse dell'agente i
	c	vettore dei prezzi dei beni prodotti

E' possibile associare al problema il gioco TU (N, v), dove:

$$v(S) = max \; \{c^T z | Az \leq b^S, z \geq 0\} \qquad S \subseteq N$$

con $b^S = \sum_{i \in S} b^i$ rappresenta le risorse possedute dalla coalizione S

Il nucleo di un gioco di produzione contiene le imputazioni x tali che $x_i = b^{iT} u^*$ dove u^* è una soluzione ottimale del duale del problema di produzione:

$$\begin{aligned} max \;\; & c^T z \\ s.t. \;\; & Az \leq b^N \\ & z \geq 0 \end{aligned}$$

- Il risultato precedente può essere esteso a tutti i giochi originati da un problema lineare (Teorema di Owen, 1975)

9.4.4 Assignment game

Problema di assegnazione

$$\mathcal{A} = (N^v, N^c, A, B)$$

dove	$N^v = \{1, ..., n^v\}$	insieme dei venditori
	$N^c = \{1, ..., n^c\}$	insieme dei compratori
	A	vettore; a_j = valutazione che $j \in N^v$ da al proprio oggetto
	B	matrice; b_{ij} = valutazione che $i \in N^c$ da all'oggetto di $j \in N^v$

- Gli oggetti non hanno un prezzo di mercato
- Ciascun venditore possiede un solo oggetto
- Ciascun compratore può acquistare un solo oggetto

E' possibile associare al problema il gioco TU semplice (N, v) con:

$$N = N^v \cup N^c$$

e v è definita nel modo seguente:

- Se $i^* \in N^c$ e $j^* \in N^v$:

$$v(i^*j^*) = c_{i^*j^*} = \begin{cases} b_{i^*j^*} - a_{j^*} & \text{se } b_{i^*j^*} - a_{j^*} \geq 0 \\ 0 & \text{se } b_{i^*j^*} - a_{j^*} < 0 \end{cases}$$

- Se S contiene più compratori che venditori, detto $i(j) \in S \cap N^c$ il compratore dell'oggetto offerto da $j \in S \cap N^v$:

$$v(S) = max \sum_{j \in S \cap N^v} c_{i(j),j}$$

- Se S contiene più venditori che compratori, detto $j(i) \in S \cap N^v$ il venditore dell'oggetto acquistato da $i \in S \cap N^c$:

$$v(S) = max \sum_{i \in S \cap N^c} c_{i,j(i)}$$

I valori c_{ij} definiscono il problema di assegnazione:

$$\begin{aligned} max \;\; z = & \sum_{i \in N^c, j \in N^v} c_{ij} x_{ij} & \\ s.t. \;\; & \sum_{i \in N^c} x_{ij} \leq 1 & j \in N^v \\ & \sum_{j \in N^v} x_{ij} \leq 1 & i \in N^c \\ & x_{ij} \in \{0,1\} & i \in N^c, j \in N^v \end{aligned}$$

dove $x_{ij} = \begin{cases} 1 & \text{se } i \text{ e } j \text{ si accordano} \\ 0 & \text{altrimenti} \end{cases}$

Per il teorema di Owen il nucleo contiene le imputazioni ottenute da una soluzione ottimale del duale:

$$\begin{aligned} min \;\; w = & \sum_{j \in N^v} y_j^v + \sum_{i \in N^c} y_i^c & \\ s.t. \;\; & y_j^v + y_i^c \geq c_{ij} & j \in N^v, i \in N^c \end{aligned}$$

- Le soluzioni ottimali duali devono avere le componenti non negative per la razionalità individuale

Esempio 9.12 (Gioco di assegnazione) *Ci sono tre giocatori, 1 (venditore, $a_1 = 10$), 2 e 3 (compratori, $b_{21} = 12, b_{31} = 15$)*

$$v(1) = v(2) = v(3) = v(23) = 0; v(12) = 2; v(13) = v(N) = 5$$

$$core(v) = \{x \in \mathbb{R}^3 | x_1 = \alpha, x_2 = 0, x_3 = 5 - \alpha, 2 \leq \alpha \leq 5\}$$

L'oggetto non viene venduto a 2 e il payoff di 1 e 3 dipende da come si accordano, ma l'utilità di 1 è almeno 2 unità. In altre parole il prezzo di vendita è almeno 12 (1 può accordarsi con 2), ma non più di 15 (3 si ritira)
Se $\bar{b}_{21} = 15$ allora $core(v) = \{(5, 0, 0)\}$ cioè il prezzo di vendita è 15 (legge della domanda e dell'offerta) $\diamondsuit$

- Considerazioni economiche

  1. La legge dell'equilibrio tra domanda e offerta dice che il prezzo deve far sì che la domanda sia uguale all'offerta, per cui se il prezzo dell'oggetto fosse inferiore a 12 vi sarebbero due acquirenti mentre se il prezzo fosse superiore a 15 non vi sarebbero acquirenti
  2. Le leggi economiche non escludono, come il nucleo, un'utilità positiva per il giocatore 2; infatti il giocatore 1 potrebbe offrire al giocatore 2 parte della sua utilità in cambio di un'offerta maggiore per far sì che il prezzo pagato dal giocatore 3 sia più alto oppure il giocatore 3 potrebbe offrire al giocatore 2 parte della sua utilità in cambio del suo ritiro per far sì che il prezzo pagato al giocatore 1 sia più basso

10 Soluzioni puntuali di un gioco TU

Prendono frequentemente il nome di *indici di potere* o *valori* perchè permettono di identificare il “potere” di ciascun giocatore all’interno del gioco
Il termine “indice di potere” si usa per i giochi semplici, mentre per un gioco qualsiasi si preferisce il termine “valore”

10.1 Valore di Shapley (1953)

Si basa sul *contributo marginale* di ogni giocatore

Definizione 10.1 *Si chiama valore di Shapley il vettore $\phi(v)$ la cui componente ϕ_i è il contributo marginale medio del giocatore i rispetto alle possibili permutazioni dei giocatori, cioè:*

$$\phi_i(v) = \frac{1}{n!} \sum_{\pi \in \Pi} [v(P(\pi, i) \cup \{i\}) - v(P(\pi, i))]$$

dove $n = |N|$, Π è l’insieme delle permutazioni di N e $P(\pi, i)$ è l’insieme dei giocatori che precedono i nella permutazione π

Il valore di Shapley per un gioco cooperativo esiste ed è unico

Se il gioco è superadditivo (subadditivo per un cost game) il valore di Shapley è un'imputazione:

$$\sum_{i \in N} \phi_i(v) = v(N)$$
$$\phi_i(v) \geq v(i) \qquad\qquad i \in N$$

ma non è necessariamente un elemento del nucleo

Dimostrare che se il gioco è convesso (concavo per un cost game) il valore di Shapley è un elemento del nucleo

Esempio 10.1 (Gioco di assegnazione) *Riferendosi all'Esempio 9.12, dove* $v(1) = v(2) = v(3) = v(23) = 0; v(12) = 2; v(13) = v(123) = 5$ *il valore di Shapley è dato da:*

	Contributi marginali		
Permutazioni	*Giocatore* 1	*Giocatore* 2	*Giocatore* 3
1 2 3	$v(1) - v(\emptyset) = 0$	$v(12) - v(1) = 2$	$v(123) - v(12) = 3$
1 3 2	$v(1) - v(\emptyset) = 0$	$v(123) - v(13) = 0$	$v(13) - v(1) = 5$
2 1 3	$v(12) - v(2) = 2$	$v(2) - v(\emptyset) = 0$	$v(123) - v(12) = 3$
2 3 1	$v(123) - v(23) = 5$	$v(2) - v(\emptyset) = 0$	$v(23) - v(2) = 0$
3 1 2	$v(13) - v(3) = 5$	$v(123) - v(13) = 0$	$v(3) - v(\emptyset) = 0$
3 2 1	$v(123) - v(23) = 5$	$v(23) - v(3) = 0$	$v(3) - v(\emptyset) = 0$
ϕ_i	$\frac{17}{6}$	$\frac{2}{6}$	$\frac{11}{6}$

Il valore di Shapley riflette il valore economico del giocatore 2 $\diamondsuit$

10.1.1 Assiomi di Shapley

Sia data una regola ψ che ad un gioco $G(N, v)$ associa un vettore di $\mathbb{R}^N$

1. Simmetria
   Se due giocatori i, j sono simmetrici, cioè $v(S \cup \{i\}) = v(S \cup \{j\}), S \subseteq N \setminus \{i, j\}$, allora $\psi_i(v) = \psi_j(v)$

2. Dummy player
   Sia i un giocatore fittizio, cioè $v(S \cup \{i\}) = v(S) + v(i), S \subseteq N \setminus \{i\}$, allora $\psi_i(v) = v(i)$

3. Additività o indipendenza (assioma controverso)
   Dati due giochi u e v, sia $(u+v)$ il gioco somma definito da $(u+v)(S) = u(S)+v(S), S \subseteq N$ allora $\psi_i(u + v) = \psi_i(u) + \psi_i(v), i \in N$

