Capitolo 1
Introduzione

1.1. Ottimalità

Spesso avviene che, dato un insieme di possibili decisioni alternative, si debba selezionarne una in particolare, di solito la 'migliore' secondo uno specificato criterio. Problemi con queste caratteristiche possono essere originati direttamente in termini matematici (ad esempio: fra tutti i rettangoli di uguale perimetro quale ha l'area più grande?), oppure possono provenire da problemi reali (ad esempio: come organizzare la produzione per ottenere il massimo profitto?). In questa sede non ci occuperemo dell'importante problema di come formalizzare un problema reale di decisione secondo un modello matematico significativo. Daremo per scontato che tale fase sia già avvenuta. L'interesse verterà unicamente su come trovare, all'interno del modello matematico formalizzato, la decisione ottima fra tutte quelle possibili.

La Ricerca Operativa è la disciplina che si occupa in generale di come tradurre un problema reale di decisione in un modello quantitativo, ricavare una 'soluzione' dal modello e infine validare la soluzione ottenuta nel contesto reale. L'Ottimizzazione è la disciplina che si occupa in particolare della seconda fase, usando soprattutto strumenti matematici ed informatici. Il tema centrale di questo testo è pertanto l'Ottimizzazione, in un contesto matematicamente formalizzato.

In termini molto generali ed astratti un problema di ottimizzazione viene definito specificando:
– un insieme E (insieme *ambiente*) i cui elementi possono venir chiamati *soluzioni*, *decisioni*, oppure *alternative*;
– un sottoinsieme $F \subset E$ (insieme *ammissibile*) i cui elementi costituiscono le soluzioni (decisioni, alternative) ammissibili. Per contro gli elementi in $E \backslash F$ sono indicati come non ammissibili. La relazione $x \in F$ viene detta *vincolo*.
– una funzione $f: E \to R$ (funzione *obiettivo*) ed un indicatore di 'min' (minimo) o 'max' (massimo) a seconda se si vuole trovare un elemento ammissibile che minimizzi o massimizzi la funzione.

Con questa struttura minima si può già definire l'elemento la cui ricerca rappresenta lo scopo di un problema di ottimizzazione.

1.1 Definizione. *Ogni elemento $x \in F$ tale che $f(x) \leq f(y)$, $\forall y \in F$, se il problema è di tipo 'min', oppure $f(x) \geq f(y)$, $\forall y \in F$, se il problema è di tipo 'max', prende il nome di* ottimo. *Il valore $v = f(x)$ della funzione nell'ottimo prende il nome di* valore ottimo. ∎

Un problema di minimo può essere banalmente ricondotto ad un problema di massimo (e viceversa) sostituendo f con $-f$. Quanto più possibile si tenderà in questo testo ad avere una

trattazione uniforme per cui si farà riferimento quasi sempre a problemi di minimo. Ogni tanto si introdurranno esplicitamente problemi di massimo quando questa è la formulazione più naturale del problema. Comunque, se non altrimenti indicato, l'ottimo deve essere inteso come un minimo.

Se l'ottimo non esiste, il valore ottimo viene comunque definito come $v = \inf\{f(x): x \in F\}$ (sup per problemi di massimo), ponendo per convenzione $v = +\infty$ $(-\infty)$ se $F = \emptyset$. Se $F = \emptyset$ il problema viene detto *non ammissibile*, mentre se $v = -\infty$ $(+\infty)$ viene detto *illimitato*.

La convenzione di definire un valore ottimo infinito per un problema non ammissibile deriva dall'osservazione che il valore ottimo normalmente aumenta (più esattamente è non decrescente) se si restringe l'insieme ammissibile, e quindi, restringendo l'insieme ammissibile fino a non avere più elementi, si estrapola postulando l'aumento più elevato possibile, cioè infinito. In molti casi questa convenzione è consistente con le relazioni che si ottengono nel caso di problemi ammissibili, ma non sempre. Si consideri la seguente relazione $\min_{x\in F} f(x) \leq \max_{x\in F} f(x)$, che è certamente vera se F non è vuoto, ma non si può estendere al caso di F vuoto adottando la convenzione, perché si avrebbe $+\infty \leq -\infty$!

È prassi comune usare la seguente notazione:

$$v = \min_{x \in F} f(x)$$

(max se si deve massimizzare) dando per scontato che il minimo esiste e che si tratta in realtà di un'operazione di inf in caso contrario. In qualche caso si userà, per maggior esattezza, inf al posto di min (sup al posto di max).

Scopo di un problema di ottimizzazione è quello di determinare almeno un ottimo, se ne esistono, oppure di stabilire che non esistono ottimi e di determinare il valore ottimo. Tipicamente questo scopo viene raggiunto per via algoritmica, ragion per cui lo studio dei problemi di ottimizzazione, cioè l'Ottimizzazione come disciplina, consiste essenzialmente nel progetto di algoritmi, il più possibile efficienti, basati sulla caratterizzazione matematica di F e f.

I problemi in cui F è un insieme finito vengono di solito indicati come problemi *discreti*. Per contro vengono indicati come problemi *continui* quelli in cui F possiede certe caratteristiche di continuità (ad esempio è un sottoinsieme infinito connesso di uno spazio lineare). Spesso sono presenti ambedue le caratteristiche, e si parla di problemi *misti* (ad esempio l'insieme ammissibile è l'unione di poliedri disgiunti). In questi casi l'aspetto che condiziona maggiormente la risoluzione è quello discreto.

La definizione di ottimalità data è quella che risponde meglio alle esigenze pratiche. Tuttavia in molti casi è utile, soprattutto ai fini algoritmici, poter disporre di definizioni più deboli di ottimalità, quale ad esempio:

1.2 Definizione. *Sia F provvisto di una topologia. Se esiste un intorno $N \subset F$ di $x \in F$ tale che $f(x) \leq f(y)$, $\forall y \in N$, allora x prende il nome di* ottimo locale. ∎

In questa sede non saranno mai considerati problemi in cui F sia un sottoinsieme di uno spazio lineare ad infinite dimensioni, ragion per cui la scelta di una topologia opportuna per F non presenta particolari problemi. Tipicamente, se $F \subset R^n$, la sua topologia sarà quella relativa indotta da una norma. Tuttavia, se F è finito (pensiamolo senza perdita di generalità immerso in R^n), la topologia che si ottiene in questo modo è quella discreta, e la definizione di ottimalità locale si svuota di contenuto, in quanto ogni $x \in F$ diventa banalmente ottimo locale, essendo $\{x\}$ intorno di x per ogni x. Per problemi discreti è più conveniente introdurre la seguente definizione:

1.3 Definizione. *Sia definita una mappa $x \mapsto N_x \subset F$. Se $f(x) \leq f(y)$, $\forall y \in N_x$, allora x prende il nome di* ottimo locale discreto. ∎

Se risulta chiaro dal contesto di quale ottimo si tratta, i termini 'discreto' oppure 'locale' possono essere omessi. Altre volte invece, per sottolineare che si tratta di un ottimo nel senso della definizione più generale, si userà il termine di *ottimo globale*.

Ovviamente un ottimo globale è anche ottimo locale (discreto o non). Qualora gli ottimi globali coincidano con gli ottimi locali discreti, si dice che gli intorni N_x sono *esatti*.

Non necessariamente l'ottimo è unico (unico è ovviamente il valore ottimo globale). Gli ottimi vengono detti *stretti* se nelle definizioni precedenti si sostituisce $f(x) \leq f(y)$, $\forall y$, con $f(x) < f(y)$, $\forall y \neq x$. Quindi un ottimo globale stretto è unico. È utile sottolineare che, in presenza di non unicità dell'ottimo globale, interessa soltanto determinare uno qualsiasi degli ottimi globali, e non già tutti quanti, richiesta che renderebbe molto più difficile la risoluzione di un problema di ottimizzazione.

1.2. Esempi

Come già detto, un particolare problema di ottimizzazione viene definito specificando le proprietà dell'insieme ammissibile e della funzione obiettivo. L'elenco di problemi diversi formalizzati e studiati in letteratura è praticamente inesauribile. Tuttavia alcuni di questi sono presenti con maggior frequenza di altri e quindi il loro studio è particolarmente importante. In questa sezione ne vengono illustrati alcuni con l'intenzione di presentare la materia di cui ci si occuperà in seguito e, intanto, di familiarizzare il lettore con i concetti e le tecniche, e in generale con il tipo di mentalità dell'Ottimizzazione.

Gli esempi riportati sono volutamente molto semplici. Tuttavia presentano le stesse caratteristiche strutturali di analoghi problemi con un numero ben superiore di dati. Gli esercizi proposti servono a consolidare l'acquisizione dei nuovi concetti. È consigliabile quindi provare a risolverli tutti anche se alcuni non sono del tutto facili.

1.4 Esempio.

$$F := [0,1]^2 \subset R^2 \qquad\qquad f(x) := x_1 + x_2$$

L'insieme ammissibile è un sottoinsieme del piano euclideo. Si tratta di un quadrato ed ha quindi la proprietà di essere convesso, chiuso, limitato con una frontiera costituita da segmenti e vertici. La funzione obiettivo è data dalla semplice somma delle variabili. È utile raffigurare i luoghi del piano che soddisfano le equazioni $f(x) = K$ per vari valori della costante K. Per ogni valore di K tali insiemi rappresentano punti dove la funzione obiettivo ha valore costante ed uguale a K e vengono chiamati *linee di livello*. In generale, quando esistono ottimi, il valore ottimo v ha la proprietà che $\{x : f(x) = v\} \cap F \neq \emptyset$ e per ogni $K < v$ si ha $\{x : f(x) = K\} \cap F = \emptyset$. Quindi si ottiene un aiuto visivo alla determinazione di v trovando l''ultima' linea di livello che abbia ancora intersezione non vuota con l'insieme ammissibile. Nell'esempio le linee di livello sono il fascio di rette parallele $x_1 + x_2 = K$.

In figura 1.1 sono raffigurati l'insieme ammissibile e le linee di livello. Come si vede, il punto $(0,0)$ è l'ottimo. Si noti che risulta essere un vertice di F. Il fatto che in questo esempio l'ottimo sia stato trovato per via grafica non deve far pensare che in generale sia questa la metodologia per trovare gli ottimi. La complessità naturale di quasi tutti i problemi d'interesse pratico rende impossibile qualsiasi approccio di tipo grafico. Rimane tuttavia utile poter visualizzare la struttura di un problema per piccole istanze, perché la comprensione che ne deriva permette di progettare algoritmi per istanze più complesse.

