Problemi di ottimizzazione
ricerca operativa
- insieme all'ottimizzazione combinatoria hanno come oggetto lo studio di metodologie al supporto di decisioni
- i problemi riguardano situazioni in cui serva massimizzare ricavi e profitti o minimizzare costi e perdite, in presenza di risorse limitate

processo decisionale
- fasi:
  - individuazione del problema
  - raccolta dati
  - costruzione modello
  - determinazione di soluzioni
  - analisi dei risultati
- ricerca operativa e ottimizzazione combinatoria si occupano delle fasi 3 e 4
  - richiedono l'impiego del linguaggio e degli strumenti di informatica e matematica

modello
- descrizione astratta e scritta con strumenti di tipo logico-matematico della parte di realtà utile al processo decisionale
- tipi:
  - basato sui giochi: la ricerca di una soluzione è risultante dall'interazione tra due o più agenti
  - di simulazione: il problema si studia riproducendo il comportamneto del sistema
  - analitico: il problema viene descritto attraverso un relazioni matematiche (o logiche) il più possibile fedeli alla situazione reale ma astratto abbastanza da permetterne la determinazione di una soluzione analitica

problema (P)
- domanda la cui risposta dipende da un numero di parametri e variabili
- descritto tramite la descrizione di: parametri (in generale indeterminati) e variabili, caratteristiche che le soluzioni devono avere
- definito dall'insieme F delle possibili soluzioni (insieme ammissibile, i cui parametri sono soluzioni ammissibili)
- può essere specificato anche con un insieme F' | F cont F' e dai vincoli che gli elementi di F devono soddisfare
  - gli elementi di F' \ F sono soluzioni non ammissibili
- istanza del problema: domanda ottenuta specificando dei valori concreti per tutti i parametri del problema
- la soluzione di un problema è in realtà sempre la soluzione della sua rappresentazione
- il modello è una descrizione limitata della realtà ma accurata negli aspetti che necessitiamo per la risoluzione del problema

descrizione di problemi
- soluzioni ammissibili (insieme Fp):
  - specificato dando $G \supseteq Fp$ e descrivendo dei vincoli che un generico g appartenente G deve soddisfare per far parte di Fp
  - G – Fp sono le soluzioni non ammissibili

problemi di ottimizzazione
- P viene descritto:
  - dando l'insieme Fp
  - specificando una funzione obiettivo cp : Fp a R che misuri costo o beneficio di ogni soluzione ammissibile
- un problema di minimo P consiste nel determinare il valore $Zp = \min\{cp(g)|g \in Fp\}$

- un problema di massimo P consiste nel determinare il valore $Zp = \max\{cp(g) | g \in Fp\}$
  - ad ogni problema di massimo P ne corrisponde uno di minimo P' tale che $cp'(g) = -cp(g)$
    - $Zp = -\min\{cp'(g) \mid g \in Fp = Fp'\}$
  - dato P, Zp è valore ottimo per P
  - dato P, un $g^* \in Fp$ tale che $Zp = cp(g^*)$ è soluzione ottima

casistica
- problema vuoto: $Fp = \emptyset$, per convenzione si assume che $Zp = \infty$
- problema illimitato: problema di massimo, $\forall x \in R$ esiste $g \in Fp$ con $cp(g) \geq x$; $Zp = +\infty$
- valore ottimo finito, ma non soluzione ottima finita: Zp esiste finito, ma $cp(g) \neq Zp\ \forall g$
- valore ottimo finito e soluzione ottima finita

ottimizzazione e decisione
- problema di decisione P: determinare una qualunque $g \in Fp$ o concludere che il problema è vuoto ($Fp = \emptyset$)
- dato un problema di decisione P, il relativo problema di certificato per $G \supseteq Fp$ consiste nel dire se $g \in Fp$ con $g \in G$
- dato un problema P di ottimizzazione
  - si può considerare R decisionale tale che $Fr = \{g \in Fp \mid cp(g) = Zp\}$
  - dato $x \in R$, si può considerare Rk decisionale tale che $Frk = \{g \in Fp \mid cp(g) \leq k\}$ se P è di minimo, altrimenti duale

ottimizzazione e algoritmi
- algoritmo esatto per P: presa in input un'istanza di P fornisce in output una soluzione ottima $g^*$ di P se esiste
- algoritmi euristici: determinano una qualsiasi soluzione ammissibile
  - calcolano implicitamente un'approssimazione superiore (se problema di minimo) e una superiore (se di massimo) del valore ottimo

qualità degli algoritmi euristici
- dato che potrebbero concludere con una non esistenza di una soluzione ammissibile anche se $Fp \neq \emptyset$, dati P e $g \in Fp$ definiamo
  - errore assoluto di g $\varepsilon p(g) = cp(g) - Zp$
  - errore relativo di g $Rp(g) = \varepsilon p(g) / |Zp| = (cp(g) - Zp) / |Zp|$
- una soluzione g è ε-ottima se $Rp(g) \leq \varepsilon$
- un algoritmo euristico si dice ε-approssimato se produce soluzione ε-ottime

rilassamenti
- quando l'errore diventa problematico, si risolve il problema che è l'approssimazione del problema in partenza
- dato P, un rilassamento di P (per esempio di minimo) è un qualunque problema $\overline{P}$ definito $\min\{c\overline{p}(g) \mid g \in F\overline{p}\}$
  - con $F\overline{p} \supseteq Fp$ e per ogn $Fp.c\overline{p}(g) \leq cp(g)$ (dualmente per i problemi di massimo)
  - $Z\overline{p}$ è inferiore a Zp
- se la soluzione ottima $g^*$ di $\overline{P}$ soddisfa $g^* \in Fp$ e $c\overline{p}(g^*) = cp(g^*)$, allora $c\overline{p}(g^*) = Z\overline{p} \leq Zp \leq cp(g^*) \leq c\overline{p}(g^*)$

modelli

da problemi a modelli
- si classificano i problemi individuati secondo categorie chiamate modelli (tra cui la programmazione lineare)

programmazione lineare (PL)
- un problema di PL è un problema di ottimizzazione definito dando:
  - un numero finito $n \in \mathbb{N}$ di variabili reali: $x = (x1, ..., xn) \in R^n$
  - una funzione obiettivo $f : R^n \to R$: $f(x) = cx$

  - un insieme di m vincoli lineari, in forma: $ax = b$, $ax \leq b$ o $ax \geq b$, con $a \in R^n$ e $b \in R$
- a volte è utile assumere che $x \in \mathbb{N}^n$ (le soluzioni ammissibili sono (vettori di) numeri naturali, PLIntera)
- un problema di PL può sempre essere espresso come max{cx | Ax ≤ b} con A ∈R^mxn e b ∈R^m
  - se P è di minimo, basta considerare f(x) = (-c)x
  - ogni vincolo ax = b diventa coppia di vincoli ax ≤ b e ax ≥ b
  - ogni vincollo ax ≥ b equivale a (-a)x ≤ (-b)