ϕ è l'unico vettore efficiente che soddisfa i precedenti assiomi

Esempio 10.2 (Giocatori simmetrici e giocatore fittizio) *Sia dato il gioco* $G = (N, v)$ *dove:*

$$N = \{1, 2, 3\}$$
$$v(1) = v(2) = v(3) = 1; v(12) = 4; v(13) = v(23) = 2; v(N) = 5$$

I giocatori 1 *e* 2 *sono simmetrici e il giocatore* 3 *è fittizio, allora* $\phi_3(v) = v(3) = 1$ *e* $\phi_1(v) = \phi_2(v) = \frac{1}{2}(v(N) - v(3)) = 2$ *e quindi* $\phi(v) = (2, 2, 1)$ $\diamondsuit$

- L'assioma di simmetria può essere sostituito dall'assioma di *anonimato*:
  Dato un gioco v e una permutazione dei giocatori π sia u il gioco definito da $u(\pi(S)) = v(S), S \subseteq N$ allora $\psi_{\pi(i)}(u) = \psi_i(v)$

- L'assioma di dummy player può essere sostituito dall'assioma di *null player*:
  Sia i un giocatore nullo, cioè $v(S \cup \{i\}) = v(S), S \subseteq N \setminus \{i\}$, allora $\psi_i(v) = 0$

10.1.2 Calcolo del valore di Shapley

Il valore di Shapley risulta molto complesso da calcolare

Definizione
E' necessario determinare i contributi marginali dei giocatori nelle $n!$ possibili coalizioni ordinate
Con 10 giocatori, per ogni giocatore ci sono $10! = 3.628.800$ permutazioni

Semplificazione
Considerare le possibili $2^n - 1$ coalizioni non vuote e per ciascuna considerare ogni giocatore come l'ultimo arrivato e quindi "pesare" il suo contributo marginale con le permutazioni degli altri giocatori della coalizione e dei giocatori non facenti parte della coalizione:

$$\phi_i(v) = \sum_{S \subseteq N, i \in S} \frac{(s-1)!(n-s)!}{n!} \left[v(S) - v(S \setminus \{i\})\right]$$

Con 10 giocatori ci sono $2^{10} - 1 = 1.023$ coalizioni

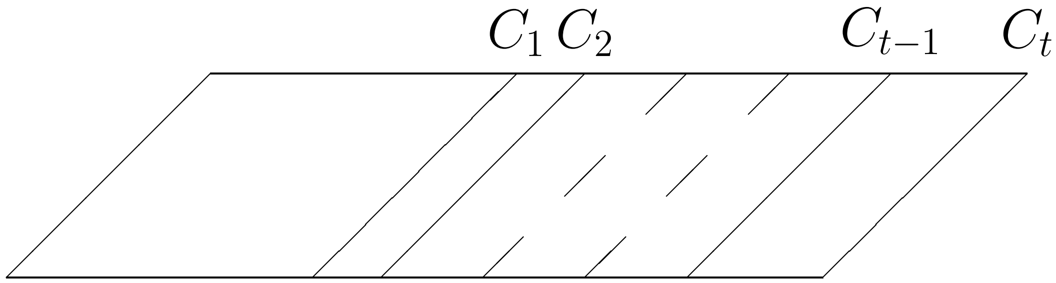
Formule "ad hoc"
Sfruttare le caratteristiche di alcune classi di giochi

Gioco dell'aeroporto (Airport game - Littlechild e Thompson, 1977)

Ripartire il costo di costruzione e manutenzione della pista tra differenti tipi di aerei

Gli aerei sono raggruppati in t sottoinsiemi disgiunti $N_1, \ldots, N_t$

Gli aerei di N_i richiedono una pista di costo C_i con $C_i < C_{i+1}$



Si definisce il gioco:

$$v(S) = C_{j(S)}$$

dove $j(S) = max\ \{i|S \cap N_i \neq \emptyset\}$

Il valore di Shapley di ogni aereo corrisponde alla seguente ripartizione dei costi:

- Il costo del primo tratto di pista C_1 è diviso tra tutti gli aerei, poichè tutti lo utilizzano;
- Il costo del secondo tratto di pista $C_2 - C_1$ è diviso tra gli aerei dei sottoinsiemi $N_2, \ldots, N_t$ che sono quelli che lo utilizzano;
- Il costo dell'ultimo tratto di pista $C_t - C_{t-1}$ che è diviso tra gli aerei del sottoinsieme N_t che sono gli unici che lo utilizzano.

Questo criterio è facilmente applicabile anche nel caso di molti aerei

Esempio 10.3 (Gioco dell'aeroporto)

$$N_1 = \{1, 2, 3\}; N_2 = \{4, 5, 6, 7\}; N_3 = \{8, 9, 10\}$$
$$C_1 = 20; C_2 = 27; C_3 = 33$$
$$\phi_1 = \frac{20}{10} = 2$$
$$\phi_2 = \frac{20}{10} + \frac{27-20}{7} = 3$$
$$\phi_3 = \frac{20}{10} + \frac{27-20}{7} + \frac{33-27}{3} = 5$$

$\diamondsuit$

La verifica utilizza gli assiomi di Shapley (Littlechild e Owen, 1973)
Si definiscono t giochi $v_1, \ldots, v_t$ con il gioco v_i relativo al tratto di pista i:

$$v_i(S) = \begin{cases} C_i - C_{i-1} & se\ i \leq j(S) \\ 0 & se\ i > j(S) \end{cases}$$

dove $C_0 = 0$
A questo punto si osserva che:

1. gli aerei di $N_1, \ldots, N_{i-1}$ sono dummy per il gioco v_i
2. gli aerei di $N_i, \ldots, N_t$ sono simmetrici per il gioco v_i
3. v è dato dalla somma dei giochi v_i

10.1.3 Un'applicazione del valore di Shapley

Esempio 10.4 (Consiglio dell'UE 1958-1973) *Il valore di Shapley permette di evidenziare un difetto nei pesi assegnati nel Consiglio dell'UE del 1958*
$quota_{1958} = 12$ $(17 \times 0.7 = 11.9)$; $quota_{1973} = 41$ $(58 \times 0.7 = 40.6)$

	1958			1973		
Paesi	Peso	%	Shapley	Peso	%	Shapley
Francia	4	23.53	0.233	10	17.24	0.179
Germania	4	23.53	0.233	10	17.24	0.179
Italia	4	23.53	0.233	10	17.24	0.179
Belgio	2	11.76	0.150	5	8.62	0.081
Paesi Bassi	2	11.76	0.150	5	8.62	0.081
Lussemburgo	1	5.88	0.000	2	3.45	0.010
Regno Unito	-	-	-	10	17.24	0.179
Danimarca	-	-	-	3	5.17	0.057
Irlanda	-	-	-	3	5.17	0.057
Totale	17	100.00	1.000	58	100.00	1.000

◇

Dimostrare che il Valore Shapley è monotono rispetto ai pesi

10.2 Nucleolo (Schmeidler, 1969)

Si basa sull'idea di minimizzare il massimo "malcontento"
Principio di Rawls: massimizzare l'utilità dell'agente che ottiene il peggior risultato

Definizione 10.2 *Dato un gioco v, sia S una coalizione e x una possibile ripartizione del valore del gioco; si dice rimpianto o eccesso di S rispetto ad x la quantità:*

$$e(S, x) = v(S) - x(S)$$

Nel caso di un cost game il rimpianto è $x(S) - c(S)$

- Nella definizione precedente x è una ripartizione del valore del gioco in quanto deve soddisfare solo l'ipotesi di efficienza; in questo caso talvolta si usano i termini preimputazione e prenucleolo per indicare che non si tiene conto della razionalità individuale

E' possibile definire il vettore $\vartheta(x) \in \mathbb{R}^{2^n}$:

$$\vartheta_1(x) = max\ \{e(S, x) | S \subset N\} = e(S_1, x)$$
$$\vartheta_i(x) = max\ \{e(S, x) | S \subset N, S \neq S_j, j = 1, \ldots, i-1\} = e(S_i, x) \qquad i = 2, \ldots, 2^n$$

Le componenti di $\vartheta(x)$ sono i rimpianti generati da x al variare di S, in ordine debolmente decrescente

Esempio 10.5 (Vettore degli eccessi) *Sia dato il gioco:*

$$N = \{1, 2, 3\}$$
$$v(1) = v(2) = v(3) = 0; v(12) = 2; v(13) = v(23) = 3; v(N) = 5$$

Data la ripartizione $x = (3, 1, 1)$ *si ha:*

$e(1, x) = -3; e(2, x) = -1; e(3, x) = -1; e(12, x) = -2; e(13, x) = -1; e(23, x) = 1; e(N, x) = 0$

e quindi:

$$\vartheta(x) = (1, 0, -1, -1, -1, -2, -3)$$

◊

Definizione 10.3 *Dati due vettori* $x, y \in \mathbb{R}^n$*, si dice che* x *è lessicograficamente minore di* y *e si indica con* $x <_L y$*, se esiste* $i \geq 1$ *per cui:*

$$x_j = y_j \qquad j < i$$
$$x_i < y_i$$

Definizione 10.4 *Dato un gioco* v *si dice nucleolo del gioco il vettore* $\nu(X)$ *che genera il minimo, secondo l'ordine lessicografico, dei vettori* $\vartheta(x)$ *al variare di* x *nell'insieme* X *delle possibili ripartizioni*