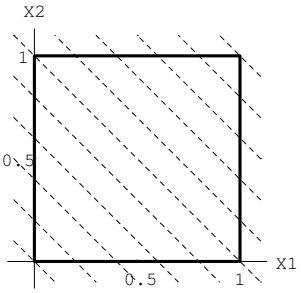

FIGURA 1.1

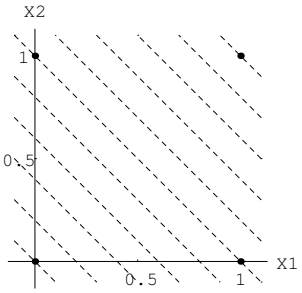

FIGURA 1.2

I problemi in cui, come nell'esempio, f è lineare e F viene definito da diseguaglianze lineari e, eventualmente, equazioni lineari prendono il nome di problemi di *Programmazione lineare* e costituiscono il nucleo centrale dell'Ottimizzazione. ∎

1.5 ESERCIZIO. Si trovi l'ottimo per le seguenti funzioni obiettivo (F come nell'esempio 1.4): $f(x) := 2\,x_1 - x_2$, e $f(x) := -x_1 + 3\,x_2$. ∎

1.6 ESERCIZIO. Si calcoli l'ottimo (non per via grafica naturalmente) per $F := [0,1]^6 \subset R^6$ e $f(x) := x_1 - 2\,x_2 - 5\,x_3 + x_4 + 2\,x_5 - x_6$. ∎

1.7 ESEMPIO.

$$F := \{0,1\}^2 \subset R^2 \qquad\qquad f(x) := x_1 + x_2$$

Rispetto all'esempio precedente l'insieme ammissibile è stato ridotto ai soli quattro vertici del quadrato (si veda la figura 1.2). Questa drastica semplificazione sembrerebbe aver reso il problema molto banale. Essendo l'insieme ammissibile costituito da un numero finito di elementi, apparentemente basta valutare la funzione ammissibile su ognuno di essi e quindi determinare l'ottimo. È certamente questo il caso nell'esempio in questione dove la cosa più semplice da fare è calcolare la funzione obiettivo nei quattro estremi del quadrato. Però se l'insieme ammissibile fosse $[0,1]^{10}$ il calcolo esplicito richiederebbe $2^{10} = 1024$ valutazioni, cosa certamente alla portata di un qualsiasi calcolatore, anche se non più eseguibile a mano. Se poi l'insieme fosse $[0,1]^{100}$ (cosa tutt'altro che infrequente in molte applicazioni) si dovrebbero eseguire $2^{100} \approx 10^{30}$ valutazioni, che, anche alla velocità, oggi non ancora ottenibile, di 10^{10} valutazioni al secondo, richiederebbe 10^{20} secondi, che è uguale a tremila miliardi di anni. Quindi un approccio che richieda la valutazione esplicita dell'insieme ammissibile in generale non si può fare nemmeno quando questo sia finito. Per rafforzare ulteriormente questo concetto si tenga presente inoltre che nessuna tecnologia futura potrà mai eseguire calcoli in tempi arbitrariamente piccoli. Il principio di indeterminazione di Heisenberg afferma che $\Delta t \cdot \Delta E \geq 4.13 \cdot 10^{-15}$ eV$\cdot$sec, dove Δt è l'incertezza nella valutazione del tempo e ΔE è l'incertezza nella valutazione dell'energia. Quindi per scendere ad esempio al di sotto di 10^{-20} secondi bisognerebbe utilizzare energie elevatissime.

La cosa da notare in questo esempio è che l'ottimo è il medesimo dell'esempio precedente. Il motivo è chiaro: l'ottimo stava sui vertici del quadrato e, avendo ora isolato i vertici, l'ottimo è il medesimo. La coincidenza degli ottimi è una conseguenza della linearità di f e del fatto che $[0,1]^2$ è una combinazione convessa di $\{0,1\}^2$. Questa osservazione, in apparenza banale, guida il progetto di algoritmi molto complessi.

I problemi in cui, come nell'esempio, f è lineare e F viene definito da diseguaglianze lineari (con eventualmente, equazioni lineari) e in più si restringono le soluzioni ammissibili

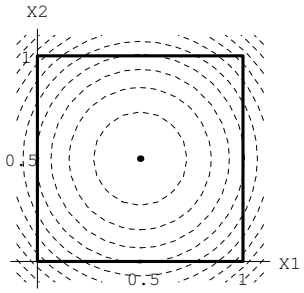

FIGURA 1.3

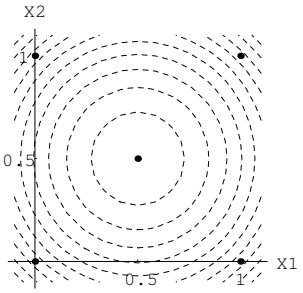

FIGURA 1.4

ai valori interi, prendono il nome di problemi di *Programmazione lineare intera* e sono fra i più usati nelle applicazioni, anche se la loro risoluzione può a volte presentare difficoltà insormontabili. ∎

1.8 ESERCIZIO. Si trovi l'ottimo per le seguenti funzioni obiettivo (F come nell'esempio 1.7): $f(x) := -2\,x_1 - 2\,x_2$, e $f(x) := x_1 + 3\,x_2$. ∎

1.9 ESERCIZIO. Si indichi come calcolare l'ottimo quando si abbia $F := \{0,1\}^n \subset R^n$ e $f(x) := \sum_{i=1}^{n} a_i\,x_i$. ∎

1.10 ESEMPIO.

$$F := [0,1]^2 \subset R^2 \qquad f(x) := (x_1 - \frac{1}{2})^2 + (x_2 - \frac{1}{2})^2$$

Rispetto all'esempio 1.4 la funzione obiettivo è ora quadratica. Le linee di livello sono circonferenze concentriche di centro $(\frac{1}{2}, \frac{1}{2})$ (si veda la figura 1.3). La funzione obiettivo è non negativa e vale zero nel centro. Il centro è ammissibile, quindi è ottimo. Si noti che, a differenza dell'esempio 1.4, l'ottimo è un punto interno di F. La perdita della linearità ha causato la perdita della proprietà (almeno in generale) di avere l'ottimo su un vertice. L'esempio costituisce un caso particolare di *Programmazione non lineare.* ∎

1.11 ESERCIZIO. Si trovi l'ottimo per le seguenti funzioni obiettivo (F come nell'esempio 1.10):

$$f(x) := (x_1 - \frac{3}{2})^2 + (x_2 - \frac{1}{2})^2$$

$$f(x) := 17\,x_1^2 - 2\,x_1\,x_2 + x_2^2 - 24\,x_1 + 9$$

$$f(x) := (x_1 - \frac{1}{2})\,(x_2 - \frac{1}{2})$$

∎

1.12 ESEMPIO.

$$F := \{0,1\}^2 \subset R^2 \qquad f(x) := (x_1 - \frac{1}{2})^2 + (x_2 - \frac{1}{2})^2$$

Come precedentemente l'insieme ammissibile è stato ristretto ai soli vertici del quadrato e quindi l'ottimo dell'esempio 1.10 non è più ammissibile e non può essere ottimo. In particolare si vede (figura 1.4) che tutti gli elementi di F sono ottimi. ∎

1.13 Esercizio. Si calcolino gli ottimi per le funzioni obiettivo assegnate nell'esercizio 1.11 (F come nell'esempio 1.12). ∎

1.14 Esempio.

$$F := [0,1]^2 \cap \{x\colon 2\,x_1 + 2\,x_2 \geq 1\} \qquad f(x) := x_1 + x_2$$

Rispetto all'esempio 1.4 l'insieme ammissibile è stato ristretto imponendo un vincolo definito da una disequazione lineare. Come si vede dalla figura 1.5 è stato tagliato via il vertice $(0,0)$ che era ottimo nell'esempio 1.4. Si noti che i coefficienti della diseguaglianza del vincolo sono proporzionali a quelli della funzione obiettivo. Questo implica che l''ultima' linea di livello interseca F su tutto un segmento e quindi gli ottimi sono tutti i punti dell'insieme $[0,1]^2 \cap \{x\colon 2\,x_1 + 2\,x_2 = 1\}$. ∎

1.15 Esercizio. Si calcolino gli ottimi per le funzioni obiettivo (F come nell'esempio 1.14)

$$f(x) := 3\,x_1 + 2\,x_2 \qquad \text{e} \qquad f(x) := 2\,x_1 + 3\,x_2$$

∎

1.16 Esercizio. Si esprima l'insieme F dell'esempio 1.14 usando soltanto diseguaglianze lineari. Si esprima l'insieme F di figura 1.6 usando soltanto diseguaglianze lineari. ∎

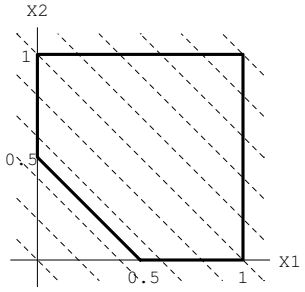

Figura 1.5

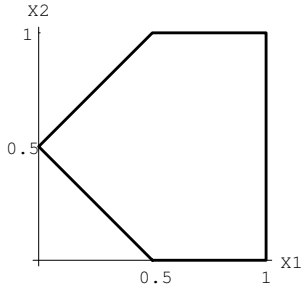

Figura 1.6

1.17 Esercizio. Si rifaccia l'esercizio 1.11 con insieme ammissibile $F := \{0,1\}^2$, notando che in questo caso $x_i^2 = x_i$ e che il prodotto $x_1\,x_2$ può essere sostituito da una nuova variabile $x_3 = x_1\,x_2$ vincolata da $x_3 \leq x_1$, $x_3 \leq x_2$ e $x_1 + x_2 \leq x_3 + 1$. A questo punto si minimizza la nuova funzione obiettivo lineare nelle variabili x_1, x_2 e x_3 e con i vincoli $0 \leq x_i$, $x_3 \leq x_1$, $x_3 \leq x_2$ e $x_1 + x_2 \leq x_3 + 1$ (i vincoli $x_i \leq 1$ sono ridondanti). I vertici dell'insieme ammissibile generato da questi vincoli (si veda in figura 1.7 tale insieme) hanno coordinate intere e quindi la minimizzazione della funzione obiettivo lineare genera lo stesso ottimo sia in $\{0,1\}^3$ come in $[0,1]^3$. ∎

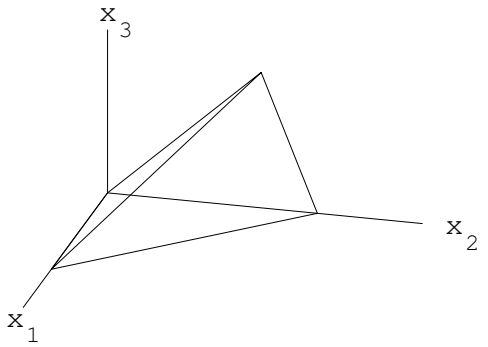

FIGURA 1.7

1.18 ESEMPIO.

$$F := \{0,1\}^2 \cap \{x\colon 2\,x_1 + 2\,x_2 \geq 1\} \qquad f(x) := x_1 + x_2$$

