Ricerca Operativa

Massimo Pappalardo
Dipartimento di Informatica
Largo B. Pontecorvo 3, Pisa
massimo.pappalardo@unipi.it

Laurea in Ingegneria Informatica
Universitá di Pisa
A.A. 2022/'23

Riferimenti

Massimo Pappalardo

Dipartimento di Informatica

Largo B. Pontecorvo 3- Pisa

Edificio C - studio 289DE

tel. 050 2212750

e-mail: massimo.pappalardo@unipi.it

ricevimento: giovedí ore 16

Riferimenti

Massimo Pappalardo

Dipartimento di Informatica

Largo B. Pontecorvo 3- Pisa

Edificio C - studio 289DE

tel. 050 2212750

e-mail: massimo.pappalardo@unipi.it

ricevimento: giovedí ore 16

Orario del corso

- lunedí 8.30-12.30
- martedí 8.30-10.30
- mercoledí 8.30-10.30
- venerdí 8.30-10.30

Problema dello zaino (knapsack problem)

Problema

Dati: un contenitore di capacitá C, n oggetti di valore $v_1, \ldots, v_n$ e peso $p_1, \ldots, p_n$.

Problema dello zaino (knapsack problem)

Problema

Dati: un contenitore di capacitá C, n oggetti di valore $v_1, \ldots, v_n$ e peso $p_1, \ldots, p_n$. Quali oggetti inserisco nel contenitore, rispettando la sua capacitá, in modo da massimizzare il valore totale?

Problema dello zaino (knapsack problem)

Problema

Dati: un contenitore di capacitá C, n oggetti di valore $v_1, \ldots, v_n$ e peso $p_1, \ldots, p_n$. Quali oggetti inserisco nel contenitore, rispettando la sua capacitá, in modo da massimizzare il valore totale?

Esempio

Budget 100. Scegliere tra 9 investimenti possibili:

Problema dello zaino (knapsack problem)

Problema

Dati: un contenitore di capacitá C, n oggetti di valore $v_1, \dots, v_n$ e peso $p_1, \dots, p_n$. Quali oggetti inserisco nel contenitore, rispettando la sua capacitá, in modo da massimizzare il valore totale?

Esempio

Budget 100. Scegliere tra 9 investimenti possibili:

Investimento	1	2	3	4	5	6	7	8	9
Ricavo atteso	50	65	35	16	18	45	45	40	25
Costo	40	50	25	10	10	40	35	30	20

Modello dello zaino binario

Variabili: $x_j = \begin{cases} 1 & \text{se oggetto } j \text{ viene inserito,} \\ 0 & \text{altrimenti.} \end{cases}$

Modello dello zaino binario

Variabili: $x_j = \begin{cases} 1 & \text{se oggetto } j \text{ viene inserito,} \\ 0 & \text{altrimenti.} \end{cases}$

$$\max \sum_{j=1}^{n} v_j x_j$$

$$\sum_{j=1}^{n} p_j x_j \leq C$$

$$x_j \in \{0,1\} \quad \forall\, j = 1, \ldots, n$$

Metodi euristici "greedy"

Metodi *greedy* per trovare una soluzione ammissibile.

Metodi *greedy* per trovare una soluzione ammissibile.

Metodo 1

Esamino gli oggetti in ordine di valore decrescente.

Metodi euristici "greedy"

Metodi *greedy* per trovare una soluzione ammissibile.

Metodo 1

Esamino gli oggetti in ordine di valore decrescente.
Ogni oggetto viene inserito purché sia rispettato il vincolo di capacitá.

Metodi euristici "greedy"

Metodi *greedy* per trovare una soluzione ammissibile.

Metodo 1

Esamino gli oggetti in ordine di valore decrescente.
Ogni oggetto viene inserito purché sia rispettato il vincolo di capacitá.

Esempio

Oggetto	1	2	3	4	5	6	7	8	9	
Valore	50	65	35	16	18	55	45	40	25	
Peso	31	39	26	21	25	28	29	27	23	$C = 100$

Metodi euristici "greedy"

Metodi *greedy* per trovare una soluzione ammissibile.

Metodo 1

Esamino gli oggetti in ordine di valore decrescente.
Ogni oggetto viene inserito purché sia rispettato il vincolo di capacitá.

Esempio

Oggetto	1	2	3	4	5	6	7	8	9	
Valore	50	65	35	16	18	55	45	40	25	
Peso	31	39	26	21	25	28	29	27	23	$C = 100$

$x_2 = 1,\ x_6 = 1,\ x_1 = 1,\ x_7 = 0,\ x_8 = 0,\ x_3 = 0,\ x_9 = 0,\ x_5 = 0,\ x_4 = 0.$

Metodi *greedy* per trovare una soluzione ammissibile.

Metodo 1

Esamino gli oggetti in ordine di valore decrescente.
Ogni oggetto viene inserito purché sia rispettato il vincolo di capacitá.

Esempio

Oggetto	1	2	3	4	5	6	7	8	9	
Valore	50	65	35	16	18	55	45	40	25	
Peso	31	39	26	21	25	28	29	27	23	$C = 100$

$x_2 = 1$, $x_6 = 1$, $x_1 = 1$, $x_7 = 0$, $x_8 = 0$, $x_3 = 0$, $x_9 = 0$, $x_5 = 0$, $x_4 = 0$.
Quindi $v_I(P) = 170$.

Metodo 2

Esamino gli oggetti in ordine di peso crescente.

Metodo 2

Esamino gli oggetti in ordine di peso crescente.
Ogni oggetto viene inserito purché sia rispettato il vincolo di capacitá.

Metodo 2

Esamino gli oggetti in ordine di peso crescente.
Ogni oggetto viene inserito purché sia rispettato il vincolo di capacitá.

Esempio

Oggetto	1	2	3	4	5	6	7	8	9	
Valore	50	65	35	16	18	55	45	40	25	
Peso	31	39	26	21	25	28	29	27	23	$C = 100$

Metodi euristici "greedy"

Metodo 2

Esamino gli oggetti in ordine di peso crescente.
Ogni oggetto viene inserito purché sia rispettato il vincolo di capacitá.

Esempio

Oggetto	1	2	3	4	5	6	7	8	9	
Valore	50	65	35	16	18	55	45	40	25	
Peso	31	39	26	21	25	28	29	27	23	$C = 100$

$x_4 = 1$, $x_9 = 1$, $x_5 = 1$, $x_3 = 1$, $x_8 = 0$, $x_6 = 0$, $x_7 = 0$, $x_1 = 0$, $x_2 = 0$.

Metodo 2

Esamino gli oggetti in ordine di peso crescente.
Ogni oggetto viene inserito purché sia rispettato il vincolo di capacitá.

Esempio

Oggetto	1	2	3	4	5	6	7	8	9	
Valore	50	65	35	16	18	55	45	40	25	
Peso	31	39	26	21	25	28	29	27	23	$C = 100$

$x_4 = 1$, $x_9 = 1$, $x_5 = 1$, $x_3 = 1$, $x_8 = 0$, $x_6 = 0$, $x_7 = 0$, $x_1 = 0$, $x_2 = 0$.
Quindi $v_I(P) = 94$.

Metodo 3

Esamino gli oggetti in ordine di rendimento (valore/peso) decrescente.

Metodo 3

Esamino gli oggetti in ordine di rendimento (valore/peso) decrescente.
Ogni oggetto viene inserito purché sia rispettato il vincolo di capacitá.

Metodo 3

Esamino gli oggetti in ordine di rendimento (valore/peso) decrescente.
Ogni oggetto viene inserito purché sia rispettato il vincolo di capacitá.

Esempio

Oggetto	1	2	3	4	5	6	7	8	9	
Valore	50	65	35	16	18	55	45	40	25	
Peso	31	39	26	21	25	28	29	27	23	$C = 100$

Metodo 3

Esamino gli oggetti in ordine di rendimento (valore/peso) decrescente.
Ogni oggetto viene inserito purché sia rispettato il vincolo di capacitá.

Esempio

Oggetto	1	2	3	4	5	6	7	8	9	
Valore	50	65	35	16	18	55	45	40	25	
Peso	31	39	26	21	25	28	29	27	23	$C = 100$

$x_6 = 1,\ x_2 = 1,\ x_1 = 1,\ x_7 = 0,\ x_8 = 0,\ x_3 = 0,\ x_9 = 0,\ x_4 = 0,\ x_5 = 0.$

Metodo 3

Esamino gli oggetti in ordine di rendimento (valore/peso) decrescente.
Ogni oggetto viene inserito purché sia rispettato il vincolo di capacitá.

Esempio

Oggetto	1	2	3	4	5	6	7	8	9	
Valore	50	65	35	16	18	55	45	40	25	
Peso	31	39	26	21	25	28	29	27	23	$C = 100$

$x_6 = 1$, $x_2 = 1$, $x_1 = 1$, $x_7 = 0$, $x_8 = 0$, $x_3 = 0$, $x_9 = 0$, $x_4 = 0$, $x_5 = 0$.
Quindi $v_I(P) = 170$.

Teorema

Supponiamo che le variabili siano in ordine di rendimento decrescente.

Sia h l'indice tale che $\sum_{j=1}^{h} p_j \leq C$ e $\sum_{j=1}^{h+1} p_j > C$.

Il rilassamento continuo $\begin{cases} \max \sum_{j=1}^{n} v_j x_j \\ \sum_{j=1}^{n} p_j x_j \leq C \\ 0 \leq x_j \leq 1 \end{cases}$ ha come soluzione ottima

$$\bar{x}_1 = 1, \ \ldots, \ \bar{x}_h = 1, \ \bar{x}_{h+1} = \frac{C - \sum_{j=1}^{h} p_j}{p_{h+1}}, \ \bar{x}_{h+2} = 0, \ \ldots, \ \bar{x}_n = 0$$

Esempio

Sia dato il seguente problema:

$$\begin{cases} \max \; 10\,x_1 + 13\,x_2 + 18\,x_3 + 24\,x_4 \\ \qquad 2\,x_1 + 3\,x_2 + 4\,x_3 + 6\,x_4 \le 7 \\ x_j \in \{0, 1\} \end{cases}$$

Esempio

Sia dato il seguente problema:

$$\begin{cases} \max \ 10\,x_1 + 13\,x_2 + 18\,x_3 + 24\,x_4 \\ \qquad\quad 2\,x_1 + 3\,x_2 + 4\,x_3 + 6\,x_4 \leq 7 \\ x_j \in \{0,1\} \end{cases}$$

Disponiamo le variabili in ordine di rendimento decrescente:

variabili	1	3	2	4
rendimenti	5	4.5	4.33	4

Esempio

Sia dato il seguente problema:

$$\begin{cases} \max \ 10\,x_1 + 13\,x_2 + 18\,x_3 + 24\,x_4 \\ \qquad 2\,x_1 + 3\,x_2 + 4\,x_3 + 6\,x_4 \leq 7 \\ x_j \in \{0, 1\} \end{cases}$$

Disponiamo le variabili in ordine di rendimento decrescente:

variabili	1	3	2	4
rendimenti	5	4.5	4.33	4

Applicando il terzo algoritmo *greedy* otteniamo la soluzione ammissibile $(1, 0, 1, 0)$ e quindi $v_I(P) = 28$.

Esempio

Sia dato il seguente problema:

$$\begin{cases} \max\ 10\,x_1 + 13\,x_2 + 18\,x_3 + 24\,x_4 \\ \qquad 2\,x_1 + 3\,x_2 + 4\,x_3 + 6\,x_4 \le 7 \\ x_j \in \{0,1\} \end{cases}$$

Disponiamo le variabili in ordine di rendimento decrescente:

variabili	1	3	2	4
rendimenti	5	4.5	4.33	4

Applicando il terzo algoritmo *greedy* otteniamo la soluzione ammissibile $(1,0,1,0)$ e quindi $v_I(P) = 28$.

L'ottimo del rilassamento continuo é $\left(1, \frac{1}{3}, 1, 0\right)$, quindi $v_S(P) = 32$.

Problema

Dati: n oggetti, ognuno di valore v_j peso p_j, un contenitore di capacitá C.

Problema dello zaino a variabili intere

Problema

Dati: n oggetti, ognuno di valore v_j peso p_j, un contenitore di capacitá C. Quanti oggetti di ogni tipo inserisco nel contenitore per massimizzare il valore totale?

Problema

Dati: n oggetti, ognuno di valore v_j peso p_j, un contenitore di capacitá C. Quanti oggetti di ogni tipo inserisco nel contenitore per massimizzare il valore totale?

Modello

$x_j =$ numero (intero) di oggetti di tipo j inseriti nel contenitore

Problema dello zaino a variabili intere

Problema

Dati: n oggetti, ognuno di valore v_j peso p_j, un contenitore di capacitá C.
Quanti oggetti di ogni tipo inserisco nel contenitore per massimizzare il valore totale?

Modello

$x_j =$ numero (intero) di oggetti di tipo j inseriti nel contenitore

$$\begin{aligned} \max \sum_{j=1}^{n} v_j x_j \\ \sum_{j=1}^{n} p_j x_j \leq C \\ x_j \in \mathbb{N} \quad \forall\, j = 1, \dots, n \end{aligned}$$

Metodi *greedy* per trovare una soluzione ammissibile.

Metodo 1

Esamino gli oggetti in ordine di valore decrescente.

Metodi euristici ”greedy”

Metodi *greedy* per trovare una soluzione ammissibile.

Metodo 1

Esamino gli oggetti in ordine di valore decrescente.
Inserisco ogni oggetto nella massima quantitá possibile, purché sia rispettato il vincolo di capacitá.

Metodi euristici "greedy"

Metodi *greedy* per trovare una soluzione ammissibile.

Metodo 1

Esamino gli oggetti in ordine di valore decrescente.
Inserisco ogni oggetto nella massima quantitá possibile, purché sia rispettato il vincolo di capacitá.

Esempio

Oggetto	1	2	3	4	5	6	7	8	9	
Valore	50	65	35	16	18	55	45	40	25	
Peso	31	39	26	21	25	28	29	27	23	$C = 100$

Metodi euristici "greedy"

Metodi *greedy* per trovare una soluzione ammissibile.

Metodo 1

Esamino gli oggetti in ordine di valore decrescente.
Inserisco ogni oggetto nella massima quantitá possibile, purché sia rispettato il vincolo di capacitá.

Esempio

Oggetto	1	2	3	4	5	6	7	8	9	
Valore	50	65	35	16	18	55	45	40	25	
Peso	31	39	26	21	25	28	29	27	23	$C = 100$

$x_2 = 2,\ x_6 = 0,\ x_1 = 0,\ x_7 = 0,\ x_8 = 0,\ x_3 = 0,\ x_9 = 0,\ x_5 = 0,\ x_4 = 1.$

Metodi *greedy* per trovare una soluzione ammissibile.

Metodo 1

Esamino gli oggetti in ordine di valore decrescente.
Inserisco ogni oggetto nella massima quantitá possibile, purché sia rispettato il vincolo di capacitá.

Esempio

Oggetto	1	2	3	4	5	6	7	8	9	
Valore	50	65	35	16	18	55	45	40	25	
Peso	31	39	26	21	25	28	29	27	23	$C = 100$

$x_2 = 2$, $x_6 = 0$, $x_1 = 0$, $x_7 = 0$, $x_8 = 0$, $x_3 = 0$, $x_9 = 0$, $x_5 = 0$, $x_4 = 1$.
Quindi $v_I(P) = 146$.

Metodo 2

Esamino gli oggetti in ordine di peso crescente.

Metodo 2

Esamino gli oggetti in ordine di peso crescente.
Inserisco ogni oggetto nella massima quantitá possibile, purché sia rispettato il vincolo di capacitá.

Esempio

Oggetto	1	2	3	4	5	6	7	8	9	
Valore	50	65	35	16	18	55	45	40	25	
Peso	31	39	26	21	25	28	29	27	23	$C = 100$

$x_4 = 4$, $x_9 = 0$, $x_5 = 0$, $x_3 = 0$, $x_8 = 0$, $x_6 = 0$, $x_7 = 0$, $x_1 = 0$, $x_2 = 0$.

Metodo 2

Esamino gli oggetti in ordine di peso crescente.
Inserisco ogni oggetto nella massima quantitá possibile, purché sia rispettato il vincolo di capacitá.

Esempio

Oggetto	1	2	3	4	5	6	7	8	9	
Valore	50	65	35	16	18	55	45	40	25	
Peso	31	39	26	21	25	28	29	27	23	$C = 100$

$x_4 = 4$, $x_9 = 0$, $x_5 = 0$, $x_3 = 0$, $x_8 = 0$, $x_6 = 0$, $x_7 = 0$, $x_1 = 0$, $x_2 = 0$.
Quindi $v_I(P) = 64$.

Metodo 3

Esamino gli oggetti in ordine di rendimento decrescente.

Metodo 3

Esamino gli oggetti in ordine di rendimento decrescente.
Inserisco ogni oggetto nella massima quantitá, rispettando il vincolo di capacitá.

Metodo 3

Esamino gli oggetti in ordine di rendimento decrescente.
Inserisco ogni oggetto nella massima quantitá, rispettando il vincolo di capacitá.

Esempio

Oggetto	1	2	3	4	5	6	7	8	9	
Valore	50	65	35	16	18	55	45	40	25	
Peso	31	39	26	21	25	28	29	27	23	$C = 100$

Metodo 3

Esamino gli oggetti in ordine di rendimento decrescente.
Inserisco ogni oggetto nella massima quantitá, rispettando il vincolo di capacitá.

Esempio

Oggetto	1	2	3	4	5	6	7	8	9	
Valore	50	65	35	16	18	55	45	40	25	
Peso	31	39	26	21	25	28	29	27	23	$C = 100$

$x_6 = 3$, $x_2 = 0$, $x_1 = 0$, $x_7 = 0$, $x_8 = 0$, $x_3 = 0$, $x_9 = 0$, $x_4 = 0$, $x_5 = 0$.

Metodo 3

Esamino gli oggetti in ordine di rendimento decrescente.
Inserisco ogni oggetto nella massima quantitá, rispettando il vincolo di capacitá.

Esempio

Oggetto	1	2	3	4	5	6	7	8	9	
Valore	50	65	35	16	18	55	45	40	25	
Peso	31	39	26	21	25	28	29	27	23	$C = 100$

$x_6 = 3$, $x_2 = 0$, $x_1 = 0$, $x_7 = 0$, $x_8 = 0$, $x_3 = 0$, $x_9 = 0$, $x_4 = 0$, $x_5 = 0$.
Quindi $v_I(P) = 165$.

Calcoliamo una $v_S(P)$ risolvendo il rilassamento continuo.

Rilassamenti

Calcoliamo una $v_S(P)$ risolvendo il rilassamento continuo.

Teorema

Se $\max_j \{\frac{v_j}{p_j}\} = \frac{v_r}{p_r}$, allora il rilassamento continuo

$$\begin{cases} \max \ \sum_{j=1}^{n} v_j x_j \\ \sum_{j=1}^{n} p_j x_j \leq C \\ x \geq 0 \end{cases}$$

Rilassamenti

Calcoliamo una $v_S(P)$ risolvendo il rilassamento continuo.

Teorema

Se $\max_j \{\frac{v_j}{p_j}\} = \frac{v_r}{p_r}$, allora il rilassamento continuo

$$\begin{cases} \max \ \sum_{j=1}^{n} v_j x_j \\ \sum_{j=1}^{n} p_j x_j \leq C \\ x \geq 0 \end{cases}$$

ha come soluzione ottima

$$\bar{x}_1 = 0, \ \ldots, \ \bar{x}_{r-1} = 0, \ \bar{x}_r = \frac{C}{p_r}, \ \bar{x}_{r+1} = 0, \ \ldots, \ \bar{x}_n = 0$$

e valore ottimo $C\, v_r/p_r$.

Esempio

Consideriamo il problema:

$$\begin{cases} \max \; 4\,x_1 + 20\,x_2 + 27\,x_3 + 26\,x_4 \\ \qquad\; 4\,x_1 + 19\,x_2 + 16\,x_3 + 14\,x_4 \leq 32 \\ x_j \in \mathbb{N} \end{cases} \tag{P}$$

Esempio

Consideriamo il problema:

$$\begin{cases} \max \; 4\,x_1 + 20\,x_2 + 27\,x_3 + 26\,x_4 \\ \qquad\; 4\,x_1 + 19\,x_2 + 16\,x_3 + 14\,x_4 \leq 32 \\ x_j \in \mathbb{N} \end{cases} \tag{P}$$

Disponiamo le variabili in ordine di rendimento decrescente:

variabili	4	3	2	1
rendimenti	1.85	1.68	1.05	1

Esempio

Consideriamo il problema:

$$\begin{cases} \max\ 4\,x_1 + 20\,x_2 + 27\,x_3 + 26\,x_4 \\ \qquad 4\,x_1 + 19\,x_2 + 16\,x_3 + 14\,x_4 \leq 32 \\ x_j \in \mathbb{N} \end{cases} \tag{P}$$

Disponiamo le variabili in ordine di rendimento decrescente:

variabili	4	3	2	1
rendimenti	1.85	1.68	1.05	1

Il terzo algoritmo *greedy* trova la soluzione $(1, 0, 0, 2)$ con $v_I(P) = 56$.

Esempio

Consideriamo il problema:

$$\begin{cases} \max\ 4\,x_1 + 20\,x_2 + 27\,x_3 + 26\,x_4 \\ \qquad 4\,x_1 + 19\,x_2 + 16\,x_3 + 14\,x_4 \le 32 \\ x_j \in \mathbb{N} \end{cases} \tag{P}$$

Disponiamo le variabili in ordine di rendimento decrescente:

variabili	4	3	2	1
rendimenti	1.85	1.68	1.05	1

Il terzo algoritmo *greedy* trova la soluzione $(1, 0, 0, 2)$ con $v_I(P) = 56$.

La soluzione ottima del rilassamento continuo é $(0, 0, 0, \frac{32}{14})$, quindi $v_S(P) = 59$.

Relazioni tra PLI e PL

Consideriamo un problema di programmazione lineare intera (PLI) in formato standard:

$$\begin{cases} \max\ c^T x \\ A x \leq b \\ x \in \mathbb{Z}^n \end{cases} \qquad (\mathcal{P})$$

Consideriamo un problema di programmazione lineare intera (PLI) in formato standard:

$$\begin{cases} \max \ c^T x \\ A x \le b \\ x \in \mathbb{Z}^n \end{cases} \qquad (\mathcal{P})$$

Definizione

Il problema di PL

$$\begin{cases} \max \ c^T x \\ A x \le b \end{cases} \qquad (RC)$$

é detto rilassamento continuo del problema $(\mathcal{P})$.

Relazioni tra PLI e PL

Quale é la relazione tra $(\mathcal{P})$ e (RC)?

Relazioni tra PLI e PL

Quale é la relazione tra $(\mathcal{P})$ e (RC)?

Teorema

- Il valore ottimo di (RC) é una valutazione superiore per il valore ottimo di $(\mathcal{P})$.

Quale é la relazione tra $(\mathcal{P})$ e (RC)?

Teorema

- Il valore ottimo di (RC) é una valutazione superiore per il valore ottimo di $(\mathcal{P})$.
- Se una soluzione ottima di (RC) é ammissibile per $(\mathcal{P})$, allora é ottima anche per $(\mathcal{P})$.

Quale é la relazione tra $(\mathcal{P})$ e (RC)?

Teorema

- Il valore ottimo di (RC) é una valutazione superiore per il valore ottimo di $(\mathcal{P})$.
- Se una soluzione ottima di (RC) é ammissibile per $(\mathcal{P})$, allora é ottima anche per $(\mathcal{P})$.

Usualmente la soluzione ottima di (RC) é inammissibile per $(\mathcal{P})$.

Relazioni tra PLI e PL

Per risolvere $(\mathcal{P})$, é sufficiente risolvere (RC) ed arrotondare la soluzione?

Relazioni tra PLI e PL

Per risolvere $(\mathcal{P})$, é sufficiente risolvere (RC) ed arrotondare la soluzione? NO

Esempio

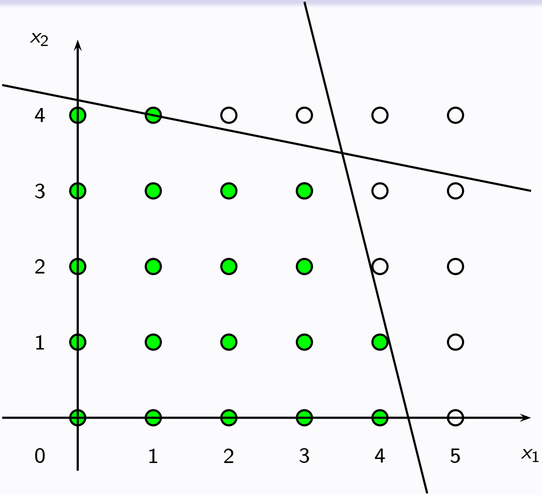
$$
\begin{cases}
\max \ x_1 + 3\,x_2 \\
x_1 + 5\,x_2 \leq 21 \\
8\,x_1 + 2\,x_2 \leq 35 \\
x \geq 0 \\
x \in \mathbb{Z}^2
\end{cases}
$$

Relazioni tra PLI e PL

Per risolvere ($\mathcal{P}$), é sufficiente risolvere (RC) ed arrotondare la soluzione? NO

Esempio

$$\begin{cases} \max\ x_1 + 3\,x_2 \\ x_1 + 5\,x_2 \leq 21 \\ 8\,x_1 + 2\,x_2 \leq 35 \\ x \geq 0 \\ x \in \mathbb{Z}^2 \end{cases}$$

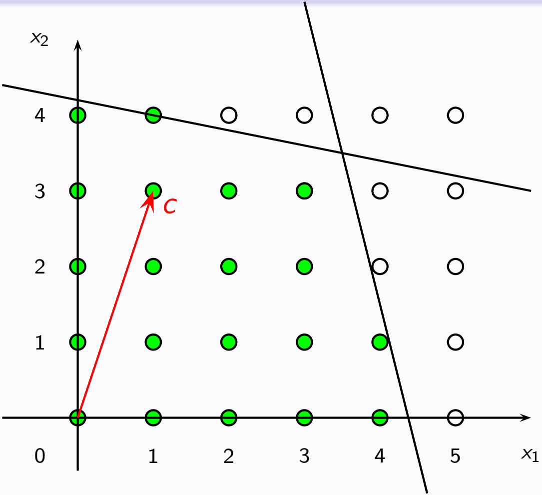


Relazioni tra PLI e PL

Per risolvere ($\mathcal{P}$), é sufficiente risolvere (RC) ed arrotondare la soluzione? NO

Esempio

$$\begin{cases} \max \; x_1 + 3\,x_2 \\ x_1 + 5\,x_2 \leq 21 \\ 8\,x_1 + 2\,x_2 \leq 35 \\ x \geq 0 \\ x \in \mathbb{Z}^2 \end{cases}$$

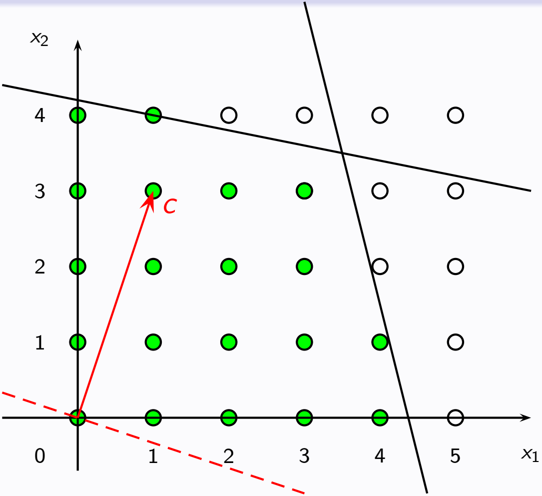


Relazioni tra PLI e PL

Per risolvere ($\mathcal{P}$), é sufficiente risolvere (RC) ed arrotondare la soluzione? NO

Esempio

$$\begin{cases} \max \ x_1 + 3\,x_2 \\ x_1 + 5\,x_2 \leq 21 \\ 8\,x_1 + 2\,x_2 \leq 35 \\ x \geq 0 \\ x \in \mathbb{Z}^2 \end{cases}$$

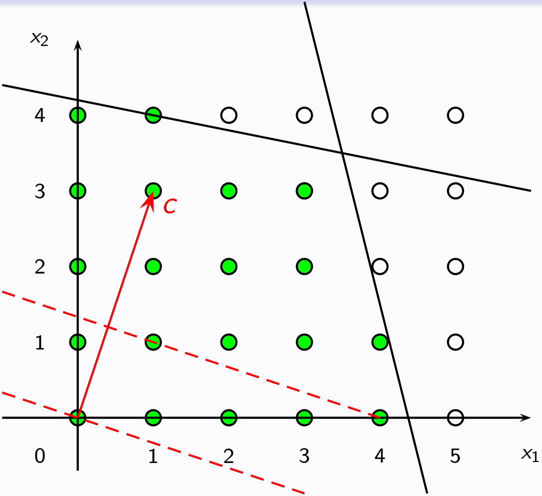


Relazioni tra PLI e PL

Per risolvere ($\mathcal{P}$), é sufficiente risolvere (RC) ed arrotondare la soluzione? NO

Esempio

$$\begin{cases} \max \; x_1 + 3\,x_2 \\ x_1 + 5\,x_2 \le 21 \\ 8\,x_1 + 2\,x_2 \le 35 \\ x \ge 0 \\ x \in \mathbb{Z}^2 \end{cases}$$

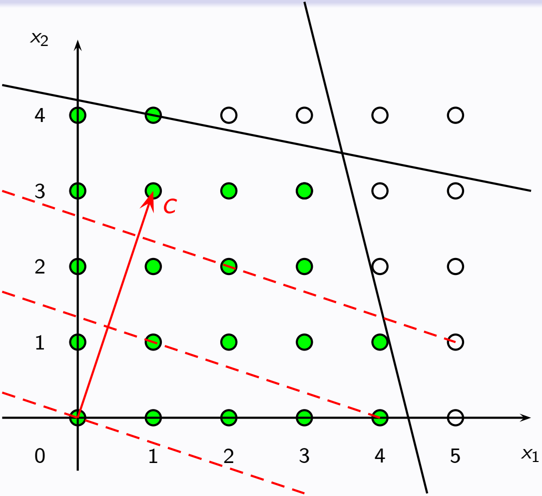


Relazioni tra PLI e PL

Per risolvere $(\mathcal{P})$, é sufficiente risolvere (RC) ed arrotondare la soluzione? NO

Esempio

$$\begin{cases} \max\ x_1 + 3\,x_2 \\ x_1 + 5\,x_2 \leq 21 \\ 8\,x_1 + 2\,x_2 \leq 35 \\ x \geq 0 \\ x \in \mathbb{Z}^2 \end{cases}$$