- Il nucleolo è un elemento del nucleo se è non vuoto, per cui costituisce un concetto di soluzione per i giochi a nucleo vuoto, ma permette anche di "scegliere" un elemento del nucleo

Dimostrare che il nucleolo appartiene al nucleo se questo è non vuoto

Esempio 10.6 (Ordine lessicografico) *Sia dato il seguente gioco:*

$$N = \{1, 2\}$$
$$v(1) = 1; v(2) = 3; v(12) = 8$$

Dati $x = (6, 2)$ *e* $y = (3, 5)$ *si ha:*

$$e(1, x) = -5; e(2, x) = 1; e(12, x) = 0$$
$$e(1, y) = -2; e(2, y) = -2; e(12, y) = 0$$

e quindi $\vartheta(x) = (1, 0, -5)$ *e* $\vartheta(y) = (0, -2, -2)$ *per cui* $\vartheta(y) <_L \vartheta(x)$
Si può verificare che $y = \nu(X)$ ♢

Proprietà

Se X è non vuoto, compatto e convesso allora $\nu(X)$ esiste ed è unico

10.2.1 Calcolo del nucleolo

Algoritmo di Kopelowitz (1967)

Il massimo rimpianto delle coalizioni è rappresentato da α:

$$v(S) - x(S) \leq \alpha \qquad S \subset N$$

E' sufficiente cercare il minimo di α

La grande coalizione viene esclusa poichè il suo rimpianto è sempre nullo

α non è vincolata in segno

$$\begin{aligned} &min \;\; \alpha \\ &s.t. \;\; \sum_{i \in N} x_i = v(N) \\ &\qquad \sum_{i \in S} x_i + \alpha \geq v(S) \qquad S \subset N \end{aligned}$$

Se la soluzione non è unica si itera l'algoritmo, conservando il massimo rimpianto ottenuto

Detto S_0 l'insieme delle coalizioni leganti, la nuova ripartizione deve minimizzare il massimo rimpianto per le altre coalizioni, senza incrementare il rimpianto per le coalizioni di S_0, per cui si riscrivono i vincoli:

$$\sum_{i \in S} x_i = v(S) - \alpha_0 \qquad S \in S_0$$

Si ottengono α_1 e S_1; iterando dopo al più n iterazioni la soluzione è unica e costituisce il nucleolo del gioco

Esempio 10.7 (Calcolo del nucleolo) *Sia dato il seguente gioco:*

$$N = \{1, 2, 3\}$$
$$v(1) = v(2) = v(3) = 0; v(12) = 2; v(13) = 3; v(23) = 5; v(N) = 6$$

Il primo problema è:

$$\begin{array}{ll} min & \alpha \\ s.t. & x_1 + x_2 + x_3 = v(N) = 6 \\ & x_1 + x_2 + \alpha \geq v(12) = 2 \\ & x_1 + x_3 + \alpha \geq v(13) = 3 \\ & x_2 + x_3 + \alpha \geq v(23) = 5 \\ & x_1 + \alpha \geq v(1) = 0 \\ & x_2 + \alpha \geq v(2) = 0 \\ & x_3 + \alpha \geq v(3) = 0 \end{array}$$

In forma tabellare:

	x_1	x_2	x_3	α^+	α^-	
v_1	1	1	1	0	0	-6
u_2	1	1	0	1	-1	-2
u_3	1	0	1	1	-1	-3
u_4	0	1	1	1	-1	-5
u_5	1	0	0	1	-1	0
u_6	0	1	0	1	-1	0
u_7	0	0	1	1	-1	0
$-z$	0	0	0	-1	1	0

	v_1	u_4	u_3	α^+	u_5	
x_1	$\frac{1}{2}$	$-\frac{1}{2}$	0	0	$\frac{1}{2}$	$\frac{1}{2}$
u_2	$\frac{1}{2}$	$\frac{1}{2}$	-1	0	$\frac{3}{2}$	$\frac{1}{2}$
x_3	0	0	1	0	-1	3
x_2	$\frac{1}{2}$	$\frac{1}{2}$	-1	0	$\frac{1}{2}$	$\frac{5}{2}$
α^-	$\frac{1}{2}$	$-\frac{1}{2}$	0	1	$-\frac{1}{2}$	$\frac{1}{2}$
u_6	0	1	-1	0	1	2
u_7	$-\frac{1}{2}$	$\frac{1}{2}$	1	0	$-\frac{1}{2}$	$\frac{5}{2}$
$-z$	$\frac{1}{2}$	$-\frac{1}{2}$	0	0	$-\frac{1}{2}$	$\frac{1}{2}$

La soluzione ottimale è $x = \left(\frac{1}{2}, \frac{5}{2}, 3\right)$ *con* $\alpha_0 = -\frac{1}{2}$ *e* $S_0 = \{\{1\}, \{2, 3\}\}$

La soluzione non è unica; si itera riscrivendo i vincoli associati alle coalizioni in S_0

Il secondo problema è:

$$
\begin{array}{ll}
min & \alpha \\
s.t. & x_1 + x_2 + x_3 = v(N) = 6 \\
& x_1 + x_2 + \alpha \geq v(12) = 2 \\
& x_1 + x_3 + \alpha \geq v(13) = 3 \\
& x_2 + x_3 = v(23) - \alpha_0 = \frac{11}{2} \\
& x_1 = v(1) - \alpha_0 = \frac{1}{2} \\
& x_2 + \alpha \geq v(2) = 0 \\
& x_3 + \alpha \geq v(3) = 0
\end{array}
$$

In forma tabellare:

	x_1	x_2	x_3	α^+	α^-	
v_1	1	1	1	0	0	-6
u_2	1	1	0	1	-1	-2
u_3	1	0	1	1	-1	-3
v_4	0	1	1	0	0	$-\frac{11}{2}$
v_5	1	0	0	0	0	$-\frac{1}{2}$
u_6	0	1	0	1	-1	0
u_7	0	0	1	1	-1	0
$-z$	0	0	0	-1	1	0

	v_1	v_4	u_3	α^+	u_2	
x_1	1	-1	0	0	0	$\frac{1}{2}$
α^-	1	$-\frac{1}{2}$	$-\frac{1}{2}$	1	$-\frac{1}{2}$	$\frac{3}{4}$
x_3	0	$\frac{1}{2}$	$\frac{1}{2}$	0	$-\frac{1}{2}$	$\frac{13}{4}$
x_2	0	$\frac{1}{2}$	$-\frac{1}{2}$	0	$\frac{1}{2}$	$\frac{9}{4}$
v_5	1	-1	0	0	0	0
u_6	-1	1	0	0	1	$\frac{3}{2}$
u_7	-1	1	1	0	0	$\frac{5}{2}$
$-z$	1	$-\frac{1}{2}$	$-\frac{1}{2}$	0	$-\frac{1}{2}$	$\frac{3}{4}$

La soluzione ottimale è $x = \left(\frac{1}{2}, \frac{9}{4}, \frac{13}{4}\right)$ *con* $\alpha_1 = -\frac{3}{4}$ *e* $S_1 = \{\{1,2\},\{1,3\}\}$

La soluzione è unica e la ripartizione trovata costituisce il nucleolo $\diamondsuit$

11 Giochi su reti

11.1 Grafi e cooperazione

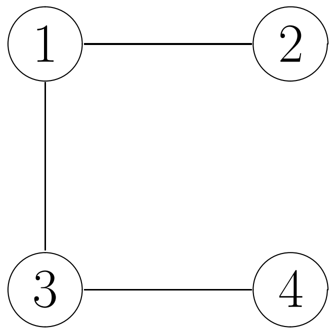
Non necessariamente tutti i giocatori possono o vogliono formare qualsiasi coalizione (coalizioni politiche)
Si considerano strutture cooperative, che possono essere rappresentate con un grafo
L'arco $a_{ij} = (v_i, v_j)$ indica che i e j sono disposti a far parte della stessa coalizione (Myerson, 1977)

Può essere necessario che ciascun giocatore debba essere disposto a unirsi a tutti gli altri (sottografo completo), o può essere sufficiente la disponibilità a formare la coalizione "attraverso alcuni giocatori intermedi" (sottografo connesso)

<u>Gioco ristretto rispetto a un grafo G</u>
I giocatori possono realizzare solo le coalizioni indotte dal grafo G

- Esiste una formulazione più restrittiva in cui i e j possono far parte della stessa coalizione solo se entrambi esprimono questa volontà (Claim game). In questo caso è necessario utilizzare un grafo orientato, in cui l'arco a_{ij} esprime che i accetta di entrare in una coalizione con j, ma non viceversa

Esempio 11.1 (Coalizioni indotte)