PLI
- nella PL, le variabili rappresentato quantità
- nella PLI, le variabili sono:
  - quantitative
  - logiche: rappresentano valori booleani; una variabile è logica se $x \in \mathbb{N}$, $0 \leq x$, $x \leq 1$
- le variabili logiche possono essere usate per modellare:
  - l'assegnamento di una risorsa a una task
  - il fatto che un'attività si debba eseguire o no

relazioni logiche
- le relazioni tra variabili logiche hanno natura logica
- le possiamo modellare tramite vincoli lineari:
  - negazione (y = !x): x = 1-y
  - implicazione (z = (x -> y)): x+z >= 1; z >= y; x+z ≤ 1+y
  - congiunzione (z = (x and y)): z ≤ x; z ≤ y; z ≤ x+y-1
  - disgiunzione (z = (x or y)): z >= x; z >= y; z ≤ x+y
- come conseguenza si ha che il problema è NP-difficile

vincoli di assegnamento
- partendo da:
  - insieme N = {1, ..., n} di oggetti
  - insieme V = {1, ..., m} di luoghi
- l'idea è di rappresentare le condizioni in cui si assegnano oggetti a luoghi
- la variabile xij (con 1 ≤ i ≤ n e 1 ≤ j ≤ m) prende valore booleano e modella il fatto che l'i-esimo oggetto è al j-esimo luogo
- vincoli di semi-assegnamento: ogni oggetto è assegnato a un luogo
  - ∑ (j tra 1 e m) xij = 1 (1 ≤ i ≤ n)
- insiemi ammissibili:
  - se ogni oggetto i ∈{1, ..., n} non può essere assegnato a uno specifico insieme B(i) ⊆V di luoghi
  - xij esiste solo se i ∈B(i)
  - ∑ (j ∈B(i)) xij = 1 (1 ≤ i ≤ n)
- vincoli di assegnamento: ogni oggetto è assegnato a un luogo e ogni luogo è assegnato a un oggetto
  - ∑ (j tra 1 e m) xij = 1 (1 ≤ i ≤ n)
  - ∑ (i tra 1 e n) xij = 1 (1 ≤ i ≤ m)
- ordinamento:
  - i vincoli di assegnamento (non semi) possono imporr che n lavori siano eseguiti in un certo ordine
  - la variabile xj indica se l'i-esim lavoro è effettuato come j-esimo (se vale 1) o meno (se vale 0)

selezione di sottoinsiemi
- sia N = {1, ..., n} un insieme finito di elementi e sia F = {F1, ..., Fm} una famiglia di

sottoinsiemi, con Fi ⊆N

- ∀Fj (con 1 ≤ j ≤ m) associamo un costo cj
- voglio determinare D ⊆F di costo minimo, tra tutti i sottoinsiemi che soddisfano certi vincoli
- rappresento la situazione con una matrice A = aij ∈{0, 1}^nxm con aij = 1 se i ∈Fj, 0 altrimenti
- il vettore delle variabili è x = (x1, ..., xn) con xj = 1 se Fj ∈D, 0 altrimenti
- la funzione obiettivo da minimizzare sarà sempre ∑ (i tra 1 e m) cj xj
- i vincoli dipendono dal problema:
  - problema di copertura: ognuno degli elementi di N sta in almeno un elemento di D
    - ∑ (j tra 1 e m) aij xij >= 1 (1 ≤ i ≤ n)
  - problema di partizione: ognuno degli elementi di N sta in un elemento di D
    - ∑ (j tra 1 e m) aij xij = 1 (1 ≤ i ≤ n)
  - problema di riempimento: ognuno degli elementi di N sta al più uno degli elementi di D
    - ∑ (j tra 1 e m) aij xij ≤ 1 (1 ≤ i ≤ n)

variabili a valori discreti

- se le variabili non sono vincolate a prendere il valore da un insieme che: non è {0, 1}, non è N o un intervallo
  - introduciamo n variabili y1, ..., yn vincolate yi ∈{0, 1}, ∑ (i tra 1 e n) yi = 1, x = ∑ (i tra 1 e n) vi yi

minima quantità positiva prefissata

- quando una variabile x rappresenta un livello di produzione, capita che il suo valore debba viaggiare in un insieme {0} un [l, u]
  - 0: assenza di produzione
  - [l, u]: possibili livelli di produzione quando il meccanismo è attivo
- per modellare serve:
  - introdurre una variabile logica y ∈{0, 1} che indica la presenza o meno di produzione
  - i vincoli sono ly ≤ x, x ≤ uy
  - correttamente, se y = 0, x = 0; altrimenti l ≤ x ≤ u

valore assoluto

- nei vincoli:
  - il vincolo |g(x)| ≤ b può essere espresso come g(x) ≤b o -g(x) ≤ b (b reale positivo)
  - in casi più complessi non è sempre possibile
- nella funzione obiettivo:
  - massimizzazione di |f(x)| con x ∈X: si può risolvere risolvendo max{f(x) | x ∈X} e max{-f(x) | x ∈X} e confrontando i valori ottimi
  - uguale per la minimizzazione

funzioni convesse

- nelle funzioni lineari a tratti la necessità di usare variabili logiche deriva dalla non-convessità
- funzione convessa: se valgono le condizioni
  - f continua ($b_{i+1}+c_{i+1}a_{i+1} = b_i+c_ia_{i+1}$) ∀1 ≤ i < 1
  - la derivata di f deve essere non descrescente, cioè $c_{i+1}$ >= ci ∀1 ≤ i < 1
- se f è convessa, la minimizzazione diventa quella di $g(z_1, ..., z_n) = b_1+c_1a_1+$∑(i tra 1 e n) cizi
  - vincoli: $0 \leq z_i \leq a_{i+1}-a_i$, $x = a_1+$∑(i tra 1 e n) $z_i$
    - se f ha un ottimo x*, questo valore ottimo può essere ricostruito usando i segmenti di indice inferiore grazie alla convessità

reti di flusso

reti

- rete: grafo G = (N, A), solitamente diretto, ai cui archi siano associati pesi
- archi: canali in cui fluiscono oggetti, rappresentati da grandezze discrete o continue
- nodi: punti di ingresso o uscita dalla rete
- ad ogni nodo i ∈N è associato un reale bi detto sbilanciamento:
  - positivo
    - il nodo i è un nodo di uscita e viene detto destinazione, pozzo o nodo di output
    - bi è detto domanda di i
  - negativo
    - il nodo i è un nodo di entrata e si chiama origine, sorgente o nodo di input
    - -bi è detto offerta di i
  - nullo
    - il nodo i è detto nodo di traferimento
- ad ogni arco (i, j) ∈A sono associati:
  - costo cij: quanto costa per un' unità di bene attraversare il canale
  - capacità inferiore lij: limite inferiore alla quantità di beni che possa fluire sul canale
  - capacità superiore uij: limite superiore alla quantità di beni che possa fluire sul canale

problemi di flusso

- sono un compromesso tra espressività (molti problemi concreti si possono esprimere come problemi di flusso) e complessità (esistono algoritmi relativamente efficienti anche per essi)
- flusso: assegnamento di valori reali agli archi di una rete G = (N, A); soluzione dei problemi di flusso
- viene formalizzato attraverso una sequenza di variabili xij, ciascuna corrispondente a un arco (i, j) ∈A
- il costo di un flusso è il costo complessivo di tutti i flussi nella rete
  - ∑ (i, j ∈A) cij xij
- vincoli:
  - domanda e offerta globale sono uguali
    - ∑ (i ∈D) bi = -∑ (i ∈O) bi ⇔∑ (i ∈N) bi = 0
    - D = {i ∈N | bi > 0}
    - O = {i ∈N | bi < 0}
  - il flusso si conserva
    - ∑ ((j, i) ∈BS(i)) $x_{ij}$ – ∑ ((i, j) ∈FS(i)) $x_{ij}$ = $b_i$ con i ∈N
    - BS(i) = {(k, i) | (k, i) ∈A}
    - FS(i) = {(i, k) | (i, k) ∈A}
  - il flusso deve essere ammissibile
    - $l_{ij}$ <= $x_{ij}$ <= $u_{ij}$ con (i, j) ∈A

problema del flusso di costo minimo (MCF)