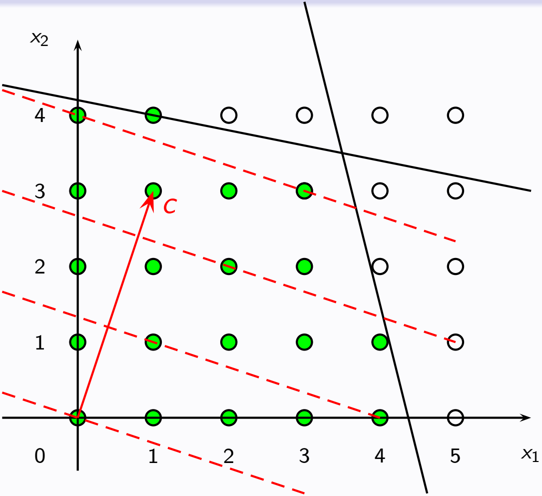


Relazioni tra PLI e PL

Per risolvere $(\mathcal{P})$, é sufficiente risolvere (RC) ed arrotondare la soluzione? NO

Esempio

$$\begin{cases} \max\ x_1 + 3\,x_2 \\ x_1 + 5\,x_2 \leq 21 \\ 8\,x_1 + 2\,x_2 \leq 35 \\ x \geq 0 \\ x \in \mathbb{Z}^2 \end{cases}$$



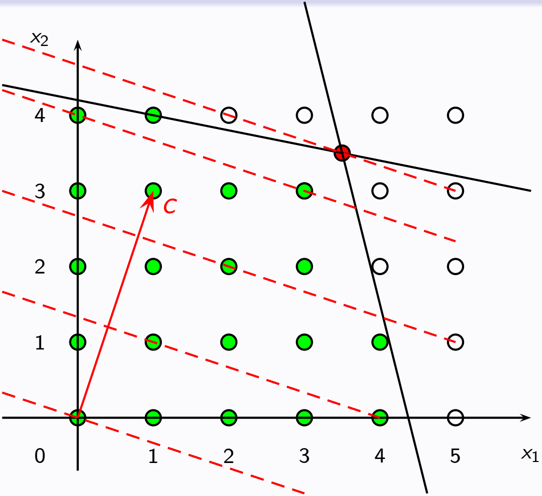
Relazioni tra PLI e PL

Per risolvere ($\mathcal{P}$), é sufficiente risolvere (RC) ed arrotondare la soluzione? NO

Esempio

$$\begin{cases} \max \ x_1 + 3\,x_2 \\ x_1 + 5\,x_2 \leq 21 \\ 8\,x_1 + 2\,x_2 \leq 35 \\ x \geq 0 \\ x \in \mathbb{Z}^2 \end{cases}$$

$\left(\frac{7}{2}, \frac{7}{2}\right)$ ottima per (RC)



Relazioni tra PLI e PL

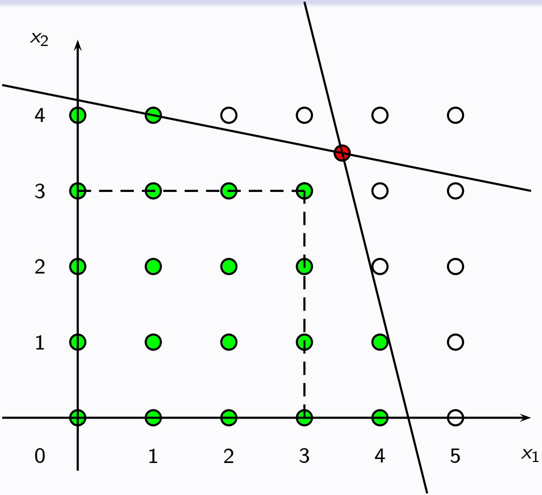
Per risolvere ($\mathcal{P}$), é sufficiente risolvere (RC) ed arrotondare la soluzione? NO

Esempio

$$\begin{cases} \max \ x_1 + 3\,x_2 \\ x_1 + 5\,x_2 \leq 21 \\ 8\,x_1 + 2\,x_2 \leq 35 \\ x \geq 0 \\ x \in \mathbb{Z}^2 \end{cases}$$

$\left(\frac{7}{2}, \frac{7}{2}\right)$ ottima per (RC)

arrotondamento $\to (3, 3)$



Relazioni tra PLI e PL

Per risolvere ($\mathcal{P}$), é sufficiente risolvere (RC) ed arrotondare la soluzione? NO

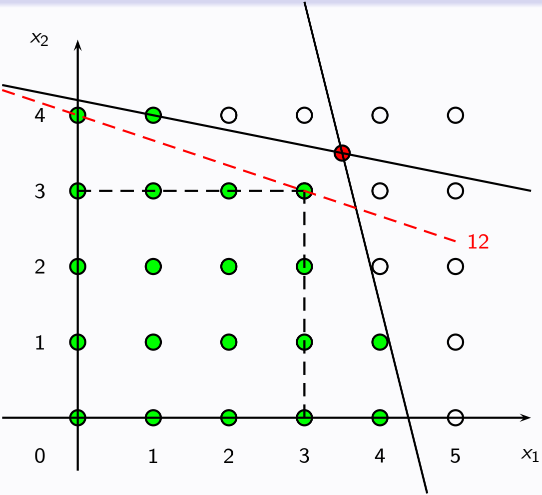
Esempio

$$\begin{cases} \max \ x_1 + 3\,x_2 \\ x_1 + 5\,x_2 \leq 21 \\ 8\,x_1 + 2\,x_2 \leq 35 \\ x \geq 0 \\ x \in \mathbb{Z}^2 \end{cases}$$

$\left(\frac{7}{2}, \frac{7}{2}\right)$ ottima per (RC)

arrotondamento $\to (3, 3)$

$(3, 3)$ non é ottima per ($\mathcal{P}$)



Per risolvere $(\mathcal{P})$, é sufficiente risolvere (RC) ed arrotondare la soluzione? NO

Esempio

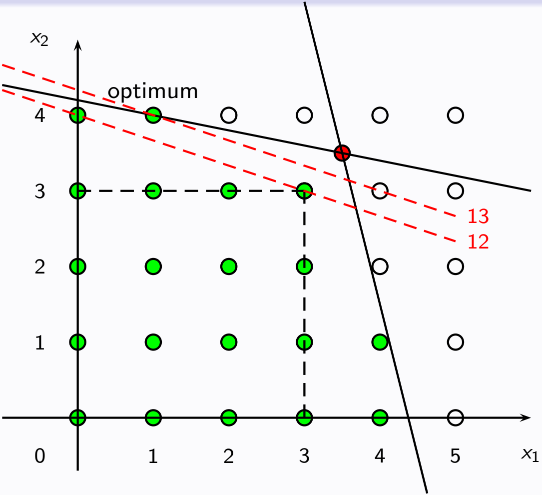
$$\begin{cases} \max\ x_1 + 3\,x_2 \\ x_1 + 5\,x_2 \leq 21 \\ 8\,x_1 + 2\,x_2 \leq 35 \\ x \geq 0 \\ x \in \mathbb{Z}^2 \end{cases}$$

$\left(\frac{7}{2}, \frac{7}{2}\right)$ ottima per (RC)

arrotondamento $\rightarrow (3,3)$

$(3,3)$ non é ottima per $(\mathcal{P})$

$(1,4)$ é ottima per $(\mathcal{P})$



Relazioni tra PLI e PL

E' sufficiente risolvere (RC) e trovare la soluzione intera ammissibile piú vicina?

E' sufficiente risolvere (RC) e trovare la soluzione intera ammissibile piú vicina?
NO

Esempio

$$\begin{cases} \max\ x_1 + 3\,x_2 \\ x_1 + 5\,x_2 \leq 21 \\ 8\,x_1 + 2\,x_2 \leq 35 \\ x \geq 0 \\ x \in \mathbb{Z}^2 \end{cases}$$

$\left(\frac{7}{2}, \frac{7}{2}\right)$ ottima per (RC).

Relazioni tra PLI e PL

E' sufficiente risolvere (RC) e trovare la soluzione intera ammissibile piú vicina?
NO

Esempio

$$\begin{cases} \max \ x_1 + 3\,x_2 \\ x_1 + 5\,x_2 \leq 21 \\ 8\,x_1 + 2\,x_2 \leq 35 \\ x \geq 0 \\ x \in \mathbb{Z}^2 \end{cases}$$

$\left(\frac{7}{2}, \frac{7}{2}\right)$ ottima per (RC).

la soluzione intera ammissibile piú vicina é $(3, 3)$.

Relazioni tra PLI e PL

E' sufficiente risolvere (RC) e trovare la soluzione intera ammissibile piú vicina?
NO

Esempio

$$\begin{cases} \max \ x_1 + 3\,x_2 \\ x_1 + 5\,x_2 \leq 21 \\ 8\,x_1 + 2\,x_2 \leq 35 \\ x \geq 0 \\ x \in \mathbb{Z}^2 \end{cases}$$

$\left(\frac{7}{2}, \frac{7}{2}\right)$ ottima per (RC).

la soluzione intera ammissibile piú vicina é $(3, 3)$.

$(3, 3)$ non é ottima per $(\mathcal{P})$.

Relazioni tra PLI e PL

E' sufficiente risolvere (RC) e trovare la soluzione intera ammissibile piú vicina? NO

Esempio

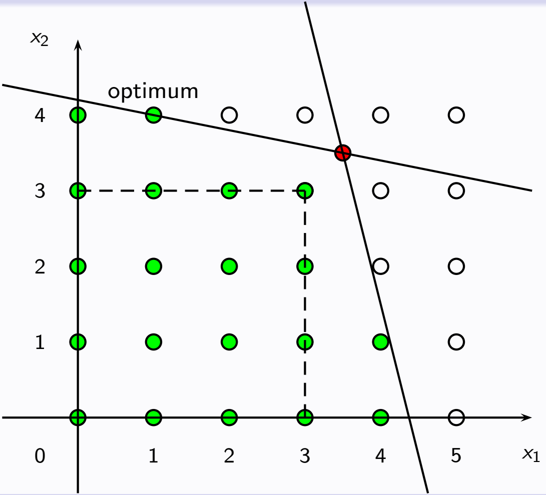
$$\begin{cases} \max \ x_1 + 3\,x_2 \\ x_1 + 5\,x_2 \leq 21 \\ 8\,x_1 + 2\,x_2 \leq 35 \\ x \geq 0 \\ x \in \mathbb{Z}^2 \end{cases}$$

$\left(\frac{7}{2}, \frac{7}{2}\right)$ ottima per (RC).

la soluzione intera ammissibile piú vicina é $(3, 3)$.

$(3, 3)$ non é ottima per $(\mathcal{P})$.

$(1, 4)$ é ottima per $(\mathcal{P})$



Metodi: enumerazione esplicita

Consideriamo il problema di PLI:

$$\begin{cases} \max \; 5\,x_1 + 6\,x_2 \\ 3\,x_1 + 4\,x_2 \leq 7 \\ x_1 \geq 0, \;\; x_2 \geq 0 \\ x \in \mathbb{Z}^2 \end{cases} \tag{$\mathcal{P}$}$$

Metodi: enumerazione esplicita

Consideriamo il problema di PLI:

$$\begin{cases} \max\ 5\,x_1 + 6\,x_2 \\ 3\,x_1 + 4\,x_2 \le 7 \\ x_1 \ge 0,\ \ x_2 \ge 0 \\ x \in \mathbb{Z}^2 \end{cases} \qquad (\mathcal{P})$$

I vincoli implicano $x_1 = 0$ o 1 o 2.

Metodi: enumerazione esplicita

Consideriamo il problema di PLI:

$$\begin{cases} \max\ 5\,x_1 + 6\,x_2 \\ 3\,x_1 + 4\,x_2 \leq 7 \\ x_1 \geq 0,\ \ x_2 \geq 0 \\ x \in \mathbb{Z}^2 \end{cases} \qquad (\mathcal{P})$$

I vincoli implicano $x_1 = 0$ o 1 o 2.
Scriviamo una partizione della regione ammissibile Ω in tre sottoinsiemi:

$$\Omega = (\Omega \cap \{x_1 = 0\}) \cup (\Omega \cap \{x_1 = 1\}) \cup (\Omega \cap \{x_1 = 2\})$$

corrispondente al primo livello dell'albero decisionale:

Metodi: enumerazione esplicita

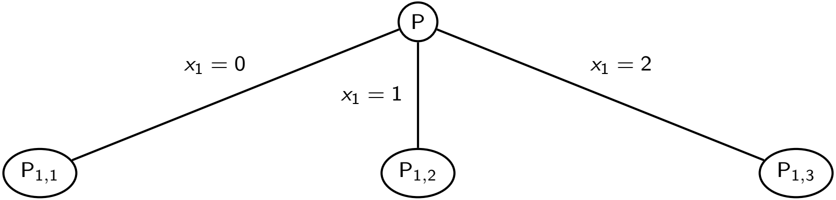
Consideriamo il problema di PLI:

$$\begin{cases} \max\ 5\,x_1 + 6\,x_2 \\ 3\,x_1 + 4\,x_2 \leq 7 \\ x_1 \geq 0,\ \ x_2 \geq 0 \\ x \in \mathbb{Z}^2 \end{cases} \qquad (\mathcal{P})$$

I vincoli implicano $x_1 = 0$ o 1 o 2.
Scriviamo una partizione della regione ammissibile Ω in tre sottoinsiemi:

$$\Omega = (\Omega \cap \{x_1 = 0\}) \cup (\Omega \cap \{x_1 = 1\}) \cup (\Omega \cap \{x_1 = 2\})$$

corrispondente al primo livello dell'albero decisionale:

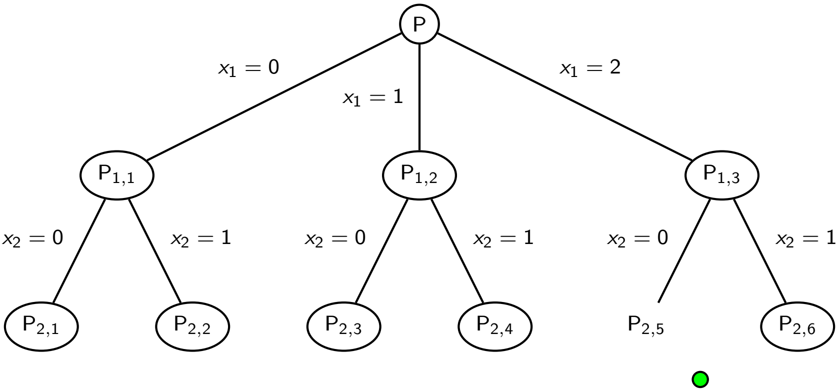


Enumerazione esplicita

Similarmente, $x_2 = 0$ or 1. Quindi, l'albero delle decisioni completo é:

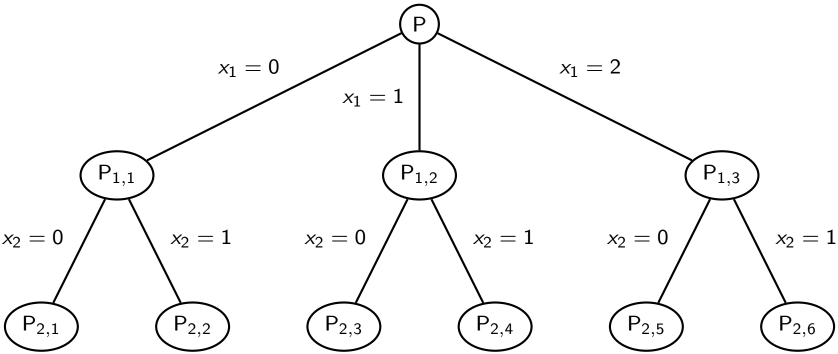
Enumerazione esplicita

Similarmente, $x_2 = 0$ or 1. Quindi, l'albero delle decisioni completo é:



Enumerazione esplicita

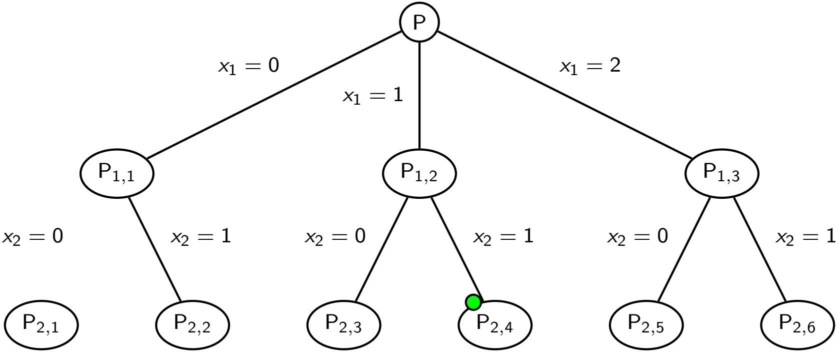
Similarmente, $x_2 = 0$ or 1. Quindi, l'albero delle decisioni completo é:



I nodi $P_{2,1}, \ldots, P_{2,5}$ corrispondono a soluzioni ammissibili di $(\mathcal{P})$, mentre il nodo $P_{2,6}$ corrisponde a $x = (2, 1)$ che é inammissibile.

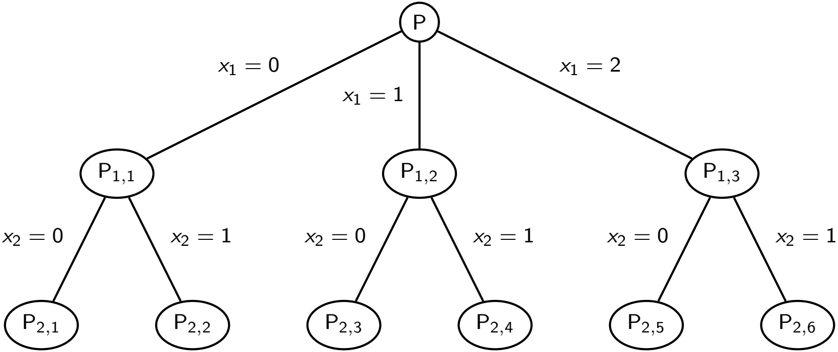
Enumerazione esplicita

Similarmente, $x_2 = 0$ or 1. Quindi, l'albero delle decisioni completo é:



I nodi $P_{2,1}, \ldots, P_{2,5}$ corrispondono a soluzioni ammissibili di $(\mathcal{P})$, mentre il nodo $P_{2,6}$ corrisponde a $x = (2, 1)$ che é inammissibile. I valori della funzione obiettivo per i nodi $P_{2,1}, \ldots, P_{2,5}$ sono 0, 6, 5, 11, 10.

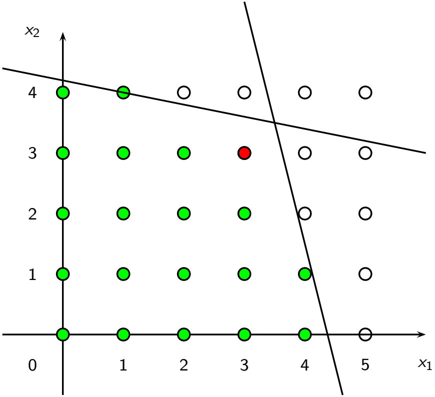
Similarmente, $x_2 = 0$ or 1. Quindi, l'albero delle decisioni completo é:



I nodi $P_{2,1}, \ldots, P_{2,5}$ corrispondono a soluzioni ammissibili di $(\mathcal{P})$, mentre il nodo $P_{2,6}$ corrisponde a $x = (2, 1)$ che é inammissibile. I valori della funzione obiettivo per i nodi $P_{2,1}, \ldots, P_{2,5}$ sono 0, 6, 5, 11, 10. Allora, la soluzione ottima é data da $P_{2,4}$, i.e., $x^* = (1, 1)$.

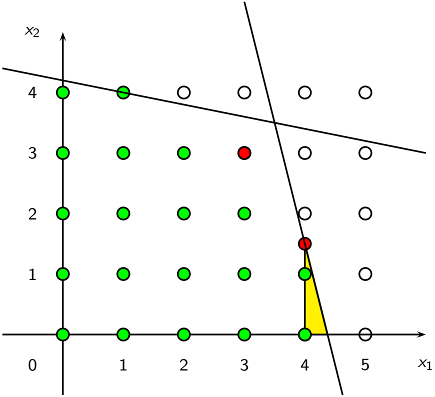
Enumerazione implicita

$$\begin{cases} \max \ x_1 + 3\,x_2 \\ x_1 + 5\,x_2 \leq 21 \\ 8\,x_1 + 2\,x_2 \leq 35 \\ x \geq 0, \ x \in \mathbb{Z}^2 \end{cases}$$



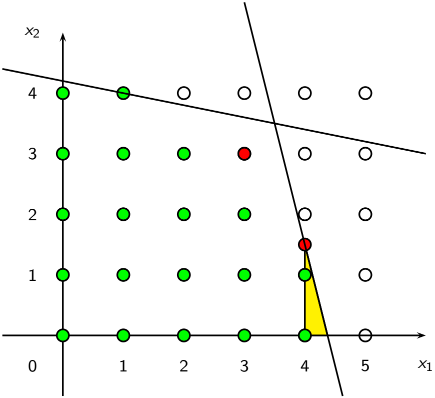
- Sappiamo che $(3,3)$ é ammissibile con valore 12.

$$\begin{cases} \max \ x_1 + 3\,x_2 \\ x_1 + 5\,x_2 \leq 21 \\ 8\,x_1 + 2\,x_2 \leq 35 \\ x \geq 0, \ x \in \mathbb{Z}^2 \end{cases}$$



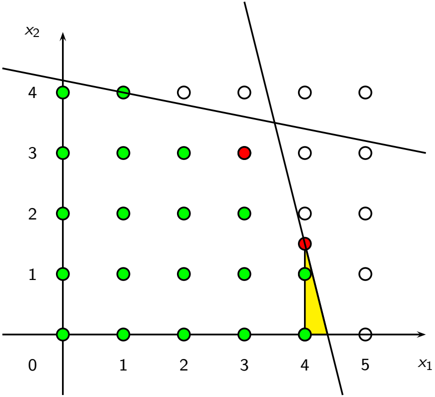
- Sappiamo che $(3,3)$ é ammissibile con valore 12.
- Consideriamo il vincolo aggiuntivo $x_1 \geq 4$:

$$\begin{cases} \max \; x_1 + 3\,x_2 \\ x_1 + 5\,x_2 \le 21 \\ 8\,x_1 + 2\,x_2 \le 35 \\ x \ge 0, \;\; x \in \mathbb{Z}^2 \end{cases}$$



- Sappiamo che $(3,3)$ é ammissibile con valore 12.
- Consideriamo il vincolo aggiuntivo $x_1 \ge 4$: l'ottimo del rilassamento continuo del sottoproblema é $(4, 3/2)$ con valore 8.5

$$\begin{cases} \max\ x_1 + 3\,x_2 \\ x_1 + 5\,x_2 \leq 21 \\ 8\,x_1 + 2\,x_2 \leq 35 \\ x \geq 0,\ x \in \mathbb{Z}^2 \end{cases}$$



- Sappiamo che $(3,3)$ é ammissibile con valore 12.
- Consideriamo il vincolo aggiuntivo $x_1 \geq 4$: l'ottimo del rilassamento continuo del sottoproblema é $(4, 3/2)$ con valore 8.5 quindi le soluzioni ammissibili del sottoproblema sono peggiori di $(3,3)$ $\rightarrow$ enumerazione implicita.

"Branch and Bound"

L'idea di base del *Branch and Bound* é:

"Branch and Bound"

L'idea di base del *Branch and Bound* é:

- La regione ammissibile é partizionata, generando un albero di ricerca.

L'idea di base del *Branch and Bound* é:

- La regione ammissibile é partizionata, generando un albero di ricerca.
- Per ogni sottoregione (corrispondente ad un sottoproblema) il valore ottimo é approssimato tramite valutazioni.

L'idea di base del *Branch and Bound* é:

- La regione ammissibile é partizionata, generando un albero di ricerca.
- Per ogni sottoregione (corrispondente ad un sottoproblema) il valore ottimo é approssimato tramite valutazioni.
- Le regioni che non possono contenere l'ottimo sono scartate.

Principali componenti

Principali componenti

- *Branching*: come partizionare la regione ammissibile per generare sottoproblemi.

Principali componenti

- *Branching*: come partizionare la regione ammissibile per generare sottoproblemi.
- *Bounding*: come stimare il valore dell'ottimo in ogni sottoregione.

Principali componenti

- *Branching*: come partizionare la regione ammissibile per generare sottoproblemi.
- *Bounding*: come stimare il valore dell'ottimo in ogni sottoregione.
- *Fathoming*: come depennare le sottoregioni che non contengono l'ottimo (come chiudere i nodi dell'albero di ricerca).

Principali componenti

- *Branching*: come partizionare la regione ammissibile per generare sottoproblemi.
- *Bounding*: come stimare il valore dell'ottimo in ogni sottoregione.
- *Fathoming*: come depennare le sottoregioni che non contengono l'ottimo (come chiudere i nodi dell'albero di ricerca).

Come partizionare la regione ammissibile

- Le sottoregioni sono generate aggiungendo vincoli.

"Branch and Bound"

Principali componenti

- *Branching*: come partizionare la regione ammissibile per generare sottoproblemi.
- *Bounding*: come stimare il valore dell'ottimo in ogni sottoregione.
- *Fathoming*: come depennare le sottoregioni che non contengono l'ottimo (come chiudere i nodi dell'albero di ricerca).

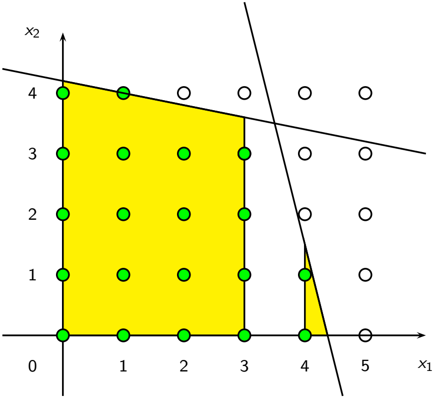
Come partizionare la regione ammissibile

- Le sottoregioni sono generate aggiungendo vincoli.
- Le sottoregioni devono essere una partizione della regione ammissibile, in modo tale da garantire che nessuna soluzione intera sia scartata.

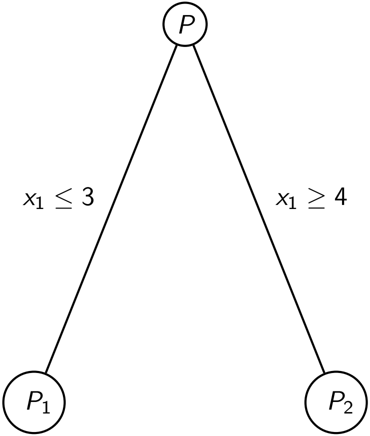
Regole di "Branching"

"Branching" bipartito

Ogni regione é partizionata in due sottoregioni, aggiungendo i vincoli $x_1 \leq 3$ e $x_1 \geq 4$



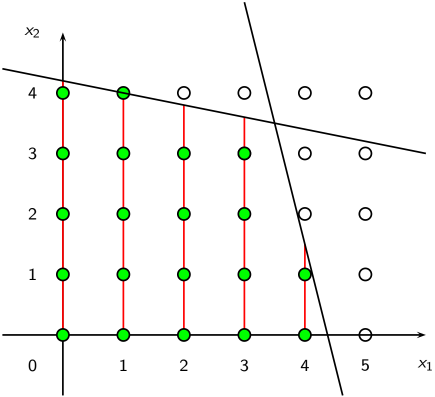
Albero di "branching" corrispondente



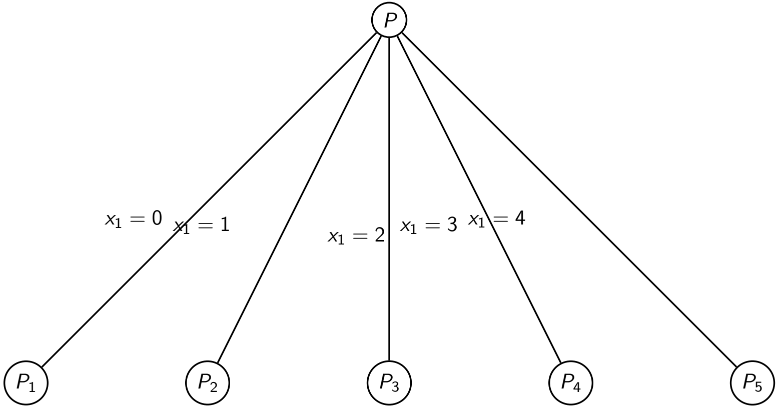
Regole di "Branching"

k-partito "branching"

Ogni regione é divisa in k sottoregioni, aggiungendo k vincoli E.g. $x_1 = 0$, $x_1 = 1$, $x_1 = 2$, $x_1 = 3$, e $x_1 = 4$



Albero di "branching" corrispondente



Valutazioni per problemi di massimo

Valutazioni inferiori LB date da soluzioni ammissibili.

Valutazioni per problemi di massimo

Valutazioni inferiori LB date da soluzioni ammissibili.

Valutazioni superiori UB date da un *rilassamento*:

Valutazioni per problemi di massimo

Valutazioni inferiori LB date da soluzioni ammissibili.

Valutazioni superiori UB date da un *rilassamento*:

- Rilassamento continuo (eliminazione del vincolo di interezza)

Valutazioni per problemi di massimo

Valutazioni inferiori LB date da soluzioni ammissibili.

Valutazioni superiori UB date da un *rilassamento*:

- Rilassamento continuo (eliminazione del vincolo di interezza)
  $x \in \{0,1\} \Rightarrow 0 \leq x \leq 1$

Valutazioni per problemi di massimo

Valutazioni inferiori LB date da soluzioni ammissibili.

Valutazioni superiori UB date da un *rilassamento*:

- Rilassamento continuo (eliminazione del vincolo di interezza)
  $x \in \{0, 1\} \Rightarrow 0 \leq x \leq 1$
  $x \in \mathbb{Z}_+ \Rightarrow x \geq 0$

Valutazioni per problemi di massimo

Valutazioni inferiori LB date da soluzioni ammissibili.

Valutazioni superiori UB date da un *rilassamento*:

- Rilassamento continuo (eliminazione del vincolo di interezza)
  $x \in \{0, 1\} \Rightarrow 0 \leq x \leq 1$
  $x \in \mathbb{Z}_+ \Rightarrow x \geq 0$
- Eliminazione di uno o piú vincoli.

Criteri di ”fathoming” o ”pruning”

Un nodo dell’albero di decisione puó essere chiuso se una delle seguenti condizioni sussite:

- il sottoproblema é inammissibile.

Criteri di ”fathoming” o ”pruning”

Un nodo dell’albero di decisione puó essere chiuso se una delle seguenti condizioni sussite:

- il sottoproblema é inammissibile.
- UB del sottoproblema é $\leq LB$ di $(\mathcal{P})$.

Criteri di "fathoming" o "pruning"

Un nodo dell'albero di decisione puó essere chiuso se una delle seguenti condizioni sussite:

- il sottoproblema é inammissibile.
- UB del sottoproblema é $\leq LB$ di $(\mathcal{P})$.
- UB del sottoproblema é $> LB$ e la soluzione ottima del sottoproblema é ammissibile per $(\mathcal{P})$. In tal caso noi aggiorniamo $LB = UB$.