Come già osservato, il vincolo aggiunto ha tagliato il vertice e cioè uno dei quattro elementi dell'insieme ammissibile dell'esempio 1.7. Ora vi sono due ottimi dati da $(1,0)$ e $(0,1)$. Se confrontiamo gli esempi 1.4 e 1.7 con gli esempi 1.14 e 1.18 possiamo notare che ora l'inviluppo convesso di $F_1 := \{0,1\}^2 \cap \{x\colon 2\,x_1 + 2\,x_2 \geq 1\}$ (si veda la figura 1.8) è strettamente contenuto in $[0,1]^2 \cap \{x\colon 2\,x_1 + 2\,x_2 \geq 1\}$ (figura 1.5). Per ottenere l'uguaglianza bisognerebbe ridefinire F_1 (senza perdita di soluzioni ammissibili) come $F_1' := \{0,1\}^2 \cap \{x\colon x_1 + x_2 \geq 1\}$. A questo punto è equivalente minimizzare una funzione lineare F_1' oppure su $[0,1]^2 \cap \{x\colon x_1 + x_2 \geq 1\}$. ∎

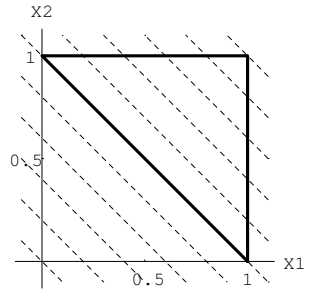

FIGURA 1.8

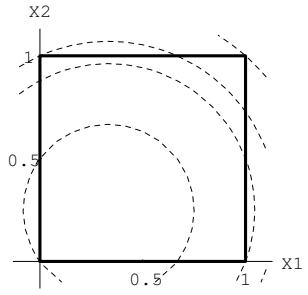

FIGURA 1.9

1.19 ESEMPIO.

$$F = [0,1]^2 \subset R^2 \qquad f(x) = -(x_1 - \frac{1}{3})^2 - (x_2 - \frac{1}{4})^2$$

L'esempio assomiglia all'esempio 1.10. L'unica differenza rilevante è il segno negativo di fronte all'espressione quadratica della funzione obiettivo. Si tratta quindi di massimizzare un'espressione quadratica anziché minimizzarla come nell'esempio 1.10, e non è una differenza da poco. Le linee di livello sono ancora circonferenze concentriche (si veda la figura 1.9) ma la funzione obiettivo migliora in direzione centrifuga e quindi l'ottimo viene a trovarsi sul vertice più distante dal centro, cioè $(1,1)$.

Se ora si considerano gli altri vertici, si può notare che per ogni vertice c'è un intorno costituito da elementi peggiori del vertice. Quindi tutti i vertici di F sono ottimi locali. Questa è un'altra differenza rilevante rispetto all'esempio 1.10. In generale si tenga presente che minimizzare una funzione convessa (come nell'esempio 1.10) è molto più facile che massimizzarla (come in questo esempio). ∎

1.20 Esercizio. In alcuni casi particolari massimizzare una funzione convessa può essere facile. Si indichi una procedura per massimizzare $\sum_{i=1}^{n}(x_i - a_i)^2$ su $F := \{0,1\}^n$. ∎

1.21 Esercizio. Si indichi una procedura per massimizzare $x\,Q\,x + c\,x$ su $F := \{0,1\}^n$. Anche questo caso è facilmente risolvibile. Si chiede di riformulare il problema come un problema di massimo lineare su un insieme di diseguaglianze lineari (si riveda l'esercizio 1.17). ∎

1.22 Esempio.

$$F = \{x\colon 2x_1 + x_2 = 1\} \qquad f(x) = x_1 + x_2$$

Non esistono ottimi! L'insieme ammissibile è illimitato e la funzione obiettivo vi può assumere valori negativi arbitrariamente grandi. Il problema è quindi illimitato. ∎

1.23 Esercizio. Si dimostri che è illimitato anche il problema

$$F = \{z \in Z^2\colon 2z_1 + z_2 = 1\} \qquad f(z) = z_1 + z_2$$

∎

1.24 Esempio.

$$F = (0,1)^2 \subset R^2 \qquad f(x) = x_1 + x_2$$

Non esistono ottimi neppure in questo caso. Il motivo è dovuto alla linearità della funzione obiettivo e alla non chiusura dell'insieme ammissibile. Si ha quindi $v = 0$. Va detto comunque che l'esempio è un po' artificioso, perché in tutti i problemi d'interesse pratico la frontiera dell'insieme ammissibile è anch'essa ammissibile. ∎

1.25 Esempio.

$$F = \{x\colon x_1 \geq 0,\ x_2 \geq 0,\ x_1 + x_2 \leq 4,\ x_2 \leq 1,\ x_1 + 3x_2 \geq 7\}$$

$$f(x) = x_1 + x_2$$

Non esistono ottimi perché $F = \emptyset$, cioè il problema è non ammissibile. ∎

1.26 Esercizio. Si dimostri che non possono esistere quattro diseguaglianze lineari in R^2 con le seguenti proprietà: a) l'insieme definito dalle quattro diseguaglianze è vuoto; b) gli insiemi definiti da tre delle quattro diseguaglianze sono ammissibili per qualsiasi terna. ∎

In tutti gli esempi visti non si è mai fatto uso di una tecnica che viene spesso invocata nel momento in cui si ricercano minimi o massimi, e cioè il calcolo di punti che annullano la derivata della funzione obiettivo. A questo riguardo bisogna rilevare che: a) ovviamente le funzioni devono essere derivabili e questo non è sempre vero; b) la derivata fa uso unicamente di informazione locale e quindi solo ottimi locali possono essere individuati in questo modo; c) la proprietà che la derivata si annulla in un minimo o un massimo è valida se il minimo o il massimo sono punti interni dell'insieme ammissibile.

Per ciò che riguarda a) e c) si possono estendere in modo non banale i risultati noti, mentre non si può far nulla riguardo a b). Ovviamente le derivate sono inapplicabili per problemi discreti. Gli esempi seguenti illustreranno questi problemi.

1.27 Esempio.

$$F := R_+ \qquad f(x) = |x|$$

Il minimo è ovviamente l'origine ed è anche l'unico punto in cui la funzione obiettivo non sia derivabile. Si possono però considerare separatamente la derivata destra e quella sinistra e, notando che la prima è positiva e la seconda negativa, concludere che l'origine deve essere punto di minimo. ∎

1.28 Esercizio. Si trovino due funzioni f_1 e f_2 in modo da avere $|x| = \max\{f_1(x)\ ;\ f_2(x)\}$. ∎

1.29 Esercizio. Si calcoli il minimo di $f(x) := \max\{1-x\ ;\ x-2\ ;\ 2\,x-6\}$, $F := R$. ∎

1.30 Esercizio. Si determini una famiglia infinita di funzioni f_α, in modo da ottenere $x^2 = \max_\alpha \{f_\alpha(x)\}$. ∎

1.31 Esempio.

$$F := R_+ \qquad f(x) = 2x^3 - 9x^2 + 12x$$

Calcolando le derivate si trova subito che $f'(1) = f'(2) = 0$ e $f''(2) > 0$ (si veda in figura 1.10 il grafico della funzione). Tuttavia 2 è soltanto ottimo locale. L'ottimo globale è 0. Il fatto che l'ottimo globale non si trovi mediante la regola di annullare la derivata è dovuto al fatto che l'ottimo globale si trova sulla frontiera dell'insieme ammissibile. Tuttavia si può notare che se la derivata destra nell'estremo sinistro dell'intervallo di ammissibilità è non negativa, l'estremo deve essere minimo locale. Allora $f'(0) = 12 > 0$ e quindi 0 è minimo locale. Per decidere quale sia l'ottimo globale dobbiamo confrontare $f(2)$ con $f(0)$. ∎

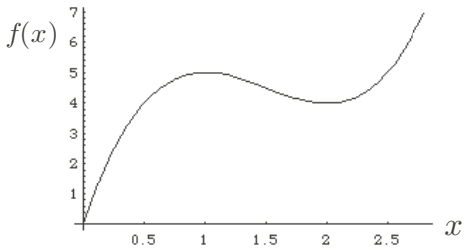

Figura 1.10

1.32 Esercizio. Si calcoli il massimo globale ed i massimi locali di $f(x) := \int_0^x t\,\sin t\,dt$, su $F := [0, 6\,\pi]$. ∎

1.33 Esempio.

$$F = R \qquad f(x) = 2x^3 - 9x^2 + 12x$$

Questa volta non esistono ottimi globali in quanto il problema è illimitato. Rimane tuttavia l'ottimo locale $x = 2$. ∎

Negli esempi precedenti l'insieme ammissibile era un sottoinsieme di uno spazio euclideo e la funzione obiettivo era naturalmente definita sul medesimo spazio. Vi sono molti esempi di problemi di ottimizzazione in cui la formulazione naturale si basa su strutture discrete e fra queste i grafi rivestono un ruolo fondamentale. Qui di seguito forniamo un'introduzione, necessariamente incompleta, a questo tipo di problemi. Per ciò che riguarda concetti e definizioni relativi ai grafi, il lettore può trovarli nel capitolo 2.

I primi tre esempi si riferiscono ad insiemi ammissibili costituiti da permutazioni sulle quali sono definite delle opportune funzioni obiettivo. A seconda del tipo di funzione obiettivo possiamo avere problemi molto diversi fra loro.

1.34 Esempio. Sia $F = E = S_n$ (S_n è l'insieme di tutte le permutazioni di n elementi). Si indichi con $(x_1, \dots, x_n)$ una generica permutazione secondo la notazione usuale, cioè l'elemento i va in posizione x_i (ad esempio sia $n = 3$, $x_1 = 2$, $x_2 = 3$, $x_3 = 1$). Siano dati n numeri reali non negativi $a_1, \dots, a_n$ (sia $a_1 = 5$, $a_2 = 2$, $a_3 = 3$). A questo punto si definisca la seguente funzione obiettivo (da massimizzare) $f(x) = \sum_{i=1}^{n} x_i\, a_i$, che può essere ridefinita come $f(x) = \sum_{i=1}^{n} i\, a_{x'_i}$, dove x' è la permutazione inversa, cioè $x'_{x_i} = i$ (nell'esempio si ottiene $x' = (3, 1, 2)$, $f((2, 3, 1)) = 2 \cdot 5 + 3 \cdot 2 + 1 \cdot 3 = 1 \cdot 3 + 2 \cdot 5 + 3 \cdot 2 = 19$).