- il costo del flusso è la funzione obiettivo da minimizzare
- le capacità inferiori sono nulle
- il problema è formalizzabile in PL: min(cx) con 0 <= x <= u, Ex = b
  - c ∈R^[A]: vettore dei costi
  - u ∈R^[A]: vettore delle capacità
  - E ∈R^[N]x[A]: matrice di incidenza tra nodi e archi
  - b ∈R^[N]: vettore degli sbilanciamenti
- rilassare assunzioni:
  - spesso conviene assumere che vi sia una sola sorgente e un solo pozzo
  - si può trasformare il problema:
    - aggiungendo due nodi fittizi: l'unica sorgente e l'unico pozzo

    - aggiungendo archi fittizi dalla sorgente a ciascun'altra sorgente: costo nullo e capacità superiore pari all'inverso dello sbilanciaento della sorgente
    - aggiungendo archi fittizi da ciascun pozzo a quello unico: costo nullo e capacità superiore pari allo sbilanciamento del pozzo
    - sbilanciamento dell'unica sorgente = somma degli sbilanciamenti delle sorgenti della rete di partenza; uguale per i pozzi
  - conviene imporre che lij = 0 per ogni arco (i, j) ∈A (capacità inferiori nulle)
  - si costruisce una rete H che sia equivalente a G ma con capacità inferiori nulle: per arco (i, j) ∈A
    - si sottrae lij a bj e uij
    - si aggiunge lij a bj
    - si aggiunge ∑ ((i, j) ∈A) cij lij alla funzione obiettivo
    - in pratica a un flusso xij in H corrisponde un flusso xij+lij in G
  - conviene imporre che i nodi (oltre agli archi) abbiano capacità (solo una quantità di flusso compresa in [li, ui] può passare per il nodo i ∈N)
  - si sdoppia ciascun nodo i in due nodi i' e i''
    - tutti gli archi entranti in i vanno in i'
    - tutti gli archi uscenti da i partono da i''
    - ci sia un arco fittizio (i', i'') con costo nullo, capacità inferiore li e capacità superire ui

problema del flusso massimo (MF)

- restrizione di MCF: l'obiettivo è di massimizzare i flussi
  - costi nulli
  - sbilanciamenti nulli
  - si aggiunge un arco fittizio da t a s di costo -1 e capacità infinita
- formalizzabile direttamente in PL
- data una rete G = (N, A)
  - fissiamo due nodi s e t
  - vogliamo massimizzare il flusso tra essi, cioè trovare il massimo valore v | se bs = -v, bt = v e bi = 0 in tutti gli altri casi, allora esiste un flusso ammissibile
- valore v ammissibile: valore del flusso x
- max v
  - som ((j, s) ∈BS(s)) $x_{js}$+v = som ((s, j) ∈FS(s)) xsj
  - som ((j, i) ∈BS(s)) xij – som((i, j) ∈FS(s)) xij = 0 con i ∈N-{s, t}
  - som ((j, t) ∈BS(t)) xjt = som ((t, j) ∈FS(t)) $x_{tj}$+v
  - o <= xij <= uij con (i, j) ∈A

tagli

- in una rete G = (N, A), un taglio è dato da una coppia (N', N'') di sottoinsiemi di N | N' ∩ N'' = ∅e N' un N'' = N
- un (s, t)-taglio in una rete G è un taglio (Ns, Nt) con s ∈Ns e t ∈Nt
- dato un (s, t)-taglio in G = (N, A), indichiamo con A+(Ns, Nt) e A-(Ns, Nt) i sottoinsiemi
  - A+(Ns, Nt) = {(i, j) ∈A | i ∈Ns ^ j ∈Nt}
  - A-(Ns, Nt) = {(i, j) ∈A | i ∈Nt ^ j ∈Ns}
- proprietà:
  - lemma: per ogni (s, t)-taglio e ogni flusso ammissibile x con valore v
    - v = som ((i, j) ∈A+(Ns, Nt)) xij – som ((i, j) ∈A-(Ns, Nt)) xij = flusso del taglio, indicato con x(Ns, Nt)
    - v <= som ((i, j) ∈A+(Ns, Nt)) uij = capacità del taglio, indicata con u(Ns, Nt)
    - dimostrazione: conseguenza di
      - v = som ((i, j) ∈A) xij – som ((i, j) ∈A) xis =

            - = som (k ∈N) (som ((k, j) ∈A) xkj – som ((i, k) ∈A) xik ) =
            - = som ((i, j) ∈A+(Ns, Nt)) xij – som ((i, j) ∈A-(Ns, Nt)) xij
        - il lemma dice che v = x(Ns, Nt) <= u(Ns, Nt)
            - il flusso ammissibile è sempre minore o uguale alla capacità di qualunque taglio

grafi residui

- data una rete G = ($N_G$, $A_G$) e un flusso ammissibile x, il grafo residuo Gx è il multigrafo ($N_{Gx}$, $A_{Gx}$) |
    - Ngx = NG
    - gli archi in Agx sono
        - concorde: per ogni arco (i, j) ∈A | xij < uij esiste un arco concorde da i a j in Gx
        - discorde: per ogni arco (i, j) ∈A | xij > 0 esiste un arco discorde da j a i in Gx
    - notiamo che in NGx possono esserci due archi tra due stessi nodi i e j

cammini aumentanti

- un cammino aumentante P in una rete G rispetto a x è un cammino orientato da s a t in Gx
    - sia P+ (insieme archi concordi) sia P- (insieme archi discordi)
- dato un cammino aumentante P rispetto a x, definiamo la capacitò di P rispetto a x
    - θ(P, x) = min{min{uij-xij | (i, j) ∈P+}, min{xij | (j, i) ∈P-}}
- dato un flusso x, un cammino P in Gx e un reale θ, definiamo x(P, θ) il flusso
    - (x(P, θ))ij = xij+θ se (i, j) ∈P+, xij-θ se (j, i) ∈P-, xij altrimenti