Schema del metodo

1. Calcola una valutazione inferiore LB del valore ottimo di $(\mathcal{P})$

Schema del metodo

1. Calcola una valutazione inferiore LB del valore ottimo di $(\mathcal{P})$
2. Se tutti i nodi sono stati esplorati allora STOP

Schema del metodo

1. Calcola una valutazione inferiore LB del valore ottimo di $(\mathcal{P})$
2. Se tutti i nodi sono stati esplorati allora STOP
3. Seleziona il nodo (sottoproblema) da esplorare

Schema del metodo

1. Calcola una valutazione inferiore LB del valore ottimo di $(\mathcal{P})$
2. Se tutti i nodi sono stati esplorati allora STOP
3. Seleziona il nodo (sottoproblema) da esplorare
4. Calcola una valutazione superiore UB del valore ottimo del sottoproblema

"Branch and Bound"

Schema del metodo

1. Calcola una valutazione inferiore LB del valore ottimo di $(\mathcal{P})$
2. Se tutti i nodi sono stati esplorati allora STOP
3. Seleziona il nodo (sottoproblema) da esplorare
4. Calcola una valutazione superiore UB del valore ottimo del sottoproblema
5. Controlla i criteri:

Schema del metodo

1. Calcola una valutazione inferiore LB del valore ottimo di ($\mathcal{P}$)
2. Se tutti i nodi sono stati esplorati allora STOP
3. Seleziona il nodo (sottoproblema) da esplorare
4. Calcola una valutazione superiore UB del valore ottimo del sottoproblema
5. Controlla i criteri:
   se il sottoproblema é inammissibile allora chiudi il nodo

Schema del metodo

1. Calcola una valutazione inferiore LB del valore ottimo di $(\mathcal{P})$
2. Se tutti i nodi sono stati esplorati allora STOP
3. Seleziona il nodo (sottoproblema) da esplorare
4. Calcola una valutazione superiore UB del valore ottimo del sottoproblema
5. Controlla i criteri:
   se il sottoproblema é inammissibile allora chiudi il nodo
   se $UB \leq LB$, allora chiudi il nodo

Schema del metodo

1. Calcola una valutazione inferiore LB del valore ottimo di $(\mathcal{P})$
2. Se tutti i nodi sono stati esplorati allora STOP
3. Seleziona il nodo (sottoproblema) da esplorare
4. Calcola una valutazione superiore UB del valore ottimo del sottoproblema
5. Controlla i criteri:
   se il sottoproblema é inammissibile allora chiudi il nodo
   se $UB \leq LB$, allora chiudi il nodo
   se $UB > LB$ e la soluzione ottima del sottoproblema é ammissibile per $(\mathcal{P})$, allora chiudi il nodo ed aggiorna $LB = UB$

Schema del metodo

1. Calcola una valutazione inferiore LB del valore ottimo di $(\mathcal{P})$
2. Se tutti i nodi sono stati esplorati allora STOP
3. Seleziona il nodo (sottoproblema) da esplorare
4. Calcola una valutazione superiore UB del valore ottimo del sottoproblema
5. Controlla i criteri:
   se il sottoproblema é inammissibile allora chiudi il nodo
   se $UB \leq LB$, allora chiudi il nodo
   se $UB > LB$ e la soluzione ottima del sottoproblema é ammissibile per $(\mathcal{P})$, allora chiudi il nodo ed aggiorna $LB = UB$
6. se il nodo non é chiuso allora scendi

Schema del metodo

1. Calcola una valutazione inferiore LB del valore ottimo di $(\mathcal{P})$
2. Se tutti i nodi sono stati esplorati allora STOP
3. Seleziona il nodo (sottoproblema) da esplorare
4. Calcola una valutazione superiore UB del valore ottimo del sottoproblema
5. Controlla i criteri:
   se il sottoproblema é inammissibile allora chiudi il nodo
   se $UB \leq LB$, allora chiudi il nodo
   se $UB > LB$ e la soluzione ottima del sottoproblema é ammissibile per $(\mathcal{P})$, allora chiudi il nodo ed aggiorna $LB = UB$
6. se il nodo non é chiuso allora scendi
7. vai al passo 2

Esempio

Applica il "Branch and Bound" per risolvere il problema

$$\begin{cases} \max\ x_1 + 3\,x_2 \\ x_1 + 5\,x_2 \leq 21 \\ 8\,x_1 + 2\,x_2 \leq 35 \\ x \geq 0,\ x \in \mathbb{Z}^2. \end{cases} \qquad (P)$$

Esempio

Applica il "Branch and Bound" per risolvere il problema

$$\begin{cases} \max \; x_1 + 3\,x_2 \\ x_1 + 5\,x_2 \leq 21 \\ 8\,x_1 + 2\,x_2 \leq 35 \\ x \geq 0, \; x \in \mathbb{Z}^2. \end{cases} \tag{P}$$

Usiamo un albero bipartito.

Esempio

Applica il "Branch and Bound" per risolvere il problema

$$\begin{cases} \max\ x_1 + 3\,x_2 \\ x_1 + 5\,x_2 \leq 21 \\ 8\,x_1 + 2\,x_2 \leq 35 \\ x \geq 0,\ x \in \mathbb{Z}^2. \end{cases} \tag{P}$$

Usiamo un albero bipartito.
Sappiamo che la soluzione ottima del rilassamento continuo é $(7/2, 7/2)$ quindi $UB(P) = 14$.

Esempio

Applica il "Branch and Bound" per risolvere il problema

$$\begin{cases} \max\ x_1 + 3\,x_2 \\ x_1 + 5\,x_2 \leq 21 \\ 8\,x_1 + 2\,x_2 \leq 35 \\ x \geq 0,\ \ x \in \mathbb{Z}^2. \end{cases} \tag{P}$$

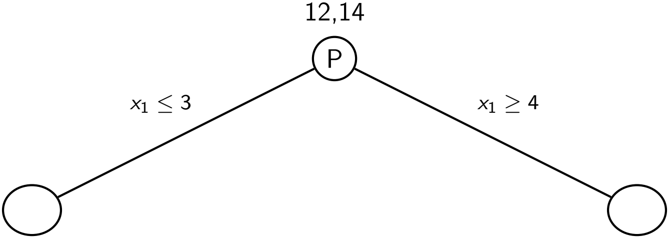
Usiamo un albero bipartito.
Sappiamo che la soluzione ottima del rilassamento continuo é $(7/2, 7/2)$ quindi $UB(P) = 14$.
Conosciamo la soluzione ammissibile $(3, 3)$ quindi $LB = 12$.

Esempio

Applica il "Branch and Bound" per risolvere il problema

$$\begin{cases} \max \; x_1 + 3\,x_2 \\ x_1 + 5\,x_2 \leq 21 \\ 8\,x_1 + 2\,x_2 \leq 35 \\ x \geq 0, \;\; x \in \mathbb{Z}^2. \end{cases} \qquad (P)$$

Usiamo un albero bipartito.
Sappiamo che la soluzione ottima del rilassamento continuo é $(7/2, 7/2)$ quindi $UB(P) = 14$.
Conosciamo la soluzione ammissibile $(3, 3)$ quindi $LB = 12$.



Esempio

La soluzione ottima del rilassamento continuo di P_1 é $(3, 18/5)$.

Esempio

La soluzione ottima del rilassamento continuo di P_1 é $(3, 18/5)$.
E' inammissibile con valore 13.8, quindi $UB(P_1) = 13 > 12 = LB$.

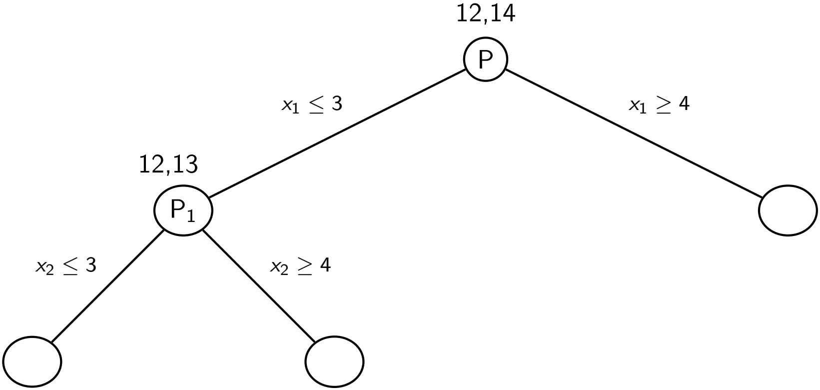
Esempio

La soluzione ottima del rilassamento continuo di P_1 é $(3, 18/5)$.
E' inammissibile con valore 13.8, quindi $UB(P_1) = 13 > 12 = LB$.
Il nodo P_1 rimane aperto.

Esempio

La soluzione ottima del rilassamento continuo di P_1 é $(3, 18/5)$.
E' inammissibile con valore 13.8, quindi $UB(P_1) = 13 > 12 = LB$.
Il nodo P_1 rimane aperto.
Partizione: $x_2 \leq 3$ e $x_2 \geq 4$.

La soluzione ottima del rilassamento continuo di P_1 é $(3, 18/5)$.
E' inammissibile con valore 13.8, quindi $UB(P_1) = 13 > 12 = LB$.
Il nodo P_1 rimane aperto.
Partizione: $x_2 \leq 3$ e $x_2 \geq 4$.



Esempio

La soluzione ottima del rilassamento continuo di P_2 é $(3, 3)$, quindi $UB(P_2) = 12 = LB$, chiudiamo il nodo P_2.

Esempio

La soluzione ottima del rilassamento continuo di P_2 é $(3, 3)$, quindi $UB(P_2) = 12 = LB$, chiudiamo il nodo P_2.
La soluzione ottima del rilassamento continuo di P_3 é $(1, 4)$ e $UB(P_3) = 13 > 12 = LB$.

Esempio

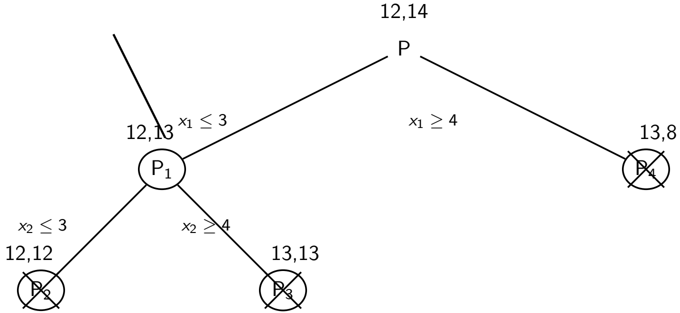
La soluzione ottima del rilassamento continuo di P_2 é $(3, 3)$, quindi $UB(P_2) = 12 = LB$, chiudiamo il nodo P_2.
La soluzione ottima del rilassamento continuo di P_3 é $(1, 4)$ e $UB(P_3) = 13 > 12 = LB$.Poiché $(1, 4)$ é ammissibile per P, aggiorniamo $LB = 13$ e chiudiamo il nodo P_3.

Esempio

La soluzione ottima del rilassamento continuo di P_2 é $(3,3)$, quindi $UB(P_2) = 12 = LB$, chiudiamo il nodo P_2.
La soluzione ottima del rilassamento continuo di P_3 é $(1,4)$ e $UB(P_3) = 13 > 12 = LB$.Poiché $(1,4)$ é ammissibile per P, aggiorniamo $LB = 13$ e chiudiamo il nodo P_3.
La soluzione ottima del rilassamento continuo di P_4 é $(4, 3/2)$ con valore 8.5, quindi $UB(P_4) = 8 < 13 = LB$. Chiudiamo il nodo P_4.

Esempio

La soluzione ottima del rilassamento continuo di P_2 é $(3,3)$, quindi $UB(P_2) = 12 = LB$, chiudiamo il nodo P_2.
La soluzione ottima del rilassamento continuo di P_3 é $(1,4)$ e $UB(P_3) = 13 > 12 = LB$.Poiché $(1,4)$ é ammissibile per P, aggiorniamo $LB = 13$ e chiudiamo il nodo P_3.
La soluzione ottima del rilassamento continuo di P_4 é $(4, 3/2)$ con valore 8.5, quindi $UB(P_4) = 8 < 13 = LB$. Chiudiamo il nodo P_4.



Tutti i nodi sono chiusi, la soluzione ottima di P é $(1,4)$ e il valore é 13.

Modello del bin packing

Variabili: $x_{ij} = \begin{cases} 1 & \text{se oggetto } j \text{ inserito contenitore } i, \\ 0 & \text{altrimenti,} \end{cases}$ $y_i = \begin{cases} 1 & \text{se } i \text{ é usato,} \\ 0 & \text{altrimenti.} \end{cases}$

Modello del bin packing

Variabili: $x_{ij} = \begin{cases} 1 & \text{se oggetto } j \text{ inserito contenitore } i, \\ 0 & \text{altrimenti,} \end{cases}$ $y_i = \begin{cases} 1 & \text{se } i \text{ é usato,} \\ 0 & \text{altrimenti.} \end{cases}$

$$\min \sum_{i=1}^{m} y_i$$

$$\sum_{i=1}^{m} x_{ij} = 1 \quad \forall\, j = 1, \ldots, n \tag{1}$$

Modello del bin packing

Variabili: $x_{ij} = \begin{cases} 1 & \text{se oggetto } j \text{ inserito contenitore } i, \\ 0 & \text{altrimenti,} \end{cases}$ $y_i = \begin{cases} 1 & \text{se } i \text{ é usato,} \\ 0 & \text{altrimenti.} \end{cases}$

$$\min \sum_{i=1}^{m} y_i$$

$$\sum_{i=1}^{m} x_{ij} = 1 \quad \forall\, j = 1, \ldots, n \tag{1}$$

$$\sum_{j=1}^{n} p_j\, x_{ij} \leq C\, y_i \quad \forall\, i = 1, \ldots, m \tag{2}$$

Modello del bin packing

Variabili: $x_{ij} = \begin{cases} 1 & \text{se oggetto } j \text{ inserito contenitore } i, \\ 0 & \text{altrimenti,} \end{cases}$ $y_i = \begin{cases} 1 & \text{se } i \text{ é usato,} \\ 0 & \text{altrimenti.} \end{cases}$

$$\min \sum_{i=1}^{m} y_i$$

$$\sum_{i=1}^{m} x_{ij} = 1 \quad \forall\ j = 1, \ldots, n \tag{1}$$

$$\sum_{j=1}^{n} p_j x_{ij} \leq C y_i \quad \forall\ i = 1, \ldots, m \tag{2}$$

$$x_{ij} \in \{0, 1\} \qquad \forall\ i, j$$

$$y_i \in \{0, 1\} \qquad \forall\ i$$

Modello del bin packing

Variabili: $x_{ij} = \begin{cases} 1 & \text{se oggetto } j \text{ inserito contenitore } i, \\ 0 & \text{altrimenti,} \end{cases}$ $y_i = \begin{cases} 1 & \text{se } i \text{ é usato,} \\ 0 & \text{altrimenti.} \end{cases}$

$$\min \sum_{i=1}^{m} y_i$$

$$\sum_{i=1}^{m} x_{ij} = 1 \quad \forall\, j = 1, \ldots, n \tag{1}$$

$$\sum_{j=1}^{n} p_j\, x_{ij} \leq C\, y_i \quad \forall\, i = 1, \ldots, m \tag{2}$$

$$x_{ij} \in \{0, 1\} \qquad \forall\, i, j$$

$$y_i \in \{0, 1\} \qquad \forall\, i$$

(1): ogni oggetto é inserito in un solo contenitore.

Modello del bin packing

Variabili: $x_{ij} = \begin{cases} 1 & \text{se oggetto } j \text{ inserito contenitore } i, \\ 0 & \text{altrimenti,} \end{cases}$ $y_i = \begin{cases} 1 & \text{se } i \text{ é usato,} \\ 0 & \text{altrimenti.} \end{cases}$

$$\min \sum_{i=1}^{m} y_i$$

$$\sum_{i=1}^{m} x_{ij} = 1 \quad \forall\ j = 1, \ldots, n \tag{1}$$

$$\sum_{j=1}^{n} p_j\, x_{ij} \leq C\, y_i \quad \forall\ i = 1, \ldots, m \tag{2}$$

$$x_{ij} \in \{0, 1\} \qquad \forall\ i, j$$

$$y_i \in \{0, 1\} \qquad \forall\ i$$

(1): ogni oggetto é inserito in un solo contenitore.
(2): capacitá contenitori.

Metodi euristici

Per calcolare una soluzione ammissibile (e quindi una valutazione superiore) descriviamo 3 algoritmi *greedy*.

Metodi euristici

Per calcolare una soluzione ammissibile (e quindi una valutazione superiore) descriviamo 3 algoritmi *greedy*.

Algoritmo Next-Fit Decreasing (NFD)

Esamina gli oggetti in ordine di peso decrescente.

Metodi euristici

Per calcolare una soluzione ammissibile (e quindi una valutazione superiore) descriviamo 3 algoritmi *greedy*.

Algoritmo Next-Fit Decreasing (NFD)

Esamina gli oggetti in ordine di peso decrescente.
Il primo contenitore é il contenitore corrente.

Metodi euristici

Per calcolare una soluzione ammissibile (e quindi una valutazione superiore) descriviamo 3 algoritmi *greedy*.

Algoritmo Next-Fit Decreasing (NFD)

Esamina gli oggetti in ordine di peso decrescente.
Il primo contenitore é il contenitore corrente.
Se possibile, assegna un oggetto al contenitore corrente;

Per calcolare una soluzione ammissibile (e quindi una valutazione superiore) descriviamo 3 algoritmi *greedy*.

Algoritmo Next-Fit Decreasing (NFD)

Esamina gli oggetti in ordine di peso decrescente.
Il primo contenitore é il contenitore corrente.
Se possibile, assegna un oggetto al contenitore corrente;
altrimenti assegnalo ad un nuovo contenitore, che diventa quello corrente.

Metodi euristici

Per calcolare una soluzione ammissibile (e quindi una valutazione superiore) descriviamo 3 algoritmi *greedy*.

Algoritmo Next-Fit Decreasing (NFD)

Esamina gli oggetti in ordine di peso decrescente.
Il primo contenitore é il contenitore corrente.
Se possibile, assegna un oggetto al contenitore corrente;
altrimenti assegnalo ad un nuovo contenitore, che diventa quello corrente.

Esempio

j	1	2	3	4	5	6	7	8	9	
p_j	70	60	50	33	33	33	11	7	3	$C = 100$

Metodi euristici

Per calcolare una soluzione ammissibile (e quindi una valutazione superiore) descriviamo 3 algoritmi *greedy*.

Algoritmo Next-Fit Decreasing (NFD)

Esamina gli oggetti in ordine di peso decrescente.
Il primo contenitore é il contenitore corrente.
Se possibile, assegna un oggetto al contenitore corrente;
altrimenti assegnalo ad un nuovo contenitore, che diventa quello corrente.

Esempio

j	1	2	3	4	5	6	7	8	9	
p_j	70	60	50	33	33	33	11	7	3	$C = 100$

Contenitori	Oggetti	Capacitá residua
1	1	
2	2	
3	3 4	
4	5	67

Per calcolare una soluzione ammissibile (e quindi una valutazione superiore) descriviamo 3 algoritmi *greedy*.

Algoritmo Next-Fit Decreasing (NFD)

Esamina gli oggetti in ordine di peso decrescente.
Il primo contenitore é il contenitore corrente.
Se possibile, assegna un oggetto al contenitore corrente;
altrimenti assegnalo ad un nuovo contenitore, che diventa quello corrente.

Esempio

j	1	2	3	4	5	6	7	8	9	
p_j	70	60	50	33	33	33	11	7	3	$C = 100$

Contenitori	Oggetti	Capacitá residua
1	1	
2	2	
3	3 4	
4	5 6	34

Per calcolare una soluzione ammissibile (e quindi una valutazione superiore) descriviamo 3 algoritmi *greedy*.

Algoritmo Next-Fit Decreasing (NFD)

Esamina gli oggetti in ordine di peso decrescente.
Il primo contenitore é il contenitore corrente.
Se possibile, assegna un oggetto al contenitore corrente;
altrimenti assegnalo ad un nuovo contenitore, che diventa quello corrente.

Esempio

j	1	2	3	4	5	6	7	8	9	
p_j	70	60	50	33	33	33	11	7	3	$C = 100$

Contenitori	Oggetti	Capacitá residua
1	1	
2	2	
3	3 4	
4	5 6 7	23

Metodi euristici

Per calcolare una soluzione ammissibile (e quindi una valutazione superiore) descriviamo 3 algoritmi *greedy*.

Algoritmo Next-Fit Decreasing (NFD)

Esamina gli oggetti in ordine di peso decrescente.
Il primo contenitore é il contenitore corrente.
Se possibile, assegna un oggetto al contenitore corrente;
altrimenti assegnalo ad un nuovo contenitore, che diventa quello corrente.

Esempio

j	1	2	3	4	5	6	7	8	9	
p_j	70	60	50	33	33	33	11	7	3	$C = 100$

Contenitori	Oggetti	Capacitá residua
1	1	
2	2	
3	3 4	
4	5 6 7 8	16

Metodi euristici

Per calcolare una soluzione ammissibile (e quindi una valutazione superiore) descriviamo 3 algoritmi *greedy*.

Algoritmo Next-Fit Decreasing (NFD)

Esamina gli oggetti in ordine di peso decrescente.
Il primo contenitore é il contenitore corrente.
Se possibile, assegna un oggetto al contenitore corrente;
altrimenti assegnalo ad un nuovo contenitore, che diventa quello corrente.

Esempio

j	1	2	3	4	5	6	7	8	9	
p_j	70	60	50	33	33	33	11	7	3	$C = 100$

Contenitori	Oggetti	Capacitá residua
1	1	
2	2	
3	3 4	
4	5 6 7 8 9	13

Algoritmo First-Fit Decreasing (FFD)

Esamina gli oggetti in ordine di peso decrescente.

Algoritmo First-Fit Decreasing (FFD)

Esamina gli oggetti in ordine di peso decrescente.
Assegna ogni oggetto al primo contenitore usato che puó contenerlo.

Metodi euristici

Algoritmo First-Fit Decreasing (FFD)

Esamina gli oggetti in ordine di peso decrescente.
Assegna ogni oggetto al primo contenitore usato che puó contenerlo.
Se nessuno di essi puó contenerlo, assegna l'oggetto ad un nuovo contenitore.

Metodi euristici

Algoritmo First-Fit Decreasing (FFD)

Esamina gli oggetti in ordine di peso decrescente.
Assegna ogni oggetto al primo contenitore usato che puó contenerlo.
Se nessuno di essi puó contenerlo, assegna l'oggetto ad un nuovo contenitore.

Esempio

j	1	2	3	4	5	6	7	8	9	
p_j	70	60	50	33	33	33	11	7	3	$C = 100$

Algoritmo First-Fit Decreasing (FFD)

Esamina gli oggetti in ordine di peso decrescente.
Assegna ogni oggetto al primo contenitore usato che puó contenerlo.
Se nessuno di essi puó contenerlo, assegna l'oggetto ad un nuovo contenitore.

Esempio

j	1	2	3	4	5	6	7	8	9	
p_j	70	60	50	33	33	33	11	7	3	$C = 100$

Contenitori	Oggetti	Capacitá residua
1	1	30
2	2	40
3	3	50

Metodi euristici

Algoritmo First-Fit Decreasing (FFD)

Esamina gli oggetti in ordine di peso decrescente.
Assegna ogni oggetto al primo contenitore usato che puó contenerlo.
Se nessuno di essi puó contenerlo, assegna l'oggetto ad un nuovo contenitore.

Esempio

j	1	2	3	4	5	6	7	8	9	
p_j	70	60	50	33	33	33	11	7	3	$C = 100$

Contenitori	Oggetti	Capacitá residua
1	1	30
2	2 4	7
3	3	50

Algoritmo First-Fit Decreasing (FFD)

Esamina gli oggetti in ordine di peso decrescente.
Assegna ogni oggetto al primo contenitore usato che puó contenerlo.
Se nessuno di essi puó contenerlo, assegna l'oggetto ad un nuovo contenitore.

Esempio

j	1	2	3	4	5	6	7	8	9	
p_j	70	60	50	33	33	33	11	7	3	$C = 100$

Contenitori	Oggetti	Capacitá residua
1	1	30
2	2 4	7
3	3 5	17

Metodi euristici

Algoritmo First-Fit Decreasing (FFD)

Esamina gli oggetti in ordine di peso decrescente.
Assegna ogni oggetto al primo contenitore usato che puó contenerlo.
Se nessuno di essi puó contenerlo, assegna l'oggetto ad un nuovo contenitore.

Esempio

j	1	2	3	4	5	6	7	8	9	
p_j	70	60	50	33	33	33	11	7	3	$C = 100$

Contenitori	Oggetti	Capacitá residua
1	1	30
2	2 4	7
3	3 5	17
4	6	67

Metodi euristici

Algoritmo First-Fit Decreasing (FFD)

Esamina gli oggetti in ordine di peso decrescente.
Assegna ogni oggetto al primo contenitore usato che puó contenerlo.
Se nessuno di essi puó contenerlo, assegna l'oggetto ad un nuovo contenitore.

Esempio

j	1	2	3	4	5	6	7	8	9	
p_j	70	60	50	33	33	33	11	7	3	$C = 100$

Contenitori	Oggetti	Capacitá residua
1	1 7	19
2	2 4	7
3	3 5	17
4	6	67

Algoritmo First-Fit Decreasing (FFD)

Esamina gli oggetti in ordine di peso decrescente.
Assegna ogni oggetto al primo contenitore usato che puó contenerlo.
Se nessuno di essi puó contenerlo, assegna l'oggetto ad un nuovo contenitore.

Esempio

j	1	2	3	4	5	6	7	8	9	
p_j	70	60	50	33	33	33	11	7	3	$C = 100$

Contenitori	Oggetti	Capacitá residua
1	1 7 8	12
2	2 4	7
3	3 5	17
4	6	67

Algoritmo First-Fit Decreasing (FFD)

Esamina gli oggetti in ordine di peso decrescente.
Assegna ogni oggetto al primo contenitore usato che puó contenerlo.
Se nessuno di essi puó contenerlo, assegna l'oggetto ad un nuovo contenitore.

Esempio

j	1	2	3	4	5	6	7	8	9	
p_j	70	60	50	33	33	33	11	7	3	$C = 100$

Contenitori	Oggetti	Capacitá residua
1	1 7 8 9	9
2	2 4	7
3	3 5	17
4	6	67

Algoritmo Best-Fit Decreasing (BFD)

Esamina gli oggetti in ordine di peso decrescente.

Algoritmo Best-Fit Decreasing (BFD)

Esamina gli oggetti in ordine di peso decrescente.
Tra tutti i contenitori usati che possono contenere un oggetto, scegli quello con la minima capacitá residua.

Algoritmo Best-Fit Decreasing (BFD)

Esamina gli oggetti in ordine di peso decrescente.
Tra tutti i contenitori usati che possono contenere un oggetto, scegli quello con la minima capacitá residua.
Se nessuno di essi puó contenerlo, assegna l'oggetto ad un nuovo contenitore.

Algoritmo Best-Fit Decreasing (BFD)

Esamina gli oggetti in ordine di peso decrescente.
Tra tutti i contenitori usati che possono contenere un oggetto, scegli quello con la minima capacitá residua.
Se nessuno di essi puó contenerlo, assegna l'oggetto ad un nuovo contenitore.

Esempio

j	1	2	3	4	5	6	7	8	9	
p_j	70	60	50	33	33	33	11	7	3	$C = 100$

Metodi euristici

Algoritmo Best-Fit Decreasing (BFD)

Esamina gli oggetti in ordine di peso decrescente.
Tra tutti i contenitori usati che possono contenere un oggetto, scegli quello con la minima capacitá residua.
Se nessuno di essi puó contenerlo, assegna l'oggetto ad un nuovo contenitore.

Esempio

j	1	2	3	4	5	6	7	8	9	
p_j	70	60	50	33	33	33	11	7	3	$C = 100$

Contenitori	Oggetti	Capacitá residua
1	1	30
2	2	40
3	3	50

Algoritmo Best-Fit Decreasing (BFD)

Esamina gli oggetti in ordine di peso decrescente.
Tra tutti i contenitori usati che possono contenere un oggetto, scegli quello con la minima capacitá residua.
Se nessuno di essi puó contenerlo, assegna l'oggetto ad un nuovo contenitore.

Esempio

j	1	2	3	4	5	6	7	8	9	
p_j	70	60	50	33	33	33	11	7	3	$C = 100$

Contenitori	Oggetti	Capacitá residua
1	1	30
2	2 4	7
3	3	50

Metodi euristici

Algoritmo Best-Fit Decreasing (BFD)

Esamina gli oggetti in ordine di peso decrescente.
Tra tutti i contenitori usati che possono contenere un oggetto, scegli quello con la minima capacitá residua.
Se nessuno di essi puó contenerlo, assegna l'oggetto ad un nuovo contenitore.

Esempio

j	1	2	3	4	5	6	7	8	9	
p_j	70	60	50	33	33	33	11	7	3	$C = 100$

Contenitori	Oggetti	Capacitá residua
1	1	30
2	2 4	7
3	3 5	17

Metodi euristici

Algoritmo Best-Fit Decreasing (BFD)

Esamina gli oggetti in ordine di peso decrescente.
Tra tutti i contenitori usati che possono contenere un oggetto, scegli quello con la minima capacitá residua.
Se nessuno di essi puó contenerlo, assegna l'oggetto ad un nuovo contenitore.

Esempio

j	1	2	3	4	5	6	7	8	9	
p_j	70	60	50	33	33	33	11	7	3	$C = 100$

Contenitori	Oggetti	Capacitá residua
1	1	30
2	2 4	7
3	3 5	17
4	6	67

Algoritmo Best-Fit Decreasing (BFD)

Esamina gli oggetti in ordine di peso decrescente.
Tra tutti i contenitori usati che possono contenere un oggetto, scegli quello con la minima capacitá residua.
Se nessuno di essi puó contenerlo, assegna l'oggetto ad un nuovo contenitore.

Esempio

j	1	2	3	4	5	6	7	8	9	
p_j	70	60	50	33	33	33	11	7	3	$C = 100$

Contenitori	Oggetti	Capacitá residua
1	1	30
2	2 4	7
3	3 5 7	6
4	6	67

Metodi euristici

Algoritmo Best-Fit Decreasing (BFD)

Esamina gli oggetti in ordine di peso decrescente.
Tra tutti i contenitori usati che possono contenere un oggetto, scegli quello con la minima capacitá residua.
Se nessuno di essi puó contenerlo, assegna l'oggetto ad un nuovo contenitore.

Esempio

j	1	2	3	4	5	6	7	8	9	
p_j	70	60	50	33	33	33	11	7	3	$C = 100$

Contenitori	Oggetti	Capacitá residua
1	1	30
2	2 4 8	0
3	3 5 7	6
4	6	67

Algoritmo Best-Fit Decreasing (BFD)

Esamina gli oggetti in ordine di peso decrescente.
Tra tutti i contenitori usati che possono contenere un oggetto, scegli quello con la minima capacitá residua.
Se nessuno di essi puó contenerlo, assegna l'oggetto ad un nuovo contenitore.