Se una coalizione richiede la reciprocità si possono formare le coalizioni $1-2-3-4-12-13-34$
Il gioco ristretto v_G *è:*

$$\begin{array}{lll} v_G(1) = v(1) & v_G(12) = v(12) & v_G(123) = max\ \{v(12) + v(3), v(13) + v(2)\} \\ v_G(2) = v(2) & v_G(13) = v(13) & v_G(124) = v(12) + v(4) \\ v_G(3) = v(3) & v_G(14) = v(1) + v(4) & v_G(134) = max\ \{v(13) + v(4), v(1) + v(34)\} \\ v_G(4) = v(4) & v_G(23) = v(2) + v(3) & v_G(234) = v(2) + v(34) \\ & v_G(24) = v(2) + v(4) & \\ & v_G(34) = v(34) & \end{array}$$

$$v_G(N) = max\ \{v(12) + v(34), v(13) + v(2) + v(4)\}$$

Se una coalizione non richiede la reciprocità si possono formare le coalizioni $1 - 2 - 3 - 4 - 12 - 13 - 34 - 123 - 134 - 1234$

Il gioco ristretto v_G *è:*

$$\begin{array}{lll}
v_G(1) = v(1) & v_G(12) = v(12) & v_G(123) = v(123) \\
v_G(2) = v(2) & v_G(13) = v(13) & v_G(124) = v(12) + v(4) \\
v_G(3) = v(3) & v_G(14) = v(1) + v(4) & v_G(134) = v(134) \\
v_G(4) = v(4) & v_G(23) = v(2) + v(3) & v_G(234) = v(2) + v(34) \\
 & v_G(24) = v(2) + v(4) & \\
 & v_G(34) = v(34) & \\
v_G(N) = v(1234) & &
\end{array}$$

$\diamondsuit$

Regola di allocazione rispetto a un grafo G

$$x : \mathcal{G} \to \mathbb{R}^n \; t.c. \sum_{i \in S} x_i(G) = v_G(S) \qquad S \in N/G$$

dove $\mathcal{G}$ è l'insieme dei grafi e N/G è l'insieme delle coalizioni indotte da una componente connessa di G

Stabilità
Una regola di allocazione x è stabile se e solo se:

$$x_i(G) \geq x_i(G \setminus (i,j)) \; e \; x_j(G) \geq x_j(G \setminus (i,j)) \qquad (i,j) \in G$$

Equità
Una regola di allocazione x è equa se e solo se:

$$x_i(G) - x_i(G \setminus (i,j)) = x_j(G) - x_j(G \setminus (i,j)) \qquad (i,j) \in G$$

Teorema 11.1 (Myerson, 1977) *L'unica regola di allocazione equa è il valore di Shapley del gioco ristretto rispetto al grafo Inoltre se il gioco è superadditivo la regola di allocazione è stabile*

E' noto come *Valore di Myerson*

Esempio 11.2 (Valore di Myerson) *Sia dato il gioco:*

$N = 1, 2, 3$

$v(1) = 1; v(2) = 2; v(3) = 3; v(12) = 6; v(13) = 8; v(23) = 10; v(N) = 12$

I giochi ristretti (non reciproci) e i valori di Shapley sono:

G	$\{1\}$	$\{2\}$	$\{3\}$	$\{12\}$	$\{13\}$	$\{23\}$	N	ϕ		
$\emptyset$	1	2	3	3	4	5	6	1	2	3
a_{12}	1	2	3	6	4	5	9	2.5	3.5	3
a_{13}	1	2	3	3	8	5	10	3	2	5
a_{23}	1	2	3	3	4	10	11	1	4.5	5.5
a_{12}, a_{13}	1	2	3	6	8	5	12	4.167	3.167	4.667
a_{12}, a_{23}	1	2	3	6	4	10	12	1.833	5.333	4.833
a_{13}, a_{23}	1	2	3	3	8	10	12	2	3.5	6.5
a_{12}, a_{13}, a_{23}	1	2	3	6	8	10	12	2.5	4	5.5

$\diamondsuit$

11.2 Giochi sugli archi e sui nodi

Classi di giochi riferiti ad un problema che può essere rappresentato tramite una rete, esogena al gioco
Appartengono agli *Operations Research Games*, interessanti da un punto di vista strutturale e computazionale in quanto "ereditano" dalla struttura del problema alcune caratteristiche che permettono di semplificare alcuni aspetti complessi della Teoria dei Giochi (funzione caratteristica e suo significato)

Si distinguono due tipi di giochi su reti:
<u>Giochi sugli archi</u>: Gli agenti controllano gli archi
<u>Giochi sui nodi</u>: Gli agenti controllano i nodi

Il controllo non è necessariamente biunivoco:

- un agente può controllare più elementi
- un elemento può essere controllato da più agenti (comitato); le decisioni possono venir prese da ogni singolo agente (disgiunzione), a maggioranza, all'unanimità, etc.
- possono esistere elementi non controllati da alcun agente (pubblici)

11.2.1 Flow Game

Gli agenti usano la rete per inviare il massimo flusso da una sorgente (origine) ad un pozzo (destinazione):

$$\mathcal{MF} = (N, V, A, s, t, c, o)$$

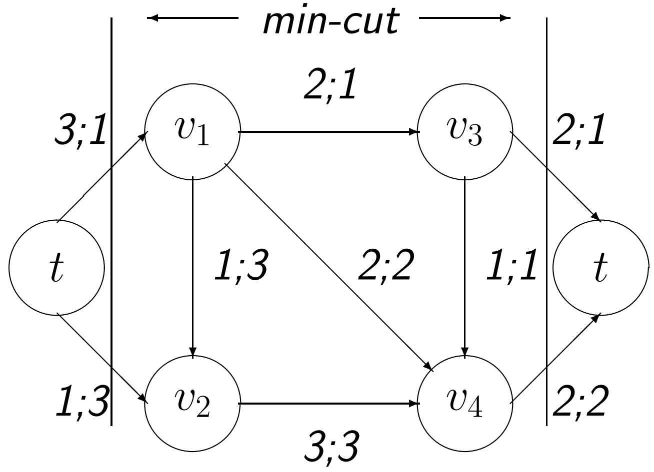
dove	$N = \{1, ..., n\}$	insieme degli agenti
	V	insieme dei nodi
	A	insieme degli archi (orientati)
	$s \in V$	sorgente
	$t \in V \setminus \{s\}$	pozzo
	$c : A \rightarrow \mathbb{R}_{>}$	capacità degli archi
	$o : A \rightarrow N$	controllo degli archi

Ogni arco è controllato da un solo agente
Ogni agente può controllare più di un arco
Non ci sono archi pubblici

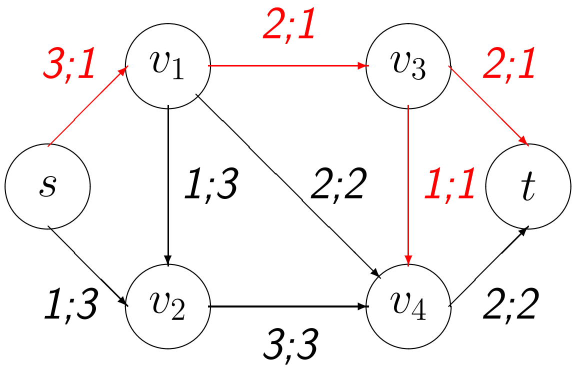
Il corrispondente TU-game (N, v) può essere definito come segue (Kalai, Zemel, 1982):

- l'insieme dei giocatori è N
- ogni coalizione $S \subseteq N$ può usare solo gli archi che controlla, quindi $v(S)$ è il valore del flusso massimo da s a t sulla sottorete generata da $o^{-1}(S)$

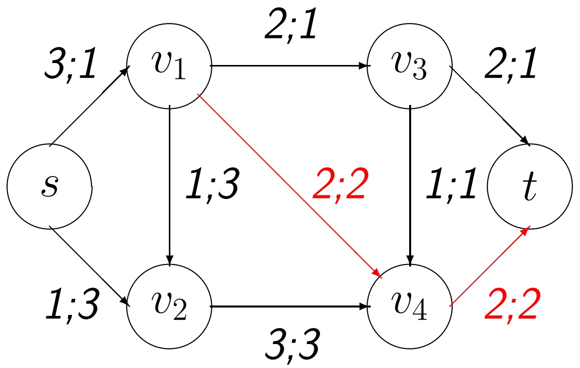
Esempio 11.3 (Flow game) *Si consideri la seguente rete con* $N = \{1, 2, 3\}$*, dove la notatione sull'arco* a *è* $c(a); o(a)$*:*



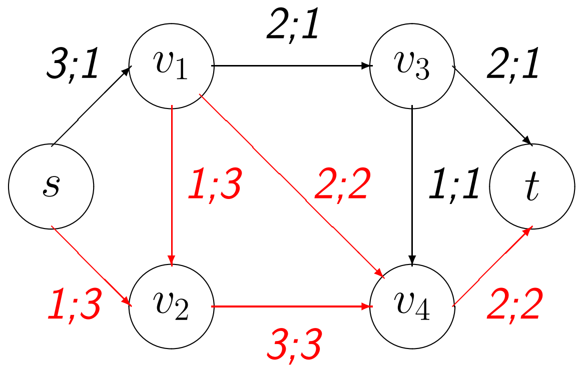
Il valore del flusso massimo è 4 e rappresenta $v(123)$*. Altri esempi di valori sono:*