algoritmo di ford – fulkerson

    - 1. x ← 0
    - 2. costruisci Gx e determina se ha un cammino aumentante P: se non esiste, termina e restituisci x
    - 3. x ← x(P, θ(P, x))
    - 4. torna al 2
- correttezza:
    - lemma: se x è ammissibile, allora anche x(P, θ(P, x)) lo è
    - lemma: se x è un flusso ammissibile massimo, allora Gx non ha cammini aumentanti
        - dimostrazione: se ci fosse un cammino aumentante in Gx, allora x non sarebbe massimo (sarebbe possibile aumentare il valore del flusso)
    - lemma: se Gx non ha cammini aumentanti, allora esiste un taglio di capacità v
        - dimostrazione: consideriamo il taglio (Ns, Nt) dove Ns contiene tutti i nodi raggiungibili da s in Gx e Nt = N-Ns
        - v = x(Ns, Nt) = som ((i, j) ∈A+(Ns, Nt)) xij – som ((i, j) ∈A-(Ns, Nt)) xij =
        - = som ((i, j) ∈A+(Ns, Nt)) uij – som ((i, j) ∈A-(Ns, Nt)) 0 =
        - = u(Ns, Nt)
    - teorema di correttezza: se l'algoritmo di ford – fulkerson termina, allora il flusso x è un flusso massimo
        - dimostrazione:
            - se ford – fulkerson termina, allora Gx non ha cammini aumentanti
            - allora esiste un taglio di capacità v
            - v è per forza massimo, in quanto se non lo fosse avremmo un taglio di capacità inferiore al valore di un flusso ammissibile
    - teorema max-flox min-cut: il valore del massimo flusso è uguale alla minima capacità dei tagli
        - dimostrazione: basta dimostrare che il valore del massimo flusso è maggiore o uguale alla capacità di un taglio
            - se x è ammissibile e massimo, Gx non ha cammini aumentanti quindi esiste un taglio di capacità v

- complessità:
  - teorema: se le capacità di G sono numeri interi, allora esiste almeno un flusso intero massimo
    - dimostrazione:
      - se le capacità sono intere, allora il flusso massimo sarà al più nU con U = max{uij | (i, j) ∈A}
      - di conseguenza, il valore del flusso aumenta almeno di 1 a ogni iterata
      - l'algoritmo termina al più nU iterate
    - la dimostrazione ci dice che se le capacità non sono intere, la complessità è O(mnU)
      - pseudopolinomiale nella dimensione della rete

algoritmo di edmonds – karp

- algoritmo di ford-fulkerson dove la ricerca del cammino aumentante è eseguita visitando in ampiezza (BFS) il grafo residuo Gx
- i cammini aumentanti saranno sempre di lunghezza minima
- proprietà:
  - correttezza: trivialmente corretto (perchè un caso particolare di ford-fulkerson)
  - complessità:
    - se in ford-fulkerson i cammini aumentanti sono di lunghezza minima, allora la distanza di un nodo i dalla sorgente s in Gx non può diminuire
    - quindi il numero di iterazioni EK non può essere più grande di N*A
  - data una rete F, un flusso ammissibile x e due nodi i, j in G, indichiamo con δx(i, j) la distanza tra i e j nel grafo residuo Gx
    - lemma: se durante l'esecuzione di EK il flusso y è ottenuto da x tramite un'operazione di aumento del flusso in un cammino aumentante, allora per ogni nodo i ∈N vale che δx(s, i) <= δy(s, i)
  - teorema: il numero di iterazioni di EK è O(N A), quindi la sua complessità è O(N A^2)
    - dimostrazione:
      - un arco (i, j) in Gx è critico per un cammino aumentante P se la sua capacità (uij-xij se concorde, xij se discorde) è uguale a θ(P, x)
        - dopo l'aumento del flusso lungo P, l'arco (i, j) sparisce dal grafo residuo
        - in ogni cammino aumentante esiste almeno un arco critico
      - dati i, j connessi da un arco in A, il numero di volte che (i, j) sia arco critico è al più O(N)
        - dato che di coppie tali ce ne sono al più O(A), in totale avremo al più O(N A) iterazioni
        - dimostrazione:
          - quando (i, j) diventa critico la prima volta, deve valere che δx(s, j) = δx(s, i)+1 dove x è il flusso e sparisce dal grafo residuo
          - l'unico modo per ricomparire è fare in modo che il flusso (reale o virtuale) da i a j diminuisca, quindi δy(s, i) = δy(s, j)+1 con y flusso
          - quindi, δy(s, i) = δy(s, j)+1 ≥ δx(s, j)+1 = δx(s, i)+2
          - di conseguenza, dal momento in cui (i, j) diventa critico al successivo, la sua distanza da s aumenta di almeno 2
          - dato che questa distanza non può essere superiore a |N|, il numero di volte in cui (i, j) può diventare critico è lineare in |N|

preflusso:

- vettore x | som ((i, j) ∈BS(i)) xij – som ((i, j) ∈FS(i)) xij ≥0 con i ∈N-{s, t}, 0 <= xij <= uij, (i, j) ∈A
- cioe, i vincoli di capacità sono soddisfatti mentre quelli di bilanciamento ai nodi possono non esserlo

- un nodo è attivo se il suo eccesso ei = som ((j, i) ∈BS(i)) – som ((i, j) ∈FS(i)) è positivo, altrimenti è bilanciato

- etichettatura: vettore d = d1, ..., dn dove di ∈R+ per ogni nodo i ∈N
  - valida se:
    - (i, j) ∈A ^ xij < uij => di-dj <= 1
    - (j, i) ∈A ^ xij > 0 => di-dj <= 1
    - dt = 0
  - data un'etichettatura valida d, un arco (i, j) è ammissibile per i sse non è saturo e di = dj+1
  - analogamente, (i, j) è ammissibile per j sse non è vuoto e dj = dj+1
- operazione di push:
  - se i è un nodo attivo ed esiste un arco (i, j) ammissibile per esso, allora è possibile inviare l'eccesso, o una sua parte, lungo l'arco tramite push
  - push forward (i, j):
    - 1. θ ← min{ei, uij-xij}
    - 2. xij ← xij+θ
    - 3. ei ← ei-θ
    - 4. ej ← ej+θ
  - push backward (i, j):
    - 1. θ ← min{ei, xij}
    - 2. xij ← xij-θ
    - 3. ei ← ei-θ
    - ej ← ej+θ
  - operazione di relabel: necessaria se il nodo i non ha nodi incidenti che siano ammissibili
    - consiste nell'aumentare l'etichetta di i: rende ammissibile almeno un aro incidente in i (quello per cui è stato ottenuto il valore minimo)
    - relabel (i): di ← 1+min{dj | ((i, j) ∈FS(i) and xij < uij) or ((j, i) ∈BS(i) and xji > 0)}
- etichettatura valida (G) d:
  - 1. supponiamo che il preflusso x sia nullo negli archi in uscita da s
  - 2. facciamo in modo che $d_i$ sia la lunghezza del cammino di lunghezza minima da i a t
    - tranne in s dove ds = n
  - 3. verifichiamo i vincoli

algoritmo di goldebrg – tarjan

- tipologia di edmonds-karp nella quale si cerca si scendere sotto la barriera di O(N A^2) per la complessità, e lo si fa rendendo la costruzione del flusso più locale
  - opponendosi così a ford-fulkerson e edmonds-karp, in cui a ogni iterazione si procede con un'analisi globale
- goldberg tarjan (G, s, t):
  - 1. x ← 0
  - 2. xsj ← usj per ogni (s, j) app FS(s)
  - 3. d ← etichettatura valida (G)
  - 4. ds ← n
  - 5. se tutti i nodi (non s e t) sono bilanciati, allora termina e restituisci x
  - 6. sia v un nodo sbilanciato
  - 7. se esiste (v, j) ammissibile per v, allora push forward (v, j) e torna a 6, altrimenti prosegui
  - 8. se esiste (i, v) ammissibile per v, allora push backward (i, v) e torna a 6, altrimenti prosegui