Esempio

j	1	2	3	4	5	6	7	8	9	
p_j	70	60	50	33	33	33	11	7	3	$C = 100$

Contenitori	Oggetti	Capacitá residua
1	1	30
2	2 4 8	0
3	3 5 7 9	3
4	6	67

Rilassamento continuo

Per calcolare una valutazione inferiore $v_I(P)$ usiamo il rilassamento continuo.

Rilassamento continuo

Per calcolare una valutazione inferiore $v_I(P)$ usiamo il rilassamento continuo.

Teorema

Il rilassamento continuo di (P):

$$\begin{cases} \min \sum_{i=1}^{n} y_i \\ \sum_{i=1}^{n} x_{ij} = 1 \quad \forall\ j = 1, \dots, n \\ \sum_{j=1}^{n} p_j x_{ij} \leq C y_i \quad \forall\ i = 1, \dots, n \\ 0 \leq x_{ij} \leq 1 \quad \forall\ i, j \\ 0 \leq y_i \leq 1 \quad \forall\ i \end{cases} \tag{RC}$$

Rilassamento continuo

Per calcolare una valutazione inferiore $v_I(P)$ usiamo il rilassamento continuo.

Teorema

Il rilassamento continuo di (P):

$$\begin{cases} \min \ \sum_{i=1}^{n} y_i \\ \sum_{i=1}^{n} x_{ij} = 1 \quad \forall\ j = 1, \dots, n \\ \sum_{j=1}^{n} p_j \, x_{ij} \leq C\, y_i \quad \forall\ i = 1, \dots, n \\ 0 \leq x_{ij} \leq 1 \quad \forall\ i, j \\ 0 \leq y_i \leq 1 \quad \forall\ i \end{cases} \tag{RC}$$

ha come soluzione ottima $x_{ij} = \begin{cases} 1 & \text{se } i = j, \\ 0 & \text{se } i \neq j, \end{cases} \quad y_i = \dfrac{p_i}{C}$ per ogni i.

Rilassamento continuo

Per calcolare una valutazione inferiore $v_I(P)$ usiamo il rilassamento continuo.

Teorema

Il rilassamento continuo di (P):

$$\begin{cases} \min \sum_{i=1}^{n} y_i \\ \sum_{i=1}^{n} x_{ij} = 1 \quad \forall\ j = 1, \dots, n \\ \sum_{j=1}^{n} p_j x_{ij} \le C y_i \quad \forall\ i = 1, \dots, n \\ 0 \le x_{ij} \le 1 \quad \forall\ i, j \\ 0 \le y_i \le 1 \quad \forall\ i \end{cases} \tag{RC}$$

ha come soluzione ottima $x_{ij} = \begin{cases} 1 & \text{se } i = j, \\ 0 & \text{se } i \neq j, \end{cases} \quad y_i = \dfrac{p_i}{C}$ per ogni i.

$L = \left\lceil \sum_{i=1}^{n} p_i / C \right\rceil$ é una $v_I(P)$.

Esempio

Sia dato il seguente problema:

j	1	2	3	4	5	6	7	
p_j	99	98	64	45	40	23	17	$C = 100$

Esempio

Sia dato il seguente problema:

j	1	2	3	4	5	6	7	
p_j	99	98	64	45	40	23	17	$C = 100$

Calcoliamo una $v_I(P)$:

Esempio

Sia dato il seguente problema:

j	1	2	3	4	5	6	7	
p_j	99	98	64	45	40	23	17	$C = 100$

Calcoliamo una $v_I(P)$:

$$L = \left\lceil \frac{386}{100} \right\rceil = 4$$

Esempio

Sia dato il seguente problema:

j	1	2	3	4	5	6	7	
p_j	99	98	64	45	40	23	17	$C = 100$

Calcoliamo una $v_I(P)$:

$$L = \left\lceil \frac{386}{100} \right\rceil = 4$$

Calcoliamo una $v_S(P)$:

Esempio

Sia dato il seguente problema:

j	1	2	3	4	5	6	7	
p_j	99	98	64	45	40	23	17	$C = 100$

Calcoliamo una $v_I(P)$:

$$L = \left\lceil \frac{386}{100} \right\rceil = 4$$

Calcoliamo una $v_S(P)$:

Oggetti	1	2	3	4	5	6	7
Algoritmo NFD	1	2	3	4	4	5	5
Algoritmo FFD	1	2	3	4	4	3	5
Algoritmo BFD	1	2	3	4	4	3	5

Esempio

Sia dato il seguente problema:

j	1	2	3	4	5	6	7	
p_j	99	98	64	45	40	23	17	$C = 100$

Calcoliamo una $v_I(P)$:

$$L = \left\lceil \frac{386}{100} \right\rceil = 4$$

Calcoliamo una $v_S(P)$:

Oggetti	1	2	3	4	5	6	7
Algoritmo NFD	1	2	3	4	4	5	5
Algoritmo FFD	1	2	3	4	4	3	5
Algoritmo BFD	1	2	3	4	4	3	5

Quindi $v_S(P) = 5$.

Problema del commesso viaggiatore (TSP)

Problema

Grafo (N, A) completo; $c_{ij} =$ costi sugli archi.

Problema del commesso viaggiatore (TSP)

Problema

Grafo (N, A) completo; c_{ij} = costi sugli archi.
Trovare un ciclo di costo minimo che passi su tutti i nodi una ed una sola volta (ciclo hamiltoniano).

Problema

Grafo (N, A) completo; $c_{ij} =$ costi sugli archi.
Trovare un ciclo di costo minimo che passi su tutti i nodi una ed una sola volta (ciclo hamiltoniano).

Quante sono le soluzioni ammissibili?

Problema del commesso viaggiatore (TSP)

Problema

Grafo (N, A) completo; c_{ij} = costi sugli archi.
Trovare un ciclo di costo minimo che passi su tutti i nodi una ed una sola volta (ciclo hamiltoniano).

Quante sono le soluzioni ammissibili? n!

Problema del commesso viaggiatore (TSP)

Problema

Grafo (N, A) completo; c_{ij} = costi sugli archi.
Trovare un ciclo di costo minimo che passi su tutti i nodi una ed una sola volta (ciclo hamiltoniano).

Quante sono le soluzioni ammissibili? n!

Applicazioni

- trasporti, logistica: (N', A') rete stradale. $S \subseteq N'$, cerco ciclo di costo minimo che passi su tutti i nodi di S. Il problema é un TSP sul grafo (N, A), dove $N = S$, $A = S \times S$, c_{ij} = costo cammino minimo da i a j sul grafo (N', A').

Problema del commesso viaggiatore (TSP)

Problema

Grafo (N, A) completo; c_{ij} = costi sugli archi.
Trovare un ciclo di costo minimo che passi su tutti i nodi una ed una sola volta (ciclo hamiltoniano).

Quante sono le soluzioni ammissibili? n!

Applicazioni

- trasporti, logistica: (N', A') rete stradale. $S \subseteq N'$, cerco ciclo di costo minimo che passi su tutti i nodi di S. Il problema é un TSP sul grafo (N, A), dove $N = S$, $A = S \times S$, c_{ij} = costo cammino minimo da i a j sul grafo (N', A').
- produzione di circuiti integrati

Problema del commesso viaggiatore (TSP)

Problema

Grafo (N, A) completo; c_{ij} = costi sugli archi.
Trovare un ciclo di costo minimo che passi su tutti i nodi una ed una sola volta (ciclo hamiltoniano).

Quante sono le soluzioni ammissibili? n!

Applicazioni

- trasporti, logistica: (N', A') rete stradale. $S \subseteq N'$, cerco ciclo di costo minimo che passi su tutti i nodi di S. Il problema é un TSP sul grafo (N, A), dove $N = S$, $A = S \times S$, c_{ij} = costo cammino minimo da i a j sul grafo (N', A').
- produzione di circuiti integrati
- data analysis

Problema del commesso viaggiatore (TSP)

Problema

Grafo (N, A) completo; c_{ij} = costi sugli archi.
Trovare un ciclo di costo minimo che passi su tutti i nodi una ed una sola volta (ciclo hamiltoniano).

Quante sono le soluzioni ammissibili? n!

Applicazioni

- trasporti, logistica: (N', A') rete stradale. $S \subseteq N'$, cerco ciclo di costo minimo che passi su tutti i nodi di S. Il problema é un TSP sul grafo (N, A), dove $N = S$, $A = S \times S$, c_{ij} = costo cammino minimo da i a j sul grafo (N', A').
- produzione di circuiti integrati
- data analysis
- sequenze DNA

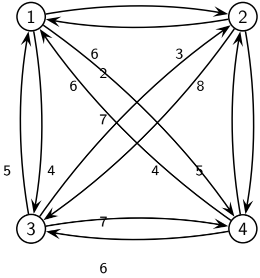
TSP simmetrico e asimmetrico

Se la matrice dei costi é simmetrica, cioé $c_{ij} = c_{ji}$ per ogni arco (i, j), il problema é detto simmetrico; altrimenti é detto asimmetrico.

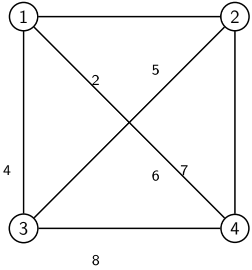
TSP simmetrico e asimmetrico

Se la matrice dei costi é simmetrica, cioé $c_{ij} = c_{ji}$ per ogni arco (i, j), il problema é detto simmetrico; altrimenti é detto asimmetrico.

TSP asimmetrico

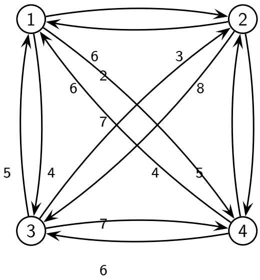


TSP simmetrico

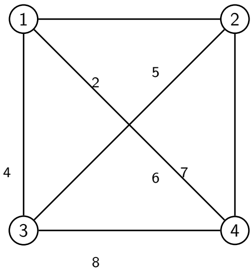


Se la matrice dei costi é simmetrica, cioé $c_{ij} = c_{ji}$ per ogni arco (i, j), il problema é detto simmetrico; altrimenti é detto asimmetrico.

TSP asimmetrico



TSP simmetrico



Prima tratteremo il problema asimmetrico (piú generale) e poi quello simmetrico (caso particolare).

Modello

Variabili: $x_{ij} = \begin{cases} 1 & \text{se arco } (i,j) \in \text{ciclo hamiltoniano,} \\ 0 & \text{altrimenti.} \end{cases}$

Modello

Variabili: $x_{ij} = \begin{cases} 1 & \text{se arco } (i,j) \in \text{ciclo hamiltoniano}, \\ 0 & \text{altrimenti.} \end{cases}$

$$
\begin{aligned}
\min \sum_{(i,j)\in A} c_{ij} x_{ij} & \\
\sum_{i \in N\setminus\{j\}} x_{ij} = 1 & \qquad \forall\, j \in N \\
\sum_{j \in N\setminus\{i\}} x_{ij} = 1 & \qquad \forall\, i \in N \\
\sum_{(i,j)\in A:\ i\in S,\ j\notin S} x_{ij} \geq 1 & \qquad \forall\, S \subseteq N, \quad S \neq \emptyset, N \\
x_{ij} \in \{0,1\} & \qquad \forall\, (i,j) \in A
\end{aligned}
$$

Rilassamenti

Eliminando i vincoli di connessione dal modello si ottiene un problema di assegnamento di costo minimo che é quindi una valutazione inferiore:

Eliminando i vincoli di connessione dal modello si ottiene un problema di assegnamento di costo minimo che é quindi una valutazione inferiore:

$$\min \sum_{i \in N} \sum_{j \in N} c_{ij} x_{ij}$$

$$\sum_{i \in N} x_{ij} = 1 \quad \forall\ j \in N$$

$$\sum_{j \in N} x_{ij} = 1 \quad \forall\ i \in N$$

$$x_{ij} \in \{0, 1\} \quad \forall\ i, j \in N$$

Rilassamenti

Eliminando i vincoli di connessione dal modello si ottiene un problema di assegnamento di costo minimo che é quindi una valutazione inferiore:

$$\begin{aligned} \min \ & \sum_{i \in N} \sum_{j \in N} c_{ij} x_{ij} \\ & \sum_{i \in N} x_{ij} = 1 \quad \forall\ j \in N \\ & \sum_{j \in N} x_{ij} = 1 \quad \forall\ i \in N \\ & x_{ij} \in \{0,1\} \quad \forall\ i,j \in N \end{aligned}$$

che é facile da risolvere.

Rilassamenti

Eliminando i vincoli di connessione dal modello si ottiene un problema di assegnamento di costo minimo che é quindi una valutazione inferiore:

$$\min \sum_{i \in N} \sum_{j \in N} c_{ij} x_{ij}$$

$$\sum_{i \in N} x_{ij} = 1 \quad \forall\ j \in N$$

$$\sum_{j \in N} x_{ij} = 1 \quad \forall\ i \in N$$

$$x_{ij} \in \{0, 1\} \quad \forall\ i, j \in N$$

che é facile da risolvere.

La soluzione ottima di tale rilassamento é, a priori, una famiglia di cicli orientati che coprono tutti i nodi.

Esempio

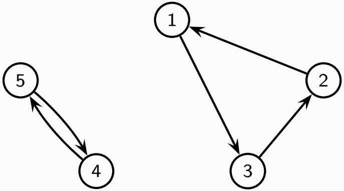
Consideriamo la seguente matrice dei costi:

	1	2	3	4	5
1	–	33	13	25	33
2	33	–	46	58	76
3	39	33	–	12	30
4	35	29	12	–	23
5	60	54	30	23	–

Esempio

Consideriamo la seguente matrice dei costi:

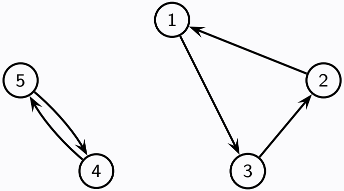
	1	2	3	4	5
1	–	33	13	25	33
2	33	–	46	58	76
3	39	33	–	12	30
4	35	29	12	–	23
5	60	54	30	23	–



Esempio

Consideriamo la seguente matrice dei costi:

	1	2	3	4	5
1	–	33	13	25	33
2	33	–	46	58	76
3	39	33	–	12	30
4	35	29	12	–	23
5	60	54	30	23	–



La soluzione ottima del rilassamento é formata da due cicli:

$$x_{13} = 1, \quad x_{32} = 1, \quad x_{21} = 1, \quad x_{45} = 1, \quad x_{54} = 1,$$

e ha valore $125 = v_I(P)$.

Osservazione

Quando la matrice dei costi é fortemente asimmetrica, il valore ottimo del problema di assegnamento é di solito una buona stima del valore ottimo del TSP.

Osservazione

Quando la matrice dei costi é fortemente asimmetrica, il valore ottimo del problema di assegnamento é di solito una buona stima del valore ottimo del TSP.

Quando la matrice dei costi é simmetrica, l'assegnamento di costo minimo solitamente é formato da un gran numero di cicli orientati.

Metodo *greedy* sugli archi

Dispongo gli archi in ordine crescente di costo.

Metodi euristici per valutazioni superiori

Metodo *greedy* sugli archi

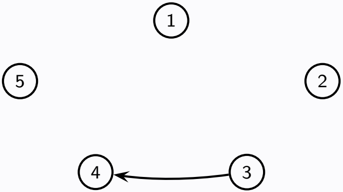
Dispongo gli archi in ordine crescente di costo.
Seguendo l'ordine, inserisco un arco se vengono rispettati tutti i vincoli.

Metodo *greedy* sugli archi

Dispongo gli archi in ordine crescente di costo.
Seguendo l'ordine, inserisco un arco se vengono rispettati tutti i vincoli.

Esempio

	1	2	3	4	5
1	–	33	13	25	33
2	33	–	46	58	76
3	39	33	–	12	30
4	35	29	12	–	23
5	60	54	30	23	–



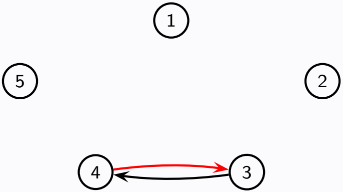
$x_{34} = 1,$

Metodo *greedy* sugli archi

Dispongo gli archi in ordine crescente di costo.
Seguendo l'ordine, inserisco un arco se vengono rispettati tutti i vincoli.

Esempio

	1	2	3	4	5
1	–	33	13	25	33
2	33	–	46	58	76
3	39	33	–	12	30
4	35	29	12	–	23
5	60	54	30	23	–



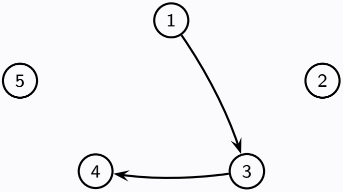
$x_{34} = 1,\ x_{43} = 0,$

Metodo *greedy* sugli archi

Dispongo gli archi in ordine crescente di costo.
Seguendo l'ordine, inserisco un arco se vengono rispettati tutti i vincoli.

Esempio

	1	2	3	4	5
1	–	33	13	25	33
2	33	–	46	58	76
3	39	33	–	12	30
4	35	29	12	–	23
5	60	54	30	23	–



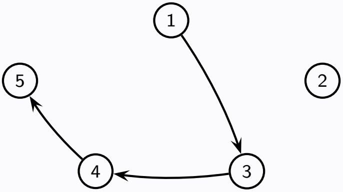
$x_{34} = 1$, $x_{43} = 0$, $x_{13} = 1$,

Metodo *greedy* sugli archi

Dispongo gli archi in ordine crescente di costo.
Seguendo l'ordine, inserisco un arco se vengono rispettati tutti i vincoli.

Esempio

	1	2	3	4	5
1	–	33	13	25	33
2	33	–	46	58	76
3	39	33	–	12	30
4	35	29	12	–	23
5	60	54	30	23	–



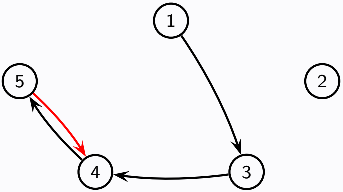
$x_{34} = 1,\ x_{43} = 0,\ x_{13} = 1,\ x_{45} = 1,$

Metodo *greedy* sugli archi

Dispongo gli archi in ordine crescente di costo.
Seguendo l'ordine, inserisco un arco se vengono rispettati tutti i vincoli.

Esempio

	1	2	3	4	5
1	–	33	13	25	33
2	33	–	46	58	76
3	39	33	–	12	30
4	35	29	12	–	23
5	60	54	30	23	–



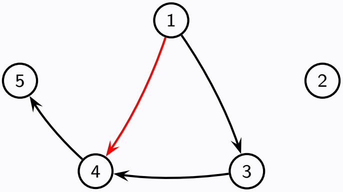
$x_{34} = 1,\ x_{43} = 0,\ x_{13} = 1,\ x_{45} = 1,\ x_{54} = 0,$

Metodo *greedy* sugli archi

Dispongo gli archi in ordine crescente di costo.
Seguendo l'ordine, inserisco un arco se vengono rispettati tutti i vincoli.

Esempio

	1	2	3	4	5
1	–	33	13	25	33
2	33	–	46	58	76
3	39	33	–	12	30
4	35	29	12	–	23
5	60	54	30	23	–



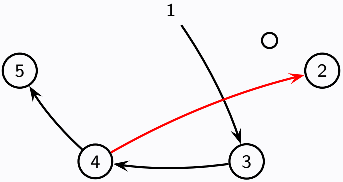
$x_{34} = 1,\ x_{43} = 0,\ x_{13} = 1,\ x_{45} = 1,\ x_{54} = 0,\ x_{14} = 0,$

Metodo *greedy* sugli archi

Dispongo gli archi in ordine crescente di costo.
Seguendo l'ordine, inserisco un arco se vengono rispettati tutti i vincoli.

Esempio

	1	2	3	4	5
1	–	33	13	25	33
2	33	–	46	58	76
3	39	33	–	12	30
4	35	29	12	–	23
5	60	54	30	23	–



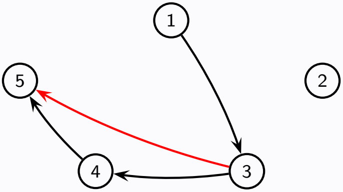
$x_{34} = 1,\ x_{43} = 0,\ x_{13} = 1,\ x_{45} = 1,\ x_{54} = 0,\ x_{14} = 0,\ x_{42} = 0,$

Metodo *greedy* sugli archi

Dispongo gli archi in ordine crescente di costo.
Seguendo l'ordine, inserisco un arco se vengono rispettati tutti i vincoli.

Esempio

	1	2	3	4	5
1	–	33	13	25	33
2	33	–	46	58	76
3	39	33	–	12	30
4	35	29	12	–	23
5	60	54	30	23	–



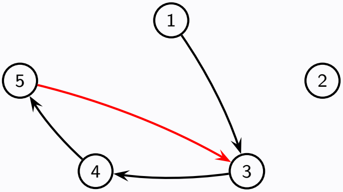
$x_{34} = 1,\ x_{43} = 0,\ x_{13} = 1,\ x_{45} = 1,\ x_{54} = 0,\ x_{14} = 0,\ x_{42} = 0,\ x_{35} = 0,$

Metodo *greedy* sugli archi

Dispongo gli archi in ordine crescente di costo.
Seguendo l'ordine, inserisco un arco se vengono rispettati tutti i vincoli.

Esempio

	1	2	3	4	5
1	–	33	13	25	33
2	33	–	46	58	76
3	39	33	–	12	30
4	35	29	12	–	23
5	60	54	30	23	–



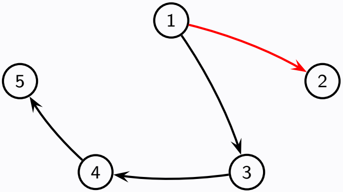
$x_{34} = 1$, $x_{43} = 0$, $x_{13} = 1$, $x_{45} = 1$, $x_{54} = 0$, $x_{14} = 0$, $x_{42} = 0$, $x_{35} = 0$, $x_{53} = 0$,

Metodo *greedy* sugli archi

Dispongo gli archi in ordine crescente di costo.
Seguendo l'ordine, inserisco un arco se vengono rispettati tutti i vincoli.

Esempio

	1	2	3	4	5
1	–	33	13	25	33
2	33	–	46	58	76
3	39	33	–	12	30
4	35	29	12	–	23
5	60	54	30	23	–



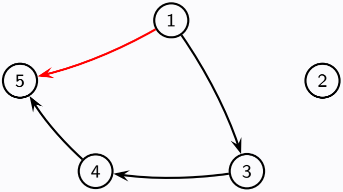
$x_{34} = 1$, $x_{43} = 0$, $x_{13} = 1$, $x_{45} = 1$, $x_{54} = 0$, $x_{14} = 0$, $x_{42} = 0$, $x_{35} = 0$, $x_{53} = 0$, $x_{12} = 0$,

Metodo *greedy* sugli archi

Dispongo gli archi in ordine crescente di costo.
Seguendo l'ordine, inserisco un arco se vengono rispettati tutti i vincoli.

Esempio

	1	2	3	4	5
1	–	33	13	25	33
2	33	–	46	58	76
3	39	33	–	12	30
4	35	29	12	–	23
5	60	54	30	23	–



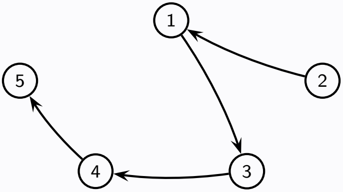
$x_{34} = 1$, $x_{43} = 0$, $x_{13} = 1$, $x_{45} = 1$, $x_{54} = 0$, $x_{14} = 0$, $x_{42} = 0$, $x_{35} = 0$, $x_{53} = 0$, $x_{12} = 0$, $x_{15} = 0$,

Metodo *greedy* sugli archi

Dispongo gli archi in ordine crescente di costo.
Seguendo l'ordine, inserisco un arco se vengono rispettati tutti i vincoli.

Esempio

	1	2	3	4	5
1	–	33	13	25	33
2	33	–	46	58	76
3	39	33	–	12	30
4	35	29	12	–	23
5	60	54	30	23	–



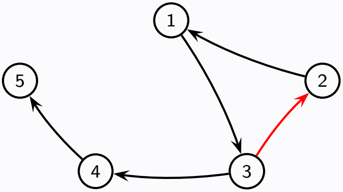
$x_{34} = 1$, $x_{43} = 0$, $x_{13} = 1$, $x_{45} = 1$, $x_{54} = 0$, $x_{14} = 0$, $x_{42} = 0$, $x_{35} = 0$, $x_{53} = 0$, $x_{12} = 0$, $x_{15} = 0$, $x_{21} = 1$,

Metodo *greedy* sugli archi

Dispongo gli archi in ordine crescente di costo.
Seguendo l'ordine, inserisco un arco se vengono rispettati tutti i vincoli.

Esempio

	1	2	3	4	5
1	–	33	13	25	33
2	33	–	46	58	76
3	39	33	–	12	30
4	35	29	12	–	23
5	60	54	30	23	–



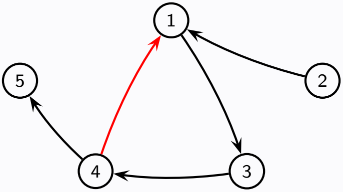
$x_{34} = 1$, $x_{43} = 0$, $x_{13} = 1$, $x_{45} = 1$, $x_{54} = 0$, $x_{14} = 0$, $x_{42} = 0$, $x_{35} = 0$, $x_{53} = 0$, $x_{12} = 0$, $x_{15} = 0$, $x_{21} = 1$, $x_{32} = 0$,

Metodo *greedy* sugli archi

Dispongo gli archi in ordine crescente di costo.
Seguendo l'ordine, inserisco un arco se vengono rispettati tutti i vincoli.

Esempio

	1	2	3	4	5
1	–	33	13	25	33
2	33	–	46	58	76
3	39	33	–	12	30
4	35	29	12	–	23
5	60	54	30	23	–



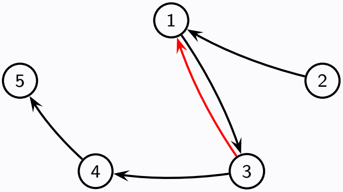
$x_{34} = 1$, $x_{43} = 0$, $x_{13} = 1$, $x_{45} = 1$, $x_{54} = 0$, $x_{14} = 0$, $x_{42} = 0$, $x_{35} = 0$, $x_{53} = 0$, $x_{12} = 0$, $x_{15} = 0$, $x_{21} = 1$, $x_{32} = 0$, $x_{41} = 0$,

Metodo *greedy* sugli archi

Dispongo gli archi in ordine crescente di costo.
Seguendo l'ordine, inserisco un arco se vengono rispettati tutti i vincoli.

Esempio

	1	2	3	4	5
1	–	33	13	25	33
2	33	–	46	58	76
3	39	33	–	12	30
4	35	29	12	–	23
5	60	54	30	23	–



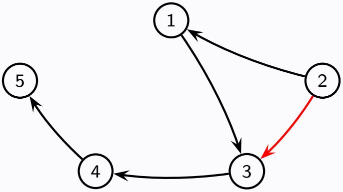
$x_{34} = 1$, $x_{43} = 0$, $x_{13} = 1$, $x_{45} = 1$, $x_{54} = 0$, $x_{14} = 0$, $x_{42} = 0$, $x_{35} = 0$, $x_{53} = 0$, $x_{12} = 0$, $x_{15} = 0$, $x_{21} = 1$, $x_{32} = 0$, $x_{41} = 0$, $x_{31} = 0$,

Metodo *greedy* sugli archi

Dispongo gli archi in ordine crescente di costo.
Seguendo l'ordine, inserisco un arco se vengono rispettati tutti i vincoli.

Esempio

	1	2	3	4	5
1	–	33	13	25	33
2	33	–	46	58	76
3	39	33	–	12	30
4	35	29	12	–	23
5	60	54	30	23	–



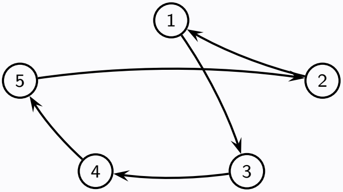
$x_{34} = 1$, $x_{43} = 0$, $x_{13} = 1$, $x_{45} = 1$, $x_{54} = 0$, $x_{14} = 0$, $x_{42} = 0$, $x_{35} = 0$, $x_{53} = 0$, $x_{12} = 0$, $x_{15} = 0$, $x_{21} = 1$, $x_{32} = 0$, $x_{41} = 0$, $x_{31} = 0$, $x_{23} = 0$,

Metodo *greedy* sugli archi

Dispongo gli archi in ordine crescente di costo.
Seguendo l'ordine, inserisco un arco se vengono rispettati tutti i vincoli.

Esempio

	1	2	3	4	5
1	–	33	13	25	33
2	33	–	46	58	76
3	39	33	–	12	30
4	35	29	12	–	23
5	60	54	30	23	–



$x_{34} = 1$, $x_{43} = 0$, $x_{13} = 1$, $x_{45} = 1$, $x_{54} = 0$, $x_{14} = 0$, $x_{42} = 0$, $x_{35} = 0$, $x_{53} = 0$, $x_{12} = 0$, $x_{15} = 0$, $x_{21} = 1$, $x_{32} = 0$, $x_{41} = 0$, $x_{31} = 0$, $x_{23} = 0$, $x_{52} = 1$.

Il ciclo trovato é 3-4-5-2-1-3 di costo 135.

Algoritmo del nodo piú vicino

Parti da un nodo i. Il nodo successivo é il piú vicino a i tra quelli non ancora visitati.

Algoritmo del nodo piú vicino

Parti da un nodo *i*. Il nodo successivo é il piú vicino a *i* tra quelli non ancora visitati.

Esempio

	1	2	3	4	5
1	–	33	13	25	33
2	33	–	46	58	76
3	39	33	–	12	30
4	35	29	12	–	23
5	60	54	30	23	–

Algoritmo del nodo piú vicino

Parti da un nodo *i*. Il nodo successivo é il piú vicino a *i* tra quelli non ancora visitati.

Esempio

	1	2	3	4	5
1	–	33	13	25	33
2	33	–	46	58	76
3	39	33	–	12	30
4	35	29	12	–	23
5	60	54	30	23	–

Partendo dal nodo 1 si ottiene il ciclo 1–3–4–5–2–1 di costo 135.

Algoritmo del nodo piú vicino

Parti da un nodo i. Il nodo successivo é il piú vicino a i tra quelli non ancora visitati.

Esempio

	1	2	3	4	5
1	–	33	13	25	33
2	33	–	46	58	76
3	39	33	–	12	30
4	35	29	12	–	23
5	60	54	30	23	–

Partendo dal nodo 1 si ottiene il ciclo 1–3–4–5–2–1 di costo 135.

Partendo dal nodo 5 si ottiene il ciclo 5–4–3–2–1–5 di costo 134.

Metodi euristici per valutazioni superiori

Algoritmi di inserimento

Costruisco un ciclo su un sottoinsieme di nodi. Estendo questo ciclo inserendo uno alla volta i nodi rimanenti fino ad inserire tutti i nodi.

Algoritmi di inserimento

Costruisco un ciclo su un sottoinsieme di nodi. Estendo questo ciclo inserendo uno alla volta i nodi rimanenti fino ad inserire tutti i nodi.

