Appunti di ricerca operativa

A cura di Gemma Martini

2 novembre 2016

Indice

Un grazie ad Aldo D'Aquino,
che mi ha fornito gli appunti
delle lezioni a cui sono mancata.

1 Introduzione

1.1 La disciplina

La definizione dell'INFORMS (INstitute For Operations Research and Management Sciences) qualifica la ricerca operativa come la disciplina che "ha lo scopo di fornire basi razionali al processo decisionale cercando di comprendere e strutturare situazioni complesse e di utilizzare questa comprensione per prevedere il comportamento dei sistemi e migliorarne le prestazioni. [...]"
Nella ricerca operativa i compiti sono molti: è necessario analizzare una situazione reale e tradurla in un modello matematico facile da utilizzare e che, a sua volta, deve essere tradotto in un algoritmo. I risultati ottenuti devono poi essere analizzati, per valutarne la consistenza con il problema reale.

1.2 Il lessico

Prima di addentrarci in un esempio concreto è utile soffermarsi sulla terminologia della materia:

Dati vs Grandezze Decisionali I primi sono valori che non possono essere modificati dall'analista, mentre le seconde sono grandezze i cui valori possono essere modificati dall'analista.

Parametri Decisionali vs Variabili Decisionali I primi possono essere settati dall'analista a piacere, mentre le seconde sono determinate dal modello.

Vincoli strutturali Sono i vincoli che legano tra loro dati, variabili e parametri.

Obiettivi Sono funzioni delle grandezze del problema da massimizzare o minimizzare.

Vincoli flessibili Ulteriori vincoli che servono per indirizzare le decisioni in una certa direzione.

1.3 Problemi e modelli

Quando un problema reale può essere tradotto mediante un modello matematico e la funzione obiettivo ed i vincoli possono essere scritti sottoforma di equazioni/disequazioni lineari si parla di un Problema di Programmazione Lineare, che verrà approfondito in seguito.

1.3.1 Problema di produzione

In una fabbrica di torte l'addetto alla gestione economica ha i seguenti dati: la torta A dà un guadagno di 1€ per quintale, mentre la torta B dà un guadagno di 2€ per quintale. Inoltre, il consumo di ingredienti è riportato in tabella in kg/q

Ingrediente	A	B
Mandorle	12	5
Nocciole	1	5

La disponibilità giornaliera degli ingredienti va come segue: Mandorle 48kg, Nocciole 15kg.
Possiamo schematizzare il problema come segue:

FUNZIONE OBIETTIVO Massimizzare il guadagno

VINCOLI DEL PROBLEMA Disponibilità giornaliera degli ingredienti

SOLUZIONE AMMISSIBILE Una soluzione che rispetta i vincoli del problema

SOLUZIONE OTTIMA La soluzione che massimizza (o minimizza) la funzione obbiettivo

Lo svolgimento del problema consta la seguente sequenza di operazioni:

RACCOLTA DI DATI ossia l'individuazione della funzione obiettivo e dei vincoli

COSTRUZIONE DI UN MODELLO

$$\begin{cases} \max cx \\ Ax \leq b \end{cases}$$

Siano x_A e x_B le quantità in quintali rispettivamente di torta A e torta B prodotte.
Chiamiamo c_1 e c_2 il guadagno al quintale rispettivamente di torta A e torta B.
Sia $A = \begin{pmatrix} 12 & 5 \\ 1 & 5 \end{pmatrix}$ la tabella del consumo di ingredienti, sottoforma di matrice e sia $b = \begin{pmatrix} 48 \\ 15 \end{pmatrix}$ il vettore delle limitazioni giornaliere. Riunendo tutte le informazioni che abbiamo sul problema si ottiene il seguente **Modello**, che, può essere portato nella forma $\begin{cases} \max cx \\ Ax \leq b \end{cases}$, se A e b sono fatti come appare in **Dati**.

Modello

$$\begin{cases} \max c_1x_A + c_2x_B \leftarrow \text{massimizza il guadagno} \\ 12x_A + 5x_B \leq 48 \leftarrow \text{consumo di mandorle} \\ x_A + 5x_B \leq 15 \leftarrow \text{consumo di noci} \\ x_A \geq 0 \leftarrow x_A = \text{quintali di torta A} \\ x_B \geq 0 \leftarrow x_B = \text{quintali di torta B} \end{cases}$$

Dati

$x = \begin{pmatrix} x_A & x_B \end{pmatrix}$ $c = \begin{pmatrix} 1000 & 2000 \end{pmatrix}$
$A = \begin{pmatrix} 12 & 5 \\ 1 & 5 \end{pmatrix}$ $b = \begin{pmatrix} 48 \\ 15 \end{pmatrix}$
Devo imporre x_A, $x_B \geq 0$

$$A = \begin{pmatrix} 12 & 5 \\ 1 & 5 \\ -1 & 0 \\ 0 & -1 \end{pmatrix} \quad b = \begin{pmatrix} 18 \\ 15 \\ 0 \\ 0 \end{pmatrix}$$

1.3.2 Problema di trasporto

Un **problema di trasporto** ha m origini ed n destinazioni, con m in generale diverso da n, ed inoltre le origini dispongono di un numero di oggetti che può essere diverso dall'unità così come anche le richieste delle destinazioni. Un problema di trasporto può essere enunciato come segue.
Supponiamo di avere:

- m centri di produzione (per esempio industrie) che indichiamo con 0_i (origini), con $i = 1, \cdots, m$.
- n centri di consumo (per esempio depositi) che indichiamo con D_j (destinazioni), con $j = 1, \cdots, n$.

Sono inoltre noti:

- La matrice $C \in \mathcal{M}^{m \times n}$ dei costi unitari di trasporto da ogni centro di produzione ad ogni centro di consumo
- La disponibilità dei vari centri di produzione d_i
- La richiesta dei vari centri di consumo r_j

Nell'ipotesi che la somma delle quantità disponibili alle origini uguagli la somma delle richieste delle destinazioni, si devono distribuire le quantità di prodotto disponibili alle origini tra i vari centri di consumo in modo da minimizzare il costo totale di trasporto, rispettando i seguenti vincoli:

1. La quantità di merce spedita da ogni origine non può superare la disponibilità in quella origine;
2. La quantità di merce ricevuta da ogni centro di consumo non può superare la sua richiesta.

Questo problema è risolvibile come un problema di programmazione lineare: Siano x_{ij}, con $i = 1, 2, .., m$ e $j = 1, 2, ..., n$ la quantità di merce spedita dal generico centro i di produzione, al generico centro j di consumo. Il problema si traduce nel minimizzare la funzione **costo**

$\min Z = \sum_{ij} c_{ij} x_{ij}$ soggetta ai seguenti vincoli: $\begin{cases} \sum_j x_{ij} = d_i \; \forall i = 1, \cdots, m \\ \sum_i x_{ij} = r_j \; \forall j = 1, \cdots, n \\ x_{ij} \geq 0 \; \forall i, j \end{cases}$

2 Programmazione lineare

2.0.1 Problema di programmazione lineare

Si definisce **problema di programmazione lineare**(**PL**) un problema con funzioni lineari e vincoli lineari $(=, \leq, \geq)$.
Sia $f : \mathbb{R}^n \to \mathbb{R}$, $f(x) = cx$, con $c \in \mathbb{R}^n$. Un problema di PL si rappresenta come:

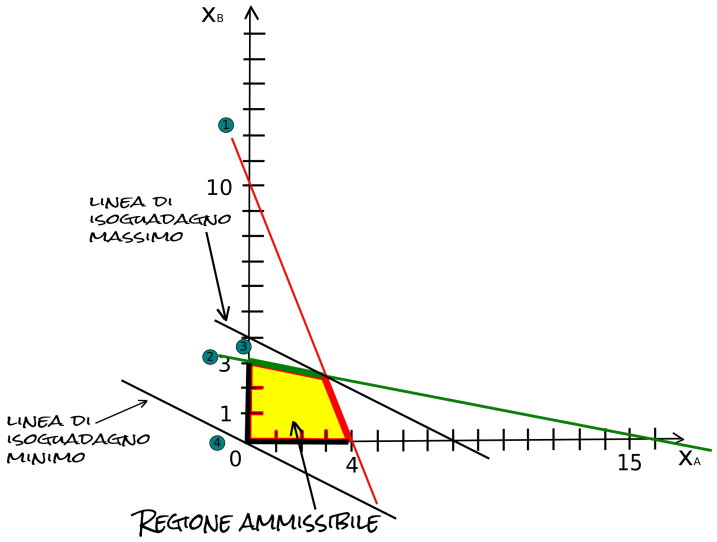
$$\begin{cases} \max f(x) \\ Ax \leq b \end{cases}$$

2.0.2 Risoluzione geometrica di un problema di programmazione lineare in due variabili

Vincoli

1. $12x_A + 5x_B \leq 48$
2. $x_A + 5x_B \leq 15$
3. $x_A \geq 0$
4. $x_B \geq 0$

$Ax \leq b$ si dice **poliedro** (o **poligono** nel caso di due dimensioni) ed è l'insieme delle soluzioni ammissibili



La **linea di isoguadagno** (nel nostro caso il fascio di rette $x_A + 2x_B = k$, in generale $f(x) = k$, con f la funzione da massimizzare) è tale che su ogni punto il guadagno è sempre lo stesso. Inoltre, nell'esempio, si ha che la linea di isoguadagno massimo è la linea interna o tangente al poligono più distante da 0. In generale, le linee di isoguadagno massimo e minimo intersecano vertice inferiore o superiore, in quanto il guadagno è continuo e vertice inferiore e superiore sono i valori estremi per k.

2.0.3 Problema di assegnamento

Lo scopo è quello di assegnare lavoro agli operai, secondo le seguenti richieste:

- Ogni operaio deve avere il suo lavoro
- Ogni lavoro deve essere effettuato da un solo operaio
- Il costo totale deve essere minimo.

Nella seguente tabella il costo di ogni operaio per effettuare i vari lavori.