Il problema di massimizzare f è equivalente a quello di trovare una permutazione $\hat{x}'$ che ordini i numeri a_i in ordine non decrescente (nell'esempio $\hat{x}' = (2, 3, 1)$ da cui $\hat{x} = (3, 1, 2)$ e $f((3, 1, 2)) = 3 \cdot 5 + 1 \cdot 2 + 2 \cdot 3 = 23$). Si può dimostrare questo fatto usando un'opportuna definizione di intorno discreto. Si definiscano i seguenti intorni discreti

$$N_x = \{y \in S_n \colon y'_i = x'_{i+1},\ y'_{i+1} = x'_i,\ y'_k = x'_k \text{ per un certo } i \text{ e } \forall k \neq i,\ k \neq i+1\}$$

cioè ogni elemento di N_x è ottenuto da x scambiando tra loro due elementi adiacenti nella permutazione. Se x è ottimo locale discreto si ha $f(x) \geq f(y)$, $\forall y \in N_x$, cioè

$$i\, a_{x'_i} + (i+1)\, a_{x'_{i+1}} \geq i\, a_{x'_{i+1}} + (i+1)\, a_{x'_i} \qquad i = 1, \dots, n-1$$

ovvero

$$a_{x'_{i+1}} \geq a_{x'_i} \qquad i = 1, \dots, n-1$$

Questa condizione implica che per ogni ottimo locale tutti gli a_i sono ordinati in modo non decrescente. Se ne deduce che il valore ottimo locale è unico e quindi ogni ottimo locale è anche ottimo globale e gli intorni N_x sono esatti. ∎

1.35 Esercizio. Siano assegnati n numeri $a_1, \dots, a_n$. Possiamo supporre che corrispondano a durate di esecuzione di n lavori diversi. Per ogni permutazione x siano definite le n quantità $C_k(x) := \sum_{i=1}^{k} a_{x'_i}$, $k = 1, \dots, n$, che corrispondono al tempo di completamento del k-mo lavoro nella permutazione x. Si definisca la seguente funzione obiettivo da minimizzare $f(x) := \sum_{k=1}^{n} C_k(x)$. Si dimostri che il problema è equivalente al problema dell'esempio 1.34. ∎

1.36 Esempio. Sia nuovamente $F = E = S_n$ e sia assegnata una matrice $n \times n$ di costi $c(i, j)$. Per ogni permutazione x (notazione come nell'esempio 1.34) sia definita la seguente funzione obiettivo

$$f(x) = \sum_{i=1}^{n} c(i, x_i)$$

Sia ad esempio $n = 4$ e sia assegnata la seguente matrice di costi c:

$$\begin{bmatrix} 0 & 2 & 1 & 3 \\ 2 & 1 & 0 & 1 \\ 2 & 1 & 1 & 0 \\ 0 & 0 & 2 & 1 \end{bmatrix}$$

Si può notare che per ogni permutazione x compaiono nella funzione obiettivo solo n elementi della matrice, e precisamente gli elementi $c(i, x_i)$. Di questi elementi ce n'è esattamente uno per ogni riga ed uno per ogni colonna. Quindi il problema può essere riformulato come quello di scegliere n numeri dalla matrice, uno per ogni riga ed uno per ogni colonna, in modo da minimizzare la loro somma.

Si può verificare che l'ottimo è dato dalla permutazione $x = (1, 3, 4, 2)$ indicata in grassetto nella seguente matrice

$$\begin{bmatrix} \mathbf{0} & 2 & 1 & 3 \\ 2 & 1 & \mathbf{0} & 1 \\ 2 & 1 & 1 & \mathbf{0} \\ 0 & \mathbf{0} & 2 & 1 \end{bmatrix}$$

Per questo tipo di problemi si usa rappresentare la soluzione anche secondo il grafo bipartito indicato in figura 1.11. I nodi di sinistra sono in corrispondenza delle righe della matrice dei costi e quelli di destra sono in corrispondenza delle colonne. Il grafo bipartito è completo e ogni arco corrisponde ad un elemento della matrice. Scegliere n elementi, uno per ogni riga e uno per ogni colonna, corrisponde a scegliere n archi in modo che ogni nodo sia incidente ad esattamente un arco. Questo problema viene chiamato *problema dell'assegnamento* (*Assignment*).

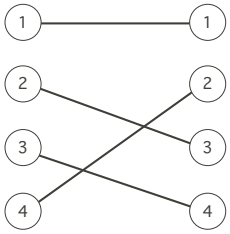

FIGURA 1.11

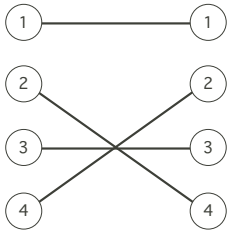

FIGURA 1.12

È prassi costante riformulare un problema discreto di natura combinatoria identificando gli elementi di E con punti di un opportuno spazio euclideo e cercando di definire F con diseguaglianze e/o equazioni lineari. Ad esempio ogni permutazione di S_n è in corrispondenza biunivoca con una matrice $n \times n$ ottenuta permutando in modo corrispondente le colonne di una matrice identica (*matrice di permutazione*). Sia $x_{ij} \in \{0, 1\}$ un elemento generico di tale matrice. Affinché gli elementi x_{ij} siano elementi di una matrice di permutazione devono soddisfare i seguenti vincoli

$$\begin{aligned} \sum_{i=1}^{n} x_{ij} &= 1 && j = 1, \ldots, n \\ \sum_{j=1}^{n} x_{ij} &= 1 && i = 1, \ldots, n \\ x_{ij} &\in \{0, 1\} && \forall i, j \end{aligned}$$

e la funzione obiettivo può essere riformulata come $\sum_{ij} c_{ij}\, x_{ij}$. In questo modo il problema acquista una struttura algebrica e geometrica che facilita lo studio di particolari proprietà del problema e il progetto di algoritmi risolutivi. Si noti inoltre che una matrice di permutazione può essere interpretata come un vettore d'incidenza del sottoinsieme di archi che definisce un assegnamento. ∎

1.37 Esercizio. Data una permutazione $(x_1, \ldots, x_n)$ si definisca il seguente intorno

$$N_x := \{y \in S_n : y_j = x_i,\ y_i = x_j,\ y_k = x_k \text{ per } k \neq i,\ k \neq j,\ \text{ per ogni } i \neq j\}$$

che corrisponde a permutazioni ottenute scambiando i nodi di destra del grafo bipartito per due archi fissati (si veda in figura 1.12 un elemento dell'intorno della permutazione in figura 1.11 ottenuto scambiando gli archi 2-3 e 3-4 con 2-4 e 3-3). Si costruisca un controesempio con $n = 3$ che faccia vedere come tali intorni non siano esatti. Come andrebbe allargato l'intorno affinché sia esatto? (suggerimento: si considerino un minimo locale non gobale e un minimo globale, e si considerino gli archi che appartengono soltanto ad una delle due soluzioni, questi archi hanno una struttura particolare...). ∎

1.38 Esercizio. Si riformuli la funzione obiettivo dell'esempio 1.34 usando le matrici di permutazione x_{ij}. ∎

L'approccio al problema dell'assegnamento può essere generalizzato. Nei problemi combinatori quasi sempre si può identificare un insieme K, che possiamo indicare come *insieme primitivo*, dal quale si costruisce l'insieme ambiente E come l'insieme dei sottoinsiemi di K e l'insieme ammissibile F come una famiglia di sottoinsiemi di 2^K opportunamente definita. Quindi la ricerca dell'ottimo conduce a trovare un sottoinsieme e risulta pertanto naturale identificare $E = 2^K$ con $\{0,1\}^{|K|}$ ed F con un suo sottoinsieme. Negli esempi che seguono si vedranno altri casi di questa impostazione generale.

1.39 Esempio. Come nell'esempio precedente sia $E = S_n$, $c(i,j)$ una matrice $n \times n$ di costi e $f(x) = \sum_{i=1}^{n} c(i, x_i)$ la funzione obiettivo. Però restringiamo l'insieme ammissibile F alle permutazioni cicliche di S_n (pertanto i valori diagonali della matrice dei costi diventano inutili e possono essere omessi).

L'ottimo dell'esempio 1.36 non è ammissibile per il nuovo vincolo. Si può verificare che la soluzione ottima è invece $x = (2, 3, 4, 1)$ con valore ottimo $v = 2$. Se consideriamo un grafo completo orientato e assegniamo la lunghezza c_{ij} all'arco generico (i, j) allora possiamo interpretare ogni permutazione ciclica come un circuito che passa per tutti i nodi esattamente una volta. Per questo motivo il problema prende il nome di *problema del commesso viaggiatore*, pensando alla minimizzazione di un percorso che visiti un certo numero di città esattamente una volta. Questo problema viene anche indicato con l'acronimo TSP (*Traveling Salesman Problem*). Si veda in figura 1.13 la soluzione ottima. La versione del TSP qui descritta viene detta *asimmetrica* perché la matrice dei costi non è simmetrica. Se assumiamo lunghezze uguali da i a j e da j a i, cioè $c_{ij} = c_{ji}$, si ha la versione cosiddetta *simmetrica*.

Anche il TSP può essere formulato algebricamente usando come insieme primitivo l'insieme E degli archi e quindi variabili $x_{ij} \in \{0,1\}$ che rappresentano vettori d'incidenza di circuiti che passano per tutti i nodi. Tuttavia la formulazione è molto più complessa che non nel problema dell'assegnamento e viene rinviata al capitolo 12. ∎

Figura 1.13

1.40 Esercizio. Spesso un TSP viene assegnato identificando i nodi del grafo con n punti in un spazio euclideo e definendo le lunghezze degli archi come le distanze euclidee corrispondenti. Questa particolare versione del TSP prende il nome di *TSP euclideo*. In figura 1.14 è riportata la soluzione di un TSP euclideo fornita da un algoritmo euristico. Si dimostri che non è ottima. ∎

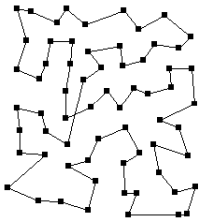

Figura 1.14

1.41 Esempio. Dato un grafo con lunghezze assegnate agli archi, un problema molto naturale è quello di trovare il cammino minimo fra due nodi assegnati. In questo caso l'insieme primitivo è quello degli archi e l'insieme ammissibile quello dei cammini fra i due nodi. I cammini possono essere semplici oppure no, a seconda della formulazione del problema. In figura 1.15 è rappresentato un grafo con le lunghezze indicate vicino agli archi e sono evidenziati i cammini minimi dal nodo in nero a tutti gli altri nodi. In questo caso i cammini minimi formano un albero di supporto.