Questo schema dipende da:

- come costruisco il ciclo iniziale: ciclo qualsiasi, ciclo sui 3 nodi che formano il triangolo piú grande, ciclo che segue l'involucro convesso dei nodi (quando $c_{ij} =$ distanza euclidea tra i e j), ...

Metodi euristici per valutazioni superiori

Algoritmi di inserimento

Costruisco un ciclo su un sottoinsieme di nodi. Estendo questo ciclo inserendo uno alla volta i nodi rimanenti fino ad inserire tutti i nodi.

Questo schema dipende da:

- come costruisco il ciclo iniziale: ciclo qualsiasi, ciclo sui 3 nodi che formano il triangolo piú grande, ciclo che segue l'involucro convesso dei nodi (quando $c_{ij} =$ distanza euclidea tra i e j), ...
- come scelgo il nodo da inserire: il piú vicino al ciclo, il piú lontano dal ciclo, quello il cui inserimento causa il minimo incremento nella lunghezza del ciclo, ...

Algoritmi di inserimento

Costruisco un ciclo su un sottoinsieme di nodi. Estendo questo ciclo inserendo uno alla volta i nodi rimanenti fino ad inserire tutti i nodi.

Questo schema dipende da:

- come costruisco il ciclo iniziale: ciclo qualsiasi, ciclo sui 3 nodi che formano il triangolo piú grande, ciclo che segue l'involucro convesso dei nodi (quando $c_{ij} =$ distanza euclidea tra i e j), ...
- come scelgo il nodo da inserire: il piú vicino al ciclo, il piú lontano dal ciclo, quello il cui inserimento causa il minimo incremento nella lunghezza del ciclo, ...
- dove inserisco il nodo scelto: di solito é inserito nel punto che causa il minimo incremento nella lunghezza del ciclo

Esempio

	1	2	3	4	5
1	–	33	13	25	33
2	33	–	46	58	76
3	39	33	–	12	30
4	35	29	12	–	23
5	60	54	30	23	–

Esempio

	1	2	3	4	5
1	–	33	13	25	33
2	33	–	46	58	76
3	39	33	–	12	30
4	35	29	12	–	23
5	60	54	30	23	–

Scelgo un ciclo: 1-2-3-1.

Esempio

	1	2	3	4	5
1	–	33	13	25	33
2	33	–	46	58	76
3	39	33	–	12	30
4	35	29	12	–	23
5	60	54	30	23	–

Scelgo un ciclo: 1-2-3-1.
Il nodo 4 ha distanza 12 dal ciclo, mentre il nodo 5 ha distanza 30. Scelgo il nodo piú vicino al ciclo: 4.

Esempio

	1	2	3	4	5
1	–	33	13	25	33
2	33	–	46	58	76
3	39	33	–	12	30
4	35	29	12	–	23
5	60	54	30	23	–

Scelgo un ciclo: 1-2-3-1.
Il nodo 4 ha distanza 12 dal ciclo, mentre il nodo 5 ha distanza 30. Scelgo il nodo piú vicino al ciclo: 4.
Dove inserisco il nodo 4?

Esempio

	1	2	3	4	5
1	–	33	13	25	33
2	33	–	46	58	76
3	39	33	–	12	30
4	35	29	12	–	23
5	60	54	30	23	–

Scelgo un ciclo: 1-2-3-1.
Il nodo 4 ha distanza 12 dal ciclo, mentre il nodo 5 ha distanza 30. Scelgo il nodo piú vicino al ciclo: 4.
Dove inserisco il nodo 4?
Se inserisco 4 tra 1 e 2, la lunghezza del ciclo aumenta di $25 + 29 - 33 = 21$

Esempio

	1	2	3	4	5
1	–	33	13	25	33
2	33	–	46	58	76
3	39	33	–	12	30
4	35	29	12	–	23
5	60	54	30	23	–

Scelgo un ciclo: 1-2-3-1.
Il nodo 4 ha distanza 12 dal ciclo, mentre il nodo 5 ha distanza 30. Scelgo il nodo piú vicino al ciclo: 4.
Dove inserisco il nodo 4?
Se inserisco 4 tra 1 e 2, la lunghezza del ciclo aumenta di $25 + 29 - 33 = 21$
Se inserisco 4 tra 2 e 3, la lunghezza del ciclo aumenta di $58 + 12 - 46 = 24$

Esempio

	1	2	3	4	5
1	–	33	13	25	33
2	33	–	46	58	76
3	39	33	–	12	30
4	35	29	12	–	23
5	60	54	30	23	–

Scelgo un ciclo: 1-2-3-1.
Il nodo 4 ha distanza 12 dal ciclo, mentre il nodo 5 ha distanza 30. Scelgo il nodo piú vicino al ciclo: 4.
Dove inserisco il nodo 4?
Se inserisco 4 tra 1 e 2, la lunghezza del ciclo aumenta di $25 + 29 - 33 = 21$
Se inserisco 4 tra 2 e 3, la lunghezza del ciclo aumenta di $58 + 12 - 46 = 24$
Se inserisco 4 tra 3 e 1, la lunghezza del ciclo aumenta di $12 + 35 - 39 = 8$

Esempio

	1	2	3	4	5
1	–	33	13	25	33
2	33	–	46	58	76
3	39	33	–	12	30
4	35	29	12	–	23
5	60	54	30	23	–

Scelgo un ciclo: 1-2-3-1.
Il nodo 4 ha distanza 12 dal ciclo, mentre il nodo 5 ha distanza 30. Scelgo il nodo piú vicino al ciclo: 4.
Dove inserisco il nodo 4?
Se inserisco 4 tra 1 e 2, la lunghezza del ciclo aumenta di $25 + 29 - 33 = 21$
Se inserisco 4 tra 2 e 3, la lunghezza del ciclo aumenta di $58 + 12 - 46 = 24$
Se inserisco 4 tra 3 e 1, la lunghezza del ciclo aumenta di $12 + 35 - 39 = 8$
Quindi inserisco il nodo 4 tra 3 e 1. Il ciclo diventa 1-2-3-4-1.

Esempio

	1	2	3	4	5
1	–	33	13	25	33
2	33	–	46	58	76
3	39	33	–	12	30
4	35	29	12	–	23
5	60	54	30	23	–

Scelgo un ciclo: 1-2-3-1.
Il nodo 4 ha distanza 12 dal ciclo, mentre il nodo 5 ha distanza 30. Scelgo il nodo piú vicino al ciclo: 4.
Dove inserisco il nodo 4?
Se inserisco 4 tra 1 e 2, la lunghezza del ciclo aumenta di $25 + 29 - 33 = 21$
Se inserisco 4 tra 2 e 3, la lunghezza del ciclo aumenta di $58 + 12 - 46 = 24$
Se inserisco 4 tra 3 e 1, la lunghezza del ciclo aumenta di $12 + 35 - 39 = 8$
Quindi inserisco il nodo 4 tra 3 e 1. Il ciclo diventa 1-2-3-4-1.
Dove inserisco il nodo 5? Conviene inserirlo tra 3 e 4, il ciclo hamiltoniano é 1-2-3-5-4-1 di costo 167.

Metodi euristici per valutazioni superiori

Algoritmo basato sulla soluzione ottima del rilassamento.

Algoritmo basato sulla soluzione ottima del rilassamento.

Algoritmo delle toppe (*patching*)

1. L'assegnamento di costo minimo é formato da una famiglia di cicli orientati $F = \{C_1, \ldots, C_p\}$.

Algoritmo basato sulla soluzione ottima del rilassamento.

Algoritmo delle toppe (*patching*)

1. L'assegnamento di costo minimo é formato da una famiglia di cicli orientati $F = \{C_1, \dots, C_p\}$.
2. Per ogni coppia di cicli $C_i, C_j \in F$, calcola l'incremento di costo γ_{ij} corrispondente alla fusione di C_i e C_j nel modo piú conveniente possibile.

Algoritmo basato sulla soluzione ottima del rilassamento.

Algoritmo delle toppe (*patching*)

1. L'assegnamento di costo minimo é formato da una famiglia di cicli orientati $F = \{C_1, \ldots, C_p\}$.
2. Per ogni coppia di cicli $C_i, C_j \in F$, calcola l'incremento di costo γ_{ij} corrispondente alla fusione di C_i e C_j nel modo piú conveniente possibile.
3. Effettua la fusione dei due cicli C_i e C_j ai quali corrisponde il minimo valore di γ_{ij}. Aggiorna F.

Algoritmo basato sulla soluzione ottima del rilassamento.

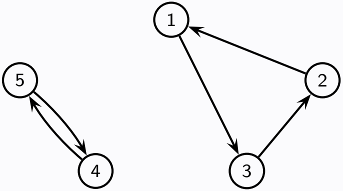
Algoritmo delle toppe (*patching*)

1. L'assegnamento di costo minimo é formato da una famiglia di cicli orientati $F = \{C_1, \dots, C_p\}$.
2. Per ogni coppia di cicli $C_i, C_j \in F$, calcola l'incremento di costo γ_{ij} corrispondente alla fusione di C_i e C_j nel modo piú conveniente possibile.
3. Effettua la fusione dei due cicli C_i e C_j ai quali corrisponde il minimo valore di γ_{ij}. Aggiorna F.
4. Se F contiene un solo ciclo allora STOP altrimenti torna al passo 2.

Metodi euristici per valutazioni superiori

Esempio

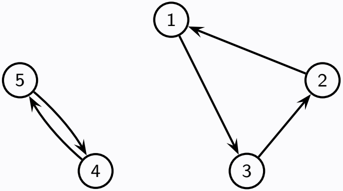
	1	2	3	4	5
1	–	33	13	25	33
2	33	–	46	58	76
3	39	33	–	12	30
4	35	29	12	–	23
5	60	54	30	23	–



Metodi euristici per valutazioni superiori

Esempio

	1	2	3	4	5
1	–	33	13	25	33
2	33	–	46	58	76
3	39	33	–	12	30
4	35	29	12	–	23
5	60	54	30	23	–

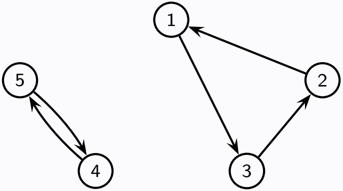


La soluzione ottima del rilassamento é formata da due cicli: 1–3–2–1 e 4–5–4, con $v_I(P) = 125$.

Metodi euristici per valutazioni superiori

Esempio

	1	2	3	4	5
1	–	33	13	25	33
2	33	–	46	58	76
3	39	33	–	12	30
4	35	29	12	–	23
5	60	54	30	23	–

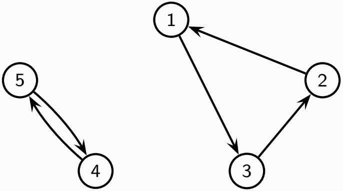


La soluzione ottima del rilassamento é formata da due cicli: 1–3–2–1 e 4–5–4, con $v_I(P) = 125$.
Le possibili fusioni dei due cicli sono le seguenti:

Metodi euristici per valutazioni superiori

Esempio

	1	2	3	4	5
1	–	33	13	25	33
2	33	–	46	58	76
3	39	33	–	12	30
4	35	29	12	–	23
5	60	54	30	23	–



La soluzione ottima del rilassamento é formata da due cicli: 1–3–2–1 e 4–5–4, con $v_I(P) = 125$.

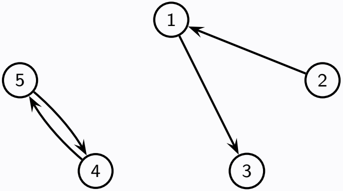
Le possibili fusioni dei due cicli sono le seguenti:

sostituire gli archi	con gli archi	si ottiene il ciclo	di costo
$(1,3)$ e $(4,5)$	$(1,5)$ e $(4,3)$	1-5-4-3-2-1	134
$(1,3)$ e $(5,4)$	$(1,4)$ e $(5,3)$	1-4-5-3-2-1	144
$(2,1)$ e $(4,5)$	$(2,5)$ e $(4,1)$	1-3-2-5-4-1	180
$(2,1)$ e $(5,4)$	$(2,4)$ e $(5,1)$	1-3-2-4-5-1	187
$(3,2)$ e $(4,5)$	$(3,5)$ e $(4,2)$	1-3-5-4-2-1	128
$(3,2)$ e $(5,4)$	$(3,4)$ e $(5,2)$	1-3-4-5-2-1	135

Metodi euristici per valutazioni superiori

Esempio

	1	2	3	4	5
1	–	33	13	25	33
2	33	–	46	58	76
3	39	33	–	12	30
4	35	29	12	–	23
5	60	54	30	23	–



La soluzione ottima del rilassamento é formata da due cicli: 1–3–2–1 e 4–5–4, con $v_I(P) = 125$.

Le possibili fusioni dei due cicli sono le seguenti:

sostituire gli archi	con gli archi	si ottiene il ciclo	di costo
$(1,3)$ e $(4,5)$	$(1,5)$ e $(4,3)$	1-5-4-3-2-1	134
$(1,3)$ e $(5,4)$	$(1,4)$ e $(5,3)$	1-4-5-3-2-1	144
$(2,1)$ e $(4,5)$	$(2,5)$ e $(4,1)$	1-3-2-5-4-1	180
$(2,1)$ e $(5,4)$	$(2,4)$ e $(5,1)$	1-3-2-4-5-1	187
$(3,2)$ e $(4,5)$	$(3,5)$ e $(4,2)$	1-3-5-4-2-1	128
$(3,2)$ e $(5,4)$	$(3,4)$ e $(5,2)$	1-3-4-5-2-1	135

La fusione piú conveniente trova il ciclo 1-3-5-4-2-1 di costo 128.

TSP simmetrico

Il modello per il TSP asimmetrico é valido ovviamente anche per il TSP simmetrico.

TSP simmetrico

Il modello per il TSP asimmetrico é valido ovviamente anche per il TSP simmetrico.
Vediamo ora un altro modello valido solo per il TSP simmetrico.

TSP simmetrico

Il modello per il TSP asimmetrico é valido ovviamente anche per il TSP simmetrico.
Vediamo ora un altro modello valido solo per il TSP simmetrico.
Nel TSP simmetrico possiamo supporre che il grafo (N, A) sia non orientato.

Il modello per il TSP asimmetrico é valido ovviamente anche per il TSP simmetrico.
Vediamo ora un altro modello valido solo per il TSP simmetrico.
Nel TSP simmetrico possiamo supporre che il grafo (N, A) sia non orientato.
Quanti sono i cicli hamiltoniani?

TSP simmetrico

Il modello per il TSP asimmetrico é valido ovviamente anche per il TSP simmetrico.
Vediamo ora un altro modello valido solo per il TSP simmetrico.
Nel TSP simmetrico possiamo supporre che il grafo (N, A) sia non orientato.
Quanti sono i cicli hamiltoniani? $n!/2$

Modello

Variabili: $x_{ij} = \begin{cases} 1 & \text{se } \{i,j\} \in \text{ciclo hamiltoniano}, \\ 0 & \text{altrimenti}, \end{cases}$ con $i < j$.

TSP simmetrico

Modello

Variabili: $x_{ij} = \begin{cases} 1 & \text{se } \{i,j\} \in \text{ciclo hamiltoniano,} \\ 0 & \text{altrimenti,} \end{cases}$ con $i < j$.

$$\min \sum_{i \in N} \sum_{\substack{j \in N \\ j > i}} c_{ij} x_{ij}$$

TSP simmetrico

Modello

Variabili: $x_{ij} = \begin{cases} 1 & \text{se } \{i,j\} \in \text{ciclo hamiltoniano,} \\ 0 & \text{altrimenti,} \end{cases}$ con $i < j$.

$$\min \sum_{i \in N} \sum_{\substack{j \in N \\ j > i}} c_{ij} x_{ij}$$

$$\sum_{\substack{j \in N \\ j > i}} x_{ij} + \sum_{\substack{j \in N \\ j < i}} x_{ji} = 2 \qquad \forall \ i \in N \tag{3}$$

TSP simmetrico

Modello

Variabili: $x_{ij} = \begin{cases} 1 & \text{se } \{i,j\} \in \text{ciclo hamiltoniano,} \\ 0 & \text{altrimenti,} \end{cases}$ con $i < j$.

$$\min \sum_{i \in N} \sum_{\substack{j \in N \\ j > i}} c_{ij}\, x_{ij}$$

$$\sum_{\substack{j \in N \\ j > i}} x_{ij} + \sum_{\substack{j \in N \\ j < i}} x_{ji} = 2 \qquad \forall\ i \in N \tag{3}$$

$$\sum_{i \in S} \sum_{\substack{j \notin S \\ j > i}} x_{ij} + \sum_{i \notin S} \sum_{\substack{j \in S \\ j > i}} x_{ij} \geq 1 \qquad \forall\ S \subset N, \tag{4}$$

$$x_{ij} \in \{0,1\} \quad \forall\ i,j \in N, \quad i < j$$

TSP simmetrico

Modello

Variabili: $x_{ij} = \begin{cases} 1 & \text{se } \{i,j\} \in \text{ciclo hamiltoniano,} \\ 0 & \text{altrimenti,} \end{cases}$ con $i < j$.

$$\min \sum_{i \in N} \sum_{\substack{j \in N \\ j > i}} c_{ij} x_{ij}$$

$$\sum_{\substack{j \in N \\ j > i}} x_{ij} + \sum_{\substack{j \in N \\ j < i}} x_{ji} = 2 \qquad \forall\ i \in N \tag{3}$$

$$\sum_{i \in S} \sum_{\substack{j \notin S \\ j > i}} x_{ij} + \sum_{i \notin S} \sum_{\substack{j \in S \\ j > i}} x_{ij} \geq 1 \qquad \forall\ S \subset N, \tag{4}$$

$$x_{ij} \in \{0,1\} \quad \forall\ i,j \in N, \quad i < j$$

(3): due archi incidenti su ogni nodo (cioé ogni nodo ha grado 2).

Modello

Variabili: $x_{ij} = \begin{cases} 1 & \text{se } \{i,j\} \in \text{ciclo hamiltoniano}, \\ 0 & \text{altrimenti}, \end{cases}$ con $i < j$.

$$\min \sum_{i \in N} \sum_{\substack{j \in N \\ j > i}} c_{ij} x_{ij}$$

$$\sum_{\substack{j \in N \\ j > i}} x_{ij} + \sum_{\substack{j \in N \\ j < i}} x_{ji} = 2 \qquad \forall\ i \in N \tag{3}$$

$$\sum_{i \in S} \sum_{\substack{j \notin S \\ j > i}} x_{ij} + \sum_{i \notin S} \sum_{\substack{j \in S \\ j > i}} x_{ij} \geq 1 \qquad \forall\ S \subset N, \tag{4}$$

$$x_{ij} \in \{0,1\} \quad \forall\ i,j \in N, \quad i < j$$

(3): due archi incidenti su ogni nodo (cioé ogni nodo ha grado 2).
(4): eliminazione di sottocicli.

Rilassamenti

- Assegnamento di costo minimo (come nel caso asimmetrico)

Rilassamenti

- Assegnamento di costo minimo (come nel caso asimmetrico)
- r-albero di costo minimo

Rilassamenti

- Assegnamento di costo minimo (come nel caso asimmetrico)
- r-albero di costo minimo

Fissiamo un nodo r nel grafo.

- Assegnamento di costo minimo (come nel caso asimmetrico)
- r-albero di costo minimo

Fissiamo un nodo r nel grafo. Un r–albero é un insieme di n archi di cui

2 sono incidenti sul nodo r.

- Assegnamento di costo minimo (come nel caso asimmetrico)
- r-albero di costo minimo

Fissiamo un nodo r nel grafo. Un r–albero é un insieme di n archi di cui

2 sono incidenti sul nodo r.

$n-2$ formano un albero di copertura sui nodi diversi da r.

Rilassamenti

- Assegnamento di costo minimo (come nel caso asimmetrico)
- r-albero di costo minimo

Fissiamo un nodo r nel grafo. Un r–albero é un insieme di n archi di cui

2 sono incidenti sul nodo r.

$n-2$ formano un albero di copertura sui nodi diversi da r.

Ogni ciclo hamiltoniano é un r–albero.

- Assegnamento di costo minimo (come nel caso asimmetrico)
- r-albero di costo minimo

Fissiamo un nodo r nel grafo. Un r–albero é un insieme di n archi di cui

2 sono incidenti sul nodo r.

$n-2$ formano un albero di copertura sui nodi diversi da r.

Ogni ciclo hamiltoniano é un r–albero.

Il problema dell'r–albero di costo minimo é facile da risolvere: basta trovare

2 archi di costo minimo incidenti sul nodo r

- Assegnamento di costo minimo (come nel caso asimmetrico)
- r-albero di costo minimo

Fissiamo un nodo r nel grafo. Un r–albero é un insieme di n archi di cui

2 sono incidenti sul nodo r.

$n-2$ formano un albero di copertura sui nodi diversi da r.

Ogni ciclo hamiltoniano é un r–albero.

Il problema dell'r–albero di costo minimo é facile da risolvere: basta trovare

2 archi di costo minimo incidenti sul nodo r

un albero di copertura di costo minimo sui nodi diversi da r

Problema

Dato un grafo (N, A) non orientato e connesso, con costi c_{ij} sugli archi, trovare un albero (insieme di archi connesso e privo di cicli) di copertura (che connetta tutti i nodi) di costo minimo.

Albero di copertura di costo minimo

Problema

Dato un grafo (N, A) non orientato e connesso, con costi c_{ij} sugli archi, trovare un albero (insieme di archi connesso e privo di cicli) di copertura (che connetta tutti i nodi) di costo minimo.

Algoritmo di Kruskal

È un algoritmo di tipo greedy:

Albero di copertura di costo minimo

Problema

Dato un grafo (N, A) non orientato e connesso, con costi c_{ij} sugli archi, trovare un albero (insieme di archi connesso e privo di cicli) di copertura (che connetta tutti i nodi) di costo minimo.

Algoritmo di Kruskal

È un algoritmo di tipo greedy:

- crea la soluzione aggiungendo un arco per volta.

Albero di copertura di costo minimo

Problema

Dato un grafo (N, A) non orientato e connesso, con costi c_{ij} sugli archi, trovare un albero (insieme di archi connesso e privo di cicli) di copertura (che connetta tutti i nodi) di costo minimo.

Algoritmo di Kruskal

È un algoritmo di tipo greedy:

- crea la soluzione aggiungendo un arco per volta.
- non ridiscute decisioni giá prese.

Problema

Dato un grafo (N, A) non orientato e connesso, con costi c_{ij} sugli archi, trovare un albero (insieme di archi connesso e privo di cicli) di copertura (che connetta tutti i nodi) di costo minimo.

Algoritmo di Kruskal

È un algoritmo di tipo greedy:

- crea la soluzione aggiungendo un arco per volta.
- non ridiscute decisioni giá prese.

1. Ordina gli archi $a_1, \ldots, a_m$ in ordine crescente di costo. $T = \emptyset$, $h = 1$.

Albero di copertura di costo minimo

Problema

Dato un grafo (N, A) non orientato e connesso, con costi c_{ij} sugli archi, trovare un albero (insieme di archi connesso e privo di cicli) di copertura (che connetta tutti i nodi) di costo minimo.

Algoritmo di Kruskal

È un algoritmo di tipo greedy:

- crea la soluzione aggiungendo un arco per volta.
- non ridiscute decisioni giá prese.

1. Ordina gli archi $a_1, \ldots, a_m$ in ordine crescente di costo. $T = \emptyset$, $h = 1$.
2. **se** $T \cup \{a_h\}$ non contiene un ciclo **allora** inserisci a_h in T

Problema

Dato un grafo (N, A) non orientato e connesso, con costi c_{ij} sugli archi, trovare un albero (insieme di archi connesso e privo di cicli) di copertura (che connetta tutti i nodi) di costo minimo.

Algoritmo di Kruskal

È un algoritmo di tipo greedy:

- crea la soluzione aggiungendo un arco per volta.
- non ridiscute decisioni giá prese.

1. Ordina gli archi $a_1, \dots, a_m$ in ordine crescente di costo. $T = \emptyset$, $h = 1$.
2. **se** $T \cup \{a_h\}$ non contiene un ciclo **allora** inserisci a_h in T
3. **se** $|T| = n - 1$ **allora** STOP

Albero di copertura di costo minimo

Problema

Dato un grafo (N, A) non orientato e connesso, con costi c_{ij} sugli archi, trovare un albero (insieme di archi connesso e privo di cicli) di copertura (che connetta tutti i nodi) di costo minimo.

Algoritmo di Kruskal

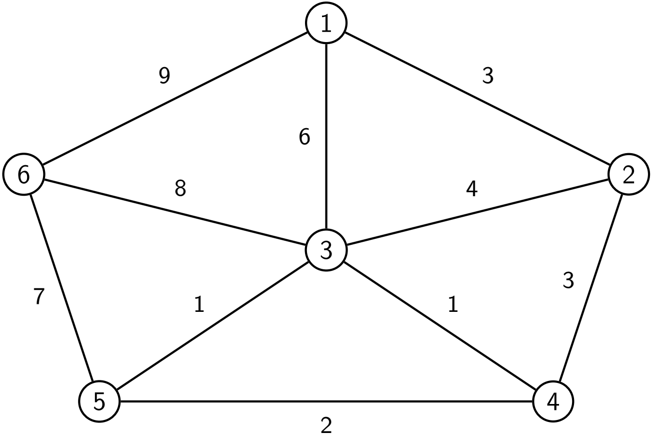
È un algoritmo di tipo greedy:

- crea la soluzione aggiungendo un arco per volta.
- non ridiscute decisioni giá prese.

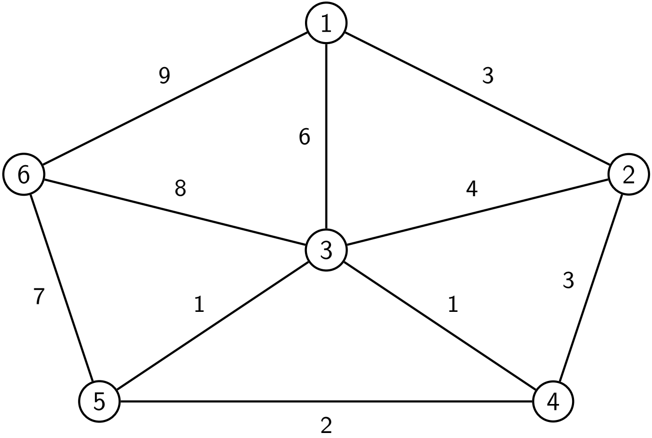
1. Ordina gli archi $a_1, \ldots, a_m$ in ordine crescente di costo. $T = \emptyset$, $h = 1$.
2. **se** $T \cup \{a_h\}$ non contiene un ciclo **allora** inserisci a_h in T
3. **se** $|T| = n - 1$ **allora** STOP