Operaio/Lavori	L1	L2	L3	L4
O1	16	22	18	16
O2	19	21	23	20
O3	15	18	20	19
O4	13	19	22	23

Procedimento

1. Raccolta dati
2. Individuazione variabili
3. Individuazione funzione obiettivo
4. Individuazione vincoli
5. Algoritmo risolutivo
6. Soluzione numerica

Esempio di soluzione ammissibile: $x = (\overset{x_{11}x_{12}x_{13}x_{14}}{1\ 0\ 0\ 0} | \overset{x_{21}x_{22}x_{23}x_{24}}{0\ 1\ 0\ 0} | \overset{x_{31}x_{32}x_{33}x_{34}}{0\ 0\ 1\ 0} | \overset{x_{41}x_{42}x_{43}x_{44}}{0\ 0\ 0\ 1})$, dove x_{ij} è una variabile booleana che indica se l'operaio j svolge il lavoro i.
Esempio di soluzione **non** *ammissibile*: $x = (1100|0\cdots)$
Vincoli

1. Righe (ogni operaio un lavoro)

$$\begin{cases} x_{11} + x_{12} + x_{13} + x_{14} = 1 \\ x_{21} + x_{22} + x_{23} + x_{24} = 1 \\ x_{31} + x_{32} + x_{33} + x_{34} = 1 \\ x_{41} + x_{42} + x_{43} + x_{44} = 1 \end{cases}$$

2. Colonne (ogni lavoro un operaio)

$$\begin{cases} x_{11} + x_{21} + x_{31} + x_{41} = 1 \\ x_{12} + x_{22} + x_{32} + x_{42} = 1 \\ x_{13} + x_{23} + x_{33} + x_{43} = 1 \\ x_{14} + x_{24} + x_{34} + x_{44} = 1 \end{cases}$$

3. $xij \geq 0$

Funzione obiettivo: $\min cx = 16x_{11} + 22x_{12} + 18x_{13} + 16x_{14} + \cdots + 23x_{44}$

Osservazione 2.0.1. Tutti i problemi di programmazione lineare si possono risolvere con lo stesso modello:

$$(c, A, b) \begin{cases} \max cx \\ Ax \leq b \end{cases}$$

Infatti vale che:

- $\begin{cases} \min f(x) = -\max(-f(x)) \\ \arg\min(f(x)) = \arg\max(-f(x)) \end{cases}$
- $Ax \geq b \equiv -Ax \leq -b$

Nota: $\arg\max(f)$ è definita come l'argomento per il quale si realizza il massimo della funzione f.

- $Ax = b \equiv \begin{cases} Ax \leq b \\ -Ax \leq -b \end{cases}$
- Aggiungendo una variabile di scarto x_3 tale che $x_1 + 2x_2 \leq 3 \equiv x_1 + 2x_2 + x_3 = 3$, con $x_3 \geq 0$

Nel problema di esempio si ottiene la seguente matrice, scrivendo due volte i vincoli 1 e 2 con i segni opportuni, per ottenere le uguaglianze e una volta il vincolo 3, come segue:

$$\overbrace{\left(\begin{array}{c|cccccccccccccccc} \text{righe (8)} & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ & -1 & -1 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ & & & & & & & & \vdots & & & & & & & & \\ & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -1 & -1 & -1 \\ \hline \text{colonne (8)} & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ & -1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ & & & & & & & & \vdots & & & & & & & & \\ & 0 & 0 & 0 & -1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & -1 \\ \hline x_{ij} \geq 0 \text{ (16)} & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ & & & & & & & & \vdots & & & & & & & & \\ & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{array}\right)}^{A} \left(\begin{array}{c} x_{11} \\ x_{12} \\ x_{13} \\ x_{14} \\ x_{21} \\ x_{22} \\ x_{23} \\ x_{24} \\ x_{31} \\ x_{32} \\ x_{33} \\ x_{34} \\ x_{41} \\ x_{42} \\ x_{43} \\ x_{44} \end{array}\right) \leq \left(\begin{array}{c} 1 \\ -1 \\ 1 \\ \vdots \\ -1 \\ \hline 1 \\ -1 \\ 1 \\ \vdots \\ -1 \\ \hline 0 \\ 0 \\ 0 \\ \vdots \\ 0 \end{array}\right)$$

3 Definizioni per introdurre gli algoritmi

3.1 Convessità

Definizione 3.1 (Insieme convesso)
In uno spazio euclideo un **insieme convesso** è un insieme nel quale, per ogni coppia di punti, il segmento che li congiunge è interamente contenuto nell'insieme.

Definizione 3.2 (Combinazione convessa)
Una **combinazione convessa** è una combinazione lineare di vettori fatta con coefficienti non negativi a somma 1. In simboli, una **combinazione convessa** degli x_i è
$\sum_{i=1}^{m} \lambda_i x_i$ con $\sum_{i=1}^{m} \lambda_i = 1$ e $\lambda_i \geq 0 \forall i$.

Definizione 3.3 (Inviluppo convesso)
L'**inviluppo convesso** di un insieme $X = \{x^1, \cdots, x^s\} \subset \mathbb{R}^n$ è l'insieme di tutte le combinazioni convesse degli elementi di X, in simboli $conv(X) = \{x_j = \sum_{i=1}^{s} \lambda_i x_i, \lambda_i \in \mathbb{R}^+ \cup \{0\} i = 1, \cdots, s : \sum_{i=1}^{s} \lambda_i = 1\}$.

Osservazione 3.1.1. Si può osservare che dire **combinazione convessa** è equivalente a dire combinazione lineare positiva e affine. Inoltre il nome **convessa** viene dal fatto che

l'insieme di tutte le combinazioni convesse di un certo insieme di punti, al variare dei coefficienti, coincide con l'inviluppo convesso di quell'insieme.

3.2 Coni e poliedri

Definizione 3.4 (Cono)
Un insieme C si dice **cono** se $\forall x \in C$, $\alpha \geq 0 \Rightarrow \alpha x \in C$.

Definizione 3.5 (Inviluppo conico)
Sia $V = \{v_1, \cdots, v_n\} \in \mathbb{R}$ un insieme finito di vettori. Si definisce **inviluppo conico** di V $cono(V) = \{v = \sum_{i=1}^{t} \nu_i v_i : \nu_i \geq 0 \text{ per } i = 1, \cdots, t\}$.

Definizione 3.6 (Combinazione conica)
Un vettore $x \in \mathbb{R}^n$ si dice **combinazione conica** dei vettori $a_1, \cdots, a_m \in \mathbb{R}^n$ con coefficienti $\lambda_1, \cdots, \lambda_m \in \mathbb{R}$ se $x = \lambda_1 a_1 + \cdots + \lambda_m a_m$ e $\lambda_1, \cdots, \lambda_m > 0$.

Osservazione 3.2.1. Si definisce **cone{**$x_1, x_2, \cdots, x_k$**}** l'insieme di tutte le possibili combinazioni coniche di $x_1, x_2, \cdots, x_k$.

Definizione 3.7 (Poliedro)
Un insieme P si dice **poliedro** se è esprimibile come intersezione di un numero finito m di semispazi chiusi. Equivalentemente, un insieme P si dice **poliedro** se $\exists A \in \mathbb{R}^{m \times n}$, $\exists b \in \mathbb{R}^m$ *t.c.* $P = \{x : Ax \geq b\}$.

Osservazione 3.2.2. I poliedri sono tutti **convessi**, ma **non** sono tutti limitati.

Definizione 3.8 (Vertice)
Sia $x \in P$. x si dice **vertice** o **punto estremo** se non può essere espresso come combinazione convessa di due punti diversi di P, ossia due soluzioni ammissibili nel caso di un problema di programmazione lineare. In simboli, x è un vertice se $x \in conv(\{x', x''\})$, $x' \in P$, $x'' \in P \Rightarrow x' = x''$.

Osservazione 3.2.3. In un poliedro, un **vertice** soddisfa due disequazioni, mentre uno **spigolo** una sola.

Definizione 3.9 (Direzione di recessione di un poliedro)
Una **direzione di recessione** per un poliedro P è un vettore d se P contiene tutte le semirette di direzione d uscenti da punti appartenenti a P. In simboli, d è una direzione di recessione se vale $x + td \in P \quad \forall x \in P, \forall t \geq 0$. L'insieme delle direzioni di recessione di P viene denotato con **rec (P)**.
È facile osservare che un poliedro limitato non ha direzioni di recessione non nulle.

Definizione 3.10 (Direzione di linealità)
Una **direzione di linealità** per un poliedro P è un vettore d *t.c.* $d \in rec(P)$, $-d \in rec(P)$.

Osservazione 3.2.4. Si può dimostrare che d è una direzione di linealità se e solo se $Ad = 0$. Inoltre se P ha direzioni di linealità allora non può avere vertici.

3.3 Forma primale standard

Definizione 3.11 (Problema in forma primale standard)
Un **problema in forma primale standard** è un problema dove si vuole massimizzare

il valore di una certa funzione rispettando dei vincoli aggiuntivi espressi sotto forma di disequazioni lineari. L'insieme dello spazio delle possibili soluzioni del problema prende il nome di **poliedro o politopo primale**, mentre la funzione che si massimizza è detta **funzione obiettivo**. La scrittura con la quale si è soliti rappresentare un problema in formato primale è

$$(P) : \begin{cases} \max c^T x \\ Ax \leq b \end{cases}$$

Che può essere scritto in forma scalare come

$$\begin{cases} \max \sum_{i=1}^{n} c_i x_i \ (funzione\ obiettivo) \\ \sum_{i=1}^{n} a_{1,i} x_i \leq b_1 \\ \dots \qquad\qquad (vincoli) \\ \sum_{i=1}^{n} a_{m,i} x_i \leq b_m \end{cases}$$

Quindi il **problema primale standard** è costituito da una certa funzione che va massimizzata (la funzione obiettivo), ed una serie di vincoli (il poliedro primale); il problema nel suo complesso viene indicato con (P), il valore ottimo del problema (P) si indica con $v(P)$ mentre P è il simbolo associato al poliedro primale.

Definizione 3.12 (Soluzione di base ammissibile)
Una soluzione di base si dice **ammissibile** se appartiene al poliedro, ossia soddisfa tutte le disuguaglianze.

Osservazione 3.3.1. Sia P un sistema di m disequazioni in n variabili, con $n \leq m$. Per ogni raggruppamento di n disequazioni esiste un unico punto che le soddisfa come equazioni, se questo punto soddisfa anche le rimanenti disequazioni (come tali) questo punto è un vertice.

Definizione 3.13 (Soluzione di base degenere)
Una soluzione di base si dice **degenere** quando tale soluzione è generata da più basi.

Definizione 3.14 (Vertice degenere)
Sia P un poliedro generato da m disequazioni in n incognite, con $n \leq m$. Un vertice che soddisfa come uguaglianze più di n disequazioni è detto **degenere**.

4 Teoremi

Teorema 4.1 (Teorema di Weyl o teorema di rappresentazione dei poliedri)
Sia $P \equiv (Ax \leq B), \exists V = \{v_1, v_2, \cdots, v_k\} \subset P$ *e* $\exists E = \{e_1, e_2, \cdots, e_p\} \subset \mathbb{R}^n$ *tali che* $P = conv(V) + cone(E) = \{a + b : a \in conv(V), b \in cone(E)\}$

Corollario 4.2
La regione ammissibile di un problema di programmazione lineare è un poliedro

Teorema 4.3 (teorema fondamentale della programmazione lineare)
Sia dato un problema in formato primale standard $(P) = \begin{cases} \max & cx \\ Ax \leq & b \end{cases}$.

Se $P \neq \varnothing$, $(P) < +\infty$, P ha vertici allora esiste almeno un vertice ottimo, ossia $\exists r \in \{1, \cdots, k\}$ *tale che* $\max_{x \in P} cx = cv_r$.