Se le lunghezze degli archi sono negative (se ad esempio si vuole trovare il cammino più lungo fra due nodi dati e implicitamente si minimizza con lunghezze negative) sorge il problema che un cammino può avvolgersi lungo un circuito di lunghezza negativa un numero infinito di volte e quindi il problema è illimitato. Se imponiamo solo cammini semplici (senza nodi ripetuti) il problema non è più illimitato (esiste solo un numero finito di cammini semplici) ma, come vedremo più avanti, il problema diventa molto più difficile.

1.42 Esercizio. Si dimostri che, dato un grafo connesso con lunghezze positive e assegnato un nodo particolare s, esiste sempre un albero di supporto tale che l'unico cammino da s a i sull'albero è un cammino minimo da s a i nel grafo, per ogni i. ∎

1.43 Esercizio. Sia dato un grafo con lunghezze positive e sia s un nodo assegnato. Per ogni albero di supporto T sia $L(T,i)$ la lunghezza del cammino sull'albero T da s a i. Si definisca $f : T \mapsto R$ come $f(T) := \sum_i L(T,i)$. Per ogni albero T, un intorno di T è costituito da alberi ottenuti da T aggiungendo un arco qualsiasi (i,j) non in T e togliendo l'ultimo arco del cammino $s \to i$ oppure $s \to j$ (in T). Si dimostri che tale intorno è esatto. ∎

1.44 Esercizio. Si consideri il problema di trovare il cammino con il maggior numero di archi fra i due nodi in nero della figura 1.16. Dimostrare che la soluzione evidenziata è ottima oppure trovare un cammino con un numero maggiore di archi. ∎

1.45 Esempio. Se un grafo è orientato e aciclico il problema del cammino (orientato) più lungo fra due nodi assegnati riveste una particolare importanza pratica. Problemi di questo genere sorgono quando i nodi del grafo rappresentano eventi, gli archi relazioni di precedenza

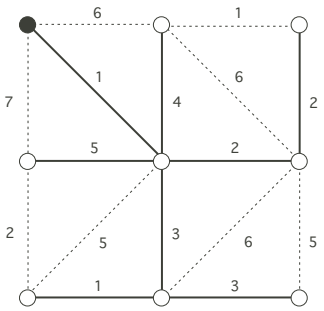

FIGURA 1.15

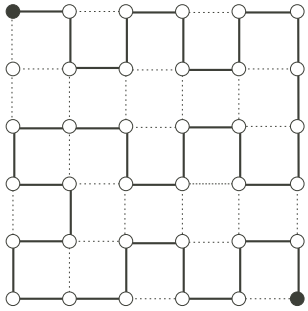

FIGURA 1.16

temporale fra coppie di eventi e i costi intervalli di tempo che devono trascorrere prima che l'evento successivo possa accadere. Il cammino di costo massimo fornisce il minimo intervallo temporale entro cui tutti gli eventi possono accadere. Si veda un esempio in figura 1.17. ∎

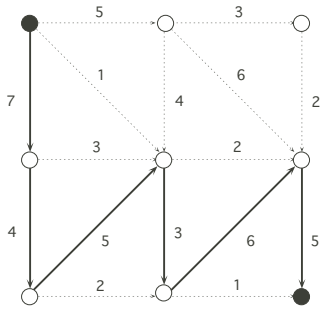

FIGURA 1.17

1.46 ESERCIZIO. Si dimostri che, dato un grafo orientato aciclico e assegnato un nodo particolare s, esiste sempre un albero tale che l'unico cammino da s a i sull'albero è un cammino massimo da s a i nel grafo, per ogni i. Si faccia anche vedere come la proprietà non sia in generale verificata in assenza di aciclicità. ∎

1.47 ESEMPIO. Sia dato un grafo (non orientato) con costi c_e per ogni arco e. Per ogni albero di supporto T si definisca la seguente funzione obiettivo $f(T) := \sum_{e \in T} c_e$. Si vuole trovare l'albero di supporto che minimizza la funzione obiettivo. Questo problema prende il nome di *minimo albero di supporto* (*Minimal Spanning Tree*). Si veda l'esempio in figura 1.18 con i valori dei costi indicati vicino agli archi interessati e la soluzione ottima. ∎

1.48 ESEMPIO. Si ottiene un caso più generale estendendo l'insieme ammissibile a tutti gli alberi che siano di supporto obbligatoriamente ad un sottoinsieme specificato di nodi e facoltativamente all'insieme complementare (nel caso precedente tutti i nodi sono obbligati). Tali alberi prendono il nome di *alberi di Steiner* e i nodi facoltativi eventualmente presenti nell'albero prendono il nome di *nodi di Steiner*. Il problema di trovare il minimo albero di Steiner è molto più difficile del problema precedente, dove il sottoinsieme di supporto coincide con tutti i nodi del grafo. Si veda in figura 1.19 lo stesso grafo con i nodi di supporto indicati in nero. ∎

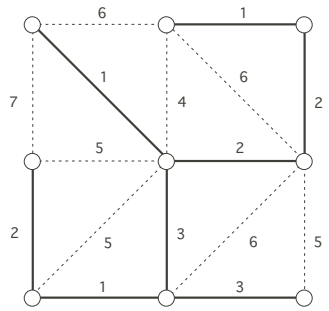

FIGURA 1.18

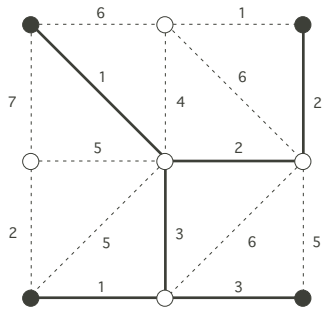

FIGURA 1.19

1.49 ESERCIZIO. Si costruisca una famiglia di grafi con numero di nodi $n = 3, 4, \ldots$, in cui gli archi del minimo albero di supporto sono complementari a quelli del minimo albero di Steiner con $n - 1$ nodi di supporto. ∎

1.50 ESEMPIO. Sia assegnato un grafo connesso (ad esempio si consideri la figura 1.20). Ci possiamo chiedere quale sia il minimo numero di archi che deve essere rimosso per rendere il grafo sconnesso. Possiamo allora scegliere come insieme primitivo l'insieme degli archi e come insieme ammissibile F la famiglia di sottoinsiemi di archi che sconnettono il grafo e il problema diventa $\min_{X \in F} |X|$.

Alternativamente però, e in modo anche più conveniente, possiamo anche formalizzare questo problema considerando come insieme primitivo l'insieme dei nodi del grafo. Ogni sottoinsieme proprio X dei nodi realizza una partizione $(X, N \backslash X)$. L'insieme degli archi che hanno un estremo in X e l'altro in $N \backslash X$ è il minimo insieme di archi che deve essere rimosso se vogliamo separare X da $N \backslash X$. Sia $c(X)$ la cardinalità di questo insieme. Il problema allora diventa:

$$\begin{aligned} \min \quad & c(X) \\ & X \subset N,\ X \neq \emptyset,\ X \neq N \end{aligned}$$

Si veda in figura 1.21 la soluzione ottima $\hat{X}$ evidenziata dai nodi neri. L'insieme degli archi da togliere è costituito dagli archi che congiungono un nodo nero con uno bianco. Come si vede $c(\hat{X}) = 2$. ∎

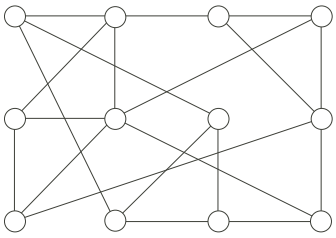

FIGURA 1.20

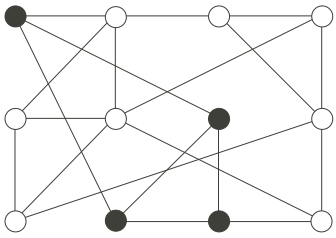

FIGURA 1.21

1.51 ESEMPIO. Sia assegnato un grafo connesso come nel caso precedente (ad esempio si riconsideri la figura 1.20). Siano inoltre assegnati due nodi particolari (ad esempio quello in alto a sinistra e quello in basso a destra). Vogliamo congiungere i due nodi con il massimo numero possibile di cammini, con la proprietà che i cammini non si intersecano se non nei due nodi estremi. Possiamo scegliere come insieme primitivo l'insieme dei cammini fra i

due nodi e come insieme ammissibile F la famiglia di sottoinsiemi di cammini che non si intersecano e il problema diventa $\max_{X \in F} |X|$. Nell'esempio di figura 1.20 vi sono al più due cammini. ∎

1.52 Esercizio. Si modifichi l'esempio 1.50 assegnando inoltre due nodi particolari e si imponga la condizione che i due nodi vengano sconnessi dalla rimozione degli archi. Si dimostri che il valore ottimo dell'esempio 1.50 modificato è sempre maggiore o uguale al valore ottimo dell'esempio 1.51 (si vedrà più avanti che i due valori ottimi devono essere uguali). ∎

1.53 Esempio. Sia dato un grafo $G = (N, E)$ e si cerchi il massimo numero possibile di archi in modo che in ogni nodo ci sia al più un arco incidente. In modo naturale l'insieme primitivo è l'insieme degli archi. Per ogni sottoinsieme X di archi sia $\deg : X \mapsto \{0, 1, \ldots, |N| - 1\}^N$ la funzione che assegna ad ogni nodo il numero di archi di X incidenti nel nodo. Allora il problema può essere formulato come

$$\begin{array}{ll} \max & |X| \\ & \deg(X) \leq \mathbf{1} \end{array}$$

Il problema prende il nome di *problema dell'accoppiamento* (*Matching*) e viene chiamato accoppiamento ogni sottoinsieme X di archi per cui $\deg(X) \leq \mathbf{1}$. Si veda la figura 1.22 in cui è anche evidenziato uno fra i diversi accoppiamenti ottimi. Un accoppiamento X si dice *perfetto* se $\deg(X) = \mathbf{1}$. ∎

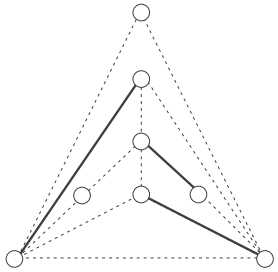

Figura 1.22

1.54 Esempio. Il problema dell'accoppiamento si presenta anche nella variante del cosiddetto accoppiamento pesato. Sono assegnati dei costi c_e agli archi e si cerca l'accoppiamento di costo massimo. In un'altra variante si cerca l'accoppiamento perfetto di costo massimo (o minimo). In questo caso è richiesto ovviamente un numero pari di nodi. Si noti che il problema dell'assegnamento (esempio 1.36) è anche un caso particolare di accoppiamento su un grafo bipartito. ∎