$v(1) = 2$



$v(2) = 0$



$v(23) = 1$

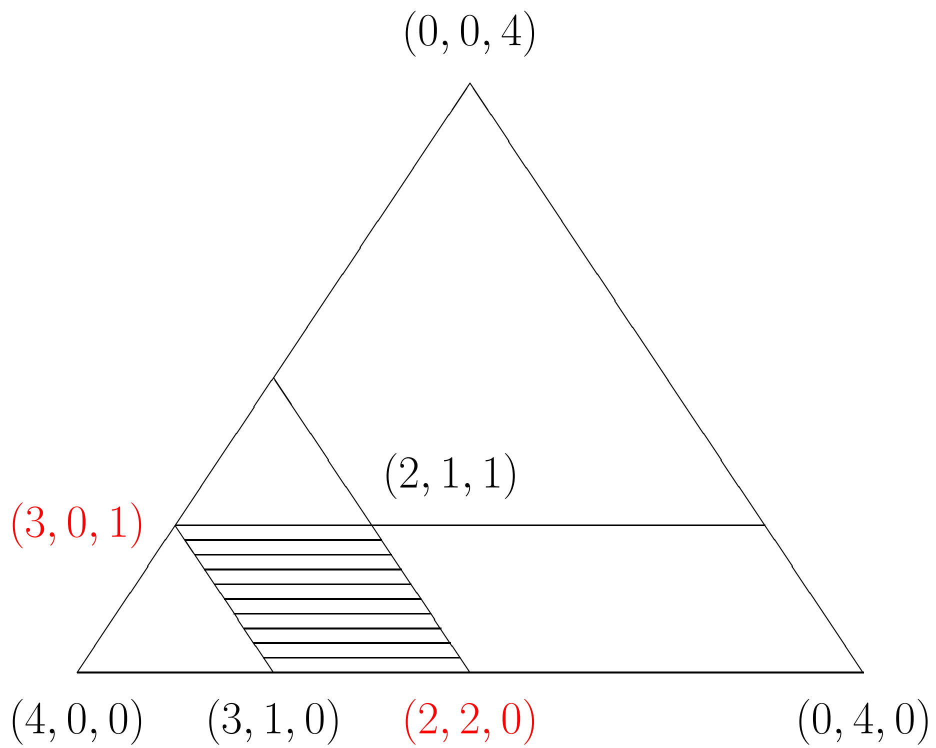
$\diamondsuit$

I flow game sono totalmente bilanciati, poiché ogni gioco ridotto è un flow game
Un'allocazione nel nucleo può essere ottenuta tramite un taglio minimo, secondo il Teorema di Owen

Esempio 11.4 (Flow game) *In riferimento all'Esempio 11.3 la funzione caratteristica e il nucleo sono:*

S	1	2	3	12	13	23	123
$v(S)$	2	0	0	3	2	1	4

$C(v) = hull\{(3,1,0),(2,2,0),(3,0,1),(2,1,1)\}$



◇

Le allocazioni duali possono non essere eque

Esempio 11.5 (Nucleo di un flow game)

$$s \xrightarrow{1;1} v_1 \xrightarrow{2;2} t$$

$$v(1) = v(2) = 0; v(12) = 1$$

L'unico taglio minimo $\{(s, v_1)\}$ *genera* $x = (1, 0)$*, mentre i giocatori sono simmetrici, ma non lo sono gli agenti e il giocatore* 2 *è in una posizione migliore* $\diamondsuit$

Teorema 11.2 (Kalai and Zemel, 1982)
Un gioco è totalmente bilanciato se e solo se è un flow game

Definizione 11.1 *Dati due giochi (N, u) e (N, v) con lo stesso insieme di giocatori, il* gioco minimo di u e v *è il gioco con insieme di giocatori N e funzione caratteristica $w = min(u, v)$ dove $w(S) = min\{u(S), v(S)\}, S \subseteq N$*

- Se i giochi (N, u) e (N, v)sono totalmente bilanciati, anche (N, w) è totalmente bilanciato; inoltre, $C(u_S) \subseteq C(w_S)$ o $C(v_S) \subseteq C(w_S)$ per ogni $S \subseteq N$

Teorema 11.3 *Un gioco è totalmente bilanciato se e solo se è il gioco minimo di un insieme finito di giochi additivi*

Dimostrazione
Per l'osservazione precedente, la condizione è sufficiente
Viceversa, sia v un gioco totalmente bilanciato e sia $\{v^S\}_{S\subseteq N}$ un insieme finito di giochi additivi con insieme di giocatori N definiti come segue:
per ogni $S \subseteq N$

- sia $(x_i)_{i\in S}$ un'allocazione nel nucleo di v_S
- siano $x_j,\ j \in N \setminus S\ n - s$ numeri reali tali che $x_j \geq v(N)$
- sia v^S il gioco additivo generato da $(x_1, ..., x_n)$

allora:

$$\min_{S\subseteq N}\{v^S(T)\} = v^T(T) = v(T), \forall\ T \subseteq N \Rightarrow v = min\{v^S\}$$

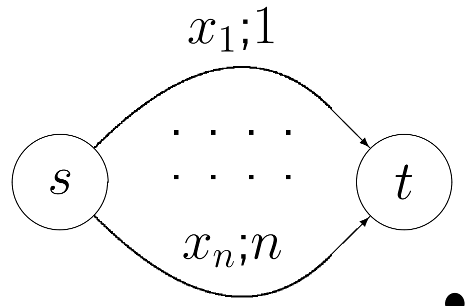
♣

- se $T \subseteq S$, $(x_i)_{i\in S} \in C(v_S) \Rightarrow v^S(T) \geq v(T)$; in particolare $v^T(T) = v(T)$
- se $T \not\subseteq S$ let $j \in (N \setminus S) \cap T$, $x_j \geq v(N) \Rightarrow v^S(T) \geq v(T)$

Lemma 11.1 *Ogni gioco additivo è un flow game*

<ins>Dimostrazione</ins>

Il gioco additivo generato da $(x_1, ..., x_n)$ corrisponde al flow game associato alla rete con n archi che collegano s e t, con $o(a_i) = i$ e $c(a_i) = x_i, i \in N$

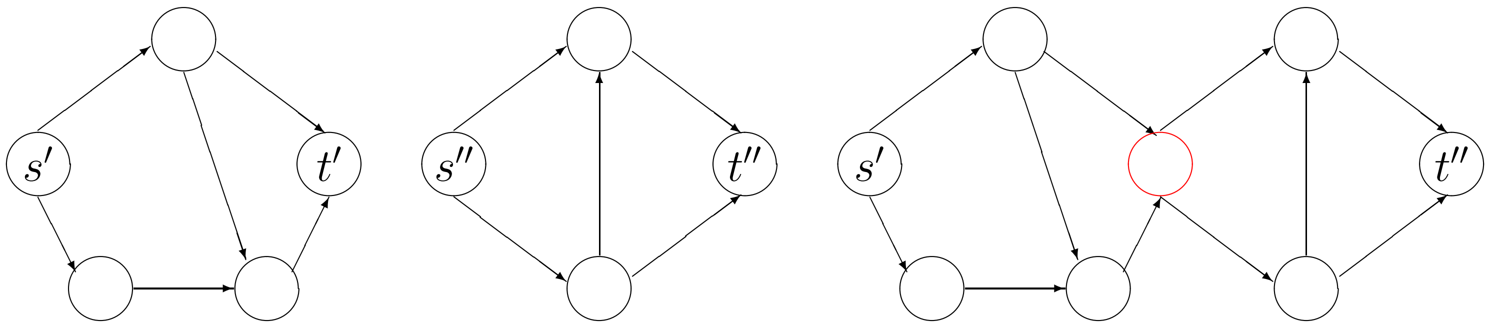


♣

Lemma 11.2 *Il gioco minimo di due flow game è un flow game*

<ins>Dimostrazione</ins>

Dati due flow game (N, v') e (N, v'') associati alle reti G' e G'', il gioco $v = min\{v', v''\}$ è associato alla rete ottenuta facendo coincidere il pozzo di G' con la sorgente di G''



♣

<ins>Dimostrazione del Teorema di Kalai e Zemel</ins>

Per i Lemmi 11.1, 11.2 e il Teorema 11.4 la condizione è sufficiente

Viceversa, un flow game è totalmente bilanciato poiché ogni sottogioco è un flow game ♣