Dimostrazione costruttiva. $x = \sum_{i=1}^{k} \lambda_i v_i + \sum_{j=1}^{p} \mu_j e_j \in P = conv(V) + cone(E)$, con $\lambda_i \in [0,1]$ e $\sum_{i=1}^{k} \lambda_i = 1, \mu_j \geq 0$

Calcolo la funzione obiettivo: $cx = c(\sum_{i=1}^{k} \lambda_i v_i + \sum_{j=1}^{p} \mu_j e_j) = c(\sum_{i=1}^{k} \lambda_i v_i) + c(\sum_{j=1}^{p} \mu_j e_j) = \sum_{i=1}^{k} \lambda_i (cv_i) + \sum_{j=1}^{p} \mu_j (ce_j) = \lambda_1 cv_1 + \cdots + \lambda_k cv_k + \mu_1 ce_1 + \cdots + \mu_p ce_p$

Poichè $v(P) < +\infty$ $ce_i < 0$, altrimenti scegliendo μ_i arbitrariamente grande si troverebbe un x arbitrariamente grande (assurdo), $\max_{x \in P} cx$ si realizza per $\mu_i = 0 \ \forall i$

Tra i v_s, che sono un numero finito $\exists r \in \{1, \cdots, k\} : cv_r \geq cv_s \forall s \in \{1, \cdots, r\}$, quindi si ha quanto segue

$$\max_{x \in P} cx = \max_{\lambda} \sum_{i=1}^{k} \lambda_i (cv_i) \leq \max_{\lambda} \sum_{i=1}^{k} \lambda_i cv_r = cv_r = \max cv_i \leq \max_{x \in P} cx \qquad \square$$

Osservazione 4.0.1. Non sarebbe una buona idea esaminare i valori della funzione obiettivo su tutti i vertici, poichè sono esponenziali nel numero di vincoli del problema, quindi richiedono asintoticamente troppi casi da valutare.

Teorema 4.4
Un punto x del poliedro primale standard P è un vertice sse esso è una soluzione di base ammissibile.

Osservazione 4.0.2. Questo teorema concilia le due visioni algebrica e geometrica di un vertice del poliedro. Infatti, dalla definizione 3.8 non è immediato dedurre che un vertice è una soluzione ammissibile del problema.

5 Esempi sulla programmazione lineare

$v_1 = \begin{pmatrix} -2 & 1 \end{pmatrix}$, $e_1 = \begin{pmatrix} 0 & -1 \end{pmatrix}$, $v_2 = \begin{pmatrix} 0 & 1 \end{pmatrix}$, $e_2 = \begin{pmatrix} 1 & -1 \end{pmatrix}$, $c = \begin{pmatrix} 1 & 1 \end{pmatrix}$.

Verifica

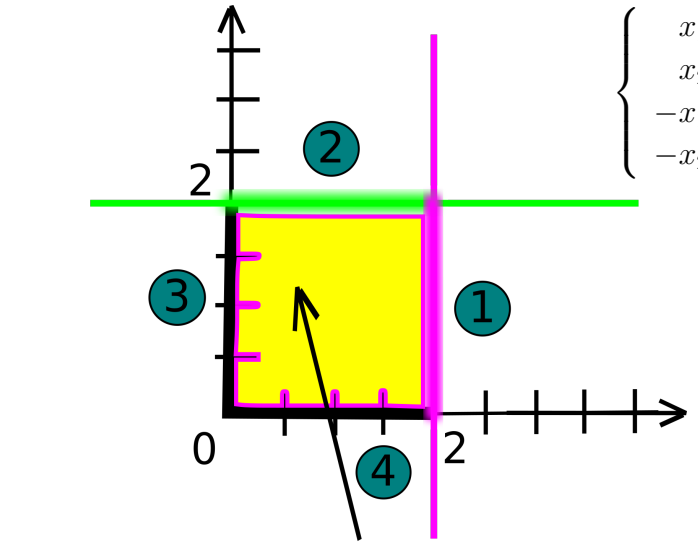
$$c \begin{cases} \nearrow \begin{cases} ce_1 = -1 \leq 0 \\ ce_2 = 0 \leq 0 \end{cases} \Rightarrow cx \neq +\infty \\ \searrow \max \begin{cases} cv_1 = -1 \\ cv_2 = 1 \end{cases} \Rightarrow v_2 \text{ è il massimo} \end{cases}$$

$(A, b) \to (V, E)$

Sia $A \in M^{m \times n}$, con $m > n$ e tale che $rk(A) = n$. Sia $\mathscr{B} \subseteq \{1, 2, \cdots, n\}$, $|\mathscr{B}| = n$, $detA_{\mathscr{B}} \neq 0$

$x = A_{\mathscr{B}}^{-1} b_{\mathscr{B}} \Rightarrow A_{\mathscr{B}} x = b_{\mathscr{B}}$

Esercizio 5.1. ***Esempio del calcolo dei vertici***



$$\begin{cases} x_1 \leq 2 & (1) \\ x_2 \leq 2 & (2) \\ -x_1 \leq 0 & (3) \\ -x_2 \leq 0 & (4) \end{cases}$$

REGIONE AMMISSIBILE

Dove, (1) *e* (2) *sono i vincoli di base* $\mathscr{B}$ *e* (3) *e* (4) *sono i vincoli non di base* $\mathscr{B}$*, che devo verificare.*

$A_{\mathscr{B}} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$, $b_{\mathscr{B}} = \begin{pmatrix} 2 \\ 2 \end{pmatrix}$, *dove* $A_{\mathscr{B}}$ *è la* ***matrice di base*** *ed è invertibile e tale che* $det(A_{\mathscr{B}}) = 1$.

Calcolo x_1 *e* x_2*:*

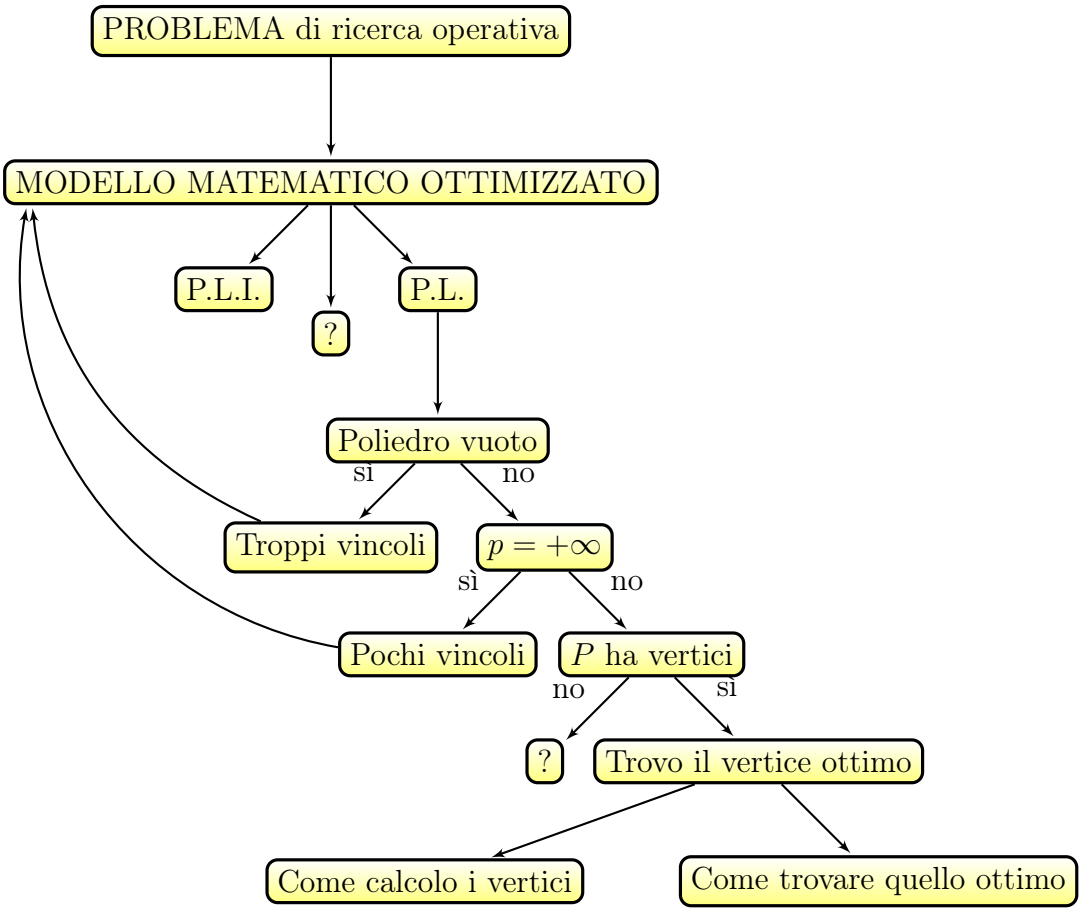
$$A_{\mathscr{B}}x = b \Rightarrow \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 2 \\ 2 \end{pmatrix} \Rightarrow x = \begin{pmatrix} 2 \\ 2 \end{pmatrix}$$

Verifico i vincoli non di base

Vogliamo dimostrare che $A_{\mathscr{N}}x \leq b_{\mathscr{N}}$*, ma sappiamo che* $x = A_{\mathscr{B}}^{-1}b_{\mathscr{B}}$*, quindi*

$$A_{\mathscr{N}}(A_{\mathscr{B}}^{-1}b_{\mathscr{B}}) \leq b_{\mathscr{N}} \begin{cases} -x_1 \leq 0 \\ -x_2 \leq 0 \end{cases} \rightarrow \begin{cases} -2 \leq 0 \\ -2 \leq 0 \end{cases}$$ *ok!* $\Rightarrow \mathscr{B}$ *è una base (non degenere).*

Approcciarsi al problema



6 Teoria della dualità

Dato un problema di ricerca operativa esiste il suo **duale**, ossia un problema strettamente collegato a quello iniziale, come segue. Dato il problema di **minimizzare** i costi di produzione di due prodotti, sapendo che devono essere prodotti un certo numero di esemplari e che le fasi di produzione avvengono su periferiche diverse a prezzi diversi, il suo problema duale è **massimizzare il guadagno** nel noleggiare la propria forza lavoro per produrre la solita tipologia di oggetti, supponendo che la concorrenza abbia gli stessi costi di produzione.
La forma di dualità più importante e più conosciuta è quella che associa due problemi di Programmazione Lineare (dualità lineare) ed è quella che serve a stabilire condizioni di ottimalità per problemi in forma primale standard. La coppia di problemi che studieremo è la seguente (dove $x, c^t \in \mathbb{R}^n$ e $y^t, b \in \mathbb{R}^m$ e $A \in \mathscr{M}^{m\times n}$):

Problema primale standard.

$$\begin{cases} \max \sum_{i=1}^{n} c_i x_i \\ \sum_{i=1}^{n} a_{1,i} x_i \leq b_1 \\ \dots \\ \sum_{i=1}^{n} a_{m,i} x_i \leq b_m \end{cases}$$

Problema duale standard.

$$\begin{cases} \min \sum_{j=1}^{m} y_j b_j \quad (funzione\, obiettivo) \\ \sum_{j=1}^{m} y_j a_{j,1} = c_1 \\ \dots \qquad\qquad (vincoli) \\ \sum_{j=1}^{m} y_j a_{j,n} = c_n \\ y_j \geq 0 \qquad j = 1, \dots, m \end{cases}$$

Quindi se consideriamo un problema di PL in formato primale standard possiamo associarvi un problema di PL in una forma duale standard; sfruttando i concetti di matrici e vettori possiamo scrivere questa corrispondenza così:

$$\begin{cases} \max \quad cx \\ Ax \leq b \end{cases}$$

$$\begin{cases} \min \quad yb \\ yA = c \\ y \geq 0 \end{cases}$$

Come si trasforma l'uno nell'altro.