1.55 Esercizio. Si assegnino costi positivi agli archi del grafo di figura 1.22 in modo che l'accoppiamento indicato non sia massimo rispetto ai costi. ∎

1.56 Esempio. Sia dato un grafo $G = (N, E)$ e si cerchi il massimo numero possibile di nodi in modo che nessun nodo sia adiacente ad un altro nodo. Un tale insieme di nodi prende il nome di insieme *stabile* oppure *indipendente*. L'insieme primitivo è ovviamente l'insieme dei nodi. Per ogni sottoinsieme X di nodi sia $g : X \mapsto \{0, 1, 2\}^E$ la funzione che assegna ad ogni arco il numero di nodi di X incidenti nell'arco. Possiamo pertanto definire il *problema del massimo insieme stabile* (o indipendente) come

$$\begin{aligned} \max \quad & |X| \\ & g(X) \leq \mathbf{1} \end{aligned}$$

Si veda in figura 1.23 un grafo con un possibile ottimo (nodi neri). ∎

1.57 Esempio. Sia sempre dato un grafo $G = (N, E)$ e si cerchi stavolta il minimo numero possibile di nodi in modo che ogni arco sia incidente ad almeno uno dei nodi scelti. Un tale insieme di nodi prende il nome di *copertura di nodi*, nel senso appunto che ogni arco è 'coperto' da almeno un nodo. Usando la stessa funzione g dell'esempio precedente, il problema della minima copertura di nodi può essere formulato come:

$$\begin{aligned} \min \quad & |X| \\ & g(X) \geq \mathbf{1} \end{aligned}$$

Si veda in figura 1.24 un grafo con un possibile ottimo (nodi neri). ∎

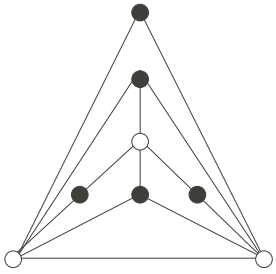

Figura 1.23

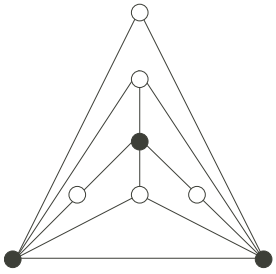

Figura 1.24

1.58 Esercizio. Confrontando le figure 1.23 e 1.24 si nota come i due ottimi siano complementari. Non è un fatto casuale. Si dimostri che l'insieme complementare di un insieme stabile è sempre una copertura di nodi. ∎

1.59 Esercizio. Si dimostri che per ogni grafo la cardinalità di una qualsiasi copertura di nodi è maggiore o uguale alla cardinalità di un qualsiasi accoppiamento. Si trovi un esempio in cui i due valori sono diversi. ∎

1.60 Esercizio. Si dimostri in modo costruttivo che per ogni grafo bipartito la cardinalità di una copertura minima di nodi è uguale alla cardinalità di un accoppiamento massimo (suggerimento: dato un accoppiamento massimo, si usino cammini che partono da nodi esposti e che alternano archi accoppiati con archi non accoppiati, e si scelgano per la copertura opportuni nodi sui cammini). ∎

1.61 Esercizio. Si sfrutti il risultato dell'esercizio 1.59 per dimostrare che le soluzioni delle figure 1.22, 1.23 e 1.24 sono ottime. ∎

1.62 Esercizio. Un sottoinsieme di nodi sia contemporaneamente stabile e copertura. Si dimostri che il grafo è necessariamente bipartito. ∎

1.63 Esercizio. Si trovi un esempio in cui esiste un insieme stabile che: a) è anche una copertura, b) non ha cardinalità massima. ∎

1.64 Esempio. Dato un grafo $G = (N, A)$, ogni sottoinsieme di nodi che induca un sottografo completo viene detto *cricca* (*Clique*). Il problema è quello di determinare la cricca massima. Questo problema è strettamente collegato con i precedenti come si fa vedere nei successivi esercizi. ∎

1.65 Esercizio. Si trovi una cricca massima nel grafo di figura 1.25. ∎

1.66 Esercizio. Si dimostri che ogni insieme stabile è una cricca nel grafo complementare. ∎

1.67 Esercizio. Si trovi un massimo insieme stabile ed una minima copertura di nodi per il grafo di figura 1.25. ∎

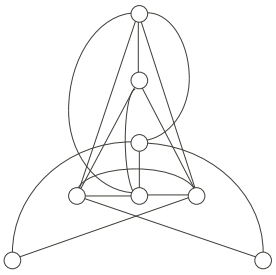

Figura 1.25

1.68 Esempio. È possibile estendere gli esempi precedenti a situazioni più generali, notando che gli archi di un grafo possono essere identificati con sottoinsiemi di esattamente due nodi. Pertanto l'insieme degli archi è una particolare famiglia di sottoinsiemi dei nodi. Analogamente un nodo può essere identificato con il sottoinsieme di archi incidenti in esso e quindi l'insieme dei nodi è una particolare famiglia di sottoinsiemi degli archi.

Dato un insieme primitivo K e una famiglia di sottoinsiemi $\mathcal{K}$ si può definire la seguente matrice d'incidenza in cui ogni riga è in corrispondenza con un elemento di K e ogni colonna con un elemento di $\mathcal{K}$:

$$a_{ij} := \begin{cases} 1 & \text{se } i \in j \\ 0 & \text{altrimenti} \end{cases}$$

Si noti che se invece interpretiamo $\mathcal{K}$ come insieme primitivo e identifichiamo ogni elemento di K con gli insiemi della famiglia che lo contengono, ribaltiamo la relazione che c'è fra righe e colonne ed abbiamo la matrice d'incidenza trasposta di quella data.

Questo tipo di problemi è ad un livello di complessità superiore rispetto ai problemi precedenti dove, dato un insieme primitivo K si cercava un sottoinsieme di K. Ora, dato

l'insieme primitivo K e data (tipicamente in modo implicito tramite la verifica di una certa proprietà) una famiglia $\mathcal{K}$ di sottoinsiemi di K, si cerca un sottoinsieme di $2^{\mathcal{K}}$ (cioè di 2^{2^K}).

Sia pertanto A una matrice d'incidenza di una generica famiglia $\mathcal{K}$ di sottoinsiemi H di un insieme primitivo K. Sia $\mathcal{X}$ una sottofamiglia di $\mathcal{K}$ e sia x il suo vettore d'incidenza. Pertanto i problemi visti precedentemente si possono inquadrare nella classe più generale di problemi di

– *Copertura d'insiemi* (*Set Covering*)

$$\begin{array}{ll} \min & |\mathcal{X}| \\ & \bigcup_{H\in\mathcal{X}} H = K \\ & \mathcal{X} \subset \mathcal{K} \end{array} \qquad \Longrightarrow \qquad \begin{array}{ll} \min & \mathbf{1}\,x \\ & A\,x \geq \mathbf{1} \\ & x \in \{0,1\}^{|\mathcal{K}|} \end{array}$$

– *Impaccamento d'insiemi* (*Set Packing*)

$$\begin{array}{ll} \max & |\mathcal{X}| \\ & \forall H_1, H_2 \in \mathcal{X},\ H_1 \neq H_2,\ H_1 \cap H_2 = \emptyset \\ & \mathcal{X} \subset \mathcal{K} \end{array} \qquad \Longrightarrow \qquad \begin{array}{ll} \max & \mathbf{1}\,x \\ & A\,x \leq \mathbf{1} \\ & x \in \{0,1\}^{|\mathcal{K}|} \end{array}$$

– *Partizione d'insiemi* (*Set Partitioning*)

$$\begin{array}{l} \exists\,?\,\mathcal{X} \\ \quad \forall H_1, H_2 \in \mathcal{X},\ H_1 \neq H_2,\ H_1 \cap H_2 = \emptyset \\ \quad \bigcup_{H\in\mathcal{X}} H = K \\ \quad \mathcal{X} \subset \mathcal{K} \end{array} \qquad \Longrightarrow \qquad \begin{array}{ll} \exists\,?\,x \\ \quad A\,x = \mathbf{1} \\ \quad x \in \{0,1\}^{|\mathcal{K}|} \end{array}$$

Il caso più frequente nelle applicazioni inoltre prevede un costo $c(H)$ per ogni $H \in \mathcal{K}$ (in qualche caso calcolabile come $\sum_{i\in H} c_i$ per dei costi definiti sugli elementi di K) e l'obiettivo è di minimizzare (o massimizzare) $\sum_{H\in\mathcal{X}} c(H)$. La duplice difficoltà di questi problemi risiede nella presenza di variabili 0-1 e nel fatto che normalmente la famiglia $\mathcal{K}$ è definita implicitamente ed è formata da un numero elevatissimo di sottoinsiemi.

Come possibile esempio si consideri il grafo di sinistra nella figura 1.26 e si voglia trovare una copertura degli archi del grafo fatta di circuiti passanti per il nodo nero. I possibili circuiti sono evidenziati nella figura. In questo semplice caso la matrice d'incidenza archi-circuiti ha 6 righe e 6 colonne e può essere generata esplicitamente, ma per grafi più grandi la cosa sarebbe fuori questione.

$$\mathrm{A} = \begin{pmatrix} 1 & 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 1 \end{pmatrix}$$

1.69 Esercizio. Si assegnino costi arbitrari a ciascuno dei 6 circuiti della figura 1.26 e si risolva il corrispondente problema di set covering. Si risolva nuovamente rilassando il vincolo $x \in \{0,1\}^6$ in $x \in [0,1]^6$. ∎

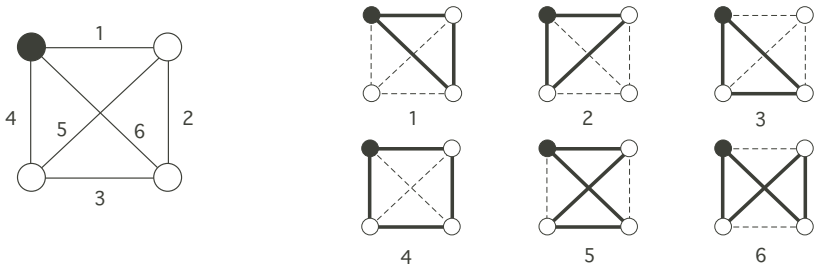

Figura 1.26

Alcuni problemi di partizione hanno una struttura abbastanza particolare e sono di frequente applicazione, per cui vale la pena indicarli esplicitamente.