    - 9. relabel (v) e torna a 6
- teorema: l'algoritmo di goldberg-tarjan è corretto e la sua complessità in tempo è O(N^2 A)
    - due invarianti sono validi all'inizio di ciascuna iterazione: l'etichettatura d è valida e x è un preflusso

problema del flusso di costo minimo
nozioni e risultati preliminari
- pseudoflusso: vettore x che soddisfa i vincoli di capacità
    - cioè | 0 <= xij <= uij con (i, j) ∈A
- dato x pseudoflusso, lo sbilanciamento di un nodo i rispetto a x è
    - ex(i) = som ((j, i) ∈BS(i)) xij – som ((i, j) ∈FS(i)) xij-bi
    - se ex è un vettore, è il vettore degli sbilanciamenti
- dato x pseudoflusso, i nodi sbilanciati rispetto a x fanno parte di:
    - Ox = {i ∈N | ex(i) > 0}, nodi con eccesso di flusso, oppure
    - Dx = {i ∈N | ex(i) < 0}, nodi con difetto di flusso
    - se Ox = Dx = ∅, allora x è un flusso
    - sbilanciamento complessimo di x:
        - g(x) = som (i ∈Ox) ex(i) = -som (j ∈Dx) ex(j)

cammini aumentanti
- quando si lavora con pseudoflussi, la nozione di cammino aumentante diventa più generale
- il grafo residuo Gx per uno pseudoflusso x si generalizza a un problema MCF, in cui però ogni arco ha un costo:
    - arco concorde (i, j) di Gx: costo cij
    - arco discorde (j, i) di Gx: costo -cij
- un cammino P tra i e j in Gx è quindi aumentante
    - gli archi possono essere partizionati in P+ e P-
    - è anche detto ciclo aumentante
- dato x pseudoflusso e P cammino aumentante, è possibile inviare 0 <= θ <= θ(P, x) unità di flusso lungo P attraverso x(P, θ) (indicato anche come x⊕Pθ)
- se P è un cammino aumentante da i a j in Gx, allora lopseudoflusso x(P, θ) ha gli stessi sbilanciamenti di x tranne in i e j
    - se i = j, il vettore degli sbilanciamenti non cambia
- costo di un cammino aumentante P:
    - c(P) = som ((i, j) ∈P+) cij – som ((i, j) ∈P-) cij
- c*(x(P, θ)) = c*(x⊕Pθ) = c*x+ θc(P)
- struttura degli pseudoflussi:
    - teorema: siano x e y due pseudoflussi, esistono k <= n+m cammini aumentanti P1, …, Pk, tutti per x, di cui al più m sono cicli | $z_1$ = x, $z_{i+1}$ = $z_i \oplus \theta_i P_i$ con 1 <= i <= k, $z_{k+1}$ = y, 0 <= $\theta_i$ <= θ($P_i$, $z_i$)
    - tutti i $P_i$ hanno come estremi dei nodi in cui o sbilanciamento di x è diverso da quello di y

pseudoflussi minimali
- a differenza del MF, in MCF non possiamo aumentare il flusso indiscriminatamente
- il problema è di determinare quali siano le operazioni di aumento lecite e quali sono le properità dei flussi che garantiscono
- pseudoflusso minimale: pseudoflusso x che ha costo minimo tra tutti gli pseudoflussi con lo stesso sbilanciamento ex
- lemma: uno pseudoflusso (flusso ammissibile) è minimale (ottimo) sse non esistono cicli aumentanti di costo negativo

- dimostrazione:
  - per contrapposizione: se esiste un ciclo aumentante di costo negativo in Gx, applicarlo diminuisce il costo senza alterare lo sbilanciamento, in contraddizione con la minimalità di x
  - per contrapposizione: supponiamo che x non sia minimale, cioè esiste y con cy <= cx e ey = ex
    - per il teorema sugli pseudoflussi, y = x⊕θ1⊕ … ⊕θnPn con θi > 0 e ogni Pi è un ciclo
    - da cy < cx discende cx > cx+θ1c(P1)+ … +θnc(Pn) e quindi c(Pi) < 0 per qualche i
- teorema: sia x uno pseudoflusso minimale e sia P un cammino aumentante rispetto a x avente costo minimo tra tutti i cammini che uniscono un nodo di Ox a uno di Dx
  - allora qualunque sia θ <= θ(x, P) abbiamo x(θ, P) = x⊕θP è ancora pseudoflusso minimale
  - dimostrazione:
    - siano s e t i vertici che P collega, supponiamo che θ <= θ(x, P) e che y sia uno pseudoflusso con vettore di sbilanciamento ex(θ, P)
    - per il teorema della struttra degli pseudoflussi esistono:
      - k cammini aumentanti P1, …, Pk rispetto a x, tutti da s a t
      - h cicli aumentanti C1, …, Ch rispetto a x
      - | y = x⊕θ1P1 ⊕... ⊕θkPk ⊕μ1C1 ⊕... ⊕μhCh (tutti gli θi, μi sono positivi)
    - per lo sbilanciamento, som (1 <= i <= k) θi = θ
    - x è minimale, c(Ci) ≥ 0
    - P ha costo minimo, c(Pi) ≥ c(P)
    - quindi, cy = cx+θ1c(P1)+ ... +θkc(Pk)+μ1c(C1)+ ... +μhc(Ch) ≥ cx+θc(P) = cx(θ, P)

algoritmi ausiliari

- cammini minimi successivi
  - cammini minimi successivi (G):
    - 1. x ← pseudoflusso minimale (G)
    - 2. se g(x) = 0, allora termina e restituisci x
    - 3. cerca un cammino di costo minimo P tra un nodo i ∈Ox e j ∈Dx:se non esiste, termina, altrimenti il problema è vuoto
    - 4. x ← x(P, min{θ(P, x), ex(i), -ex(j)})
    - 5. torna a 2
  - correttezza:
    - a ogni passo il flusso x rimane minimale
    - se l'algoritmo termina, allora g(x) = 0, quindi x è uno pseudoflusso minimale con sbilanciamento nullo, cioè un flusso di costo minimo
  - terminazione:
    - se b e u sono vettori di numeri interi osserviamo che:
      - lo pseudoflusso iniziale è intero
      - se x è uno pseudoflusso intero, allora la capacità θ(P, x) rimane intera
      - quindi lo pseudoflusso x rimane sempre intero
      - ad ogni passo g(x) diminuisce di almeno 1
  - complessità:
    - lo sbilanciamento iniziale $\bar{g}$ è al più $\bar{g}$ <= som (bi > 0) bi + som (cij < 0) uij
    - dato che lo sbilanciamento g(x) cala almeno di 1 ad ogni interazione, le iterazioni saranno al più $\bar{g}$
    - il costo computazionale di ogni iterazione è dominato dalla ricerca di un cammino