$$\begin{cases} \max(\overset{c_1}{5} x_1 \ \overset{c_2}{-6}\, x_2) \\ 3x_1 + 2x_2 \leq 8 \\ -x_1 + 2x_2 \leq 5 \\ x_1 - 3x_2 \leq 7 \\ x_1 + 7x_2 \leq 9 \end{cases} \rightarrow \begin{cases} \min(8y_1 + 5y_2 + 7y_3 + 9y_4) \\ 3y_1 - y_2 + y_3 + y_4 \leq 5 \\ 2y_1 + 2y_2 - 3y_3 + 7y_4 \leq -6 \\ y \geq 0 \end{cases}$$

Dove $A = \begin{pmatrix} 3 & 2 \\ -1 & 2 \\ 1 & -3 \\ 1 & 7 \end{pmatrix}$, $x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$, $y = \begin{pmatrix} y_1 & y_2 & y_3 & y_4 \end{pmatrix}$, $c = \begin{pmatrix} 5 & -6 \end{pmatrix}$, $b = \begin{pmatrix} 8 \\ 5 \\ 7 \\ 9 \end{pmatrix}$

Ai fini della creazione di un test di ottimalità introduciamo solamente alcuni brevi elementi teorici a proposito del problema duale standard:

- Mentre il problema in forma primale standard si denotava (P), quello in forma duale standard si chiama (D)
- Il valore ottimo del duale è indicato con $v(D)$
- Il poliedro duale è definito come $D = \{yA = c,\, y \geq 0\}$
- Per minimizzare la funzione obiettivo del problema duale si ricercano delle soluzioni di base $y = \begin{pmatrix} y_{\mathscr{B}} & y_{\mathscr{N}} \end{pmatrix}$, tali che $y_{\mathscr{B}} = cA_{\mathscr{B}}^{-1}$, $y_{\mathscr{N}} = 0$
- Le soluzioni di base, relative ad una base $\mathscr{B}$, possono essere **ammissibili** (se rispettano $\forall i \in \mathscr{B}\ y_i \geq 0$), prendendo il nome di vertici del poliedro duale
- I vertici duali possono essere **degeneri** (se rispettano la condizione $\exists i \in \mathscr{B} : y_i = 0$).

Un'importante proprietà dei problemi duali lineari è che il duale del duale è il primale, cioè riconducendo il problema duale alla forma primale e applicando la corrispondenza di cui sopra si riotterrà una formulazione equivalente del problema primale. Questo significa che l'operazione di associazione primale-duale è un'operazione involutoria, cioè se applicata a se stessa torna la situazione di base.

Teorema 6.1 (Teorema della dualità debole)

Siano $P,\ D \neq \emptyset$, *tali che* $P = \begin{cases} \max cx \\ Ax \leq b \end{cases}$ *e* $D = \begin{cases} \min yb \\ yA = c \\ y \geq 0 \end{cases}$.

Sia X *la regione ammissibile di* P *e sia* Y *la regione ammissibile di* D.
Allora $\forall x \in X,\ \forall y \in Y$ *vale*

$$cx \leq yb$$

Teorema 6.2 (Teorema della dualità forte)

Siano $P,\ D \neq \emptyset$. *Allora* $P = \begin{cases} \max cx \\ Ax \leq b \end{cases} \equiv D = \begin{cases} \min yb \\ yA = c \\ y \geq 0 \end{cases}$

Trovare i vertici del poliedro duale.
Sia $y = \begin{pmatrix} y_{\mathscr{B}} & y_{\mathscr{N}} \end{pmatrix}$:

1. pongo $y_{\mathscr{N}} = 0$
2. risolvo $y_{\mathscr{B}} A_{\mathscr{B}} = c$, con $y \geq 0 \rightarrow y_{\mathscr{B}} = cA_{\mathscr{B}}^{-1} \geq 0$
3. verifico se è ammissibile

Analogamente a quanto detto nel teorema 4.4, si ha il 6.3

Teorema 6.3

Un punto y *del poliedro* D *è un vertice del poliedro duale standard sse esso è una soluzione di base duale ammissibile.*

Definizione 6.1 (Base degenere)
Una base $\mathscr{B}$ si dice **degenere** quando una delle componenti di $y_{\mathscr{B}}$ è zero, ovvero quando due basi diverse generano la stessa soluzione di base.

Definizione 6.2 (Soluzioni complementari)
Le soluzioni $\bar{x} \in \mathbb{R}^n$ e $\bar{y}^t \in \mathbb{R}^m$ formano una **coppia di soluzioni complementari** se $\bar{y}A = c$ e $A\bar{x} \leq b$ e viene verificata la seguente proprietà, detta degli scarti complementari:

$$\bar{y}(b - A\bar{x}) = 0$$

Teorema 6.4 (Teorema degli scarti complementari)
Date due soluzioni $\bar{x}$ e $\bar{y}$, ammissibili rispettivamente per P e D, esse sono ottime se e solo se verificano le condizioni degli scarti complementari.

Corollario 6.5
Date due soluzioni $\bar{x}$ e $\bar{y}$, ammissibili rispettivamente per P e D, esse sono ottime se vale $\forall i \in \mathscr{B}$

$$\bar{y}_i(b_i - A_i\bar{x}) = 0$$

Dimostrazione. Infatti, siano $\bar{x} \in \mathbb{R}^n$ e $\bar{y}^t \in \mathbb{R}^m$, esplicitando il prodotto tra vettori, la proprietà degli scarti complementari può essere riscritta come

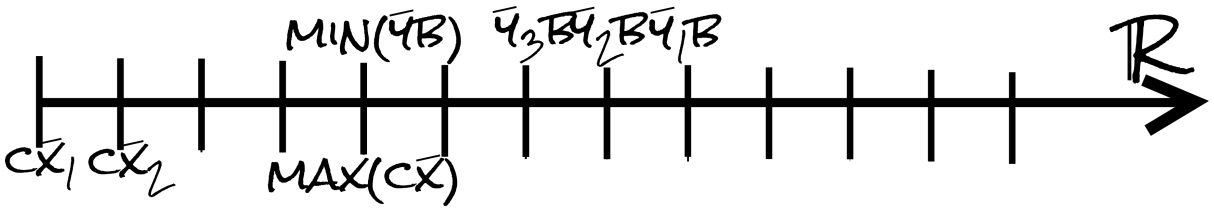
$$\bar{y}(b - A\bar{x}) = \sum_{i=1}^{m} \bar{y}_i(b_i - A_i\bar{x}) = 0$$

Poichè la somma di termini nulli dà a sua volta zero si ha la tesi. □

Test di ottimalità(criterio di STOP).

Sia $\mathscr{B}$ una base si ha

$$P \equiv \begin{cases} \max cx \\ Ax \leq b \end{cases} \quad \rightarrow \bar{x} = A_{\mathscr{B}}^{-1} b_{\mathscr{B}} \qquad D \equiv \begin{cases} \min yb \\ yA = c \\ y \geq 0 \end{cases} \quad \rightarrow \bar{y} = (cA_{\mathscr{B}}^{-1}, 0)$$



$\max_{x \in P} cx = \min_{y \in D} yb \Rightarrow c\bar{x} \leq \bar{y}b$
$c\bar{x} = c(A_{\mathscr{B}}^{-1} b_{\mathscr{B}}) = (cA_{\mathscr{B}}^{-1}) b_{\mathscr{B}}$ (1)
$\bar{y}b = (y_{\mathscr{B}}, y_{\mathscr{N}}) \begin{pmatrix} b_{\mathscr{B}} \\ b_{\mathscr{N}} \end{pmatrix} = \bar{y}_{\mathscr{B}} b_{\mathscr{B}}$ (2)
Uguagliando (1) e (2) si ha che $y_{\mathscr{B}} \geq 0 \Rightarrow cA_{\mathscr{B}}^{-1} \geq 0$ (**criterio di stop**). Se questa condizione è verificata $\bar{x}$ è il vertice ottimo.

7 Algoritmo del simplesso

L'algoritmo del simplesso è stato inventato da Dantzig nel 1947 e fino al 1979 è stato l'unico algoritmo risolutivo noto. La sua complessità è polinomiale al caso medio, anche se ciò chiaramente non garantisce che non si esamineranno un numero di nodi esponenziale nel numero di vincoli.

7.1 Come funziona

L'idea di fondo è di **esplorare i vertici** in modo da migliorare di volta in volta. Un vertice viene rappresentato mediante l'insieme degli indici delle disequazioni **attive** (ossia che soddisfa come uguaglianze) e tali indici prendono il nome di **base**. È quindi evidente che nel caso di un vertice degenere c'è il rischio di muoversi da una base ad un'altra che rappresenta lo stesso vertice.
Il passaggio tra un vertice ed un altro avviene tra **vertici adiacenti**, perciò (se un vertice non è degenere) le due basi differiscono di un indice. La **scelta della base** iniziale e la **valutazione dell'esistenza di una base ammissibile** vengono trattati mediante la tecnica del

Definizione 7.1 (Problema artificiale)
Sia (P) un problema di programmazione linare. P' è detto **problema artificiale** se P' è derivato dal problema originale aggiungendo una variabile artificiale ad ogni vincolo, come segue.

$$\sum_j a_{i,j}x_j \leq (\text{risp.} \geq \text{ risp.} =)b_i \rightarrow \sum_j a_{i,j}x_j + z_i \leq (\text{risp.} \geq \text{risp.} =)b_i, \text{ con } z_i \geq 0.$$

Lo scopo è quindi di trasmormare il problema iniziale in un problema artificiale e poi applicare l'algoritmo del simplesso sul problema artificiale, cercando di far tendere le variabili z_i a 0, ossia cercando il $\min z_i$. È infatti molto facile trovare una base ammissibile per il problema artificiale e, da quella, si ottiene una prima base ammissibile per il problema originale, che consente di applicare l'algoritmo del simplesso.
Riassumendo:

1. Obiettivo: Data una base $\mathscr{B}$ di partenza se ne vuole calcolare un'altra migliore per trovare il vertice ottimo. Di fatto l'algoritmo va di base in base, in particolare, assumendo basi non degeneri, va di vertice in vertice **adiacente** (e questo assicura anche che il vertice sia ammissibile).