1.70 Esempio. Siano assegnati n numeri non negativi $a_1, \ldots, a_n$ (ad esempio sia $a = \{31, 15, 23, 11, 17, 43, 49, 7, 36\}$) e si vogliano suddividere in due sottoinsiemi in modo che le somme dei numeri nei due sottoinsiemi differiscano il meno possibile. Formalmente l'insieme primitivo è $K = \{1, 2, \ldots, n\}$ e si vuole trovare un sottoinsieme $X \subset K$ tale che la quantità

$$|\sum_{i \in X} a_i - \sum_{i \notin X} a_i|$$

sia minima. Il problema si può riformulare identificando ogni sottoinsieme X con un vettore $x \in \{0,1\}^n$ e risolvendo

$$\begin{aligned} \max \quad & \sum_{i=1}^{n} a_i \, x_i \\ & \sum_{i=1}^{n} a_i \, x_i \leq \sum_{i=1}^{n} a_i / 2 \\ & x_i \in \{0,1\} \quad \forall i \end{aligned}$$

Nell'esempio numerico l'ottimo è dato da $\hat{X} = \{1, 2, 3, 4, 9\}$ e si ha $\sum_{i \in \hat{X}} a_i = \sum_{i \notin \hat{X}} a_i = 116$, quindi le due somme sono esattamente uguali. Il problema di suddividere gli n numeri in due sottoinsiemi di uguale somma prende il nome di *problema della partizione*. ∎

1.71 Esercizio. Si formuli il problema della partizione, con il vincolo aggiuntivo che uno dei due sottoinsiemi abbia una cardinalità fissata, usando variabili 0-1. ∎

1.72 Esempio. Il problema della partizione è un caso particolare di una più ampia famiglia di problemi in cui sono assegnati due insiemi di numeri non negativi $a_1, \ldots, a_n$ e $b_1, \ldots, b_n$ ed un numero c. L'insieme primitivo è sempre $K = \{1, 2, \ldots, n\}$ e i sottoinsiemi ammissibili X sono quelli per cui $\sum_{i \in X} b_i \leq c$. Fra questi si cerca quello che massimizza $\sum_{i \in X} a_i$. Tale problema prende il nome di *problema dello zaino* (*knapsack*) perché si può interpretare come la scelta di un carico di oggetti, ciascuno con un peso ed un valore, in modo da massimizzare

il valore caricato nello zaino al di sotto di un limite superiore di peso. Formulato con variabili 0-1 il problema è

$$\begin{aligned} \max \quad & \sum_{i=1}^{n} a_i\, x_i \\ & \sum_{i=1}^{n} b_i\, x_i \le c \\ & x_i \in \{0,1\} \quad \forall i \end{aligned} \tag{1.1}$$

Si consideri ad esempio la seguente istanza: $a = (24, 5, 26, 28)$, $b = (8, 5, 13, 7)$, $c = 26$. L'ottimo è $\hat{x} = (0, 1, 1, 1)$, ovvero $\hat{X} = (2, 3, 4)$, con valore ottimo $v = 59$. ∎

1.73 Esercizio. Si ammettano anche valori negativi nella definizione di un problema di knapsack. Come ci si può ricondurre al caso di valori non negativi? ∎

1.74 Esempio. Se in (1.1) rilassiamo il vincolo $x \in \{0,1\}^n$ in $0 \le x \le 1$, otteniamo un nuovo problema che viene indicato come *knapsack Continuo*. La soluzione è tipicamente diversa. Infatti riprendendo l'istanza dell'esempio 1.72 si ottiene l'ottimo $\hat{x} = (1\,,\, 0\,,\, \frac{11}{13}\,,\, 1)$ con valore ottimo $v = 74$. ∎

1.75 Esercizio. Nel problema del knapsack continuo si effettui la trasformazione di variabile $y_i = b_i\, x_i$, in modo da risolvere il seguente problema:

$$\begin{aligned} \max \quad & \sum_{i=1}^{n} \frac{a_i}{b_i}\, y_i \\ & \sum_{i=1}^{n} \quad y_i \le c \\ & 0 \le y_i \le b_i \quad \forall i \end{aligned} \tag{1.2}$$

Si deduca la struttura della soluzione ottima. ∎

1.76 Esempio. Siano assegnati n numeri interi positivi $a_1, \ldots, a_n$ ed un intero positivo C. Si vuole trovare una partizione di $\{1, \ldots, n\}$ in sottoinsiemi $X_1, \ldots, X_m$, in modo che $\sum_{i \in X_j} a_i \le C$, $j := 1, \ldots, m$, ed m sia minimo. Tale problema può corrispondere alla situazione in cui vi sono n oggetti (ad esempio file), ciascuno di dimensione a_i (ad es. la lunghezza in byte del file), vi sono dei contenitori di capacità C (dischetti di capacità nota e fissata) e si vuole trovare il minimo numero di contenitori in cui inserire tutti gli oggetti rispettando le capacità dei contenitori. Per questo motivo il problema prende il nome di *Bin Packing*.

Si ottiene un problema simmetrico se si pensa di fissare ad un valore K il numero dei contenitori e di togliere il vincolo sulla capacità. Si vuole allora trovare una partizione $X_1, \ldots, X_K$, tale che $\max_j \sum_{i \in X_j} a_i$ sia minimo. In questa forma il problema può corrispondere alla situazione in cui gli oggetti sono dei lavori da eseguire, i valori a_i rappresentano le durate delle esecuzioni, sono a disposizione K processori e si vuole distribuire i lavori sui processori in modo da completare tutti i lavori nel minor tempo possibile. Per questo motivo il problema prende il nome di *Multiprocessor Scheduling*. ∎

In molti problemi ciò che interessa non è tanto trovare l'ottimo all'interno di un insieme di elementi quanto più semplicemente verificare che un opportuno insieme A non sia vuoto e trovare un suo qualsiasi elemento. Anche se questi problemi sono apparentemente più semplici di quelli di ottimizzazione, la loro risoluzione presenta spesso lo stesso tipo di difficoltà. Formalmente $\emptyset \neq A \subset E$ (con E compatto, caso abbastanza frequente) se e solo se $v = 0$ per una funzione obiettivo continua $f\colon E \to R$ tale che $f(x) = 0$ se $x \in A$ e $f(x) > 0$ se $x \notin A$. Quindi un problema di ammissibilità è teoricamente riconducibile ad un problema di ottimizzazione. D'altro lato il valore ottimo di un problema di ottimizzazione può essere calcolato mediante ripetuti problemi di ammissibilità del tipo

$$A := \{x \in E\colon f(x) \leq K,\, x \in F\}$$

per diversi valori di una costante K.

1.77 Esempio. Siano date funzioni $g : R^n \to R^p$ e $h : R^n \to R^q$ e sia

$$A := \{x :\ g(x) \leq 0,\, h(x) = 0\}$$

Per determinare un elemento di A si può ad esempio risolvere (perché?)

$$\begin{aligned} v = \min \quad & \mathbf{1} \cdot z_g + \mathbf{1} \cdot z_h \\ & g(x) + z_g \leq 0 \\ & h(x) + z_h = 0 \\ & z_g \geq 0,\ z_h \geq 0 \end{aligned}$$

∎

1.78 Esempio. Si supponga di avere a disposizione un algoritmo di ammissibilità che, assegnata una matrice H, fornisca in uscita un semplice 'sì' oppure un 'no', a seconda se l'insieme $\{x \in \{0,1\}^n : H\,x \leq b\}$ sia ammissibile oppure no. Si voglia risolvere:

$$\begin{aligned} v = \min\, & c\,x \\ & A\,x \leq b \\ & x \in \{0,1\}^n \end{aligned}$$

dove c e b sono vettori e A è una matrice, tutti di dimensioni opportune e a valori interi. Sia $F := \{x \in \{0,1\}^n : A\,x \leq b\}$. Supponiamo, per comodità che $c \geq 0$. Quindi $0 \leq v \leq \sum_i c_i$. Sia $F(K) := \{x \in F : c\,x \leq K\}$. L'algoritmo è in grado di dire se $F(K)$ è vuoto o no per ogni valore intero di K. Se $F(0) \neq \emptyset$ allora $x = 0$ è ottimo. Se $F(\sum_i c_i) = \emptyset$ allora $F = \emptyset$ e quindi il problema di minimizzazione è non ammissibile. Altrimenti si procede con ricerca binaria fino a determinare il valore ottimo v (cioè quel valore per cui $F(v) \neq \emptyset$ e $F(v-1) = \emptyset$). A questo punto si impone $x_1 := 0$. Definiamo $F_0 := \{x \in \{0,1\}^n : A\,x \leq b,\, x_1 = 0\}$. Se $F_0(v) \neq \emptyset$ allora $\hat{x}_1 := 0$, altrimenti $\hat{x}_1 := 1$. Si procede allo stesso modo, una variabile alla volta, finché è determinato tutto il vettore.

L'ottimo è stato pertanto determinato con $O(\log \sum_i c_i + n)$ chiamate dell'algoritmo di ammissibilità. ∎

1.79 Esercizio. Si supponga di avere a disposizione un algoritmo di ammissibilità per il problema del TSP. In altre parole l'algoritmo fornisce in uscita un 'sì' oppure un 'no', a seconda se esistono soluzioni ammissibili non peggiori di un valore fissato K. Si indichi come trovare l'ottimo usando questo algoritmo. ∎

I seguenti esempi sono di natura combinatoria. I primi due sono formulati come problemi logici. Si vedrà più avanti come in realtà molti problemi si riescano a convertire l'uno nell'altro.