   altrimenti $h = h + 1$ e torna al passo 2

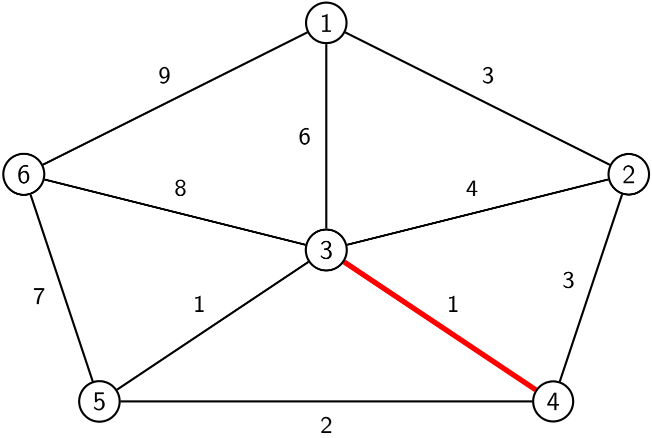
Algoritmo di Kruskal: esempio



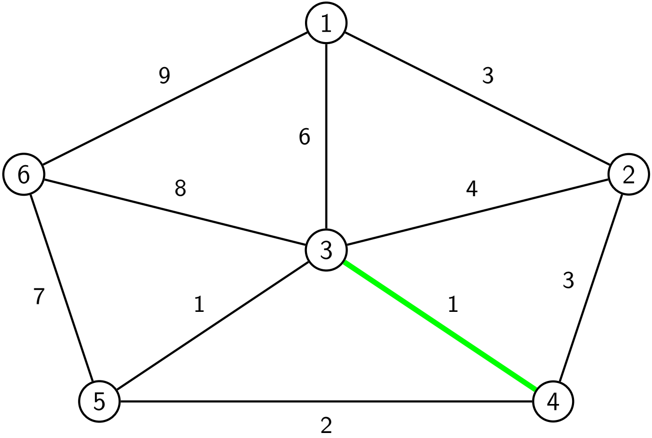
Algoritmo di Kruskal: esempio



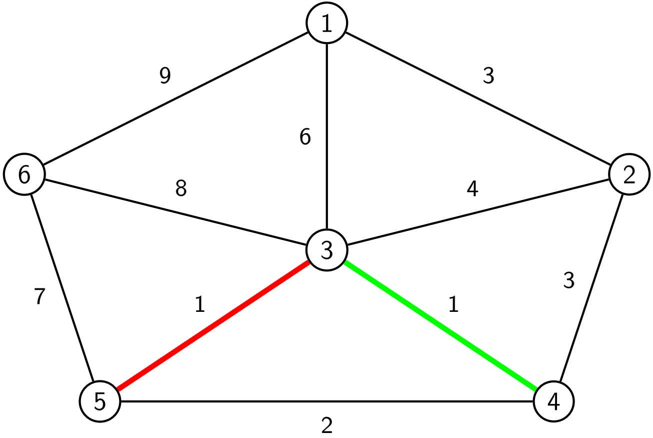
Algoritmo di Kruskal: esempio



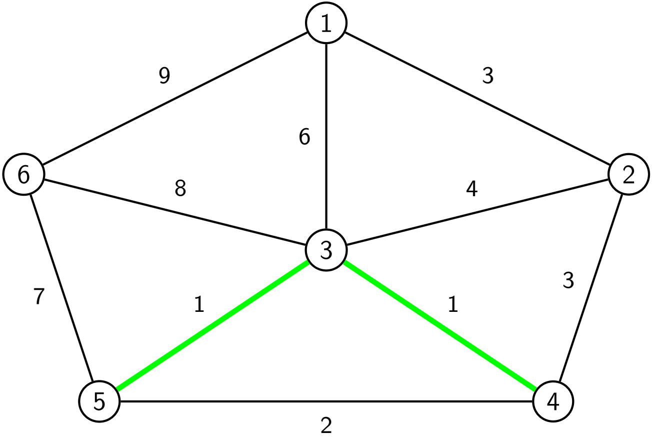
Algoritmo di Kruskal: esempio



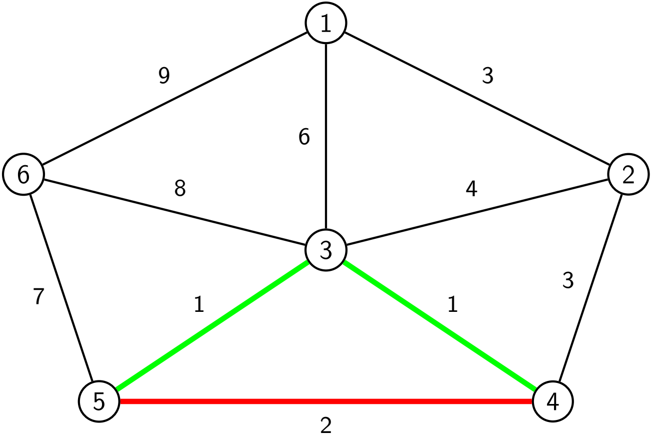
Algoritmo di Kruskal: esempio



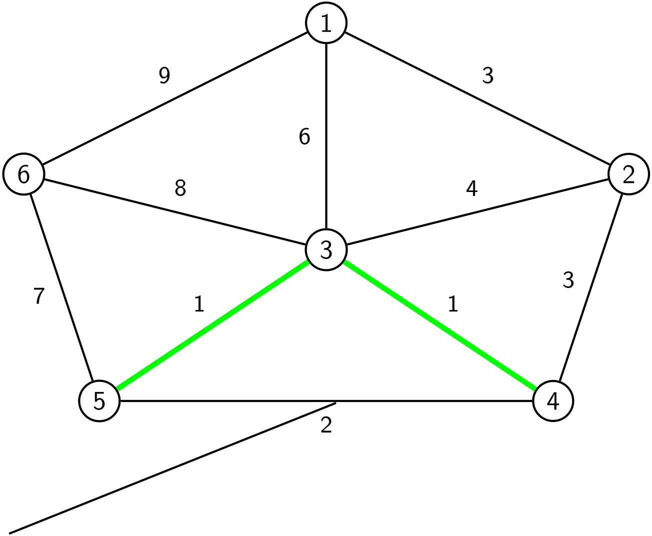
Algoritmo di Kruskal: esempio



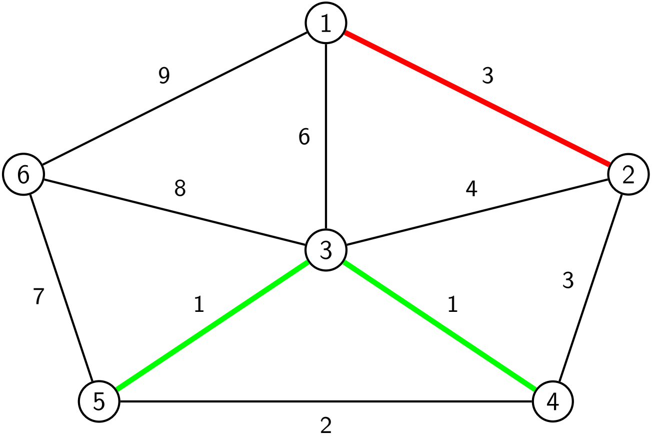
Algoritmo di Kruskal: esempio

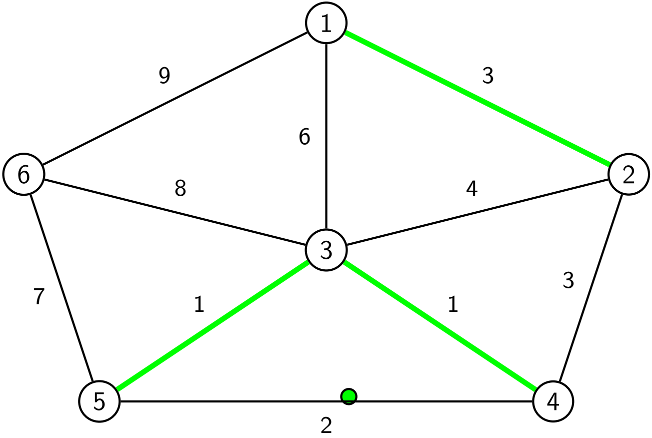


Algoritmo di Kruskal: esempio

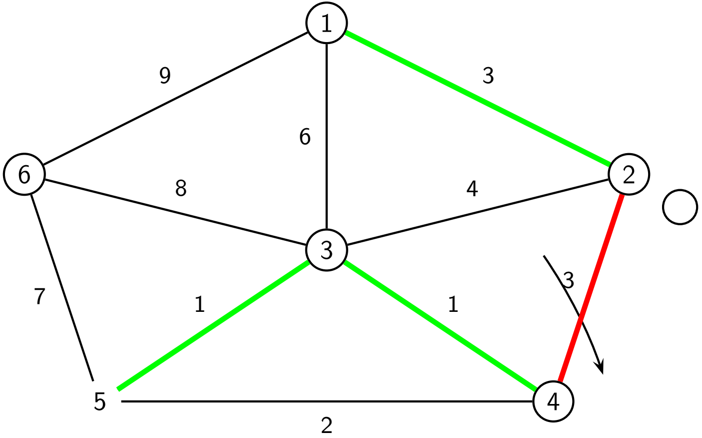


Algoritmo di Kruskal: esempio

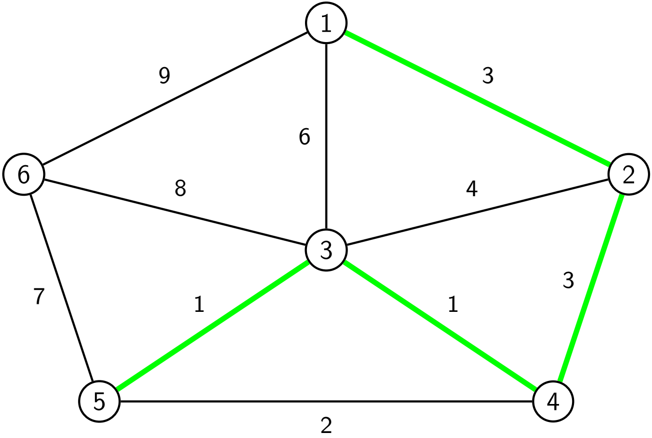



1
2
3
4
5
6
9
3
6
8
4
7
1
1
3
2

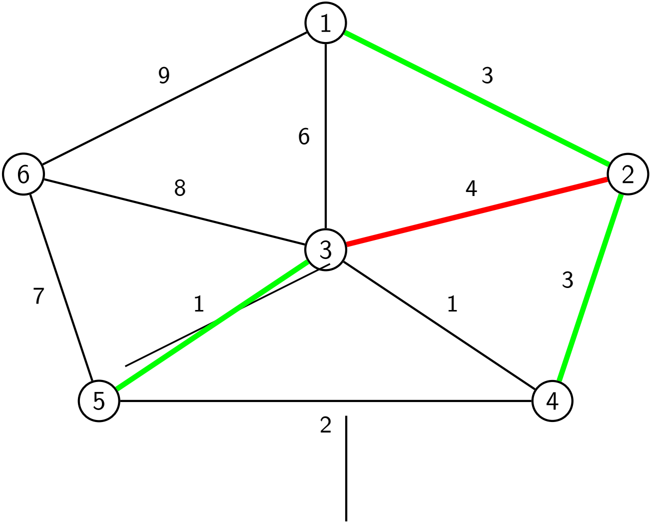
Algoritmo di Kruskal: esempio



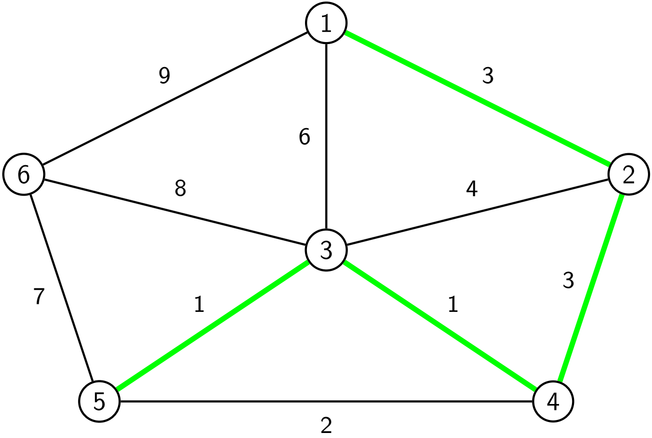
Algoritmo di Kruskal: esempio



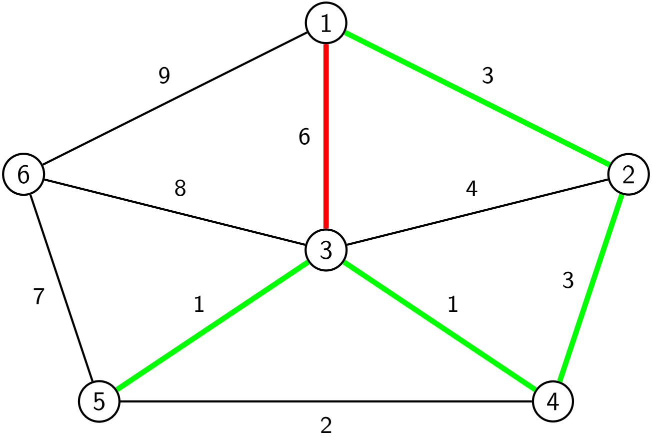
Algoritmo di Kruskal: esempio



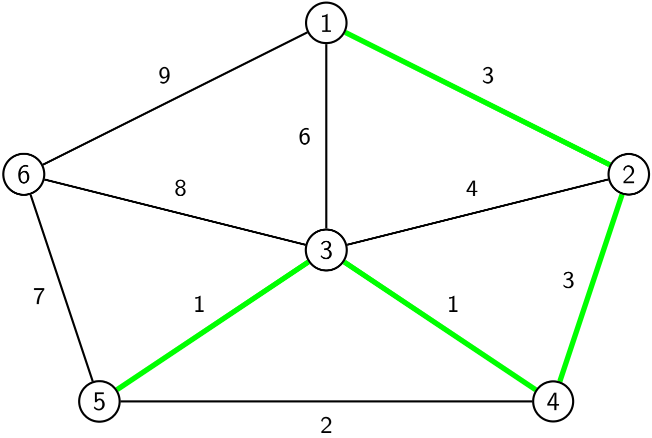
Algoritmo di Kruskal: esempio



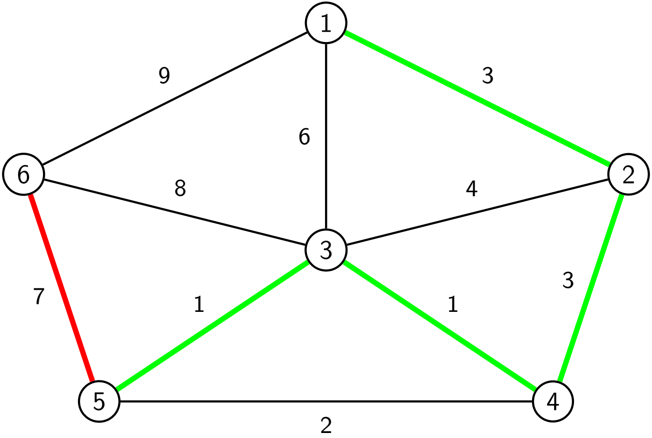
Algoritmo di Kruskal: esempio



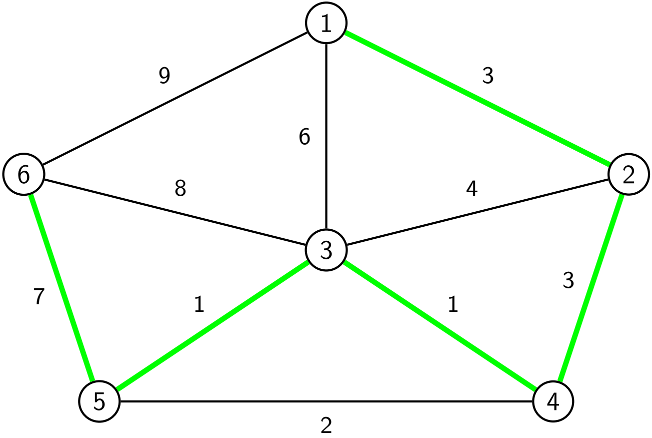
Algoritmo di Kruskal: esempio



Algoritmo di Kruskal: esempio



Algoritmo di Kruskal: esempio



$\{1,2\}$	$\{1,3\}$	$\{1,6\}$	$\{2,3\}$	$\{2,4\}$	$\{3,4\}$	$\{3,5\}$	$\{3,6\}$	$\{4,5\}$	$\{5,6\}$
3	6	9	4	3	1	1	8	2	7

$\{3,4\}$	$\{3,5\}$	$\{4,5\}$	$\{1,2\}$	$\{2,4\}$	$\{2,3\}$	$\{1,3\}$	$\{5,6\}$	$\{3,6\}$	$\{1,6\}$
1	1	2	3	3	4	6	7	8	9

archi in ordine crescente di costo

$\{3,4\}$	$\{3,5\}$	$\{4,5\}$	$\{1,2\}$	$\{2,4\}$	$\{2,3\}$	$\{1,3\}$	$\{5,6\}$	$\{3,6\}$	$\{1,6\}$
1	1	2	3	3	4	6	7	8	9

$\{3,4\}$ non forma un ciclo con $T = \emptyset$

$\{3,4\}$	$\{3,5\}$	$\{4,5\}$	$\{1,2\}$	$\{2,4\}$	$\{2,3\}$	$\{1,3\}$	$\{5,6\}$	$\{3,6\}$	$\{1,6\}$
1	1	2	3	3	4	6	7	8	9

$\{3,4\}$ non forma un ciclo con $T = \emptyset$

$\{3,4\}$	$\{3,5\}$	$\{4,5\}$	$\{1,2\}$	$\{2,4\}$	$\{2,3\}$	$\{1,3\}$	$\{5,6\}$	$\{3,6\}$	$\{1,6\}$
1	1	2	3	3	4	6	7	8	9

$\{3,5\}$ non forma un ciclo con T

$\{3,4\}$	$\{3,5\}$	$\{4,5\}$	$\{1,2\}$	$\{2,4\}$	$\{2,3\}$	$\{1,3\}$	$\{5,6\}$	$\{3,6\}$	$\{1,6\}$
1	1	2	3	3	4	6	7	8	9

$\{3,5\}$ non forma un ciclo con T

$\{3,4\}$	$\{3,5\}$	$\{4,5\}$	$\{1,2\}$	$\{2,4\}$	$\{2,3\}$	$\{1,3\}$	$\{5,6\}$	$\{3,6\}$	$\{1,6\}$
1	1	2	3	3	4	6	7	8	9

$\{4,5\}$ forma un ciclo con T!

$\{3,4\}$	$\{3,5\}$	$\{4,5\}$	$\{1,2\}$	$\{2,4\}$	$\{2,3\}$	$\{1,3\}$	$\{5,6\}$	$\{3,6\}$	$\{1,6\}$
1	1	2	3	3	4	6	7	8	9

$\{4,5\}$ forma un ciclo con T!

$\{3,4\}$	$\{3,5\}$	$\{4,5\}$	$\{1,2\}$	$\{2,4\}$	$\{2,3\}$	$\{1,3\}$	$\{5,6\}$	$\{3,6\}$	$\{1,6\}$
1	1	2	3	3	4	6	7	8	9

$\{1,2\}$ non forma un ciclo con T

$\{3,4\}$	$\{3,5\}$	$\{4,5\}$	$\{1,2\}$	$\{2,4\}$	$\{2,3\}$	$\{1,3\}$	$\{5,6\}$	$\{3,6\}$	$\{1,6\}$
1	1	2	3	3	4	6	7	8	9

$\{1,2\}$ non forma un ciclo con T

$\{3,4\}$	$\{3,5\}$	$\{4,5\}$	$\{1,2\}$	$\{2,4\}$	$\{2,3\}$	$\{1,3\}$	$\{5,6\}$	$\{3,6\}$	$\{1,6\}$
1	1	2	3	3	4	6	7	8	9

$\{2,4\}$ non forma un ciclo con T

$\{3,4\}$	$\{3,5\}$	$\{4,5\}$	$\{1,2\}$	$\{2,4\}$	$\{2,3\}$	$\{1,3\}$	$\{5,6\}$	$\{3,6\}$	$\{1,6\}$
1	1	2	3	3	4	6	7	8	9

$\{2,4\}$ non forma un ciclo con T

$\{3,4\}$	$\{3,5\}$	$\{4,5\}$	$\{1,2\}$	$\{2,4\}$	$\{2,3\}$	$\{1,3\}$	$\{5,6\}$	$\{3,6\}$	$\{1,6\}$
1	1	2	3	3	4	6	7	8	9

$\{2,3\}$ forma un ciclo con T!

$\{3,4\}$	$\{3,5\}$	$\{4,5\}$	$\{1,2\}$	$\{2,4\}$	$\{2,3\}$	$\{1,3\}$	$\{5,6\}$	$\{3,6\}$	$\{1,6\}$
1	1	2	3	3	4	6	7	8	9

$\{2,3\}$ forma un ciclo con T!

$\{3,4\}$	$\{3,5\}$	$\{4,5\}$	$\{1,2\}$	$\{2,4\}$	$\{2,3\}$	$\{1,3\}$	$\{5,6\}$	$\{3,6\}$	$\{1,6\}$
1	1	2	3	3	4	6	7	8	9

$\{1,3\}$ forma un ciclo con T!

$\{3,4\}$	$\{3,5\}$	$\{4,5\}$	$\{1,2\}$	$\{2,4\}$	$\{2,3\}$	$\{1,3\}$	$\{5,6\}$	$\{3,6\}$	$\{1,6\}$
1	1	2	3	3	4	6	7	8	9

$\{1,3\}$ forma un ciclo con T!

$\{3,4\}$	$\{3,5\}$	$\{4,5\}$	$\{1,2\}$	$\{2,4\}$	$\{2,3\}$	$\{1,3\}$	$\{5,6\}$	$\{3,6\}$	$\{1,6\}$
1	1	2	3	3	4	6	7	8	9

$\{5,6\}$ non forma un ciclo con T.

$\{3,4\}$	$\{3,5\}$	$\{4,5\}$	$\{1,2\}$	$\{2,4\}$	$\{2,3\}$	$\{1,3\}$	$\{5,6\}$	$\{3,6\}$	$\{1,6\}$
1	1	2	3	3	4	6	7	8	9

$\{5,6\}$ non forma un ciclo con T. STOP

Per ottenenere il rilassamento, nel modello del TSP simmetrico possiamo eliminare i vincoli in rosso.

$$
\begin{cases}
\min \; \sum_{i} \sum_{j>i} c_{ij} x_{ij} \\
\sum_{j>i} x_{ij} + \sum_{j<i} x_{ji} = 2 \qquad \forall \; i \neq r \\
\sum_{j>r} x_{rj} + \sum_{j<r} x_{jr} = 2 \\
\sum_{i \in S} \sum_{\substack{j \notin S \\ j>i}} x_{ij} + \sum_{i \notin S} \sum_{\substack{j \in S \\ j>i}} x_{ij} \geq 1 \qquad \forall \; S \subset N, \quad S \neq \emptyset, N \\
x_{ij} \in \{0,1\} \quad i < j
\end{cases}
$$

Rilassamenti: r-albero di costo minimo

Per ottenenere il rilassamento, nel modello del TSP simmetrico possiamo eliminare i vincoli in rosso.

$$\begin{cases} \min \displaystyle\sum_{i} \sum_{j>i} c_{ij} x_{ij} \\ \displaystyle\sum_{j>i} x_{ij} + \sum_{j<i} x_{ji} = 2 \qquad \forall\ i \neq r \\ \displaystyle\sum_{j>r} x_{rj} + \sum_{j<r} x_{jr} = 2 \\ \displaystyle\sum_{i\in S} \sum_{\substack{j\notin S \\ j>i}} x_{ij} + \sum_{i\notin S} \sum_{\substack{j\in S \\ j>i}} x_{ij} \geq 1 \qquad \forall\ S \subset N, \quad S \neq \emptyset, N \\ x_{ij} \in \{0,1\} \quad i<j \end{cases}$$

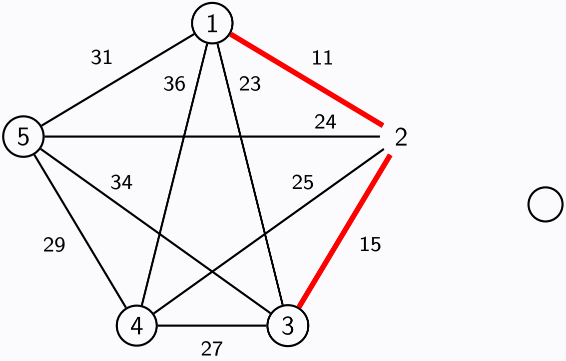
Eliminando quindi i vincoli sul grado dei nodi diversi da r si ottiene l'r-albero di costo minimo.

Esempio

Il 2–albero di costo minimo sul grafo

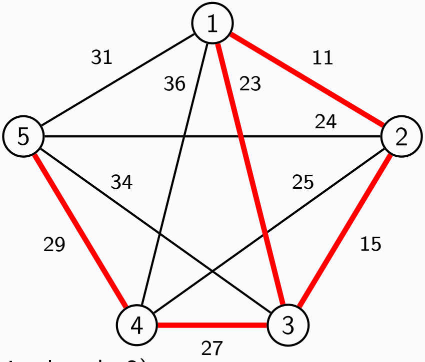
Esempio

Il 2–albero di costo minimo sul grafo



Esempio

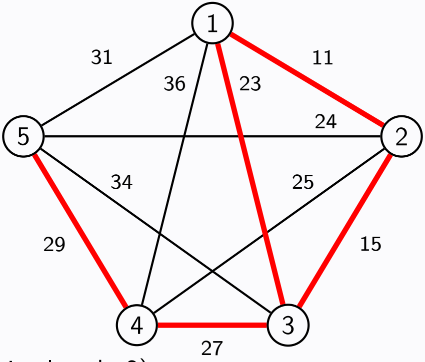
Il 2–albero di costo minimo sul grafo



$\{1, 2\}$, $\{2, 3\}$ (incidenti sul nodo 2)

Esempio

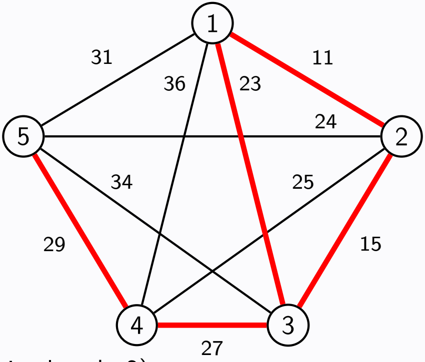
Il 2–albero di costo minimo sul grafo



$\{1, 2\}$, $\{2, 3\}$ (incidenti sul nodo 2)
$\{1, 3\}$, $\{3, 4\}$, $\{4, 5\}$ (albero di copertura sui nodi diversi da 2).

Esempio

Il 2–albero di costo minimo sul grafo



$\{1,2\}$, $\{2,3\}$ (incidenti sul nodo 2)
$\{1,3\}$, $\{3,4\}$, $\{4,5\}$ (albero di copertura sui nodi diversi da 2).
Ha costo $105 = v_I(P)$.

Metodi euristici

Per trovare un ciclo hamiltoniano possiamo usare i metodi visti per il TSP asimmetrico:

Metodi euristici

Per trovare un ciclo hamiltoniano possiamo usare i metodi visti per il TSP asimmetrico:

- metodo *greedy* sugli archi.

Metodi euristici

Per trovare un ciclo hamiltoniano possiamo usare i metodi visti per il TSP asimmetrico:

- metodo *greedy* sugli archi.
- algoritmo del nodo piú vicino.

Metodi euristici

Per trovare un ciclo hamiltoniano possiamo usare i metodi visti per il TSP asimmetrico:

- metodo *greedy* sugli archi.
- algoritmo del nodo piú vicino.
- algoritmi di inserimento.

Per trovare un ciclo hamiltoniano possiamo usare i metodi visti per il TSP asimmetrico:

- metodo *greedy* sugli archi.
- algoritmo del nodo piú vicino.
- algoritmi di inserimento.

Dopo aver trovato una soluzione ammissibile, provo a migliorarla.

Per trovare un ciclo hamiltoniano possiamo usare i metodi visti per il TSP asimmetrico:

- metodo *greedy* sugli archi.
- algoritmo del nodo piú vicino.
- algoritmi di inserimento.

Dopo aver trovato una soluzione ammissibile, provo a migliorarla.

1. Trovo una soluzione ammissibile x

Per trovare un ciclo hamiltoniano possiamo usare i metodi visti per il TSP asimmetrico:

- metodo *greedy* sugli archi.
- algoritmo del nodo piú vicino.
- algoritmi di inserimento.

Dopo aver trovato una soluzione ammissibile, provo a migliorarla.

1. Trovo una soluzione ammissibile x
2. Genero un insieme $N(x)$ di soluzioni "vicine" ad x (intorno)

Metodi euristici

Per trovare un ciclo hamiltoniano possiamo usare i metodi visti per il TSP asimmetrico:

- metodo *greedy* sugli archi.
- algoritmo del nodo piú vicino.
- algoritmi di inserimento.

Dopo aver trovato una soluzione ammissibile, provo a migliorarla.

1. Trovo una soluzione ammissibile x
2. Genero un insieme $N(x)$ di soluzioni "vicine" ad x (intorno)
3. Se in $N(x)$ esiste una soluzione ammissibile x' migliore di x
   allora $x := x'$ e torno al passo 2

Per trovare un ciclo hamiltoniano possiamo usare i metodi visti per il TSP asimmetrico:

- metodo *greedy* sugli archi.
- algoritmo del nodo piú vicino.
- algoritmi di inserimento.

Dopo aver trovato una soluzione ammissibile, provo a migliorarla.

1. Trovo una soluzione ammissibile x
2. Genero un insieme $N(x)$ di soluzioni "vicine" ad x (intorno)
3. Se in $N(x)$ esiste una soluzione ammissibile x' migliore di x
   allora $x := x'$ e torno al passo 2

4 altrimenti STOP (x é una soluzione ottima locale)

Nel caso del TSP, come definisco un intorno $N(x)$?

Nel caso del TSP, come definisco un intorno $N(x)$?

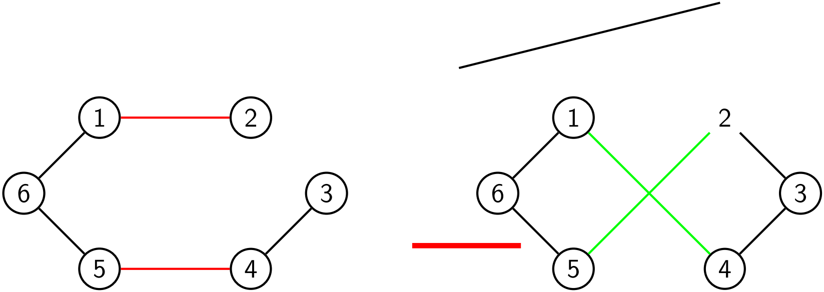
Una possibile scelta é:

$$N(x) = \{\text{cicli hamiltoniani che hanno 2 archi diversi da } x\}.$$

Nel caso del TSP, come definisco un intorno $N(x)$?

Una possibile scelta é:

$$N(x) = \{\text{cicli hamiltoniani che hanno 2 archi diversi da } x\}.$$



x $\qquad$ $x' \in N(x)$

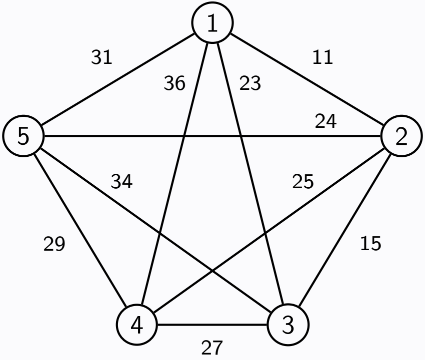
Esempio

Consideriamo il TSP sul grafo

Metodi euristici: ricerca locale

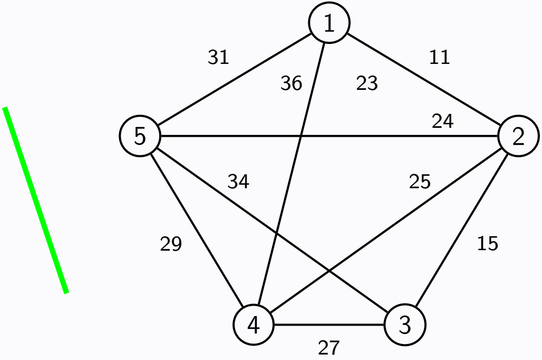
Esempio

Consideriamo il TSP sul grafo



Esempio

Consideriamo il TSP sul grafo



Effettuiamo la ricerca locale a partire dal ciclo 1-3-5-2-4 di costo 142.

Esempio

x	costo	elimino archi	inserisco archi	x'	costo
1-3-5-2-4-1	142	(1,3) (5,2)	(1,5) (3,2)	1-5-3-2-4-1	141
1-5-3-2-4-1	141	(1,5) (2,4)	(1,2) (5,4)	1-2-3-5-4-1	125
1-2-3-5-4-1	125	(1,2) (3,5)	(1,3) (2,5)	1-3-2-5-4-1	127
		(1,2) (5,4)	(1,5) (2,4)	1-5-3-2-4-1	141
		(2,3) (5,4)	(2,5) (3,4)	1-2-5-3-4-1	132
		(2,3) (4,1)	(2,4) (3,1)	1-3-5-4-2-1	122
1-3-5-4-2	122	...	...	...	...

Problemi di localizzazione

I problemi di *facility location* riguardano il posizionamento ottimale di servizi.

I problemi di *facility location* riguardano il posizionamento ottimale di servizi.
Molte applicazioni sia nel settore pubblico che in quello privato:

Problemi di localizzazione

I problemi di *facility location* riguardano il posizionamento ottimale di servizi.
Molte applicazioni sia nel settore pubblico che in quello privato:

- porti, aeroporti, scuole, ospedali, fermate autobus, stazioni metropolitana, piscine, ...

Problemi di localizzazione

I problemi di *facility location* riguardano il posizionamento ottimale di servizi.
Molte applicazioni sia nel settore pubblico che in quello privato:

- porti, aeroporti, scuole, ospedali, fermate autobus, stazioni metropolitana, piscine, ...
- stabilimenti produttivi, magazzini, punti vendita, centri di calcolo ...

Il problema di posizionare ambulanze

- Ambulanze possono essere posizionate in prefissati luoghi.

Il problema di posizionare ambulanze

- Ambulanze possono essere posizionate in prefissati luoghi.
- Malati importanti devono essere raggiunti in al piú 8 minuti.

Il problema di posizionare ambulanze

- Ambulanze possono essere posizionate in prefissati luoghi.
- Malati importanti devono essere raggiunti in al piú 8 minuti.

Problema

Come posizionare il minimo numero di ambulanze per arrivare in ogni posto in al piú 8 minuti?

Il problema di posizionare ambulanze

- Ambulanze possono essere posizionate in prefissati luoghi.
- Malati importanti devono essere raggiunti in al piú 8 minuti.

Problema

Come posizionare il minimo numero di ambulanze per arrivare in ogni posto in al piú 8 minuti?

- Dati:

$J =$ insieme di possibili posizioni delle ambulanze.

Il problema di posizionare ambulanze

- Ambulanze possono essere posizionate in prefissati luoghi.
- Malati importanti devono essere raggiunti in al piú 8 minuti.