2. L'algoritmo calcola la funzione costo e determina l'indice di base uscente e quello entrante.

3. calcolare l'indice uscente h
   Sia h l'indice uscente. Esso è l'indice caratterizzante e indica su quale spigolo del polinomio spostarsi
   $$x(\lambda) = \bar{x} + \lambda W^h$$
   Dove $\bar{x} \in \mathbb{R}^n$ ed è il vertice uscente (quello che si abbandona) $\lambda \geq 0$ (quindi è una semiretta), W^h è la h-esima colonna di $W = A_{\mathscr{B}}^{-1}$, il tutto è una parametrizzazione

di una semiretta di origine in $\bar{x}$, di direzione W^h e modulo λ.
Per la **regola anticiclo di Bland** $h = \min\{i \in \mathscr{B} : \bar{y}_{\mathscr{B}}, i < 0\}$
$cx(\lambda) = c(\bar{x} + \lambda^h) = c\bar{x} + \lambda cW^h$
$y_{\mathscr{B}} = cA_{\mathscr{B}}^{-1} \Rightarrow cW^h = -y_h$

4. Calcolare l'indice entrante k
   $Ax(\lambda) \leq b$ (da notare che i λ che soddisfano questa condizione appartengono al poliedro) $\rightarrow A(\bar{x} + \lambda W^h) \leq b \rightarrow A_i\bar{x} + A_i\lambda^h \leq b_i, i \in \mathscr{B}$
   Poichè $A_i\bar{x} = b_i \Rightarrow A_i\lambda W^h \leq 0$
   $W = -A_{\mathscr{B}}^{-1} \Rightarrow A_i\lambda(-A_{\mathscr{B}}),\ A_i(-A_{\mathscr{B}}^{-1}) \in \{0, 1\} \Rightarrow \forall\lambda$
   Cerco i λ tali che il vertice appartiene al poliedro: $A_i\bar{x} + \lambda a_i W^h \leq b_i$, con $i \in \mathscr{N}$ (non di base) $\lambda(\geq 0)A_iW^h \leq b_i - A_i\bar{x} \geq 0$, se $A_iW^h \leq 0$ va sempre bene. E adesso:

   - Consideriamo $a_iw^h \geq 0 \Rightarrow 0 \leq \lambda \leq \frac{b_i - A_i\bar{x}}{A_iW^h} = r_i$
   - Calcoliamo i rapporti r_i al variare di i
   - Calcoliamo solo quelli tali che $A_iW^h \geq 0$
   - Sono al più $m - N$, dove m sono i vincoli e N sono i vincoli non di base
   - Se $\nexists i$ *t.c.* $A_iW^h \geq 0$, posso scegliere qualunque $\lambda \Rightarrow \lambda \rightarrow \infty$ e la funzione obiettivo non ha massimo, $k = \min r_i$, dove $r_i = \frac{b_i - A_i\bar{x}}{A_iW^h}$, con $i \in \mathscr{B}$ e $A_iW^h \geq 0$.

7.2 Algoritmo per passi

1. Trovare una base $\mathscr{B}$ **ammissibile** e calcolare $A_{\mathscr{B}}$, $b_{\mathscr{B}}$ e c.
2. Invertire la matrice $A_{\mathscr{B}}$: sia $A_{\mathscr{B}} = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$, $A_{\mathscr{B}}^{-1} = \frac{1}{det(A_{\mathscr{B}})} \begin{pmatrix} d & -b \\ -c & a \end{pmatrix}$
3. Calcolare la soluzione di base primale $\bar{x} = A_{\mathscr{B}}^{-1}b_{\mathscr{B}}$ e la soluzione di base duale $\bar{y} = \begin{pmatrix} \bar{y}_{\mathscr{B}} \\ \bar{y}_{\mathscr{N}} \end{pmatrix}$, $\bar{y}_{\mathscr{B}} = c^tA_{\mathscr{B}}^{-1}$, $\bar{y}_{\mathscr{N}} = 0$.
   $\bar{x}$ si dice **ammissibile** se $\forall i \in \mathscr{N}\, A_i\bar{x} \leq b_i$.
   $\bar{y}$ si dice **ammissibile** se $\forall i \in \mathscr{B}\bar{y}_i \geq 0$.
   $\bar{x}$ si dice **degenere** se $\exists i \in \mathscr{N} : A_i\bar{x} = b_i$.
   $\bar{y}$ si dice **degenere** se $\exists i \in \mathscr{B} : \bar{y}_i = 0$.
4. Se $\bar{y}_{\mathscr{B}} > 0$ STOP. Altrimenti calcolare l'indice uscente $h = \min\{i \in \mathscr{B} : \bar{y}_i < 0\}$
5. Sia $W = -A_{\mathscr{B}}^{-1}$ e sia W^i l'i-esima colonna di W, con $i \in \mathscr{B}$.
   Se $A_iW^h \leq 0\ \forall i \in \mathscr{N}$ STOP.
   Altrimenti calcolare $\theta = \min\{\frac{b_i - A_i\bar{x}}{A_iW^h} : i \in \mathscr{N}, A_iW^h > 0\}$.
6. Calcolare l'indice entrante $k = \min\{i \in \mathscr{N} : A_iW^h > 0, \frac{b_i - A_i\bar{x}}{A_iW^h} = \theta\}$.
7. Aggiornare la base $\mathscr{B}' = \mathscr{B} - \{h\} \cup \{k\}$.

7.3 Esempi di applicazione dell'algoritmo del simplesso primale

Esercizio 7.1. *Dato il problema:* $\begin{cases} \max(2x_1) \\ x_1 + x_2 \leq 6 \\ x_1 \leq 4 \\ -x_1 + x_2 \leq 2 \\ x_2 \leq 4 \end{cases}$

Svolgimento 7.2. *Risolviamo il problema per passi:*

PRIMO PASSO. *Scrivere i dati:*

$$c^t = \begin{pmatrix} 2 & 0 \end{pmatrix}, \; A = \begin{pmatrix} 1 & 1 \\ 1 & 0 \\ -1 & 1 \\ 0 & 1 \end{pmatrix}, \; b = \begin{pmatrix} 6 \\ 4 \\ 2 \\ 4 \end{pmatrix}$$

SECONDO PASSO. *Prendere una base $\mathscr{B} = \{1, 4\}$, che è scelta in base ad un algoritmo (si prende l'indice delle righe che formano un minore invertibile di rango massimo della matrice).*

TERZO PASSO. *Vedere se è una base ammissible, in questo caso lo è perchè soddisfa tutte le disuguaglianze.*

QUARTO PASSO. *Presa la matrice di base, costruita prendendo le righe di A corrispondenti ai vettori di base scelti, risolvere il sistema $A_{\mathscr{B}}x = b_{\mathscr{B}}$, invertendo la matrice $A_{\mathscr{B}}$:*

$$A_{\mathscr{B}}^{-1} = -\begin{pmatrix} 1 & -1 \\ 0 & -1 \end{pmatrix} = \begin{pmatrix} -1 & 1 \\ 0 & 1 \end{pmatrix}$$

$$\bar{x} = A_{\mathscr{B}}^{-1} b_{\mathscr{B}} = \begin{pmatrix} -1 & 1 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 2 \\ 4 \end{pmatrix} = \begin{pmatrix} 2 \\ 4 \end{pmatrix}$$

QUINTO PASSO. *Prendere la soluzione duale associata alla soluzione primale trovata e valutare se è verificata o meno la condizione di ottimalità.*

$$\bar{y}_{\mathscr{B}}^t = c^t A_{\mathscr{B}}^{-1} = \begin{pmatrix} 2 & 0 \end{pmatrix} \begin{pmatrix} -1 & 1 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} -2 & 2 \end{pmatrix}$$

Dove, $\bar{y}^t = \begin{pmatrix} 0 & 0 & -2 & 2 \end{pmatrix}$, quindi la condizione di ottimalità ***non*** *è verificata. Da notare che* ***questo è un caso dubbio****, perchè la soluzione è degenere anche per il primo vincolo (ovvero lo soddisfa), quindi potrebbe succedere che si trovi una condizione di ottimalità anche in un'altra base. Dunque, è nesessario cambiare base per trovare il vertice ottimo.*

SESTO PASSO. *Selezionare l'indice uscente dalla base, ossia l'indice di riga della prima componente strettamente negativa: $h = 3$*

SETTIMO PASSO. *Scegliere una direzione di crescita, che si ottiene prendendo la prima colonna della matrice invertita e cambiandola di segno $W^h = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$.*

OTTAVO PASSO *Calcolare* $A_n = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}$.

NONO PASSO. *Calcolare* $A_n W^h = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$.

DECIMO PASSO. *Calcolare* θ *e* k*:* $\theta = \min\{\frac{b_i - A_i\bar{x}}{A_i W^h}, A_i w^h > 0, i \in \mathbb{N}\} = \min\{\frac{6-6}{1}, \frac{4-2}{1}\} = 0$*, da cui si evince che non cambierà vertice;* k *associato al minimo:* $k = -1$

UNDICESIMO PASSO. *Scegliere un'altra base* $\mathscr{B}' = \{1, 4\}$.

DODICESIMO PASSO. *Risolvere il sistema:* $A_{\mathscr{B}'} = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$; $A_{\mathscr{B}'}^{-1} = \begin{pmatrix} 1 & -1 \\ 0 & 1 \end{pmatrix}$; $\bar{x} = \begin{pmatrix} 1 & -1 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 6 \\ 4 \end{pmatrix} = \begin{pmatrix} 2 \\ 4 \end{pmatrix}$

TREDICESIMO PASSO. *Calcolare la soluzione duale e controllare se la soluzione è ottima:* $\bar{y}^t = \begin{pmatrix} 2 & 0 & 0 & -2 \end{pmatrix}$*. Si evince che neanche con questa base si ha la condizione di ottimalità, inoltre la soluzione è* ***non*** *degenere, perchè nessuna delle componenti di base* $\begin{pmatrix} 2 & -2 \end{pmatrix}$ *è nulla.*

8 Algoritmo del simplesso duale

L'**algoritmo del simplesso duale** si effettua prendendo una base ammissibile per il problema duale, calcolando la base primale e vedendo se è ottima. A livello logico non c'è differenza, quello che cambia è il procedimento. Questo algoritmo è indipendente dall'altro, la differenza è che i vertici del problema duale hanno una caratterizzazione diversa da quelli del problema primale, quindi è necessario trovare un modo diverso di cambiare base.

Il problema è scritto nella forma $\begin{cases} \min cx \\ Ax \leq b \end{cases}$

8.1 Come funziona

PRIMO PASSO. Si determina una base $\mathscr{B}$ duale ammissibile.

SECONDO PASSO. Si calcolano le soluzioni di base primale associate a $\mathscr{B}$, $\bar{y}^t = (\bar{y}^t_{\mathscr{B}}, \bar{y}^t_{\mathscr{N}})$, $\bar{y}^t_{\mathscr{B}} = c^t A_{\mathscr{B}}^{-1}, \bar{y}^t_{\mathscr{N}} = 0$. Se risulta $A_{\mathscr{N}}\bar{x} \leq b_{\mathscr{N}}$, allora $\bar{y}$ e $\bar{x}$ sono ottime per D e P **STOP**. Altrimenti passo 3.

TERZO PASSO. Si calcola l'indice entrante in base $k = min\{i \in \mathscr{N}; A_i\bar{x} > b_i\}$, si pone W^i la colonna i-esima di $-A_{\mathscr{B}}^{-1}, i \in \mathscr{B}$. Se $A_k W^i \geq 0$, $\forall i \in \mathscr{B}$, allora D è infinitamente illimitato, STOP, ossia che esiste una direzione di decrescita indefinita. Altrimenti passo 4.