11.2.2 Shortest Path Game

Gli agenti usano la rete per trasportare un bene da un'origine ad una destinazione:

$$\mathcal{SP} = (N, V, A, S, T, c, o, \pi)$$

dove	$N = \{1, ..., n\}$	insieme degli agenti
	V	insieme dei nodi
	A	insieme degli archi (orientati)
	$O \subset V$	insieme delle origini
	$D \subset V \setminus O$	insieme delle destinazioni
	$c : A \longrightarrow \mathbb{R}_{\geq}$	costo degli archi
	$o : V \longrightarrow N$	controllo dei nodi
	$\pi \in \mathbb{R}_{\geq}$	ricavo per trasportare il bene

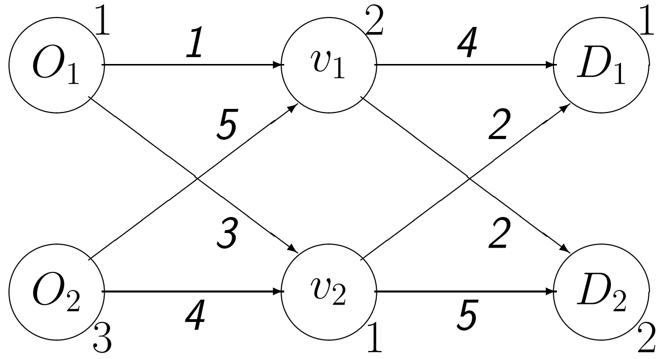
Ogni nodo è controllato da un solo agente
Ogni agente può controllare più di un nodo
Non ci sono nodi pubblici

Il corrispondente TU-game (N, v) può essere definito come segue (Fragnelli, Garcia-Jurado, Mendez-Naya, 2000):

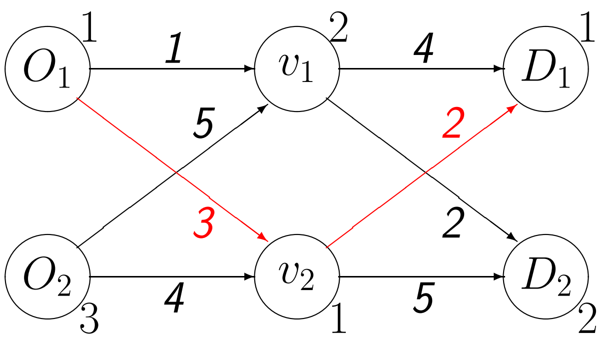
- l'insieme dei giocatori è N
- ogni coalizione $S \subseteq N$ può usare solo cammini attraverso i nodi in $o^{-1}(S)$ che collegano un'origine e una destinazione; Il costo di un cammino è la somma dei costi degli archi, e il costo c_S per la coalizione S è il costo del cammino minimo controllato da S, se esiste, per cui:

$$v(S) = \begin{cases} \pi - c_S & \text{se } S \text{ controlla un cammino e } c_S \leq \pi \\ 0 & \text{altrimenti} \end{cases}$$

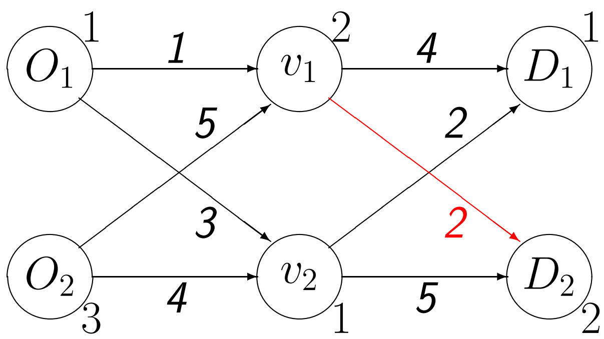
Esempio 11.6 (Shortest path game) *Si consideri la seguente rete con* $N = \{1, 2, 3\}$*, dove i numeri vicino ai nodi indicano il controllore:*



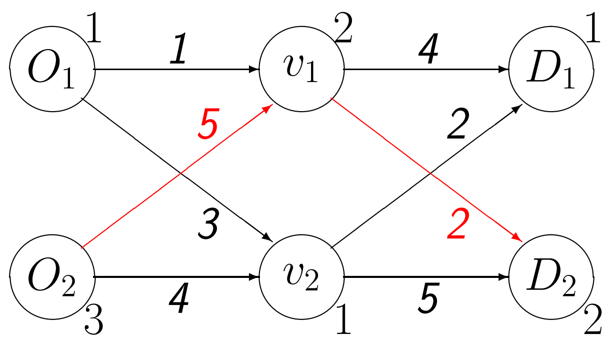
Il valore del cammino minimo è 3 e se $\pi = 6$ *allora* $v(123) = 3$*. Altri esempi di valori sono:*



$v(1) = 1$



$v(2) = 0$ $(nessun\ cammino)$



$v(23) = 0$ $(c_{\{23\}} > \pi)$ ♢

Proposizione 11.1 *La classe degli shortest path game coincide con la classe dei giochi monotoni*

Il nucleo di uno shortest path game può essere vuoto
Il bilanciamento dipende strettamente dal valore di π

Teorema 11.4 *Uno shortest path game è bilanciato per ogni $\pi \in \mathbb{R}_{\geq}$ oppure esiste $\widetilde{\pi} \in \mathbb{R}_{\geq}$ tale che il gioco è bilanciato se e solo se $\pi \leq \widetilde{\pi}$*

Caratterizzazione degli shortest path game bilanciato

Definizione 11.2 *Dato uno shortest path game* (N, v)*, un cammino* è proficuo *per* (N, v) *se il suo costo è minore di* π

(N, v) è non banale *se ci sono cammini proficui o, equivalentemente, se* $v(N) > 0$

Un giocatore è s-veto *per* (N, v) *se controlla almeno un nodo in ogni cammino minimo*

Si noti che, per un gioco non banale (N, v), $i \in N$ è s-veto se e solo se $v(N \setminus i) < v(N)$

Proposizione 11.2 *Dato uno shortest path game non banale* (N, v)*, sia* V *l'insieme dei giocatori s-veto, e sia* x *un'imputazione tale che* $x_i > 0$ *per qualche* $i \in N \setminus V$*. Allora* $x \notin C(v)$

Teorema 11.5 *Uno shortest path game non banale* (N, v) *con insieme dei giocatori s-veto* V *è bilanciato se e solo se valgono le due condizioni:*

1. V *è non vuoto e il gioco ristretto* $(V, \bar{v})$ *è bilanciato*

2. *Ogni cammino proficuo contiene un nodo controllato da un giocatore s-veto*

Corollario 11.1 *Sia* $\bar{x}$ *la restrizione di* $x \in \mathbb{R}^N$ *a* V*, allora* $x \in C(v)$ *se e solo se* $\bar{x} \in C(\bar{v})$

Esempio 11.7 (Nucleo di uno shortest path game) *In riferimento all'Esempio 11.6, i cammini proficui sono $O_1 - v_1 - D_1$ and $O_1 - v_1 - D_2$; il secondo è l'unico cammino minimo, per cui $V = \{1, 2\}$ La funzione caratteristica e il nucleo sono:*

S	1	2	3	12	13	23	123
$v(S)$	1	**0**	**0**	3	1	0	3
$\bar{v}(S)$	1	**0**	$-$	3	$-$	$-$	$-$

$C(\bar{v}) = \{(1 + \alpha, 2 - \alpha)\ s.t.\ 0 \leq \alpha \leq 2\}$

$C(v) = \{(1 + \alpha, 2 - \alpha, 0)\ s.t.\ 0 \leq \alpha \leq 2\}$ ♢

Il valore di Shapley

Poiché il nucleo può essere vuoto, una soluzione alternativa è il valore di Shapley

Il valore di Shapley è l'unica regola di allocazione che soddisfa le seguenti proprietà:

- *Efficienza* - tutta l'utilità viene allocata
- *Giocatori irrilevanti* - i giocatori che controllano solo nodi isolati devono ricevere 0
- *Equità per giocatori adiacenti* - aggiungendo o eliminnado un arco, i giocatori che controllano i suoi estremi hanno la stessa variazione
- *Equità per giocatori non connessi* - due giocatori non connessi hanno la stessa variazione se si elimina l'altro giocatore

11.2.3 Minimum Cost Spanning Tree Game

Gli agenti necessitano di una rete per collegarsi ad una sorgente da cui ricevono un servizio:

$$\mathcal{MCST} = (N, V, A, 0, c, o)$$

dove	$N = \{1, ..., n\}$	insieme degli agenti
	V	insieme dei nodi
	A	insieme degli archi (non orientati)
	$0 \in V$	sorgente
	$c : A \to \mathbb{R}_{\geq}$	costo di costruzione degli archi
	$o : V \setminus \{0\} \to N$	controllo dei nodi

Il grafo (V, A) è supposto completo (clique)

Ogni agente controlla un solo nodo

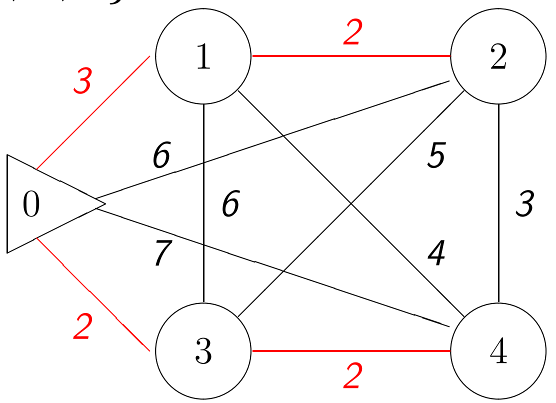
Ogni nodo può essere controllato da più agenti, ma solitamente è uno solo