Problema

Come posizionare il minimo numero di ambulanze per arrivare in ogni posto in al piú 8 minuti?

- Dati:

  $J =$ insieme di possibili posizioni delle ambulanze.

  $I =$ insieme delle richieste.

Il problema di posizionare ambulanze

- Ambulanze possono essere posizionate in prefissati luoghi.
- Malati importanti devono essere raggiunti in al piú 8 minuti.

Problema

Come posizionare il minimo numero di ambulanze per arrivare in ogni posto in al piú 8 minuti?

- Dati:

$J =$ insieme di possibili posizioni delle ambulanze.

$I =$ insieme delle richieste.

$$a_{ij} = \begin{cases} 1 & \text{se é possibile andare da } i \text{ a } j \text{ in al piú 8 minuti} \\ 0 & \text{altrimenti} \end{cases}$$

Struttura del modello matematico

- Decisioni (dove porre le ambulanze) $\longrightarrow$ variabili

Struttura del modello matematico

- Decisioni (dove porre le ambulanze) $\longrightarrow$ variabili

$$x_j = \begin{cases} 1 & \text{se un'ambulanza é posizionata al posto } j \\ 0 & \text{altrimenti} \end{cases}$$

Struttura del modello matematico

- Decisioni (dove porre le ambulanze) $\longrightarrow$ variabili

$$x_j = \begin{cases} 1 & \text{se un'ambulanza é posizionata al posto } j \\ 0 & \text{altrimenti} \end{cases}$$

- Il piú piccolo numero di ambulanze $\longrightarrow$ funzione obiettivo

- Richieste: tutte gli utenti raggiunti in al piú 8 minuti $\longrightarrow$ vincoli

Modello matematico

Funzione obiettivo

Minimizzare il numero di ambulanze

$$\min \sum_{j \in J} x_j$$

Modello matematico

Funzione obiettivo

Minimizzare il numero di ambulanze

$$\min \sum_{j \in J} x_j$$

Vincoli

Per ogni posizione j almeno un'ambulanza deve arrivare in al piú 8 minuti

$$\sum_{j \in J} a_{ij} x_j \geq 1, \qquad \forall\ i \in I$$

Modello matematico

Funzione obiettivo

Minimizzare il numero di ambulanze

$$\min \sum_{j \in J} x_j$$

Vincoli

Per ogni posizione j almeno un'ambulanza deve arrivare in al piú 8 minuti

$$\sum_{j \in J} a_{ij} x_j \geq 1, \qquad \forall\ i \in I$$

Dominio delle variabili

$$x_j \in \{0, 1\}, \qquad \forall\ j \in J$$

Problema

In generale, dati:
$I =$ insieme di nodi domanda

Problema

In generale, dati:
$I =$ insieme di nodi domanda
$J =$ insieme di siti candidati ad ospitare il servizio.

Problema

In generale, dati:
$I =$ insieme di nodi domanda
$J =$ insieme di siti candidati ad ospitare il servizio.
$c_j =$ costo per aprire il servizio nel sito j

Problema

In generale, dati:
$I =$ insieme di nodi domanda
$J =$ insieme di siti candidati ad ospitare il servizio.
$c_j =$ costo per aprire il servizio nel sito j
$d_{ij} =$ distanza tra nodo di domanda i e sito candidato j

Problema di copertura

Problema

In generale, dati:
$I =$ insieme di nodi domanda
$J =$ insieme di siti candidati ad ospitare il servizio.
$c_j =$ costo per aprire il servizio nel sito j
$d_{ij} =$ distanza tra nodo di domanda i e sito candidato j
$D =$ distanza di copertura, cioé i é coperto da j se $d_{ij} \leq D$

Problema di copertura

Problema

In generale, dati:
$I =$ insieme di nodi domanda
$J =$ insieme di siti candidati ad ospitare il servizio.
$c_j =$ costo per aprire il servizio nel sito j
$d_{ij} =$ distanza tra nodo di domanda i e sito candidato j
$D =$ distanza di copertura, cioé i é coperto da j se $d_{ij} \leq D$

Decidere in quali siti aprire il servizio in modo da coprire tutti i nodi di domanda con l'obiettivo di minimizzare il costo totale.

Problema di copertura

Problema

In generale, dati:
$I =$ insieme di nodi domanda
$J =$ insieme di siti candidati ad ospitare il servizio.
$c_j =$ costo per aprire il servizio nel sito j
$d_{ij} =$ distanza tra nodo di domanda i e sito candidato j
$D =$ distanza di copertura, cioé i é coperto da j se $d_{ij} \leq D$

Decidere in quali siti aprire il servizio in modo da coprire tutti i nodi di domanda con l'obiettivo di minimizzare il costo totale.

Esempio

Posizionamento di servizi di emergenza (ambulanze, caserme dei pompieri, ...)

Modello di copertura

Definiamo $a_{ij} = \begin{cases} 1 & \text{se } d_{ij} \leq D, \\ 0 & \text{altrimenti.} \end{cases}$

Modello di copertura

Definiamo $a_{ij} = \begin{cases} 1 & \text{se } d_{ij} \leq D, \\ 0 & \text{altrimenti.} \end{cases}$ Variabili:

$x_j = \begin{cases} 1 & \text{se nel sito } j \text{ apriamo il servizio,} \\ 0 & \text{altrimenti.} \end{cases}$

Modello di copertura

Definiamo $a_{ij} = \begin{cases} 1 & \text{se } d_{ij} \leq D, \\ 0 & \text{altrimenti.} \end{cases}$ Variabili:
$x_j = \begin{cases} 1 & \text{se nel sito } j \text{ apriamo il servizio,} \\ 0 & \text{altrimenti.} \end{cases}$

$$\begin{aligned} \min \sum_{j \in J} c_j x_j & \\ \sum_{j \in J} a_{ij} x_j \geq 1 & \quad \forall\ i \in I \\ x_j \in \{0, 1\} & \quad \forall\ j \in J \end{aligned} \tag{5}$$

Modello di copertura

Definiamo $a_{ij} = \begin{cases} 1 & \text{se } d_{ij} \leq D, \\ 0 & \text{altrimenti.} \end{cases}$ Variabili:
$x_j = \begin{cases} 1 & \text{se nel sito } j \text{ apriamo il servizio,} \\ 0 & \text{altrimenti.} \end{cases}$

$$\begin{aligned} \min \sum_{j \in J} c_j\, x_j & \\ \sum_{j \in J} a_{ij}\, x_j \geq 1 & \quad \forall\ i \in I \\ x_j \in \{0, 1\} & \quad \forall\ j \in J \end{aligned} \tag{5}$$

(5): ogni nodo di domanda deve essere coperto dal servizio.

Esempio

Una ASL deve decidere dove posizionare alcune ambulanze nella cittá.

Esempio

Una ASL deve decidere dove posizionare alcune ambulanze nella cittá.
Ci sono 5 siti diversi, ognuno dei quali puó ospitare un'ambulanza ed é necessario servire 10 zone della cittá.

Esempio

Una ASL deve decidere dove posizionare alcune ambulanze nella cittá.
Ci sono 5 siti diversi, ognuno dei quali puó ospitare un'ambulanza ed é necessario servire 10 zone della cittá.
I tempi (in minuti) per raggiungere le zone da ogni sito sono riassunti nella seguente tabella:

Esempio

Una ASL deve decidere dove posizionare alcune ambulanze nella cittá.
Ci sono 5 siti diversi, ognuno dei quali puó ospitare un'ambulanza ed é necessario servire 10 zone della cittá.
I tempi (in minuti) per raggiungere le zone da ogni sito sono riassunti nella seguente tabella:

	siti				
zone	1	2	3	4	5
1	2	7	13	11	12
2	5	13	9	12	11
3	7	15	2	16	4
4	19	7	6	18	2
5	11	2	12	8	19
6	12	18	9	10	11
7	13	17	6	18	7
8	18	12	12	13	6
9	11	6	7	8	9
10	8	14	5	6	13

Esempio

I costi per posizionare le ambulanze nei vari siti sono:

Esempio

I costi per posizionare le ambulanze nei vari siti sono:

siti	1	2	3	4	5
costo	6	9	10	8	7

Esempio

I costi per posizionare le ambulanze nei vari siti sono:

siti	1	2	3	4	5
costo	6	9	10	8	7

L'ASL deve decidere quante ambulanze posizionare e dove posizionarle in modo che ogni zona possa essere raggiunta da un'ambulanza entro 10 minuti, con l'obiettivo di minimizzare i costi.

Problema di massima copertura

Quando il budget é limitato, si puó non riuscire a coprire tutti i nodi di domanda.

Problema di massima copertura

Quando il budget é limitato, si puó non riuscire a coprire tutti i nodi di domanda.

Problema

Dati:
$I =$ insieme di nodi domanda

Problema di massima copertura

Quando il budget é limitato, si puó non riuscire a coprire tutti i nodi di domanda.

Problema

Dati:
$I =$ insieme di nodi domanda
$h_i =$ domanda associata al nodo i

Problema di massima copertura

Quando il budget é limitato, si puó non riuscire a coprire tutti i nodi di domanda.

Problema

Dati:
$I =$ insieme di nodi domanda
$h_i =$ domanda associata al nodo i
$J =$ insieme di siti candidati ad ospitare il servizio

Quando il budget é limitato, si puó non riuscire a coprire tutti i nodi di domanda.

Problema

Dati:
$I =$ insieme di nodi domanda
$h_i =$ domanda associata al nodo i
$J =$ insieme di siti candidati ad ospitare il servizio
$p =$ numero prefissato di posti dove aprire il servizio

Problema di massima copertura

Quando il budget é limitato, si puó non riuscire a coprire tutti i nodi di domanda.

Problema

Dati:
$I =$ insieme di nodi domanda
$h_i =$ domanda associata al nodo i
$J =$ insieme di siti candidati ad ospitare il servizio
$p =$ numero prefissato di posti dove aprire il servizio
$d_{ij} =$ distanza tra nodo di domanda i e sito candidato j

Problema di massima copertura

Quando il budget é limitato, si puó non riuscire a coprire tutti i nodi di domanda.

Problema

Dati:
$I =$ insieme di nodi domanda
$h_i =$ domanda associata al nodo i
$J =$ insieme di siti candidati ad ospitare il servizio
$p =$ numero prefissato di posti dove aprire il servizio
$d_{ij} =$ distanza tra nodo di domanda i e sito candidato j
$D =$ distanza di copertura

Problema di massima copertura

Quando il budget é limitato, si puó non riuscire a coprire tutti i nodi di domanda.

Problema

Dati:
$I =$ insieme di nodi domanda
$h_i =$ domanda associata al nodo i
$J =$ insieme di siti candidati ad ospitare il servizio
$p =$ numero prefissato di posti dove aprire il servizio
$d_{ij} =$ distanza tra nodo di domanda i e sito candidato j
$D =$ distanza di copertura

Decidere in quali p siti aprire il servizio in modo da massimizzare la domanda coperta.

Modello

Variabili: $x_j = \begin{cases} 1 & \text{se apriamo il servizio nel sito } j, \\ 0 & \text{altrimenti.} \end{cases}$ $z_i = \begin{cases} 1 & \text{se il nodo } i \text{ é coperto,} \\ 0 & \text{altrimenti.} \end{cases}$

Modello

Variabili: $x_j = \begin{cases} 1 & \text{se apriamo il servizio nel sito } j, \\ 0 & \text{altrimenti.} \end{cases}$ $z_i = \begin{cases} 1 & \text{se il nodo } i \text{ é coperto,} \\ 0 & \text{altrimenti.} \end{cases}$

$$\max \sum_{i \in I} h_i z_i$$

$$\sum_{j \in J} a_{ij} x_j \geq z_i \quad \forall\ i \in I \tag{6}$$

$$\sum_{j \in J} x_j = p \tag{7}$$

$$x_j \in \{0, 1\} \quad \forall\ j \in J$$

$$z_i \in \{0, 1\} \quad \forall\ i \in I$$

Modello

Variabili: $x_j = \begin{cases} 1 & \text{se apriamo il servizio nel sito } j, \\ 0 & \text{altrimenti.} \end{cases}$ $z_i = \begin{cases} 1 & \text{se il nodo } i \text{ é coperto,} \\ 0 & \text{altrimenti.} \end{cases}$

$$\max \sum_{i \in I} h_i \, z_i$$

$$\sum_{j \in J} a_{ij} \, x_j \geq z_i \quad \forall \; i \in I \tag{6}$$

$$\sum_{j \in J} x_j = p \tag{7}$$

$$x_j \in \{0, 1\} \quad \forall \; j \in J$$

$$z_i \in \{0, 1\} \quad \forall \; i \in I$$

(6): per coprire il nodo *i* dobbiamo aprire almeno un servizio in un posto che lo possa coprire.

Modello

Variabili: $x_j = \begin{cases} 1 & \text{se apriamo il servizio nel sito } j, \\ 0 & \text{altrimenti.} \end{cases}$ $z_i = \begin{cases} 1 & \text{se il nodo } i \text{ é coperto,} \\ 0 & \text{altrimenti.} \end{cases}$

$$\max \sum_{i \in I} h_i z_i$$

$$\sum_{j \in J} a_{ij} x_j \geq z_i \quad \forall\ i \in I \tag{6}$$

$$\sum_{j \in J} x_j = p \tag{7}$$

$$x_j \in \{0,1\} \quad \forall\ j \in J$$

$$z_i \in \{0,1\} \quad \forall\ i \in I$$

(6): per coprire il nodo i dobbiamo aprire almeno un servizio in un posto che lo possa coprire.
(7): deve essere aperto il servizio in p luoghi.

Esempio

Una ASL ha a disposizione un budget sufficiente per posizionare 2 centri di prenotazione (CUP) nella cittá e sa che ci sono 5 siti che possono ospitare tali centri.

Esempio

Una ASL ha a disposizione un budget sufficiente per posizionare 2 centri di prenotazione (CUP) nella cittá e sa che ci sono 5 siti che possono ospitare tali centri.

La cittá é divisa in 10 zone e i tempi (in minuti) per raggiungere le zone da ogni sito sono indicati nella tabella dell'esempio precedente.

Esempio

Una ASL ha a disposizione un budget sufficiente per posizionare 2 centri di prenotazione (CUP) nella cittá e sa che ci sono 5 siti che possono ospitare tali centri.

La cittá é divisa in 10 zone e i tempi (in minuti) per raggiungere le zone da ogni sito sono indicati nella tabella dell'esempio precedente.

La tabella seguente indica il numero di abitanti che vivono nelle 10 zone della cittá:

Esempio

Una ASL ha a disposizione un budget sufficiente per posizionare 2 centri di prenotazione (CUP) nella cittá e sa che ci sono 5 siti che possono ospitare tali centri.

La cittá é divisa in 10 zone e i tempi (in minuti) per raggiungere le zone da ogni sito sono indicati nella tabella dell'esempio precedente.

La tabella seguente indica il numero di abitanti che vivono nelle 10 zone della cittá:

zone	1	2	3	4	5	6	7	8	9	10
abitanti	1500	3000	1000	1800	2500	2100	1900	1700	2200	2700

Esempio

Una ASL ha a disposizione un budget sufficiente per posizionare 2 centri di prenotazione (CUP) nella cittá e sa che ci sono 5 siti che possono ospitare tali centri.

La cittá é divisa in 10 zone e i tempi (in minuti) per raggiungere le zone da ogni sito sono indicati nella tabella dell'esempio precedente.

La tabella seguente indica il numero di abitanti che vivono nelle 10 zone della cittá:

zone	1	2	3	4	5	6	7	8	9	10
abitanti	1500	3000	1000	1800	2500	2100	1900	1700	2200	2700

Il dirigente deve decidere dove posizionare i 2 centri in modo da massimizzare il numero di abitanti che possono raggiungere un CUP in al piú 10 minuti.

Metodo greedy per soluzione ammissibile

1. $I = \{1, \ldots, m\}$, $J = \{1, \ldots, n\}$, $x := 0$.

Metodo greedy per soluzione ammissibile

1. $I = \{1, \dots, m\}$, $J = \{1, \dots, n\}$, $x := 0$.
2. Per ogni $j \in J$ calcola la domanda u_j coperta da j:

Metodo greedy per soluzione ammissibile

1. $I = \{1, \ldots, m\}$, $J = \{1, \ldots, n\}$, $x := 0$.
2. Per ogni $j \in J$ calcola la domanda u_j coperta da j:

$$u_j = \sum_{i \in I : d_{ij} \leq D} h_i$$

3. Trova l'indice $k \in J$ tale che $u_k = \max_{j \in J} u_j$

Metodo greedy per soluzione ammissibile

1. $I = \{1, \dots, m\}$, $J = \{1, \dots, n\}$, $x := 0$.
2. Per ogni $j \in J$ calcola la domanda u_j coperta da j:

$$u_j = \sum_{i \in I : d_{ij} \leq D} h_i$$

3. Trova l'indice $k \in J$ tale che $u_k = \max_{j \in J} u_j$

   Poni $x_k := 1$, da J elimina k, da I elimina $\{i : \; d_{ik} \leq D\}$
4. Se $\sum_{j=1}^{n} x_j = p$ allora STOP; altrimenti torna al passo 2.

Esempio

Applichiamo il metodo greedy al problema precedente.

Esempio

Applichiamo il metodo greedy al problema precedente.

La domanda coperta da ogni sito é

Esempio

Applichiamo il metodo greedy al problema precedente.

La domanda coperta da ogni sito é

siti	1	2	3	4	5
domanda coperta	8200	8000	14700	9500	8600

Esempio

Applichiamo il metodo greedy al problema precedente.

La domanda coperta da ogni sito é

siti	1	2	3	4	5
domanda coperta	8200	8000	14700	9500	8600

Scegliamo il sito 3 ($x_3 = 1$) e copriamo le zone 2,3,4,6,7,9,10. Per le zone rimanenti 1,5,8 la domanda coperta dai 4 siti rimanenti é:

Esempio

Applichiamo il metodo greedy al problema precedente.

La domanda coperta da ogni sito é

siti	1	2	3	4	5
domanda coperta	8200	8000	14700	9500	8600

Scegliamo il sito 3 ($x_3 = 1$) e copriamo le zone 2,3,4,6,7,9,10. Per le zone rimanenti 1,5,8 la domanda coperta dai 4 siti rimanenti é:

siti	1	2	4	5
domanda coperta	1500	4000	2500	1700

Esempio

Applichiamo il metodo greedy al problema precedente.

La domanda coperta da ogni sito é

siti	1	2	3	4	5
domanda coperta	8200	8000	14700	9500	8600

Scegliamo il sito 3 ($x_3 = 1$) e copriamo le zone 2,3,4,6,7,9,10. Per le zone rimanenti 1,5,8 la domanda coperta dai 4 siti rimanenti é:

siti	1	2	4	5
domanda coperta	1500	4000	2500	1700

Scegliamo il sito 2 ($x_2 = 1$) e copriamo le zone 1 e 5. In totale copriamo 18700 abitanti (la zona 8 con 1700 abitanti rimane scoperta).

Schema generale di localizzazione

Insieme $I = \{1, \ldots, m\}$.

Schema generale di localizzazione

Insieme $I = \{1, \ldots, m\}$.
Famiglia $S_1, \ldots, S_n$ di sottoinsiemi di I, ognuno ha costo (valore) c_j.

Schema generale di localizzazione

Insieme $I = \{1, \dots, m\}$.
Famiglia $S_1, \dots, S_n$ di sottoinsiemi di I, ognuno ha costo (valore) c_j.

Problema di copertura: determinare una sottofamiglia $\mathcal{F}$ di costo minimo tale che ogni elemento di I appartenga ad almeno un sottoinsieme di $\mathcal{F}$.

Schema generale di localizzazione

Insieme $I = \{1, \ldots, m\}$.
Famiglia $S_1, \ldots, S_n$ di sottoinsiemi di I, ognuno ha costo (valore) c_j.

Problema di copertura: determinare una sottofamiglia $\mathcal{F}$ di costo minimo tale che ogni elemento di I appartenga ad almeno un sottoinsieme di $\mathcal{F}$.

Problema di partizione: determinare una sottofamiglia $\mathcal{F}$ di costo minimo tale che ogni elemento di I appartenga esattamente ad un sottoinsieme di $\mathcal{F}$.

Insieme $I = \{1, \dots, m\}$.
Famiglia $S_1, \dots, S_n$ di sottoinsiemi di I, ognuno ha costo (valore) c_j.

Problema di copertura: determinare una sottofamiglia $\mathcal{F}$ di costo minimo tale che ogni elemento di I appartenga ad almeno un sottoinsieme di $\mathcal{F}$.

Problema di partizione: determinare una sottofamiglia $\mathcal{F}$ di costo minimo tale che ogni elemento di I appartenga esattamente ad un sottoinsieme di $\mathcal{F}$.

Problema di riempimento: determinare una sottofamiglia $\mathcal{F}$ di valore massimo tale che ogni elemento di I appartenga ad al piú un sottoinsieme di $\mathcal{F}$.

Schema generale di localizzazione

Insieme $I = \{1, \ldots, m\}$.
Famiglia $S_1, \ldots, S_n$ di sottoinsiemi di I, ognuno ha costo (valore) c_j.

Problema di copertura: determinare una sottofamiglia $\mathcal{F}$ di costo minimo tale che ogni elemento di I appartenga ad almeno un sottoinsieme di $\mathcal{F}$.

Problema di partizione: determinare una sottofamiglia $\mathcal{F}$ di costo minimo tale che ogni elemento di I appartenga esattamente ad un sottoinsieme di $\mathcal{F}$.

Problema di riempimento: determinare una sottofamiglia $\mathcal{F}$ di valore massimo tale che ogni elemento di I appartenga ad al piú un sottoinsieme di $\mathcal{F}$.

Schema generale di localizzazione

Un esempio:

Un esempio:
$I = \{1, 2, 3, 4, 5, 6\}$,
$S_1 = \{1, 3\}$, $S_2 = \{2, 4\}$, $S_3 = \{2, 5, 6\}$, $S_4 = \{5, 6\}$, $S_5 = \{3, 4\}$.

Schema generale di localizzazione

Un esempio:
$I = \{1, 2, 3, 4, 5, 6\}$,
$S_1 = \{1, 3\}$, $S_2 = \{2, 4\}$, $S_3 = \{2, 5, 6\}$, $S_4 = \{5, 6\}$, $S_5 = \{3, 4\}$.

copertura: $\mathcal{F} = \{S_1, S_2, S_3\}$

Schema generale di localizzazione

Un esempio:
$I = \{1, 2, 3, 4, 5, 6\}$,
$S_1 = \{1, 3\}$, $S_2 = \{2, 4\}$, $S_3 = \{2, 5, 6\}$, $S_4 = \{5, 6\}$, $S_5 = \{3, 4\}$.

copertura: $\mathcal{F} = \{S_1, S_2, S_3\}$
partizione: $\mathcal{F} = \{S_1, S_2, S_4\}$

Schema generale di localizzazione

Un esempio:
$I = \{1, 2, 3, 4, 5, 6\}$,
$S_1 = \{1, 3\}$, $S_2 = \{2, 4\}$, $S_3 = \{2, 5, 6\}$, $S_4 = \{5, 6\}$, $S_5 = \{3, 4\}$.

copertura: $\mathcal{F} = \{S_1, S_2, S_3\}$
partizione: $\mathcal{F} = \{S_1, S_2, S_4\}$
riempimento: $\mathcal{F} = \{S_3, S_5\}$

Modelli

Definiamo la matrice: $a_{ij} = \begin{cases} 1 & \text{se } i \in S_j, \\ 0 & \text{altrimenti.} \end{cases}$

Modelli

Definiamo la matrice: $a_{ij} = \begin{cases} 1 & \text{se } i \in S_j, \\ 0 & \text{altrimenti.} \end{cases}$

Ad ogni sottofamiglia $\mathcal{F}$ associamo una variabile x, dove $x_j = \begin{cases} 1 & \text{se } S_j \in \mathcal{F}, \\ 0 & \text{altrimenti.} \end{cases}$

Modelli

Definiamo la matrice: $a_{ij} = \begin{cases} 1 & \text{se } i \in S_j, \\ 0 & \text{altrimenti.} \end{cases}$

Ad ogni sottofamiglia $\mathcal{F}$ associamo una variabile x, dove $x_j = \begin{cases} 1 & \text{se } S_j \in \mathcal{F}, \\ 0 & \text{altrimenti.} \end{cases}$

Copertura	Partizione	Riempimento
$\begin{cases} \min \ \sum_{j=1}^{n} c_j x_j \\ \sum_{j=1}^{n} a_{ij} x_j \geq 1 \quad \forall\ i \\ x_j \in \{0,1\} \quad \forall\ j \end{cases}$		

Modelli

Definiamo la matrice: $a_{ij} = \begin{cases} 1 & \text{se } i \in S_j, \\ 0 & \text{altrimenti.} \end{cases}$

Ad ogni sottofamiglia $\mathcal{F}$ associamo una variabile x, dove $x_j = \begin{cases} 1 & \text{se } S_j \in \mathcal{F}, \\ 0 & \text{altrimenti.} \end{cases}$

Copertura

$$\begin{cases} \min \ \sum_{j=1}^{n} c_j x_j \\ \sum_{j=1}^{n} a_{ij} x_j \geq 1 \quad \forall \ i \\ x_j \in \{0,1\} \quad \forall \ j \end{cases}$$

Partizione

$$\begin{cases} \min \ \sum_{j=1}^{n} c_j x_j \\ \sum_{j=1}^{n} a_{ij} x_j = 1 \quad \forall \ i \\ x_j \in \{0,1\} \quad \forall \ j \end{cases}$$

Riempimento

Modelli

Definiamo la matrice: $a_{ij} = \begin{cases} 1 & \text{se } i \in S_j, \\ 0 & \text{altrimenti.} \end{cases}$

Ad ogni sottofamiglia $\mathcal{F}$ associamo una variabile x, dove $x_j = \begin{cases} 1 & \text{se } S_j \in \mathcal{F}, \\ 0 & \text{altrimenti.} \end{cases}$

Copertura	Partizione	Riempimento
$\begin{cases} \min \sum_{j=1}^{n} c_j x_j \\ \sum_{j=1}^{n} a_{ij} x_j \geq 1 \quad \forall\, i \\ x_j \in \{0,1\} \quad \forall\, j \end{cases}$	$\begin{cases} \min \sum_{j=1}^{n} c_j x_j \\ \sum_{j=1}^{n} a_{ij} x_j = 1 \quad \forall\, i \\ x_j \in \{0,1\} \quad \forall\, j \end{cases}$	$\begin{cases} \max \sum_{j=1}^{n} c_j x_j \\ \sum_{j=1}^{n} a_{ij} x_j \leq 1 \quad \forall\, i \\ x_j \in \{0,1\} \quad \forall\, j \end{cases}$

Dislocazione mezzi soccorso

Insieme U di siti in cui sono presenti utenti.

Dislocazione mezzi soccorso

Insieme U di siti in cui sono presenti utenti.
Insieme M di siti dove é possibile dislocare ambulanze.

Dislocazione mezzi soccorso

Insieme U di siti in cui sono presenti utenti.
Insieme M di siti dove é possibile dislocare ambulanze.
$t_{ij} =$ tempo necessario utente in $i \in U$ sia raggiunto da ambulanza posta in $j \in M$.

Dislocazione mezzi soccorso

Insieme U di siti in cui sono presenti utenti.
Insieme M di siti dove é possibile dislocare ambulanze.
$t_{ij} =$ tempo necessario utente in $i \in U$ sia raggiunto da ambulanza posta in $j \in M$.
$c_j =$ costo di attivazione del sito $j \in M$, $T =$ massima attesa per ogni utente.

Dislocazione mezzi soccorso

Insieme U di siti in cui sono presenti utenti.
Insieme M di siti dove é possibile dislocare ambulanze.
$t_{ij} =$ tempo necessario utente in $i \in U$ sia raggiunto da ambulanza posta in $j \in M$.
$c_j =$ costo di attivazione del sito $j \in M$, $T =$ massima attesa per ogni utente.

Obiettivo: decidere in quali siti dislocare ambulanze in modo che ogni utente sia raggiunto in al piú T minuti, minimizzando il costo totale.

Dislocazione mezzi soccorso

Insieme U di siti in cui sono presenti utenti.
Insieme M di siti dove é possibile dislocare ambulanze.
t_{ij} = tempo necessario utente in $i \in U$ sia raggiunto da ambulanza posta in $j \in M$.
c_j = costo di attivazione del sito $j \in M$, T = massima attesa per ogni utente.

Obiettivo: decidere in quali siti dislocare ambulanze in modo che ogni utente sia raggiunto in al piú T minuti, minimizzando il costo totale.

Puó essere formulato come problema di copertura:

Dislocazione mezzi soccorso

Insieme U di siti in cui sono presenti utenti.
Insieme M di siti dove é possibile dislocare ambulanze.
$t_{ij} =$ tempo necessario utente in $i \in U$ sia raggiunto da ambulanza posta in $j \in M$.
$c_j =$ costo di attivazione del sito $j \in M$, $T =$ massima attesa per ogni utente.

Obiettivo: decidere in quali siti dislocare ambulanze in modo che ogni utente sia raggiunto in al piú T minuti, minimizzando il costo totale.

Puó essere formulato come problema di copertura:
sottoinsiemi $S_j = \{$utenti raggiunti entro T minuti da ambulanza posta in $j\}$.

Ricerca di informazioni

Bisogna soddisfare *m* richieste di dati.

Ricerca di informazioni

Bisogna soddisfare m richieste di dati.
n archivi a disposizione, ognuno contiene alcune delle informazioni richieste.

Ricerca di informazioni

Bisogna soddisfare m richieste di dati.
n archivi a disposizione, ognuno contiene alcune delle informazioni richieste.
$c_j =$ costo per interrogare archivio j.

Ricerca di informazioni

Bisogna soddisfare m richieste di dati.
n archivi a disposizione, ognuno contiene alcune delle informazioni richieste.
$c_j =$ costo per interrogare archivio j.

Obiettivo: selezionare un sottoinsieme di archivi capace di soddisfare tutte le richieste e che richieda il minimo costo totale.

Ricerca di informazioni

Bisogna soddisfare m richieste di dati.
n archivi a disposizione, ognuno contiene alcune delle informazioni richieste.
$c_j =$ costo per interrogare archivio j.

Obiettivo: selezionare un sottoinsieme di archivi capace di soddisfare tutte le richieste e che richieda il minimo costo totale.

Puó essere formulato come problema di copertura:

Ricerca di informazioni

Bisogna soddisfare m richieste di dati.
n archivi a disposizione, ognuno contiene alcune delle informazioni richieste.
$c_j =$ costo per interrogare archivio j.

Obiettivo: selezionare un sottoinsieme di archivi capace di soddisfare tutte le richieste e che richieda il minimo costo totale.

Puó essere formulato come problema di copertura:
$I = \{\text{richieste}\}$, sottoinsiemi $S_j =$ archivi.

Pianificazione di equipaggi

Compagnia aerea ha m rotte (punto partenza, punto arrivo, durata volo).

Pianificazione di equipaggi

Compagnia aerea ha m rotte (punto partenza, punto arrivo, durata volo).
n combinazioni possibili di rotte (*round trip*) che puó fare un equipaggio in un giorno.