QUARTO PASSO. Si calcola l'indice uscente di base (si calcola rispettivamente agli elementi del vettore $A_k W^i$, che sarà negativo). Nella direzione di discesa sia $\theta = min\{\frac{\bar{y}_i}{-A_k W^i}, A_k W^i < 0, i \in \mathscr{B}\}$. Sia $h = min\{i \in \mathscr{B}\ tc \frac{\bar{y}_i}{-A_k W^i} = \theta, A_k W^i < 0\}$, si

pone $\mathscr{B} = (\mathscr{B} - \{h\}) \cup \{k\}$ e si torna al passo 2 (notare che si sfrutta la **regola anticiclo di Bland**, che serve per non passare di nuovo dalle stessi basi).

Definito in questa maniera l'algoritmo è finito, perchè il numero di basi che si possono trovare è finito, quindi il numero di passi è finito.

Osservazione 8.1.1. $A_k W^i \geq 0, i \in \mathscr{B} \Leftrightarrow A_k A_{\mathscr{B}}^{-1} \leq 0$

8.2 Algoritmo per passi

1. Trovare una base $\mathscr{B}$ **ammissibile** e calcolare $A_{\mathscr{B}}$, $b_{\mathscr{B}}$ e c.
2. Invertire la matrice $A_{\mathscr{B}}$: sia $A_{\mathscr{B}} = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$, $A_{\mathscr{B}}^{-1} = \frac{1}{det(A_{\mathscr{B}})} \begin{pmatrix} d & -b \\ -c & a \end{pmatrix}$
3. Calcolare la soluzione di base primale $\bar{x} = A_{\mathscr{B}}^{-1} b_{\mathscr{B}}$ e la soluzione di base duale $\bar{y} = \begin{pmatrix} \bar{y}_{\mathscr{B}} \\ \bar{y}_{\mathscr{N}} \end{pmatrix}$, $\bar{y}_{\mathscr{B}} = c^t A_{\mathscr{B}}^{-1}$, $\bar{y}_{\mathscr{N}} = 0$.

   $\bar{x}$ si dice **ammissibile** se $\forall i \in \mathscr{N} A_i \bar{x} \leq b_i$.

   $\bar{y}$ si dice **ammissibile** se $\forall i \in \mathscr{B} \bar{y}_i \geq 0$.

   $\bar{x}$ si dice **degenere** se $\exists i \in \mathscr{N} : A_i \bar{x} = b_i$.

   $\bar{y}$ si dice **degenere** se $\exists i \in \mathscr{B} : \bar{y}_i = 0$.
4. Se $b_i - A_i \bar{x} \geq 0 \ \forall i \in \mathscr{N}$ STOP. Altrimenti calcolare l'indice entrante $k = \min\{i \in \mathscr{N} : b_i - A_i \bar{x} < 0\}$
5. Sia $W = -A_{\mathscr{B}}^{-1}$ e sia W^i l'i-esima colonna di W, con $i \in \mathscr{B}$.

   Se $A_k W^i \geq 0 \ \forall i \in \mathscr{B}$ STOP.

   Altrimenti calcolare $\theta = \min\{\frac{\bar{y}_i}{-A_k W^i} : i \in \mathscr{B}, A_k W^i < 0\}$.
6. Calcolare l'indice uscente $h = \min\{i \in \mathscr{B} : A_k W^i < 0, \frac{\bar{y}_i}{-A_k W^i} = \theta\}$
7. Aggiornare la base $\mathscr{B}' = \mathscr{B} - \{h\} \cup \{k\}$.

8.3 Esempi di applicazione dell'algoritmo del simplesso duale

Esercizio 8.1. *Risolvere il seguente problema di programmazione lineare mediante l'algoritmo del simplesso duale.*
$min(13y_3 + 9y_4 + 7y_5)$

$$\begin{cases} -y_1 + y_3 + y_4 + y_5 = 3 \\ -y_2 + 2y_3 + y_4 = -4 \\ y \geq 0 \end{cases}$$

Svolgimento 8.2. *Esempio di schematizzazione dell'esercizio per punti:*

Scrivere i dati. $A = \begin{pmatrix} =1 & 0 \\ 0 & -1 \\ 1 & 2 \\ 1 & 1//1^0 \end{pmatrix}$, $c^t = \begin{pmatrix} 3 & -4 \end{pmatrix}$, $b = \begin{pmatrix} 0 \\ 0 \\ 13 \\ 9 \\ 7 \end{pmatrix}$ $\mathscr{B} = \{2, 3\}$

Calcolare la soluzione duale. $A_{\mathscr{B}} = \begin{pmatrix} 0 & -1 \\ 1 & 2 \end{pmatrix}$, $A_{\mathscr{B}}^{-1} = \begin{pmatrix} 2 & 1 \\ -1 & 0 \end{pmatrix}$, $\bar{y}_{\mathscr{B}}^t = \begin{pmatrix} 3 & -4 \end{pmatrix} \begin{pmatrix} 2 & 1 \\ -1 & 0 \end{pmatrix} = \begin{pmatrix} 10 & 3 \end{pmatrix}$

$\bar{y}^T = \begin{pmatrix} 0 & 10 & 3 & 0 & 0 \end{pmatrix}$, $\bar{x} = A_{\mathscr{B}}^{-1} b_{\mathscr{B}} = \begin{pmatrix} 2 & 1 \\ -1 & 0 \end{pmatrix} \begin{pmatrix} 0 \\ 13 \end{pmatrix}$

Calcolare $A_{\mathscr{N}}$ e k. $A_{\mathscr{N}} = \begin{pmatrix} -1 & 0 \\ 1 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 13 & 0 \end{pmatrix} = \begin{pmatrix} -13 \\ 13 \\ 13 \end{pmatrix} \not\leq \begin{pmatrix} 0 \\ 9 \\ 7 \end{pmatrix} \Rightarrow k = 4$

Calcolare $A_k W$, θ, h e la nuova base . $\forall i \in \mathscr{B}\ A_k W = \begin{pmatrix} 1 & 1 \end{pmatrix} (-A_{\mathscr{B}}^{-1}) = - \begin{pmatrix} -1 & 1 \end{pmatrix} \begin{pmatrix} 2 & 1 \\ 1 & 0 \end{pmatrix} = - \begin{pmatrix} 1 & 1 \end{pmatrix}$ $\theta = \min\{\frac{10}{1}, \frac{3}{1}\} = 3 \Rightarrow h = 3 \Rightarrow nuovabase \mathscr{B}' = \{2, 4\}$

Ripetere il procedimento con la base $\mathscr{B}'$. $\mathscr{B}' = \{2, 4\}$, $A = \begin{pmatrix} 0 & -1 \\ 1 & 1 \end{pmatrix}$, $A_{\mathscr{B}'}^{-1} = \begin{pmatrix} 1 & 1 \\ -1 & 0 \end{pmatrix}$, $\bar{y}_{\mathscr{B}'}^t = \begin{pmatrix} 3 & -4 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ -1 & 0 \end{pmatrix} = \begin{pmatrix} 7 & 3 \end{pmatrix}$, $\bar{y}^T = \begin{pmatrix} 0 & 7 & 0 & 3 & 0 \end{pmatrix}$, $\bar{x} = A_{\mathscr{B}'}^{-} 1 b_{\mathscr{B}'} = \begin{pmatrix} 1 & 1 \\ -1 & 0 \end{pmatrix} \begin{pmatrix} 0 \\ 9 \end{pmatrix} = \begin{matrix} 9 \\ 0 \end{matrix}$, $A_{\mathscr{N}} = \begin{pmatrix} -1 & 0 \\ 1 & 2 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 9 & 0 \end{pmatrix} = \begin{pmatrix} -9 \\ 9 \\ 9 \end{pmatrix} \not\leq \begin{pmatrix} 0 \\ 13 \\ 7 \end{pmatrix}$. *L'indice per il quale non è verificata è il terzo, quindi* $k = 5$.

Calcolare $A_k W$, θ e k. W^i: $\forall i \in \mathscr{B}\ A_k W = \begin{pmatrix} 1 & 0 \end{pmatrix} (-A_{\mathscr{B}}^{-1}) = - \begin{pmatrix} 1 & 0 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ -1 & 0 \end{pmatrix} = - \begin{pmatrix} 1 & 1 \end{pmatrix}$ *Se avessi avuto una componente del vettore* 0*, non avrei calcolato la frazione corrispondente per trovare* θ*, ma lo avrei preso uguale all'unico rimasto.* $\theta = \min\{\frac{7}{1}, \frac{3}{1}\} = 3 \Rightarrow h = 4$*, che è l'elemento di base relativo alla posizione del minimo*

Calcolare la nuova base. $\Rightarrow$ *nuova base* $\mathscr{B}'' = \{2, 5\}$

Ripetere il procedimento con la base $\mathscr{B}''$. $A_{\mathscr{B}''} = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}$, $A_{\mathscr{B}''}^{-1} = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}$,

$\bar{y}_{\mathscr{B}''}^t = \begin{pmatrix} 3 & -4 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} = \begin{pmatrix} 4 & 3 \end{pmatrix}$, $\bar{y}^T = \begin{pmatrix} 0 & 4 & 0 & 0 & 3 \end{pmatrix}$, $\bar{x} = A_{\mathscr{B}''}^{-1} b_{\mathscr{B}''} = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} \begin{pmatrix} 0 \\ 7 \end{pmatrix} = \begin{matrix} 7 \\ 0 \end{matrix}$, $A_{\mathscr{N}} = \begin{pmatrix} -1 & 0 \\ 1 & 2 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} 7 & 0 \end{pmatrix} = \begin{pmatrix} -7 \\ 7 \\ 7 \end{pmatrix} \leq \begin{pmatrix} 0 \\ 13 \\ 9 \end{pmatrix}$.

Considerazioni. *Dato che la condizione è verificata ne deduco che* $\bar{x}$ *e* $\bar{y}$ *sono ottime. La soluzione duale è non degenere, perchè le sue due componenti sono tutte e due*

diverse da zero. Neanche la soluzione primale lo è, infatti $\begin{pmatrix}-7\\7\\7\end{pmatrix} < \begin{pmatrix}0\\13\\9\end{pmatrix}$*, c'è il* ***minore stretto***.

Osservazione 8.3.1. Se la soluzione duale è non degenere la soluzione ottima primale è unica. Analogamente, se la soluzione primale è non degenere la soluzione duale è unica, per il teorema degli scarti complementari.

8.4 Elementi teorici sull'algoritmo del simplesso duale

N.B. Non è obbligatorio saperla per l'orale.