        - minimo eseguibile in tempo O(N A)
        - quindi la complessità è nel caso peggiore $O(\overline{g}\, N A)$, pseudopolinomiale nella dimensione del grafo

flusso ammissibile (G)

- risolviamo il problema MF sulla rete ottenuta aggiungendo due nodi s e t

cancellazione cicli

- cancellazione cicli (G):
    - 1. se flusso ammissibile (G) restituisce un flusso ammissibile, allora mettilo in x, altrimenti termina (problema vuoto)
    - 2. cerca un ciclo di costo negativo in Gx: se non lo trovi, termina e restituisci x, altrimenti metti il ciclo in C
    - 3. x ← x(C, θ(C, x))
    - 4. torna al 2
- correttezza: conseguenza del lemma sull'equivalenza tra assenza di cicli aumentati e ottimalità
- se le capacità sono numeri interi, allora il costo diminuisce di almeno 1 a ogni iterazione e l'algoritmo termina
- il costo di qualunque flusso ammissibile è compreso tra $-A\overline{uc}$ e $A\overline{uc}$, dove
    - $\overline{u} = \max\{u_{ij} \mid (i, j) \in N\}$
    - $\overline{c} = \max\{c_{ij} \mid (i, j) \in N\}$
- complessità: pseudopolinomiale; $O(N A) * O(A\overline{uc}) = O(N A^2\overline{uc})$

problemi di accoppiamento

nozioni preliminari

- i grafi sono bipartiti non orientati, cioè grafi nella forma G = (O un D, A) con
    - O = {1, ..., n} insieme dei nodi origine
    - D = {n+1, ..., n+d} insieme dei nodi destinazione
    - $A \subseteq O \times D$ insieme degli archi, ciascuno dei quali ha un costo
- accoppiamento per grafo bipartito G = (O un D, A): sottoinsieme M di A i cui archi non hanno nodi in comune
    - gli archi in M sono interni, quelli in A-M esterni
    - i nodi che compaiono in qualche arco di M sono accoppiati, gli altri sono esposti
- accoppiamento perfetto: sse non vi sono nodi esposti in M
- costo di un accoppiamento M: c(M) = som ((i, j) ∈M) cij
- dato M, l'arco (i, j) ∈M di costo massimo è detto arco bottleneck e ha valore di bottleneck $\max\{c_{ij} \mid (i, j) \in M\}$

problemi

- accoppiamento di massima cardinalità
- accoppiamento di costo minimo: tra tutti gli accoppiamenti perfetti
- accoppiamento di massima cardinalità bottleneck: tra tutti gli accopiamenti di massima cardinalità si vuole determinare quello con valore di bottleneck minimo

accoppiamento di massima cardinalità

- il problema può essere visto come di flusso massimo con più sorgenti (i nodi in O) e più pozzi (nodi in D)
    - tutte le capacità pari a 1
    - teniamo conto solo dei flussi interi
- flussi ammissibili interi e accoppiamenti sono in corrispondenza biunivoca (a un flusso M corrisponde un grafo residuo $G_M$)
- si possono utilizzare algoritmi classici, ma è possibile anche sfruttare le caratteristiche di questi problemi

- ogni cammino aumentante in GM deve:
  - essere alternante, cioè consistere di archi interni, seguiti da archi esterni
  - partire da un'origine esposta e arrivare a una destinazione esposta
  - cioè, se PE = P-M e P1 = M int P sono rispettivamente gli archi esterni e interni di un cammino aumentante P, allora |PE|-|P1| = 1
- la capacità θ(M, P) di un cammino aumentante P è sempre 1, quindi M⊕(θ(M, P))P = (M-PI) un PE
- otteniamo così un algoritmo simile a ford-fulkerson, ma in cui la ricerca del cammino aumentante è eseguita solo atttraverso una procedura di visita
- complessità: O(mn) perchè U = max{cij | (i, j) ∈A} = 1
- nel caso il problema sia l'accoppiamento di costo minimo, si può procedere uguale ma specializzando gli algoritmi per MCF
  - i cammini minimi aumentanti possono essere visti come cammini esposti tra due vertici esposti, rispettivamente in O e D

geometria della PL
restringere lo spazio di ricerca
- lo spazio di ricerca nei problemi PL è infinito (ha anche cardinalità del continuo)
- in alcuni casi lo si può ridurre a un insieme  finito (insieme dei vertici del poliedro che definisce la regione ammissibile)

nozioni preliminari
- iperpiano: insieme {x ∈R^n | ax = b} delle soluzioni dell'equazione lineare ax = b, con a ∈ R^n e b ∈R
- semispazio: insieme {x ∈R^n | ax <= b} delle soluzioni dell'equazione lineare ax <= b con a ∈R^n e b ∈R
  - un iperpiano è il confine del corrispondente semispazio
- poliedro: intersezione P di un numero finito m di semispazi
  - devono esistere una matrice A ∈R^mxn e un vettore b ∈R^m | P = {x | Ax <= b}
- insieme convesso: insieme C cont R^n | tutti i punti che connettono x, y ∈C sono anch'essi in C
  - cioè, per ogni x, y ∈C per ogni α ∈[0, 1] αx+(1-α)y ∈C
  - semispazi e poliedri sono insiemi convessi
- se consideriamo il poliedro P = {x | Ax <= b} (A ∈R^mxn) e fissiamo un sottoinsieme I di {1, ..., m}:
  - $\bar{I}$ = {1, ..., m}-I complementare di I
  - $A_I$ sottomatrice di A ottenuta considerando solo le righe con indice in I
  - $P_I$ = {x | $A_I$x = $b_I$ ^ $A_{\bar{I}}$x <= $b_{\bar{I}}$} poliedro
- faccia: il poliedro PI se I | PI non è vuoto
  - il numero di facce distinte di un poliedro {x | Ax <= b} (A ∈R^mxn) è al massimo 2^m
  - P stesso è la faccia P∅
  - faccette: facce proprie (non banali) e massimali
  - dimensione della faccia: dimensione del più piccolo sottospazio che la contiene
    - una faccia determinata da una matrice $A_I$ di rango k ha dimensione n-k o inferiore

vertici
- vertici: facce determinate da matrici AI di rango n quindi con dimensione 0
  - per ipotesi sul rango di $A_I$, $A_I$x = $b_I$ ha una sola soluzione
  - le facce sono sempre non vuote
- spigoli: facce individuate da sottomatrci AI di rango n-1 e dimensione al massimo di 1

soluzioni di base

- supponiamo B | AB sia matrice quadrata e invertibile:
  - B = base
  - AB = matrice di base
  - xB = AB^-1 bB = soluzione di base
- soluzione di base xB | xB ∈P è ammissibile, altrimenti non lo è
- i vertici di P sono tutte le soluzioni di base ammissibili

vincoli attivi

- vincoli attivi in x: se x ∈P, sono i vincoli che vengono soddisfatti come uguaglianze
- I(x) = {i | $A_i x = b_i$} insieme degli indici dei vincoli attivi in x
- per ogni J ⊆I(x), l'insieme $P_J$ è una faccia di P e $P_{I(x)}$ è la faccia minimale tra esse