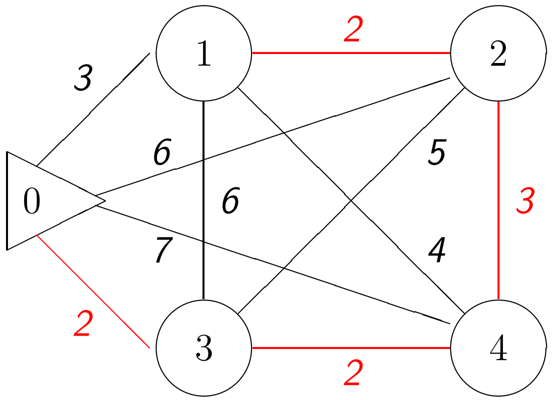
La sorgente è l'unico nodo pubblico

Il corrispondente TU-game (N, c) può essere definito come segue (Granot, Huberman, 1981 and 1984):

- l'insieme dei giocatori è N
- ogni coalizione $S \subseteq N$ paga il costo degli archi necessari a collegare tutti i giocatori di S con la sorgente, per cui $c(S)$ è il costo minimo degli alberi che ricoprono $o^{-1}(S) \cup \{0\}$
  Si noti che l'albero di costo minimo di una coalizione S deve ricoprire solo $o^{-1}(S) \cup \{0\}$ oppure può ricoprire anche altri nodi se questo permette di ridurre il costo, ottenendo un gioco monotono

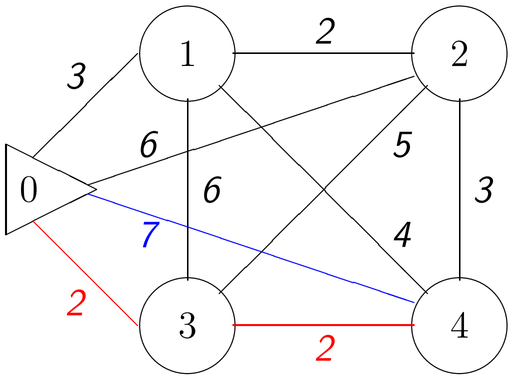
Esempio 11.8 (Minimum cost spanning tree game) *Si consideri la seguente rete con* $N = \{1, 2, 3, 4\}$*:*





Costo del minimo spanning tree = 9 ($= v(1234)$*)*

Per esempio, per la coalizione $\{4\}$ *il costo è:*



$v(4) = 7$; $v(4) = 4$ $(= v(34))$

◇

- I minimum cost spanning tree game sono bilanciati

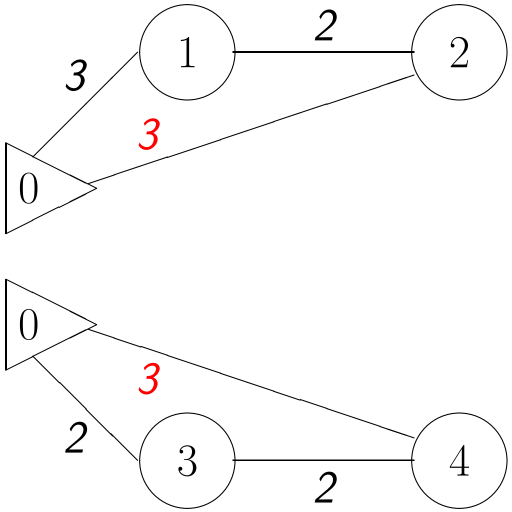
- Un'allocazione nel nucleo può essere ottenuta con la regola di Bird (1976), che assegna al giocatore $i \in N$ il costo dell'arco "entrante" nel nodo nello spanning tree di costo minimo relativo alla grande coalizione
L'allocazione di Bird dipende dallo spanning tree se ne esiste più di uno; è favorevole ai giocatori che hanno meno successori nello spanning tree di costo minimo

- Il nucleo e il nucleolo possono essere determinati più semplicemente decomponendo il gioco in sottogiochi secondo una *struttura efficiente* $\{P_1, ..., P_m\}$, cioè una partizione di N identificata dai sottoalberi con radice nella sorgente $\{0\}$
I costi degli archi $(0, j)$ devono essere ridefiniti come:

$$c'_{0j} = min \left\{ c_{0j}, \min_{k \notin P_j} \{c_{jk}\} \right\}, \qquad j \in P_j$$

Il nucleo e il nucleolo sono dati dal prodotto cartesiano dei corrispondenti nuclei e nucleoli dei sottogiochi.

Esempio 11.9 (Decomposizione) *In riferimento all'Esempio 11.8, si consideri la decomposizione:*



$$
\begin{array}{ll}
c_{12}(1) = 3; c_{12}(2) = 3; c_{12}(12) = 5 & \\
c_{34}(3) = 2; c_{34}(4) = 3; c_{34}(34) = 4 & \\
C(c_{12}) = \{(2+\alpha, 3-\alpha) \ s.t. \ 0 \leq \alpha \leq 1\} & \nu(c_{12}) = (2.5, 2.5) \\
C(c_{34}) = \{(1+\beta, 3-\beta) \ s.t. \ 0 \leq \beta \leq 1\} & \nu(c_{34}) = (1.5, 2.5) \\
C(c) = \{(2+\alpha, 3-\alpha, 1+\beta, 3-\beta) \ s.t. \ 0 \leq \alpha \leq 1; 0 \leq \beta \leq 1\} & \nu(c) = (2.5, 2.5, 1.5, 2.5)
\end{array}
$$

$\diamondsuit$

11.2.4 Fixed Tree Game

Gli agenti usano una rete ad albero per collegarsi ad una sorgente da cui ricevono un servizio:

$$\mathfrak{FT} = (N, V, A, 0, c, o)$$

dove	$N = \{1, ..., n\}$	insieme degli agenti
	V	insieme dei nodi
	A	insieme degli archi (non orientati)
	$0 \in V$	sorgente
	$c : A \to \mathbb{R}_{\geq}$	costo di manutenzione degli archi
	$o : V \setminus \{0\} \to N$	controllo dei nodi

Ogni agente controlla un solo nodo

Ogni nodo può essere controllato da più agenti, ma solitamente è uno solo

La sorgente è l'unico nodo pubblico

Il corrispondente TU-game (N, c) può essere definito come segue (Maschler, Potters, Reijnierse, 1995 e Biørndal, Koster, Tijs, 1999):

- l'insieme dei giocatori è N
- ogni coalizione $S \subseteq N$ paga il costo di manutenzione degli archi necessari a collegare tutti i giocatori di S con la sorgente, per cui $c(S)$ è il costo del sottoalbero che ricopre $o^{-1}(S) \cup \{0\}$:

$$c(S) = \min_{T \supseteq S} \left\{ \sum_{i \in T} c(a_i) \right\} \qquad \forall S \subseteq N$$

  dove a_i è l'unico arco entrante in $o^{-1}(i)$ e T è la componente connessa dell'albero che comprende la sorgente

Ogni allocazione nel nucleo può essere ottenuta dividendo il costo di ciascun arco tra i giocatori che controllano i nodi della componente connessa dell'albero che non comprende la sorgente dopo aver rimosso l'arco

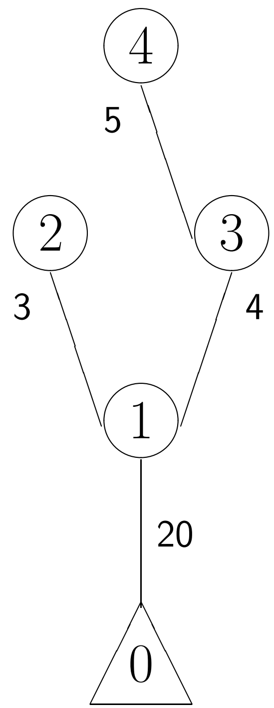
Dimostrare che la condizione non è necessaria

Le regole di allocazione possono essere descritte tramite una *painting story*

Valore di Shapley di un fixed tree game

- Tutti i giocatori dipingono alla stessa velocità; tutti cominciano allo stesso istante, partendo dalla sorgente verso i nodi che controllano, dove terminano
- Quando un arco è completato i giocatori vengono riassegnati con le stesse regole

Esempio 11.10 (Valore di Shapley di un fixed tree game)