Vogliamo risolvere il problema $\begin{cases} min(y^t b) \\ y^t A = c^t \\ y^t \geq 0 \end{cases}$. L'algoritmo dice che siamo in possesso di una base duale ammissible, questo si traduce nel fatto che $y^t A = c^t \Leftrightarrow (y^t_{\mathscr{B}}, my^t_{\mathscr{N}}) \begin{pmatrix} A_{\mathscr{B}} \\ A_{\mathscr{N}} \end{pmatrix} = c^t$

$y^t_{\mathscr{B}} A_{\mathscr{B}} + y^t_{\mathscr{N}} A_{\mathscr{N}} = c^t$,$y^t_{\mathscr{B}} A_{\mathscr{B}} = c^t - y^t_{\mathscr{N}} A_{\mathscr{N}} \Rightarrow y^t_{\mathscr{B}}$(che è maggiore o uguale di zero in partenza)$= c^t A^{-1}_{\mathscr{B}} - y_{\mathscr{N}} A_{\mathscr{N}} A^{-1}_{\mathscr{B}}$

$$\begin{cases} c^t A^{-1}_{\mathscr{B}} - y^t_{\mathscr{N}} A_{\mathscr{N}} A^{-1}_{\mathscr{B}} \geq 0 \\ y_{\mathscr{N}} \geq 0 \end{cases}$$

$y^t_b = (y^t_{\mathscr{B}}, y^t_{\mathscr{N}}) \begin{pmatrix} b_{\mathscr{B}} \\ b_{\mathscr{N}} \end{pmatrix} = y^t_{\mathscr{B}} b_{\mathscr{B}} + y^t_{\mathscr{N}} b_{\mathscr{N}} = (c^t A^{-1}_{\mathscr{B}} - y^t_{\mathscr{N}} A_{\mathscr{N}} A^{-1}_{\mathscr{B}}) b_{\mathscr{B}} + y^t_{\mathscr{N}} b_{\mathscr{N}} = c^t A^{-1}_{\mathscr{B}} b_{\mathscr{B}} +$
$y^t_{\mathscr{N}}(b_{\mathscr{N}} - A) N A^{-1}_{\mathscr{B}} b_{\mathscr{B}})$

Quindi, riscrivendo i vincoli, si ha:

$$\begin{cases} min(c^t A^{-1}_{\mathscr{B}} - y^t_{\mathscr{N}}(b_{\mathscr{N}} - A_{\mathscr{N}} A^{-1}_{\mathscr{B}} b_{\mathscr{B}}) \\ c^t A^{-1}_{\mathscr{B}} - y^t_{\mathscr{N}} A_{\mathscr{N}} A^{-}_{\mathscr{B}} 1 \geq 0 \\ y_{\mathscr{N}} \geq 0 \end{cases}$$

Osservazione 8.4.1. Se risultasse che questo coefficiente $b_{\mathscr{N}} - A_{\mathscr{N}} A^{-1}_{\mathscr{B}} b_{\mathscr{B}}$ avesse componenti tutte maggiori o uaguali a zero la soluzione sarebbe ottima.

Osservazione 8.4.2. Se $b_{\mathscr{N}} - A_{\mathscr{N}} A^{-1}_{\mathscr{B}} b_{\mathscr{B}} \geq 0$ allora $\bar{y}_{\mathscr{N}}$ va come 0, quindi è ottimo per D_r. Infatti, sia k come al passo 3 e definiamo una direzione di ricerca $e_{\mathscr{N}} = (0, \cdots, 0, \underbrace{1}_{k}, 0, \cdots, 0)$, con k che è una direzione di decrescita per D_r, perchè (boh!!). Consideriamo $y_{\mathscr{N}}(\theta) = \bar{y}_{\mathscr{N}} + \theta e_{\mathscr{N}} = \theta e_{\mathscr{N}}, \theta \geq 0$. I vincoli sono verificati, tranne il secondo, quindi deve essere

$$c^t A^{-1}_{\mathscr{B}} - y^t_{\mathscr{N}}(\theta) A_{\mathscr{N}} A^{-1}_{\mathscr{B}} \geq 0 \Leftrightarrow c^t A^{-1}_{\mathscr{B}} - \theta e_{\mathscr{N}} A_{\mathscr{N}} A^{-1}_{\mathscr{B}} \geq 0$$

$$\Leftrightarrow c^t A^{-1}_{\mathscr{B}} - \theta A_k A^{-1}_n \geq 0 \Leftrightarrow c^t A^{-1}_{\mathscr{B}} + \theta A_k W \geq 0 \Leftrightarrow \bar{y}_i + \theta A_k W^i \geq 0, i \in B$$

$A_K W^i \geq 0, \forall i \in B \Rightarrow \theta \to \infty$ (le precedenti disuguaglianze sono verificate $\forall \theta \geq 0$).
Se $A_k W^i \leq 0$ allora si può scrivere che $\theta(A_k W^i) \leq \bar{y}_i$, questo vuol dire che $\theta \leq \frac{\bar{y}_i}{A_k W^i} < 0, i \in B \Rightarrow \theta \leq min\{\frac{\bar{y}_i}{-A_K W^i}, A_k W^i \leq 0, i \in B\}$.

Trovare una prima base duale ammissibile
Dato D problema duale, tale che

$$\begin{cases} \min yb \\ yA = c \\ y \geq 0 \end{cases}$$

scrivere il problema ausiliario: sia w (come anche c) $\in \mathbb{R}^n$

$$\begin{cases} \min yb \\ yA + w = c \\ y, w \geq 0 \end{cases}$$

Una base ammissibile di questo problema è $\mathscr{T} = \{w_{m+1}, \cdots, w_n\}$.
Da qui eseguire l'algoritmo del simplesso per poi ottenere una base $\mathscr{T}^f$ associata al valore ottimo. Se alla fine $w = 0$ $\mathscr{T}^f$ è una base ammissibile per il problema duale originale. Altrimenti il problema non ha soluzione.

9 Programmazione lineare intera

9.1 Algoritmo per passi

Calcolare una valutazione superiore del valore ottimo risolvendo il rilassamento continuo.

Approcciare il problema da un punto di vista grafico, trovando la regione ammissibile e disegnando la funzione obiettivo (retta per l'origine).

Il vertice ottimo si trova intersecando le rette ortogonali alla funzione obiettivo con i vertici: il vertice che viene incontrato più verso il bordo è il vertice **ottimo**. ATTENZIONE. Funziona solo trovando algebricamente i vertici e provandoli. Una volta trovato quello che minimizza/massimizza la funzione obiettivo la "sol. ottima del rilassamento" è il vertice e $v_i(p)$ è il valore della funzione obiettivo in quel punto.

Inoltre, per verificare che il vertice trovato sia il vertice ottimo, basta disegnare i vettori ortogonali ai vincoli adiacenti al vertice e vedere se la retta appartenente al fascio della funzione obiettivo passante per il vertice appartiene.

La soluzione ottima è quindi il vertice.

La valutazione superiore è la parte intera inferiore della funzione obiettivo calcolata in quel vertice.

Cacolare una valutazione inferiore del valore ottimo arrotondando la soluzione ottima del rilassamento.

Prendere la parte intera inferiore del vertice trovato e quella è la **soluzione ammissibile**, nel frattempo calcolare la valutazione in quel punto.

Calcolare un taglio di Gomory
1. Trasformare i vincoli in uguaglianze
2. Trovare $\bar{x} \in \mathbb{R}^n$, conoscendo x_1 e x_2 del rilassamento continuo.

3. Prendere la base, ossia gli indici delle componenti non nulle.
4. Scrivere A, b, $A_{\mathscr{B}}$, $A_{\mathscr{B}}^{-1}$, con A matrice del sistema nuovo.
5. Calcolare $\tilde{A} = A_{\mathscr{B}}^{-1} A_{\mathscr{N}}$ e $\tilde{b} = A_{\mathscr{B}}^{-1} b$.
6. Il numero di tagli possibili è il numero di componenti frazionarie di $\tilde{b}$.
7. I piani possibili sono $\sum_{j \in \mathscr{N}} \{\tilde{a}_{rj}\} x_j \geq \{\tilde{b}_r\}$, con $\{\}$ parte frazionaria dell'argomento, r la prima componente di base se è stato preso $\tilde{b}_1$ e l'indice della seconda componente di base se è stato preso $\tilde{b}_2$ nel vettore $\tilde{b} = \begin{pmatrix} \tilde{b}_1 \\ \tilde{b}_2 \end{pmatrix}$

10 Grafi

10.1 Definizioni

Definizione 10.1 (Grafo)
Un **grafo** è una coppia di insiemi (V, E), dove V è l'insieme dei nodi ed E è un insieme di coppie non ordinate di nodi. Un grafo si dice **orientato** se le coppie sono ordinate.

Definizione 10.2 (Cammino)
Un **cammino** è una sequenza di nodi collegati da archi, che vanno da un nodo ad un altro. Un cammino è **orientato** se, percorrendolo, viene rispettato l'orientamento degli archi.

Definizione 10.3 (Ciclo)
Un **ciclo** è cammino che ha come nodo di partenza e di arrivo lo stesso nodo. Un ciclo è **orientato** se, percorrendolo, viene rispettato l'orientamento degli archi.

Definizione 10.4 (Connessione e forte connessione)
Un grafo si dice **connesso** se per ogni coppia di nodi esiste un cammino che li collega. Un grafo si dice **fortemente connesso** se per ogni coppia di nodi esiste un cammino orientato che li collega.

Definizione 10.5 (Albero di copertura)
Dato un grafo G si definisce **albero di copertura** un albero T con gli stessi nodi e alcuni archi, in modo che non ci siano cicli e che sia connesso. Equivalentemente, sia G un grafo con n nodi, T è un **albero di copertura** di G se ha gli stessi nodi, $n-1$ archi ed è connesso.

11 Flusso su reti

Teorema 11.1 (Teorema di interezza)
Sia $G = (V, E)$ un grafo e sia dato il seguente problema di flusso capacitato di costo minimo

$$\begin{cases} \min cx \\ Mx = b \\ x \leq u \\ x \geq 0 \end{cases}$$

Se $b \in \mathbb{Z}^m$ e $u \in \mathbb{Z}^n$ allora $x \in \mathbb{Z}^n$.

Dimostrazione. Poichè il sistema ha più incognite che equazioni fisso le $n-m$ componenti a 0 o $u_i, \forall i$, secondo la tripartizione. Sia M' un minore $(n-1) \times (n-1)$ invertibile di M. Una permutazione delle sue righe permette di ottenere una matrice triangolare con, sulla diagonale, soltanto 1 o -1. Dall'algebra lineare sappiamo che, con una matrice in questa forma, essendo b intero non è necessario effettuare divisioni, pertanto si ha la tesi. □

Osservazione 11.0.1. Dato un grafo G ed un problema di flusso capacitato di costo minimo

$$\begin{cases} \min cx \\ Mx = b \\ x \leq u \\ x \geq 0 \end{cases}$$

esiste una soluzione ottima se esiste una soluzione ammissibile e non esiste un ciclo di costo (somma dei costi degli archi) negativo e capacità (intesa come minimo delle capacità) infinita.

11.1 Algoritmo per passi

(a)

1. Fare l'albero con solo gli archi in T e i bilanci
2. Aggiungere gli archi in U con flusso uguale alla capacità di quell'arco e aggiornare i bilanci degli estremi.
3. Prendere una foglia e capire il flusso necessario per azzerare il bilancio della foglia (il bilancio stesso se l'arco è entrante, il suo opposto se l'arco è uscente).
4. Aggiornare i bilanci del nodo all'altro capo dell'arco (al nodo da cui esce il flusso sommare il flusso, dal nodo entrante sottrarre).
5. Togliere l'arco e ripetere 2. e 3..
6. $x_{ij} = 0 \ \forall (i,j) \notin T$, $x_{ij} =$ numero sull'arco nell'albero disegnato $\forall (i,j) \in T$.
7. La soluzione è **ammissibile** $\Leftrightarrow$ i flussi sono tutti non negativi e $\leq$ della capacità, in simboli $x \geq 0, x_{ij} \leq u_{ij} \ \forall (i,j) \in E$.
8. La soluzione è **degenere** $\Leftrightarrow \exists (i,j) \in T : x_{ij} = 0 \vee x_{ij} = u_{ij}$, con x_{ij} flusso e u_{ij} capacità.