inviluppi convessi

- i poliedri possono anche essere rappresentati per punti, usando l'insieme dei vertici
- inviluppo converso: insieme conv(X) = {x = som (i da 1 a s) λixi | som (i da 1 a s) λi = 1^λi ≥ 0}
  - dato un insieme di punti X = {x1, ..., xs} ⊆R^n
- conv(X) è il più piccolo insieme convesso che contiene tutti i punti di X
- politopo: poliedro limitato i cui vertii sono tutti in X
  - conv(X) è un politopo
  - non tutti i poliedri lo sono (possono essere illimitati)

coni convessi

- cono: insieme C ⊆R^n sse per ogni x ∈C e per ogni α ∈R+ vale αx ∈C
- coni convessi: coni che sono anche insiemi convessi
  - sono caratterizzabili come gli insiemi C | x, y ∈C ^ λ, μ ∈R => λx+μy ∈C
- rappresentazione basata sule direzioni: cono finitamente generato da V
  - cono(V) = {v = som (i tra1 e t) νivi | νi ∈R+}
  - insieme V = {v1, ..., vt}
  - cono(V) è il più piccolo cono convesso che contiene tutti i vettori di V

teorema di motzkin

- P ⊆R^n è un poliedro sse esistono X, V finiti | P = conv(X)+cono(V)
- in questo contesto, P è generato dai punti in X e dalle direzioni in V
- se P è poliedro generato dai punti in X e X è minimale, allora tutti i suoi elementi sono i vertici di P
- analogamente: se P è un poliedro generato dalle direzioni in V e V è minimale, allora i sui elementi (raggi esterni) corrispondono alle direzioni degli spigoli illimitati

due rappresentazioni

- poliedri come intersezioni di semispazi o come somma di un politopo e un cono
- le due rappresentazioni sono equivalenti (grazie al teorema di motzkin) ma non hanno la stessa dimensione

teorema

- sia P = {x | Ax <= b} e siano x1, ..., xs, v1, ..., vt ∈R^n | P = conv({x1, ..., xs})+cono({v1, ..., vt})
- allora il problema max{cs | Ax <= b} ha ottimo finito sse cvj <= 0 per ogni j ∈{1, ..., t}
- in tal caso esiste un k ∈{1, ..., s} | xk è soluzione ottima
- dimostrazione:
  - per il teorema di decomposizione abbiamo che max{cx | Ax <= b} è equivalente al problema sulle variabili λ1, ..., λs e ν1, ..., νt
  - max(som (i tra 1 e s) λi(cxi)) + som (j tra 1 e t) νj(cvj)
  - som (i tra 1 e s) λi = 1; λi ≥ 0; νj ≥ 0
  - questo problema sarebbe finito sse cvj <= 0 per ogni j ∈{1, ..., t}, infatti:

        - 1. se fosse $cv_j > 0$ per qualche $j \in \{1, ..., t\}$, allora si potrebbe far crescere a piacimento la funzione obiettivo pompando $v_j$
        - 2. supponiamo $cv_j <= 0$ per ogni $j \in \{1, ..., t\}$ e prendiamo $y \in P$
        - se $\lambda_i$ e $v_j$ sono i corrispondenti coefficienti del teorema di decomposizione
        - $cy$ = som (i tra 1 e s) $\lambda_i(cx_i)$ + som (j tra 1 e t) $v_j(cv_j) <=$ som (i tra 1 e s) $\lambda_i(cx_i) <=$ som (i tra 1 e s) $\lambda_i(cx_k) = cx_k$
        - $x_k$ = vettore | $x_k = \max\{cx_i \mid i = 1, ..., s\}$ = soluzione ottima finita

massimizzare il rango

- spesso assumiamo che in $\max\{cx \mid Ax <= b\}$ il rango di $A \in R^{m \times n}$ sia uguale a n (cioè è massimo)
- se la matrice A ha rango inferiore di n:
    - $m < n$: ci sono pochi vincoli
    - $m \geq n$ due o più righe di A sono linearmente indipendenti
- teorema: se il rango di $A \in R^{m \times n}$ è strettamente inferiore a n, allora $\max\{cx \mid Ax <= b\}$ può essere ricondotto a un diverso problema di PL la cui matrice dei coefficienti si ottiene eliminando una colonna di A (elimino quindi una variabile)
    - dimostrazione: se il rango di A è inferiore a n, esiste almeno una colonna di A che è combinazione lineare delle altre
        - esempio: l'ultima $A = (A', a_n)$; $c = (c', c_n)$; $x = (x', x_n)$; $A'\mu = a_n$ con $\mu \in R^{n-1}$
        - studiamo il nostro problema tramite $\max\{c'x' \mid A'x' <= b\}$ (problema ridotto), per poi tradurre i risultati sul problema iniziale
        - si verifica che
            - $x = (x', x_n)$ è ammissibile per il problema di partenza sse $x'+\mu x_n$ è ammissibile per il problema ridotto
            - se il problema ridotto è superiormente limitato, anche il vecchio lo è
            - se $x'$ è soluzione ottima per il problema ridotto, allora ogni vettore $x(\alpha) = (x'-\alpha\mu, \alpha)$ è soluzione ottima per il problema di partenza

dualità

teoria

- si basa sulla definizione di un'involuzione (funzione inversa) che mappa ogni problema PL nel suo duale

primale e duale

- coppie asimmetriche:
    - primale: $\max\{cx \mid Ax <= b\}$
    - duale: $\min\{yb \mid (yA = c) \wedge (y \geq 0)\}$
- coppie simmetriche:
    - primale: $\max\{cx \mid (Ax <= b) \wedge (x \geq 0)\}$
    - duale: $\min\{yb \mid (yA \geq c) \wedge (y \geq 0)\}$
- dimostrazione (il duale del duale è il primale): nel caso di coppia simmetrica
    - esprimiamo il duale come $-\max\{y(-b) \mid (yA \geq c) \wedge (y \geq 0)\} = -\max\{(-b^T)y \mid ((-A^T)y <= -c) \wedge (y <= 0)\}$
    - il cui duale è $-\min\{-cx \mid ((x(-A^T) \geq (-b)) \wedge (x \geq 0)\} = \max\{cx \mid (Ax <= b) \wedge (x \geq 0)\}$

teorema debole di dualità

- se $\bar{x}$ e $\bar{y}$ sono soluzioni ammissibili per il prima e il duale, rispettivamente, allora $c\bar{x} <= \bar{y}b$
- dimostrazione:
    - coppia asimmetrica: $A\bar{x} <= b$; $\bar{y}A = c$, $\bar{y} \geq 0 \Rightarrow \bar{y}A\bar{x} <= \bar{y}b$; $\bar{y}A\bar{x} = c\bar{x} \Rightarrow c\bar{x} <= \bar{y}b$
    - coppia simmetrica: $A\bar{x} <= b$, $\bar{x} \geq 0$; $\bar{y}A = c$, $\bar{y} \geq 0 \Rightarrow \bar{y}A\bar{x} <= \bar{y}b$; $\bar{y}A\bar{x} \geq c\bar{x} \Rightarrow c\bar{x} <= \bar{y}b$
- corollario: se il primale è illimitato, allora il duale è vuoto
    - dimostrazione:

        - se il primale è illimitato, allora per ogni M app R esiste una soluzione ammissibile x per il primale con cx > M
        - se per assurdo ci fosse y ammissibile per il duale, troveremmo x ammissibile per il primale con cx > yb (in contrasto col teorema deble di dualità)
- corollario: se $\bar{x}$ e $\bar{y}$ sono soluzioni ammissibili per primale e duale, rispettivamente, e $c\bar{x} = \bar{y}b$, allora $\bar{x}$ e $\bar{y}$ sono soluzioni ottime
    - dimostrazione: se $c\bar{x} = \bar{y}b$ non fosse ottima, troveremmo z ammissibile per il primale con $cx > c\bar{x}$ e quindi $cz > \bar{y}b$ (in contrasto col teorema debole di dualità)

direzioni ammissibili
- direzione ammissibile: vettore $\xi \in R^n$ se esiste $\bar{\lambda} > 0 \mid x(\lambda) = \bar{x}+\lambda\xi$ è ammissibile nel primale per ogni $\lambda \in [0, \bar{\lambda}]$
- ci chiediamo se data una coppia asimmetrica, consideriamo una soluzione ammissibile $\bar{x}$ per il primale, se ci spostiamo lungo una direzione dell'iperspazio a partire da $\bar{x}$ si resta o no nella regione ammissibile
- lemma: il vettore $\xi$ è direzione ammissibile per $\bar{x}$ sse $A_{I(\bar{x})}\xi <= 0$
    - dimostrazione: $\xi$ come direzione ammissibile se per ogni $i \in \{1, ..., m\}$ $A_i\bar{x}(\lambda) = A_ix+\lambda A_i\xi <= b_i$
        - se $i \in I(\bar{x})$, allora $A_i\bar{x} = b_i$, quindi l'equazione è verificata sse $\lambda A_i\xi \leq 0$
        - se $i \notin I(\bar{x})$, allora l'equazione è verificata da qualunque $\xi$, purchè $\lambda$ sia abbastanza piccolo

direzioni di crescita
- direzione di crescita per $\bar{x}$: direzione $\xi \in R^N$ se uno spostamento $\lambda$ lungo $\xi$ fa crescere il valore della funzione obiettivo
    - cioè se $c\bar{x}(\lambda) = c\bar{x}+\lambda\, c\xi > c\bar{x} \Leftrightarrow c\xi > 0$
- la nozione di direzione di crescita non dipende dal punto x
- se c = 0, allora la funzione obiettivo vale sempre 0 e quindi tutte le soluzioni ammissibili sono ottime
- se $c \neq 0$, allora se esiste una direzione ammissibile per $\bar{x}$ che sia anche di crescita, allora $\bar{x}$ non può essere ottimo

algoritmo del simplesso

struttura
- l'algoritmo procede iteritivamente, visitando alcuni tra i vertici del poliedro che definisce l'insieme delle soluzioni ammissibili
- dato un vertice x si cerca di determinare se tale vertice sia o no una soluzione ottima, cercando di determinare se esiste una soluzione $\bar{y}$ per il duale con lo stesso valore della funzione obiettivo
- nel caso x non sia ottima, si cerca di seguire una direzione di crescita ammissibile con un altro vertice
- se non si può spostarsi indefinitamente lungo questa direzione di crescita, allora il problema è illimitato
    - altrimenti si incontra un altro vertice e ci si sposta
- simplesso primale(A, b, c, B)
    - 1. $N \leftarrow \{1, ..., m\}$-B
    - 2. $x \leftarrow A_B$^-1bB
    - 3. $yB \leftarrow$ cAB^-1
    - 4. $yN \leftarrow 0$
    - 5. se $yB \geq 0$, allora termina con successo e restituisci x e y
    - 6. $h \leftarrow \min\{i \in B \mid \bar{y}_i < 0\}$
    - 7. sia $\xi$ la colonna di indice h in -(AB^-1)

- 8. se $A_N\xi \le 0$, allora termina e restituisci $\xi$ (il problema è illimitato)
- 9. $k \leftarrow \arg\min\{(b_i - A_i x)/A_i\xi \mid A_i\xi > 0 \wedge i \in N\}$
- 10. $B \leftarrow B \cup \{k\} - \{h\}$
- 11. torna a 1

correttezza

- l'algoritmo lavora con tre invarianti
  - B = base ammissibile
  - $\bar{x}$ = soluzione ammissibile per il problema primale $\max\{cx \mid Ax \le b\}$ mentra $\bar{y}A = c$
  - quindi, $\bar{x}$ è sempre vertice
  - per y la condizione yA = c vale per come yB e yN vengono inizializzati
  - quindi, $\bar{y}$ è soluzione per il duale $\max\{yb \mid yA = c, y \ge 0\}$ sse $\bar{y}B \ge 0$
- allora l'algoritmo termina correttamente restituendo $\bar{x}$ e $\bar{y}$ soluzioni ottime per primale e duale
- se invece c'è y strettamente negativo, allora non vale l'ottimalità
  - cerchiamo una direzione ammissibile e di crescita per x
    - $\xi$ è sempre direzione di crescita perchè $c\xi = c(-A_B\text{^-}1u_h) = -(cA_B\text{^-}1)u_h = -yu_h = -y_h > 0$
    - con $u_h$ vettore nullo ovunque tranne nella componente corrispondente a i (in cui vale 1)
  - $\xi$ potrebbe non essere direzione ammissibile
- il vettore $A_B\xi$ è una delle colonne della matrice identica cambiata di segno, e quindi $A_B\boldsymbol{\xi} \le 0$
- se $i \in N$ e $A_i\xi \le 0$, allora $\bar{x}(\lambda)$ soddisfa l'i-esimo vincolo per ogni valore non negativo di $\lambda$
- se $i \in N$ e $A_i\xi > 0$, allora $A_i\bar{x}(\lambda) = A_i\bar{x} + \lambda A_i\xi \le b_i \Leftrightarrow \lambda \le (b_i - A_i\bar{x})/A_i\xi$
- scegliamo l'indice i che rende $\lambda$ minimo, lo chiamiamo k
- $\bar{\lambda} = \min\{\lambda_i \mid i \in N\}$
- 1. se $\lambda = +\infty$ (cioè se $A_N\xi < 0$), allora il problema è illimitato
- 2. se 0 < delta < $+\infty$, allora x(delta) è ammissibile per ogni delta $\in$[0, delta] e non ammissibile in altri casi
  - quindi, possiamo spostarci da B a B un {k}-{h} (che corrisponde a un altro vertice)
- se delta = 0, allora la direzione non è ammissibile, ma possiamo effettuare un cambio di base verso B un {k}-{h} (rimaniamo sullo stesso vertice)

complessità

- si può dimostrare che ogni base viene trattata al massimo una volta durante l'esecuzione dell'algoritmo
- quindi, vi saranno al massimo un numero di iterazioni che può diventare esponenziale in n
- quindi
  - dal punto di vista teorico, la complessità nel caso medio è polinomiale
  - dal punto di vista pratico, si osserva che il simplesso è l'algoritmo più efficiente e che si comporta meglio di altri