Pianificazione di equipaggi

Compagnia aerea ha m rotte (punto partenza, punto arrivo, durata volo).
n combinazioni possibili di rotte (*round trip*) che puó fare un equipaggio in un giorno.
c_j = costo del *round trip* j.

Pianificazione di equipaggi

Compagnia aerea ha m rotte (punto partenza, punto arrivo, durata volo).
n combinazioni possibili di rotte (*round trip*) che puó fare un equipaggio in un giorno.
$c_j =$ costo del *round trip* j.

Obiettivo: determinare un insieme di round trip con il minimo costo totale in modo che ogni rotta sia coperta da esattamente un *round trip*.

Pianificazione di equipaggi

Compagnia aerea ha m rotte (punto partenza, punto arrivo, durata volo).
n combinazioni possibili di rotte (*round trip*) che puó fare un equipaggio in un giorno.
$c_j =$ costo del *round trip* j.

Obiettivo: determinare un insieme di round trip con il minimo costo totale in modo che ogni rotta sia coperta da esattamente un *round trip*.

Puó essere formulato come problema di partizione:

Pianificazione di equipaggi

Compagnia aerea ha m rotte (punto partenza, punto arrivo, durata volo).
n combinazioni possibili di rotte (*round trip*) che puó fare un equipaggio in un giorno.
c_j = costo del *round trip* j.

Obiettivo: determinare un insieme di round trip con il minimo costo totale in modo che ogni rotta sia coperta da esattamente un *round trip*.

Puó essere formulato come problema di partizione:
$I = \{\text{rotte}\}$, sottoinsiemi $S_j =$ *round trip*.

Formazione di distretto elettorale

Esempio

Formazione di distretto elettorale

Un territorio formato da vari comuni deve essere diviso in un insieme di distretti elettorali.

Esempio

Formazione di distretto elettorale

Un territorio formato da vari comuni deve essere diviso in un insieme di distretti elettorali.

Sono specificate n possibili suddivisioni che rispondono a determinati requisiti.

Esempio

Formazione di distretto elettorale

Un territorio formato da vari comuni deve essere diviso in un insieme di distretti elettorali.

Sono specificate n possibili suddivisioni che rispondono a determinati requisiti.

Se si sceglie di costruire il distretto j si paga un costo c_j.

Esempio

Formazione di distretto elettorale

Un territorio formato da vari comuni deve essere diviso in un insieme di distretti elettorali.

Sono specificate n possibili suddivisioni che rispondono a determinati requisiti.

Se si sceglie di costruire il distretto j si paga un costo c_j.

Ogni comune deve essere incluso esattamente in un solo distretto elettorale.

Esempio

Formazione di distretto elettorale

Un territorio formato da vari comuni deve essere diviso in un insieme di distretti elettorali.

Sono specificate n possibili suddivisioni che rispondono a determinati requisiti.

Se si sceglie di costruire il distretto j si paga un costo c_j.

Ogni comune deve essere incluso esattamente in un solo distretto elettorale.

La matrice A ha una riga per ciascun comune ed una colonna per ciascun distretto.

Esempio

Formazione di distretto elettorale

Un territorio formato da vari comuni deve essere diviso in un insieme di distretti elettorali.

Sono specificate n possibili suddivisioni che rispondono a determinati requisiti.

Se si sceglie di costruire il distretto j si paga un costo c_j.

Ogni comune deve essere incluso esattamente in un solo distretto elettorale.

La matrice A ha una riga per ciascun comune ed una colonna per ciascun distretto.

Puó essere formulato come problema di partizione:

Esempio

Formazione di distretto elettorale

Un territorio formato da vari comuni deve essere diviso in un insieme di distretti elettorali.

Sono specificate n possibili suddivisioni che rispondono a determinati requisiti.

Se si sceglie di costruire il distretto j si paga un costo c_j.

Ogni comune deve essere incluso esattamente in un solo distretto elettorale.

La matrice A ha una riga per ciascun comune ed una colonna per ciascun distretto.

Puó essere formulato come problema di partizione:

$I = \{\text{comuni}\}$, sottoinsiemi $S_j = \{\text{insiemi dei possibili distretti }\}$.

Squadre di operai

Insieme di operai, insieme di squadre di operai.

Squadre di operai

Insieme di operai, insieme di squadre di operai.
La squadra j é in grado di svolgere un'attivitá che fornisce profitto p_j.

Squadre di operai

Insieme di operai, insieme di squadre di operai.
La squadra j é in grado di svolgere un'attivitá che fornisce profitto p_j.
Le squadre devono lavorare contemporaneamente.

Squadre di operai

Insieme di operai, insieme di squadre di operai.
La squadra j é in grado di svolgere un'attivitá che fornisce profitto p_j.
Le squadre devono lavorare contemporaneamente.

Obiettivo: formare le squadre in modo da massimizzare il profitto.

Squadre di operai

Insieme di operai, insieme di squadre di operai.
La squadra j é in grado di svolgere un'attivitá che fornisce profitto p_j.
Le squadre devono lavorare contemporaneamente.

Obiettivo: formare le squadre in modo da massimizzare il profitto.

Puó essere formulato come problema di riempimento:

Squadre di operai

Insieme di operai, insieme di squadre di operai.
La squadra j é in grado di svolgere un'attivitá che fornisce profitto p_j.
Le squadre devono lavorare contemporaneamente.

Obiettivo: formare le squadre in modo da massimizzare il profitto.

Puó essere formulato come problema di riempimento:
$I = \{\text{operai}\}$, sottoinsiemi $S_j =$ squadre di operai.

Esempio

Localizzazione di impianti ad elevato impatto ambientale

Esempio

Localizzazione di impianti ad elevato impatto ambientale

Insieme di cittá, gruppi di cittá inquinate dal sito j.

Esempio

Localizzazione di impianti ad elevato impatto ambientale

Insieme di cittá, gruppi di cittá inquinate dal sito j.

Obiettivo: Localizzare gli impianti massimizzando il profitto ma con il vincolo che ogni cittá subisca al piú un impianto inquinante.

Esempio

Localizzazione di impianti ad elevato impatto ambientale

Insieme di cittá, gruppi di cittá inquinate dal sito j.

Obiettivo: Localizzare gli impianti massimizzando il profitto ma con il vincolo che ogni cittá subisca al piú un impianto inquinante.

Puó essere formulato come problema di riempimento:

Esempio

Localizzazione di impianti ad elevato impatto ambientale

Insieme di cittá, gruppi di cittá inquinate dal sito j.

Obiettivo: Localizzare gli impianti massimizzando il profitto ma con il vincolo che ogni cittá subisca al piú un impianto inquinante.

Puó essere formulato come problema di riempimento:
$I = \{\text{cittá}\}$, sottoinsiemi $S_j =$ di cittá che subiscono disagi dall'impianto j.

Esempio

$$A = \begin{array}{|c|cccc|}
\hline
c_j & 7 & 6 & 4 & 2 \\
\hline
{}_i\backslash^j & 1 & 2 & 3 & 4 \\
\hline
1 & 1 & 1 & 0 & 0 \\
2 & 0 & 0 & 1 & 0 \\
3 & 1 & 1 & 1 & 0 \\
4 & 0 & 0 & 0 & 1 \\
5 & 1 & 0 & 0 & 1 \\
6 & 1 & 1 & 0 & 0 \\
7 & 1 & 0 & 0 & 1 \\
8 & 1 & 0 & 0 & 0 \\
9 & 0 & 1 & 0 & 0 \\
\hline
\end{array}$$

Esempio

$$A = \begin{array}{|c|cccc|}
\hline
c_j & 7 & 6 & 4 & 2 \\
\hline
{}_i\backslash^j & 1 & 2 & 3 & 4 \\
\hline
1 & 1 & 1 & 0 & 0 \\
2 & 0 & 0 & 1 & 0 \\
3 & 1 & 1 & 1 & 0 \\
4 & 0 & 0 & 0 & 1 \\
5 & 1 & 0 & 0 & 1 \\
6 & 1 & 1 & 0 & 0 \\
7 & 1 & 0 & 0 & 1 \\
8 & 1 & 0 & 0 & 0 \\
9 & 0 & 1 & 0 & 0 \\
\hline
\end{array}$$

La colonna 1 significa che posizionare l'impianto nel sito 1 si guadagna 7 e si creano disagi alle cittá $1, 3, 5, 6, 7, 8$

Riempimento $\longrightarrow$ Partizione

Ogni problema di riempimento puó essere trasformato in un problema di partizione cambiando segno ai costi e aggiungendo variabili di scarto:

Ogni problema di riempimento puó essere trasformato in un problema di partizione cambiando segno ai costi e aggiungendo variabili di scarto:

$$\begin{cases} \max \sum_{j=1}^{n} c_j x_j \\ \sum_{j=1}^{n} a_{ij} x_j \leq 1 \quad \forall\ i \\ x_j \in \{0,1\} \quad \forall\ j \end{cases} \longrightarrow \begin{cases} \min\ -\sum_{j=1}^{n} c_j x_j \\ \sum_{j=1}^{n} a_{ij} x_j + s_i = 1 \quad \forall\ i \\ x_j \in \{0,1\} \quad \forall\ j \\ s_i \in \{0,1\} \quad \forall\ i \end{cases}$$

Ogni problema di riempimento puó essere trasformato in un problema di partizione cambiando segno ai costi e aggiungendo variabili di scarto:

$$\begin{cases} \max \ \sum_{j=1}^{n} c_j x_j \\ \sum_{j=1}^{n} a_{ij} x_j \leq 1 \quad \forall\ i \\ x_j \in \{0,1\} \quad \forall\ j \end{cases} \longrightarrow \begin{cases} \min \ -\sum_{j=1}^{n} c_j x_j \\ \sum_{j=1}^{n} a_{ij} x_j + s_i = 1 \quad \forall\ i \\ x_j \in \{0,1\} \quad \forall\ j \\ s_i \in \{0,1\} \quad \forall\ i \end{cases}$$

ossia aggiungendo i sottoinsiemi $S_i = \{i\}$ di costo nullo, per ogni $i = 1, \ldots, m$.

Partizione $\longrightarrow$ Copertura

Teorema

Ogni problema di partizione puó essere trasformato in un problema di copertura.

Partizione $\longrightarrow$ Copertura

Teorema

Ogni problema di partizione puó essere trasformato in un problema di copertura.

Siano $\bar{c}_j = c_j + M \sum_{i=1}^{m} a_{ij}$, dove $M > \sum_{j:c_j>0} c_j - \sum_{j:c_j<0} c_j$.

Partizione $\longrightarrow$ Copertura

Teorema

Ogni problema di partizione puó essere trasformato in un problema di copertura.

Siano $\bar{c}_j = c_j + M \sum_{i=1}^{m} a_{ij}$, dove $M > \sum_{j:c_j>0} c_j - \sum_{j:c_j<0} c_j$.

Allora ogni soluzione ottima del problema di copertura

$$\begin{cases} \min \ \sum_{j=1}^{n} \bar{c}_j x_j \\ \sum_{j=1}^{n} a_{ij} x_j \geq 1 \quad \forall \ i \\ x_j \in \{0,1\} \quad \forall \ j \end{cases}$$

Partizione $\longrightarrow$ Copertura

Teorema

Ogni problema di partizione puó essere trasformato in un problema di copertura.

Siano $\bar{c}_j = c_j + M \sum_{i=1}^{m} a_{ij}$, dove $M > \sum_{j:c_j>0} c_j - \sum_{j:c_j<0} c_j$.

Allora ogni soluzione ottima del problema di copertura

$$\begin{cases} \min \ \sum_{j=1}^{n} \bar{c}_j \, x_j \\ \sum_{j=1}^{n} a_{ij} \, x_j \geq 1 \quad \forall \ i \\ x_j \in \{0,1\} \quad \forall \ j \end{cases}$$

é una soluzione ottima del problema di partizione

Partizione $\longrightarrow$ Copertura

Teorema

Ogni problema di partizione puó essere trasformato in un problema di copertura.

Siano $\bar{c}_j = c_j + M \sum_{i=1}^{m} a_{ij}$, dove $M > \sum_{j:c_j>0} c_j - \sum_{j:c_j<0} c_j$.

Allora ogni soluzione ottima del problema di copertura

$$\begin{cases} \min \sum_{j=1}^{n} \bar{c}_j x_j \\ \sum_{j=1}^{n} a_{ij} x_j \geq 1 \quad \forall\ i \\ x_j \in \{0,1\} \quad \forall\ j \end{cases}$$

é una soluzione ottima del problema di partizione

$$\begin{cases} \min \sum_{j=1}^{n} c_j x_j \\ \sum_{j=1}^{n} a_{ij} x_j = 1 \quad \forall\ i \\ x_j \in \{0,1\} \quad \forall\ j \end{cases}$$

Riduzione di un problema di copertura

Si possono eliminare variabili e vincoli applicando le seguenti regole:

Riduzione di un problema di copertura

Si possono eliminare variabili e vincoli applicando le seguenti regole:

1. Se $c_j \leq 0$ allora pongo $x_j = 1$ ed elimino tutti i vincoli in cui compare x_j (perché esiste una soluzione ottima con $x_j = 1$).

Riduzione di un problema di copertura

Si possono eliminare variabili e vincoli applicando le seguenti regole:

1. Se $c_j \leq 0$ allora pongo $x_j = 1$ ed elimino tutti i vincoli in cui compare x_j (perché esiste una soluzione ottima con $x_j = 1$).
2. Se esiste un vincolo r tale che $a_{rj} = \begin{cases} 1 & \text{se } j = p, \\ 0 & \text{altrimenti,} \end{cases}$ allora pongo $x_p = 1$ ed elimino tutti i vincoli in cui compare x_p (perché ogni sol. ammissibile ha $x_p = 1$).

Riduzione di un problema di copertura

Si possono eliminare variabili e vincoli applicando le seguenti regole:

1. Se $c_j \leq 0$ allora pongo $x_j = 1$ ed elimino tutti i vincoli in cui compare x_j (perché esiste una soluzione ottima con $x_j = 1$).
2. Se esiste un vincolo r tale che $a_{rj} = \begin{cases} 1 & \text{se } j = p, \\ 0 & \text{altrimenti,} \end{cases}$ allora pongo $x_p = 1$ ed elimino tutti i vincoli in cui compare x_p (perché ogni sol. ammissibile ha $x_p = 1$).
3. Se esistono 2 vincoli r, s tali che $a_{rj} \leq a_{sj}$ per ogni j, allora elimino il vincolo s (perché il vincolo r implica il vincolo s).

Riduzione di un problema di copertura

Si possono eliminare variabili e vincoli applicando le seguenti regole:

1. Se $c_j \leq 0$ allora pongo $x_j = 1$ ed elimino tutti i vincoli in cui compare x_j (perché esiste una soluzione ottima con $x_j = 1$).
2. Se esiste un vincolo r tale che $a_{rj} = \begin{cases} 1 & \text{se } j = p, \\ 0 & \text{altrimenti,} \end{cases}$ allora pongo $x_p = 1$ ed elimino tutti i vincoli in cui compare x_p (perché ogni sol. ammissibile ha $x_p = 1$).
3. Se esistono 2 vincoli r, s tali che $a_{rj} \leq a_{sj}$ per ogni j, allora elimino il vincolo s (perché il vincolo r implica il vincolo s).
4. Se esiste un sottoinsieme S_p e una famiglia F tali che

$$\begin{aligned} a_{ip} &\leq \sum_{j \in F} a_{ij} \qquad \text{per ogni } i \in I \\ c_p &\geq \sum_{j \in F} c_j \end{aligned}$$

Riduzione di un problema di copertura

Si possono eliminare variabili e vincoli applicando le seguenti regole:

1. Se $c_j \leq 0$ allora pongo $x_j = 1$ ed elimino tutti i vincoli in cui compare x_j (perché esiste una soluzione ottima con $x_j = 1$).
2. Se esiste un vincolo r tale che $a_{rj} = \begin{cases} 1 & \text{se } j = p, \\ 0 & \text{altrimenti,} \end{cases}$ allora pongo $x_p = 1$ ed elimino tutti i vincoli in cui compare x_p (perché ogni sol. ammissibile ha $x_p = 1$).
3. Se esistono 2 vincoli r, s tali che $a_{rj} \leq a_{sj}$ per ogni j, allora elimino il vincolo s (perché il vincolo r implica il vincolo s).
4. Se esiste un sottoinsieme S_p e una famiglia F tali che

$$\begin{aligned} a_{ip} &\leq \sum_{j \in F} a_{ij} \qquad \text{per ogni } i \in I \\ c_p &\geq \sum_{j \in F} c_j \end{aligned}$$

allora pongo $x_p = 0$ (perché gli elementi di S_p sono coperti anche dalla famiglia F ed il costo di S_p non é inferiore a quello di F).

Riduzione di un problema di copertura: esempio

Consideriamo il problema di copertura:

Riduzione di un problema di copertura: esempio

Consideriamo il problema di copertura:

$$A = \begin{array}{|c|cccccccc|}
\hline
c_j & 10 & 6 & 4 & 2 & 8 & 5 & 3 & 7 \\
\hline
{}_i\backslash^j & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\
\hline
1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 \\
2 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 \\
3 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 \\
4 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
5 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\
6 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\
7 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\
8 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\
9 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\
\hline
\end{array}$$

Consideriamo il problema di copertura:

$$A = \begin{array}{|c|cccccccc|}
\hline
c_j & 10 & 6 & 4 & 2 & 8 & 5 & 3 & 7 \\
\hline
{}_i\diagdown^j & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\
\hline
1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 \\
2 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 \\
3 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 \\
4 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
5 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\
6 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\
7 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\
8 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\
9 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\
\hline
\end{array}$$

Dalla riga 4 si ottiene $x_5 = 1$ e si eliminano le righe 2, 4 e 9.

Riduzione di un problema di copertura: esempio (segue)

Il problema ridotto diventa:

Riduzione di un problema di copertura: esempio (segue)

Il problema ridotto diventa:

$$A = \begin{array}{|c|ccccccc|}
\hline
c_j & 10 & 6 & 4 & 2 & 5 & 3 & 7 \\
\hline
{}_i\diagdown^j & 1 & 2 & 3 & 4 & 6 & 7 & 8 \\
\hline
1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\
3 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\
5 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\
6 & 1 & 1 & 0 & 0 & 1 & 1 & 1 \\
7 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\
8 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\
\hline
\end{array}$$

Riduzione di un problema di copertura: esempio (segue)

Il problema ridotto diventa:

$$A = \begin{array}{|c|ccccccc|} \hline c_j & 10 & 6 & 4 & 2 & 5 & 3 & 7 \\ \hline {}_i\backslash^j & 1 & 2 & 3 & 4 & 6 & 7 & 8 \\ \hline 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ 3 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 5 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ 6 & 1 & 1 & 0 & 0 & 1 & 1 & 1 \\ 7 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 8 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ \hline \end{array}$$

Poiché $a_{1j} \leq a_{6j}$ per ogni j, si pu0̸ eliminare la riga 6.

Riduzione di un problema di copertura

$A =$

c_j	10	6	4	2	5	3	7
$_i\backslash^j$	1	2	3	4	6	7	8
1	1	1	0	0	1	1	0
3	1	1	1	0	0	0	1
5	1	0	0	1	0	1	0
7	1	0	0	1	1	0	0
8	1	0	0	0	1	1	1

Riduzione di un problema di copertura

$A =$

c_j	10	6	4	2	5	3	7
$_i\backslash^j$	1	2	3	4	6	7	8
1	1	1	0	0	1	1	0
3	1	1	1	0	0	0	1
5	1	0	0	1	0	1	0
7	1	0	0	1	1	0	0
8	1	0	0	0	1	1	1

La colonna 1 é dominata dalle colonne 3, 4 e 7, quindi si pone $x_1 = 0$.

Riduzione di un problema di copertura

$A =$

c_j	10	6	4	2	5	3	7
$_i\backslash^j$	1	2	3	4	6	7	8
1	1	1	0	0	1	1	0
3	1	1	1	0	0	0	1
5	1	0	0	1	0	1	0
7	1	0	0	1	1	0	0
8	1	0	0	0	1	1	1

La colonna 1 é dominata dalle colonne 3, 4 e 7, quindi si pone $x_1 = 0$.

$A =$

c_j	6	4	2	5	3	7
$_i\backslash^j$	2	3	4	6	7	8
1	1	0	0	1	1	0
3	1	1	0	0	0	1
5	0	0	1	0	1	0
7	0	0	1	1	0	0
8	0	0	0	1	1	1

Metodi per problemi di copertura

D'ora in poi consideriamo un problema di copertura:

Metodi per problemi di copertura

D'ora in poi consideriamo un problema di copertura:

$$\begin{cases} \min \ \sum_{j=1}^{n} c_j x_j \\ \sum_{j=1}^{n} a_{ij} x_j \geq 1 \quad \forall \ i \\ x_j \in \{0,1\} \quad \forall \ j \end{cases}$$

Per applicare un metodo risolutivo é necessario:

Metodi per problemi di copertura

D'ora in poi consideriamo un problema di copertura:

$$\begin{cases} \min \sum_{j=1}^{n} c_j x_j \\ \sum_{j=1}^{n} a_{ij} x_j \geq 1 \quad \forall\ i \\ x_j \in \{0,1\} \quad \forall\ j \end{cases}$$

Per applicare un metodo risolutivo é necessario:

- calcolare le $v_I(P_i)$
- calcolare una $v_S(P)$

Metodi euristici: algoritmo di Chvátal

$v_S(P)$: un algoritmo *greedy*.

Metodi euristici: algoritmo di Chvátal

$v_S(P)$: un algoritmo *greedy*.

Algoritmo di Chvátal

1. $I = \{1, \ldots, m\}$, $J = \{1, \ldots, n\}$, $x := 0$.

$v_S(P)$: un algoritmo *greedy*.

Algoritmo di Chvátal

1. $I = \{1, \dots, m\}$, $J = \{1, \dots, n\}$, $x := 0$.
2. Per ogni $j \in J$ calcola i costi unitari di copertura

$$u_j = \frac{c_j}{\sum_{i \in I} a_{ij}}$$

Metodi euristici: algoritmo di Chvátal

$v_S(P)$: un algoritmo *greedy*.

Algoritmo di Chvátal

1. $I = \{1, \ldots, m\}$, $J = \{1, \ldots, n\}$, $x := 0$.
2. Per ogni $j \in J$ calcola i costi unitari di copertura

$$u_j = \frac{c_j}{\sum\limits_{i \in I} a_{ij}}$$

3. Trova un indice $k \in J$ tale che $u_k = \min\limits_{j \in J} u_j$

Metodi euristici: algoritmo di Chvátal

$v_S(P)$: un algoritmo *greedy*.

Algoritmo di Chvátal

1. $I = \{1, \dots, m\}$, $J = \{1, \dots, n\}$, $x := 0$.
2. Per ogni $j \in J$ calcola i costi unitari di copertura

$$u_j = \frac{c_j}{\sum_{i \in I} a_{ij}}$$

3. Trova un indice $k \in J$ tale che $u_k = \min_{j \in J} u_j$

   Poni $x_k := 1$, da J elimina k, da I elimina $\{i : \ a_{ik} = 1\}$
4. Se $I = \emptyset$ allora STOP (x é una copertura)

Metodi euristici: algoritmo di Chvátal

$v_S(P)$: un algoritmo *greedy*.

Algoritmo di Chvátal

1. $I = \{1, \dots, m\}$, $J = \{1, \dots, n\}$, $x := 0$.
2. Per ogni $j \in J$ calcola i costi unitari di copertura
$$u_j = \frac{c_j}{\sum\limits_{i \in I} a_{ij}}$$
3. Trova un indice $k \in J$ tale che $u_k = \min\limits_{j \in J} u_j$
   Poni $x_k := 1$, da J elimina k, da I elimina $\{i : \ a_{ik} = 1\}$
4. Se $I = \emptyset$ allora STOP (x é una copertura)
5. Se $J = \emptyset$ allora STOP (non esistono coperture)

$v_S(P)$: un algoritmo *greedy*.

Algoritmo di Chvátal

1. $I = \{1, \dots, m\}$, $J = \{1, \dots, n\}$, $x := 0$.
2. Per ogni $j \in J$ calcola i costi unitari di copertura
$$u_j = \frac{c_j}{\sum_{i \in I} a_{ij}}$$
3. Trova un indice $k \in J$ tale che $u_k = \min_{j \in J} u_j$
   Poni $x_k := 1$, da J elimina k, da I elimina $\{i : \; a_{ik} = 1\}$
4. Se $I = \emptyset$ allora STOP (x é una copertura)
5. Se $J = \emptyset$ allora STOP (non esistono coperture)
   altrimenti torna al passo 2.

Metodi euristici: algoritmo di Chvátal

Esempio

$I = \{1, \dots, 9\}$, $J = \{1, \dots, 8\}$.

Esempio

$I = \{1, \ldots, 9\}$, $J = \{1, \ldots, 8\}$.

	c_j	10	6	4	2	8	5	3	7
	$_i\backslash^j$	1	2	3	4	5	6	7	8
	1	1	1	0	0	0	1	1	0
	2	0	0	1	0	1	0	0	1
	3	1	1	1	0	0	0	0	1
	4	0	1	0	0	1	0	0	0
A	5	1	0	0	1	0	0	1	0
	6	1	1	0	0	0	1	1	1
	7	1	0	0	1	0	1	0	0
	8	1	0	0	0	0	1	1	1
	9	0	1	0	0	1	0	1	0

Metodi euristici: algoritmo di Chvátal

Esempio

$I = \{1, \dots, 9\}$, $J = \{1, \dots, 8\}$.

	c_j	10	6	4	2	8	5	3	7
	$i \backslash j$	1	2	3	4	5	6	7	8
	1	1	1	0	0	0	1	1	0
	2	0	0	1	0	1	0	0	1
	3	1	1	1	0	0	0	0	1
	4	0	1	0	0	1	0	0	0
A	5	1	0	0	1	0	0	1	0
	6	1	1	0	0	0	1	1	1
	7	1	0	0	1	0	1	0	0
	8	1	0	0	0	0	1	1	1
	9	0	1	0	0	1	0	1	0

Costi unitari:

j	1	2	3	4	5	6	7	8
u_j	$\frac{10}{6}$	$\frac{6}{5}$	2	1	$\frac{8}{3}$	$\frac{5}{4}$	$\frac{3}{5}$	$\frac{7}{4}$

$u_7 = \min u_j \to x_7 = 1$

Metodi euristici: algoritmo di Chvátal

Elimino la colonna 7 e le righe 1,5,6,8,9:

Elimino la colonna 7 e le righe 1,5,6,8,9:

c_j	10	6	4	2	8	5	7
$_i\backslash^j$	1	2	3	4	5	6	8
2	0	0	1	0	1	0	1
3	1	1	1	0	0	0	1
4	0	1	0	0	1	0	0
7	1	0	0	1	0	1	0

Elimino la colonna 7 e le righe 1,5,6,8,9:

c_j	10	6	4	2	8	5	7
$_i\backslash^j$	1	2	3	4	5	6	8
2	0	0	1	0	1	0	1
3	1	1	1	0	0	0	1
4	0	1	0	0	1	0	0
7	1	0	0	1	0	1	0

Costi unitari:

j	1	2	3	4	5	6	8
u_j	5	3	2	2	4	5	7/2

$u_3 = \min u_j \to x_3 = 1.$

Elimino la colonna 7 e le righe 1,5,6,8,9:

c_j	10	6	4	2	8	5	7
$i \backslash j$	1	2	3	4	5	6	8
2	0	0	1	0	1	0	1
3	1	1	1	0	0	0	1
4	0	1	0	0	1	0	0
7	1	0	0	1	0	1	0

Costi unitari:

j	1	2	3	4	5	6	8
u_j	5	3	2	2	4	5	$7/2$

$u_3 = \min u_j \to x_3 = 1$.

In seguito si trova $x_4 = 1$ e $x_2 = 1$.

Metodi euristici: algoritmo di Chvátal

Elimino la colonna 7 e le righe 1,5,6,8,9:

c_j	10	6	4	2	8	5	7
$_i\backslash^j$	1	2	3	4	5	6	8
2	0	0	1	0	1	0	1
3	1	1	1	0	0	0	1
4	0	1	0	0	1	0	0
7	1	0	0	1	0	1	0

Costi unitari:

j	1	2	3	4	5	6	8
u_j	5	3	2	2	4	5	7/2

$u_3 = \min u_j \to x_3 = 1$.

In seguito si trova $x_4 = 1$ e $x_2 = 1$. L'algoritmo di Chvátal trova la copertura $(0, 1, 1, 1, 0, 0, 1, 0)$ di costo 15.

Metodi euristici: arrotondamento

Un altro metodo euristico é risolvere il rilassamento continuo e arrotondare per eccesso le componenti frazionarie.

Metodi euristici: arrotondamento

Un altro metodo euristico é risolvere il rilassamento continuo e arrotondare per eccesso le componenti frazionarie.

Esempio

	c_j	10	6	4	2	8	5	3	7
	$_i\backslash^j$	1	2	3	4	5	6	7	8
	1	1	1	0	0	0	1	1	0
	2	0	0	1	0	1	0	0	1
	3	1	1	1	0	0	0	0	1
	4	0	1	0	0	1	0	0	0
A	5	1	0	0	1	0	0	1	0
	6	1	1	0	0	0	1	1	1
	7	1	0	0	1	0	1	0	0
	8	1	0	0	0	0	1	1	1
	9	0	1	0	0	1	0	1	0

Metodi euristici: arrotondamento

Un altro metodo euristico é risolvere il rilassamento continuo e arrotondare per eccesso le componenti frazionarie.

Esempio

	c_j	10	6	4	2	8	5	3	7
	$_i\backslash^j$	1	2	3	4	5	6	7	8
	1	1	1	0	0	0	1	1	0
	2	0	0	1	0	1	0	0	1
	3	1	1	1	0	0	0	0	1
	4	0	1	0	0	1	0	0	0
A	5	1	0	0	1	0	0	1	0
	6	1	1	0	0	0	1	1	1
	7	1	0	0	1	0	1	0	0
	8	1	0	0	0	0	1	1	1
	9	0	1	0	0	1	0	1	0

L'ottimo del rilassamento continuo é $\left(0, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, 0\right)$,

Metodi euristici: arrotondamento

Un altro metodo euristico é risolvere il rilassamento continuo e arrotondare per eccesso le componenti frazionarie.

Esempio

	c_j	10	6	4	2	8	5	3	7
	$_i\backslash^j$	1	2	3	4	5	6	7	8
	1	1	1	0	0	0	1	1	0
	2	0	0	1	0	1	0	0	1
	3	1	1	1	0	0	0	0	1
	4	0	1	0	0	1	0	0	0
A	5	1	0	0	1	0	0	1	0
	6	1	1	0	0	0	1	1	1
	7	1	0	0	1	0	1	0	0
	8	1	0	0	0	0	1	1	1
	9	0	1	0	0	1	0	1	0

L'ottimo del rilassamento continuo é $\left(0, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, 0\right)$,quindi $(0, 1, 1, 1, 1, 1, 1, 0)$ é una copertura di costo 28.