(b)

1. Fare l'albero con gli archi in T e solo i costi. Mettere a 0 il potenziale di 1 ($\pi_1 = 0$).
2. Calcolare il potenziale π_j del nodo j come $\pi_i + c_{ij}$ se $(i,j) \in T, \pi_i - c_{ij}$ se $(j,i) \in T$.
3. Ripetere finchè tutti i nodi hanno potenziale.
4. La soluzione è **ammissibile** se $\forall (i,j) \in L \ c_{ij}^{\bar{\pi}} = \pi_i - \pi_j + c_{ij} \geq 0$, $\forall (i,j) \in U \ c_{ij}^{\bar{\pi}} \leq 0$.
5. La soluzione è **degenere** se $\exists (i,j) \in L \cup U : c_{ij}^{\bar{\pi}} = 0$.

12 Algoritmo del simplesso su reti

12.1 Algoritmo per passi

NOTA: Arco entrante ed arco uscente possono coincidere

(a)
1. Fare l'albero con solo gli archi in T e i bilanci
2. Aggiungere gli archi in U con flusso uguale alla capacità di quell'arco e aggiornare i bilanci degli estremi.
3. Prendere una foglia e capire il flusso necessario per azzerare il bilancio della foglia (il bilancio stesso se l'arco è entrante, il suo opposto se l'arco è uscente).
4. Aggiornare i bilanci del nodo all'altro capo dell'arco (al nodo da cui esce il flusso sommare il flusso, dal nodo entrante sottrarre).
5. Togliere l'arco e ripetere 2. e 3..
6. $x_{ij} = 0 \ \forall (i,j) \notin T$, x_{ij} = numero sull'arco nell'albero disegnato $\forall (i,j) \in T$.
7. La soluzione è **ammissibile** $\Leftrightarrow$ i flussi sono tutti non negativi e $\leq$ della capacità, in simboli $x \geq 0, x_{ij} \leq u_{ij} \ \forall (i,j) \in E$.
8. La soluzione è **degenere** $\Leftrightarrow \exists (i,j) \in T : x_{ij} = 0 \vee x_{ij} = u_{ij}$, con x_{ij} flusso e u_{ij} capacità.

(b)
1. Fare l'albero con gli archi in T e solo i costi. Mettere a 0 il potenziale di 1 ($\pi_1 = 0$).
2. Calcolare il potenziale π_j del nodo j come $\pi_i + c_{ij}$ se $(i,j) \in T, \pi_i - c_{ij}$ se $(j,i) \in T$.
3. Ripetere finchè tutti i nodi hanno potenziale.
4. La soluzione è **ammissibile** se $\forall (i,j) \in L \ c_{ij}^{\bar{\pi}} = \pi_i - \pi_j + c_{ij} \geq 0$, $\forall (i,j) \in U \ c_{ij}^{\bar{\pi}} \leq 0$.
5. La soluzione è **degenere** se $\exists (i,j) \in L \cup U : c_{ij}^{\bar{\pi}} = 0$.

(c)
1. Se la soluzione è **ammissibile** STOP.
2. Altrimenti calcolare l'arco entrante $(p,q) = \min\left(\{(i,j) \in L : c_{ij}^{\bar{\pi}} < 0\} \cup \{(i,j) \in U : c_{ij}^{\bar{\pi}} > 0\}\right)$
3. Scrivere il ciclo $T \cup \{p,q\}$. Il suo verso è concorde con (p,q) se $(p,q) \in L$, discorde altrimenti.
4. $\mathcal{C}^+$ è l'insieme degli archi concordi con il verso di $\mathcal{C}$ e $\mathcal{C}^-$ è l'insieme degli archi discordi con il verso di $\mathcal{C}$
5. Sapendo che u_{ij} rappresenta le capacità calcolare:
   - $\theta^+ = \min\{u_{ij} - \bar{x}_{ij} : (i,j) \in \mathcal{C}^+\}$
   - $\theta^- = \min\{\bar{x}_{ij} : (i,j) \in \mathcal{C}^-\}$
   - $\theta = \min\{\theta^+, \theta^-\}$.
6. Se $\theta = +\infty$ STOP. Altrimenti calcolare l'arco uscente $(r,s) = \min\left(\{(i,j) \in \mathcal{C}^+ : u_{ij} - \bar{x}_{ij} = \theta\} \cup \{(i,j) \in \mathcal{C}^- : \bar{x}_{ij} = \theta\}\right)$.

7. Aggiornare la tripartizione: se l'arco entrante e l'arco uscente coincidono, l'arco passa da L a U o viceversa. Altrimenti:

   L'arco entrante va in T

   L'arco uscente va:

   - nell'insieme da cui ho tolto l'arco entrante se arco entrante ed arco uscente erano discordi
   - nell'altro insieme altrimenti

8. Tornare al passo 1, il nuovo flusso è "aumentare il flusso degli archi in $\mathcal{C}^+$ di θ e diminuire il flusso degli archi di $\mathcal{C}^-$ di θ".

13 Algoritmo di Dijkstra

13.1 Algoritmo per passi

1. $Q = \{r\}$, $\pi_i = \begin{cases} +\infty & i \neq r \\ 0 & i = r \end{cases}$

   $p_i = \begin{cases} -1 & i \neq r \\ 0 & i = r \end{cases}$

2. Estrarre da Q un nodo u con π_u minima.

3. $\forall (u, v)$ arco se $\pi_v > \pi_u + c_{uv}$ allora $p_v = u$, $\pi_v = \pi_u + c_{uv}$, $Q = Q \cup \{v\}$.

4. $Q = \varnothing \Rightarrow$ STOP, altrimenti tornare al passo "2.".

14 Algoritmo di Ford-Fulkerson

14.1 Algoritmo per passi

1. Trovare un cammino dalla sorgente alla destinazione di lunghezza minima, senza tener conto delle capacità, ossia lunghezza minima nel numero di archi (a parità di lunghezza scegliere quello che viene prima in ordine lessicografico), con gli archi percorsi nel verso giusto.

2. Trovare la minima capacità δ su quel cammino e $\forall$ arco del cammino $x_{ij} = \delta$, tutti gli altri sono 0 e $v = \delta$.

3. Costruire il grafo residuo $G(x)$, ossia il grafo fatto dagli stessi nodi e gli archi fatti così: $\forall i, j \in A$

   - $x_{ij} = 0 \Rightarrow$ mettere un arco tra i e j con capacità u_{ij}.
   - $x_{ij} = u_{ij} \Rightarrow$ mettere un arco tra j e i di capacità u_{ij}.
   - In tutti gli altri casi mettere un arco da i a j di capacità $u_{ij} - x_{ij}$ e un arco da j a i di capacità x_{ij}.

4. Trovare un cammino di lunghezza minima (a parità di lunghezza scegliere quello che viene prima in ordine lessicografico) nel verso degli archi dall'origine alla destinazione. Se non c'è STOP.

5. δ = capacità minima $\forall$ arco del cammino se l'ho percorso nel senso in cui è nel grafo originale $x_{ij} = x_{ij} + \delta$, altrimenti $x_{ij} = x_{ij} - \delta$.

   $v = \delta + v$. Tornare al passo "3.".

6. $N_s = \{i \in N : \exists$ cammino orientato dalla sorgente s a i in $G(x)\}$, con $G(x)$, l'ultimo grafo residuo. $N_t = N - N_s$.

15 Algoritmo Bellman-Ford, per il problema SPT (albero dei cammini minimi)

15.1 Algoritmo per passi

1. $k = 0$, $\pi_i^0 = \begin{cases} +\infty & i \neq r \\ 0 & i = r \end{cases}$

   $p_i = \begin{cases} -1 & i \neq r \\ 0 & i = r \end{cases}$

2. $k = k+1$ $\forall j = 1, \cdots, n$ $BS(j) = \{i \in \mathcal{N} : \exists (i,j) \in A\}$, u nodo t.c. $\min\{\pi_i^{k-1} + c_{ij} : i \in BS(j)\}$. Se $\pi_j^{k-1} > \pi_u^{k-1} + c_{uj}$ allora $\pi_j^k = \pi_u^{k-1} + c_{uj}$ e $p_j = u$, altrimenti $\pi_j^k = \pi_j^{k-1}$.

3. Se $\pi^k = \pi^{k-1}$ STOP.

4. Se $k \leq n-1 \Rightarrow$ Passo "2.". Altrimenti STOP.

16 Esercizio del commesso viaggiatore

16.1 Algoritmo per passi

(a)

1. Fare la lista degli archi con pesi crescenti
2. Prendere l'arco di costo minimo che non colleghi il nodo i da cui il problema dice di partire.
3. Capire se i due estremi sono già collegati. Se non lo sono aggiungere l'arco.
4. Ripetere "3". fino a quando gli archi non sono finiti.
5. Mettere due archi di costo minimo che collegano i ai nodi presi.
6. $v_I(P)$ = somma dei pesi.

(b)

1. Trovare il nodo più vicino non visitato.
2. Ripetere "(b) 1." da quel nodo.

3. Alla fine mettere l'ultimo arco e scrivere la sequenza di nodi senza l'ultimo, ma fare la somma con l'ultimo.

(c) 1. Istanziare la prima variabile data con 0 o 1

2. Se il sottoproblema è **impossibile** (ossia facendo il disegno dei SOLI archi oggligati: *caso (a)* n archi con un vertice in comune, cioè si ha un ciclo che non passa per tutti i vertici, *caso (b)* formano un ciclo) passare al successivo. Altrimenti calcolare la valutazione inferiore come in "(a)". Se è peggio della **valutazione superiore** (con valutazione superiore si intende la più bassa che è stata scritta nell'albero) passare a "5.".
3. Se la soluzione ottenuta è un ciclo e la valutazione inferiore trovata è $< v_{sup} \Rightarrow v_{sup} = v_1$, ossia aggiornare la **valutazione superiore** (con valutazione superiore si intende la più bassa che è stata scritta nell'albero).
4. Se la valutazione inferiore in questo sottoproblema è minore della **valutazione superiore** (con valutazione superiore si intende la più bassa che è stata scritta nell'albero) ripetere da "(c) 1.". Altrimenti passo 5.
5. Chiudere il nodo.

17 Ripasso di teoria

17.1 Domanda 1

Spiega a chi non sa nulla di Ricerca Operativa il teorema fondamentale della programmazione lineare

17.2 Domanda 2

Spiega il significato di soluzione di base su grafi (in particolare spiegare il significato delle basi e sulla tripartizione)

17.3 Domanda 3

Formulazione del problema del flusso massimo e suo duale (taglio di capacità minima), che relazione c'è tra i due?

17.4 Domanda 4

Enuncia il teorema MaxFlow - MinCut

17.5 Domanda 5

I vertici di un poliedro esistono sempre?