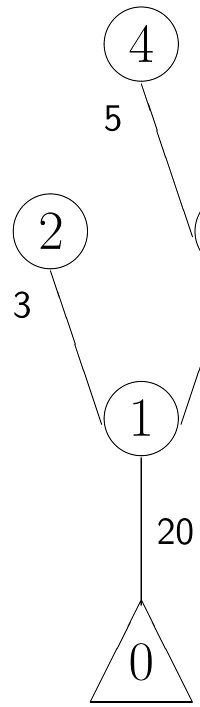
- *Asseggnzione iniziale:*
  $1 \to (0,1), 2 \to (0,1), 3 \to (0,1), 4 \to (0,1)$
  dopo 5 unità di tempo, l'arco $(0,1)$ *è completo e* 1 *termina*
- *Nuove assegnazioni:*
  $2 \to (1,2), 3 \to (1,3), 4 \to (1,3)$
  dopo 2 unità di tempo, l'arco $(1,3)$ *è completo e* 3 *termina*
- *Ultime assegnazioni:*
  $2 \to (1,2), 4 \to (3,4)$
  2 *termina dopo 1 unità di tempo*
  4 *termina dopo 5 unità di tempo*

Il valore di Shapley è $\phi = (5, 8, 7, 12)$ $\diamondsuit$

Nucleolo di un fixed tree game

- Tutti i giocatori dipingono alla stessa velocità; tutti cominciano allo stesso istante
- Ogni giocatore parte dal secondo arco più vicino non completato; l'arco entrante nel nodo che controlla è l'ultimo
- Quando un arco è completato i giocatori vengono riassegnati con le stesse regole

Esempio 11.11 (Nucleolo di un fixed tree game)



- *Assegnazione iniziale:*
  $1 \rightarrow (0,1), 2 \rightarrow (0,1), 3 \rightarrow (0,1), 4 \rightarrow (1,3)$
  dopo 4 unità di tempo 4 *completa l'arco* $(1,3)$
- *Nuove assegnazioni:*
  $1 \rightarrow (0,1), 2 \rightarrow (0,1), 3 \rightarrow (0,1), 4 \rightarrow (0,1)$
  dopo 2 unità di tempo, l'arco $(0,1)$ *è completo;* 1 *e* 3 *terminano;*
- *Ultime assegnazioni:*
  $2 \rightarrow (1,2), 4 \rightarrow (3,4)$
  2 *termina dopo 3 unità di tempo*
  4 *termina dopo 5 unità di tempo*

Il nucleolo è $\nu = (6,9,6,11)$ ♢

11.2.5 Minimum Cost Forest Game

Gli agenti necessitano di una rete per collegarsi a varie sorgenti che forniscono servizi che possono richiedere:

$$\mathcal{MCF} = (N, V, A, O, c, o, R)$$

dove	$N = \{1, ..., n\}$	insieme degli agenti
	V	insieme dei nodi
	A	insieme degli archi (non orientati)
	$O \subset V$	insieme delle sorgenti
	$c : A \to \mathbb{R}_{\geq}$	costo di costruzione degli archi
	$o : V \setminus O \to N$	controllo dei nodi
	R	matrice booleana $n \times card(O)$ delle richieste di collegamento

Ogni agente controlla un solo nodo
Ogni nodo è controllato da un solo agente
Le sorgenti sono gli unici nodi pubblici
Gli archi possono essere utilizzati per ogni collegamento

Il corrispondente TU-game (N, c) può essere definito come segue (Kuiper, 1997):

- l'insieme dei giocatori è N
- ogni coalizione $S \subseteq N$ paga il costo di costruzione degli archi necessari a collegare tutti i giocatori di S con tutte le sorgenti che almeno uno richiede, per cui $c(S)$ è il costo della foresta di costo minimo che soddisfa le richieste di tutti i giocatori di S (Algoritmo di Kruskal modificato)

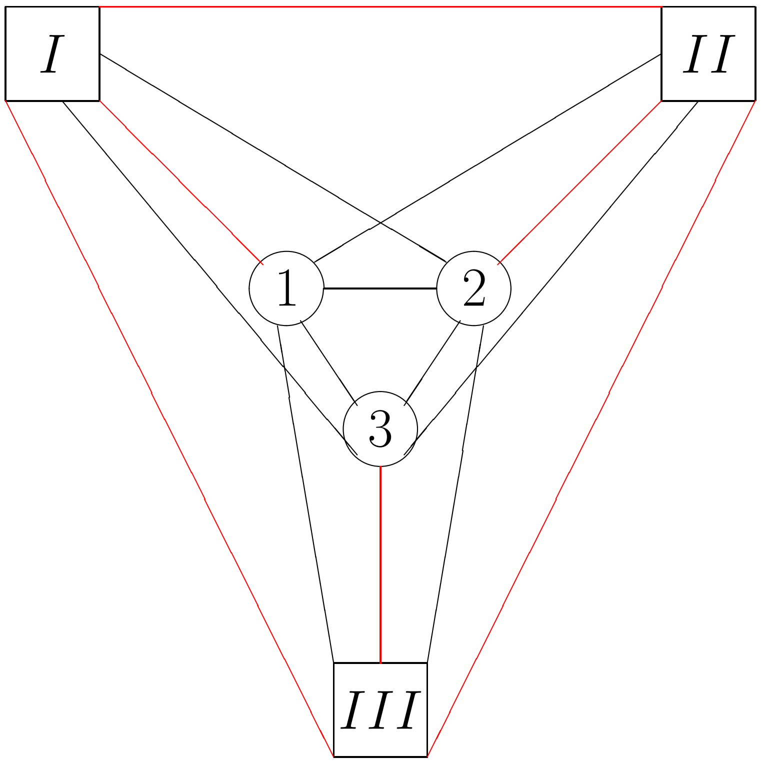
Teorema 11.6 (Kuipers, 1977)
Il nucleo di un Minimum Cost Forest Game è non vuoto se:

1. *esiste almeno un nodo connesso con tutte le sorgenti (la soluzione è un albero)*
2. *nessun nodo è connesso con più di una sorgente (la soluzione è un insieme di alberi)*

- Se il gioco ha al più due sorgenti o al più due giocatori il nucleo è non vuoto

Esempio 11.12 (Nucleo vuoto) *I giocatori sono* $N = \{1, 2, 3\}$*; le sorgenti sono* $O = \{I, II, III\}$*; le richieste giocatore-sorgente sono:*

	I	II	III
1	1	0	0
2	0	1	0
3	0	0	1



Gli archi 'neri' *hanno costo* 3 *e quelli* 'rossi' *hanno costo* 5

La collezione bilanciata $\{\{12\}, \{13\}, \{23\}\}$ *porta a:*

$$c(N) = C(I-2, 2-1, 1-II) + C(III-3) = 14$$
$$c(12) = C(I-2, 2-1, 1-II) = 9$$
$$c(13) = C(I-3, 3-1, 1-III) = 9$$
$$c(23) = C(II-3, 3-2, 2-III) = 9$$

e quindi:

$$c(N) = 14 > \frac{27}{2} = \frac{1}{2}\, c(12) + \frac{1}{2}\, c(13) + \frac{1}{2}\, c(23)$$

per cui il nucleo è vuoto

$\diamondsuit$

11.2.6 Traveling Salesman Game

Gli agenti necessitano di un circuito hamiltoniano per collegarsi ad una sorgente che fornisce un servizio:

$$\mathcal{TS} = (N, V, A, 0, c, o)$$

dove	$N = \{1, ..., n\}$	insieme degli agenti
	V	insieme dei nodi
	A	insieme degli archi (orientati)
	$0 \in V$	sorgente
	$c : A \to \mathbb{R}_{\geq}$	costo degli archi
	$o : V \setminus \{0\} \to N$	controllo dei nodi

Il grafo (V, A) deve essere fortemente

Ogni agente controlla un solo nodo

Ogni nodo è controllato da un solo agente

La sorgente è l'unico nodo pubblico

Il corrispondente TU-game (N, c) può essere definito come segue (Potters, Curiel, Tijs, 1997):

- l'insieme dei giocatori è N
- ogni coalizione $S \subseteq N$ paga il costo degli archi del circuito hamiltoniano di costo minimo che collega tutti i giocatori di S con la sorgente

Teorema 11.7 (Potters, Curiel e Tijs, 1992)
Il nucleo di un Traveling Salesman Game è non vuoto se ci sono al più tre giocatori

Esempio 11.13 (Nucleo vuoto) *I giocatori sono* $N = \{1, 2, 3, 4\}$*; i costi degli archi sono:*

	0	1	2	3	4
0	–	1	2	2	1
1	1	–	1	2	2
2	2	1	–	1	2
3	1	2	2	–	2
4	1	2	1	1	–

La collezione bilanciata $\{\{123\}, \{124\}, \{34\}\}$ *porta a:*

$$c(N) = C(0-1-2-3-4-0) = 6$$
$$c(123) = C(0-1-2-3-0) = 4$$
$$c(124) = C(0-4-2-1-0) = 4$$
$$c(34) = C(0-4-3-0) = 3$$

e quindi:

$$c(N) = 6 > \frac{11}{2} = \frac{1}{2}\,c(123) + \frac{1}{2}\,c(124) + \frac{1}{2}\,c(34)$$

per cui il nucleo è vuoto $\diamondsuit$

Teorema 11.8 (Kuipers, 1991 e Tamir, 1989)
Se il problema è simmetrico il nucleo di un Traveling Salesman Game è non vuoto se ci sono al più cinque giocatori