1.80 Esempio. Sia data una formula logica booleana in forma congiuntiva. Ad esempio:

$$(x_1 \vee x_2 \vee \neg x_3) \wedge (\neg x_1 \vee x_2 \vee \neg x_4) \wedge (\neg x_1 \vee \neg x_3 \vee x_4) \wedge (\neg x_2 \vee x_3 \vee \neg x_4)$$

dove le variabili booleane x_j possono assumere il valore vero (V) o falso (F). La domanda è: esiste un assegnamento di valori alle variabili tale che la formula è soddisfatta (cioè è vera)? Ci sono 2^4 assegnamenti possibili. Ci si chiede quindi se ne esiste uno ammissibile, intendendo per ammissibile uno che renda vera la formula. Tale problema prende il nome di *problema della soddisfattibilità*. Nell'esempio dato la formula è soddisfattibile con molti assegnamenti fra cui ad esempio $x_i := V$, $\forall i$. ∎

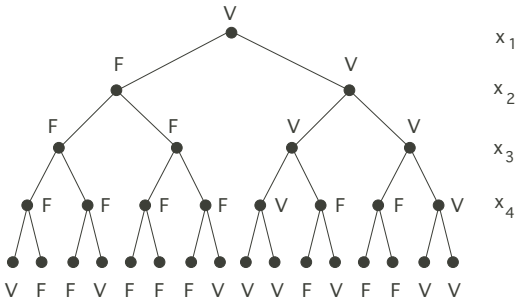

Figura 1.27

1.81 Esempio. Sia data una formula booleana in forma congiuntiva come nell'esempio precedente. Questa volta ci chiediamo però se esiste un valore di x_1 tale che per ogni valore di x_2 (cioè indipendentemente dal fatto che x_2 sia vero o falso) esiste un valore per x_3 (dipendente dal valore di x_2 e x_1) tale che per ogni valore di x_4 la formula sia soddisfattibile. Tale problema prende il nome di *problema della soddisfattibilità quantificata*. In formula:

$$\exists x_1\, \forall x_2\, \exists x_3\, \forall x_4,\ (x_1 \vee x_2 \vee \neg x_3) \wedge (\neg x_1 \vee x_2 \vee \neg x_4) \wedge (\neg x_1 \vee \neg x_3 \vee x_4) \wedge (\neg x_2 \vee x_3 \vee \neg x_4) = V\,?$$

Per capire la natura del problema si veda in figura 1.27 l'albero che rappresenta tutti i 16 assegnamenti di valori vero-falso alle variabili. La variabile che viene assegnata ad ogni livello è indicata a destra dell'albero. Ogni ramo di sinistra corrisponde all'assegnamento 'vero' e ogni ramo di destra a 'falso'. Sotto le foglie, nella riga più bassa, i valori V o F indicano se la formula è vera o falsa, per ogni assegnamento.

Si immagini ora un gioco fra due giocatori P e D (per pari e dispari). D assegna i valori alle variabili dispari e P a quelle pari. I valori vengono assegnati in sequenza secondo gli indici delle variabili. D vuole rendere la formula vera e P falsa. Per capire se vincerà D o P consideriamo l'ultima mossa che spetta a P. Ovviamente P sceglierà in modo che la formula sia falsa, se ne esiste la possibilità. Una volta che P ha eseguito le sue scelte in

tutte le posizioni dell'albero dove si assegna x_4, il valore finale della formula è noto (valori V e F nel livello superiore a quello dello foglie). Ora D deve scegliere l'assegnamento per x_3 in tutte le posizioni dove si assegna x_3. Siccome D vuole rendere vera la formula, sceglie in modo conseguente. Proseguendo ricorsivamente si vede che D può vincere. Basta che assegni $x_1 := F$ e poi se $x_2 := V$ bisogna assegnare $x_3 := V$, mentre se $x_2 := F$ bisogna assegnare $x_3 := F$. A questo punto il valore di x_4 è irrilevante. ∎

1.82 Esempio. Si riconsideri il problema dell'assegnamento (esempio 1.36). Spesso si deve risolvere una versione del problema in cui il grafo bipartito non è completo e ci si chiede semplicemente se esiste un accoppiamento perfetto. Un'altra variante del problema chiede di trovare l'assegnamento (cioè l'accoppiamento) di cardinalità massima. ∎

1.83 Esempio. Il problema dell'assegnamento può essere riformulato come: sono dati due insiemi A e B disgiunti e di uguale cardinalità. È data inoltre una famiglia di coppie $\mathcal{K} := \{(a_1, b_1), (a_2, b_2), \ldots, (a_m, b_m)\}$ con $a_i \in A$ e $b_i \in B$. Si vuole trovare una sottofamiglia $\mathcal{X} \subset \mathcal{K}$ che costituisca una partizione di $A \cup B$ (interpretando le coppie come sottoinsiemi di due elementi). In altre parole ogni elemento di A deve figurare in esattamente una coppia di $\mathcal{X}$, e altrettanto per B.

Formulato in questo modo il problema dell'assegnamento può essere facilmente generalizzato. Il seguente problema prende il nome di *assegnamento tridimensionale*: sono dati tre insiemi A, B e C disgiunti e di uguale cardinalità. È data inoltre una famiglia di triple $\mathcal{K} := \{(a_1, b_1, c_1), (a_2, b_2, c_2), \ldots, (a_m, b_m, c_m)\}$ con $a_i \in A$, $b_i \in B$ e $c_i \in C$. Si vuole trovare una sottofamiglia $\mathcal{X} \subset \mathcal{K}$ che costituisca una partizione di $A \cup B \cup C$. In altre parole ogni elemento di A deve figurare in esattamente una tripla di $\mathcal{X}$, e altrettanto per B e C. ∎

1.84 Esercizio. Si formuli il problema dell'assegnamento tridimensionale come un problema di programmazione lineare 0-1. ∎

1.85 Esercizio. Si risolva il particolare problema di assegnamento tridimensionale definito in figura 1.28, dove le terne di $\mathcal{K}$ sono definite implicitamente a coppie, ovvero $(a, b, c) \in \mathcal{K}$ se e solo se $(a, b) \in \mathcal{K}_1$, $(b, c) \in \mathcal{K}_2$ e $(a, c) \in \mathcal{K}_3$, e $\mathcal{K}_1$, $\mathcal{K}_2$ e $\mathcal{K}_3$ sono definiti dalle caselle bianche in figura 1.28. ∎

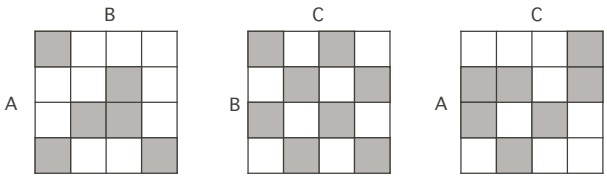

Figura 1.28

1.86 Esempio. Il problema della partizione (vedi esempio 1.70) è un problema di ammissibilità: esiste un insieme $J \subset \{1, \ldots, n\}$ tale che $\sum_{i \in J} a_i = \sum_{i \notin J} a_i$? Un problema simile ma più complesso prende il nome di *tripartizione* ed è importante per questioni di complessità computazionale (vedi capitolo 3): dato un intero B, dati $3\,m$ interi positivi $a_1, \ldots, a_{3m}$ tali che $B/4 < a_i < B/2$, $\forall i$, e $\sum_{i=1}^{3m} a_i = m\,B$, esiste una partizione $X_1, \ldots, X_m$ di $\{1, \ldots, 3\,m\}$

tale che $\sum_{i \in X_j} a_i = B$ per ogni j? (si noti che ogni X_j deve contenere esattamente tre elementi).

Una variante del problema di tripartizione, anche detta *assegnamento tridimensionale numerico*, prevede tre insiemi di numeri $a_1, \ldots, a_m$, $b_1, \ldots, b_m$ e $c_1, \ldots, c_m$ e ci si chiede se esistono due permutazioni π e μ tali che $a_i + b_{\pi(i)} + c_{\mu(i)} = B$ per ogni i. ∎

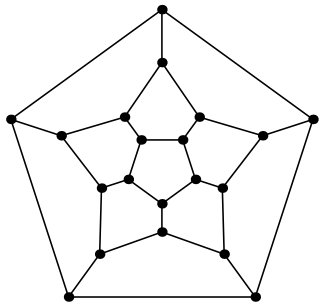

FIGURA 1.29

1.87 ESEMPIO. Sia dato un grafo $G = (N, E)$. Esiste un circuito che passa per tutti i nodi di G esattamente una volta? Storicamente tale problema fu posto per la prima volta dal matematico irlandese William Rowan Hamilton (1805-1865) in connessione con un gioco, inventato dallo stesso e fisicamente realizzato con una tavola di legno, pioli e un pezzo di corda, in cui si chiedeva appunto di far passare la corda su tutti i pioli senza ripetizioni. Il gioco aveva la forma del grafo in figura 1.29 che rappresenta la struttura di vertici e spigoli dell'icosaedro (e così infatti il gioco venne battezzato da Hamilton). Apparentemente il gioco non ebbe un gran successo, ma il concetto matematico rimase ben legato al nome di Hamilton, tant'è che i circuiti che passano per tutti i nodi esattamente una volta vengono detti *circuiti hamiltoniani* e i grafi per i quali esiste un circuito hamiltoniano vengono detti anch'essi hamiltoniani.

Il problema di determinare se un grafo è hamiltoniano è strettamente legato al TSP. Infatti, dato un grafo $G = (N, E)$ si può costruire un grafo completo in cui viene assegnato un costo c_1 agli archi in E e un costo $c_2 > c_1$ agli archi non in E. Se il valore ottimo è $n\,c_1$ il grafo è hamiltoniano, altrimenti non lo è.

Una variante del problema del circuito hamiltoniano è il problema del cammino hamiltoniano, in cui si richiede di trovare un cammino che passi per tutti i nodi esattamente una volta. Questo problema si presenta a sua volta in tre versioni a seconda se: a) non sono fissati i nodi estremi del cammino, b) è fissato un nodo estremo, c) sono fissati ambedue i nodi estremi. ∎

1.88 ESERCIZIO. Sia dato un grafo G. Si trovi una trasformazione $\tau : G \mapsto G'$, dove G' è un altro grafo, con la proprietà che G è hamiltoniano se e solo se in G' esiste un cammino hamiltoniano senza nodi estremi fissati. Si ripeta l'esercizio per gli altri casi di cammino hamiltoniano (i nodi estremi fissati vengono decisi dalla trasformazione).

Viceversa si trovi una trasformazione $\tau' : G \mapsto G'$ con la proprietà che G' è hamiltoniano se e solo se in G esiste un cammino hamiltoniano senza nodi estremi fissati. Si ripeta l'esercizio per gli altri casi di cammino hamiltoniano. ∎

1.89 Esempio. Sia dato un grafo $G = (N, E)$. Esiste un circuito che passa per tutti gli archi di G esattamente una volta? Anche questo problema fu posto per la prima volta (nel 1736) da un grande matematico, Leonhard Euler (1707-1783). In questo caso il problema riguardava la possibilità di fare una passeggiata attraversando un'unica volta tutti i ponti che connettono due isole sul Pregel a Königsberg (figura 1.30). Si considera questo lavoro di Eulero come la nascita della Teoria dei Grafi.

Un tale circuito venne successivamente chiamato euleriano ed euleriani sono i grafi che ne posseggono uno. Una immediata condizione di necessità affinché un grafo sia euleriano è data dal fatto che ogni nodo deve avere grado pari (ovviamente da ogni nodo si entra e si esce). Questa condizione fu fornita da Eulero [1736]. La condizione è anche sufficiente come fu provato molti anni più tardi da Hierholzer [1876].

Esiste la variante del cammino euleriano. In questo caso la condizione necessaria e sufficiente all'esistenza di un cammino euleriano è data dall'avere esattamente due nodi di gradi dispari. Nel noto giochino di disegnare una figura senza staccare mai la matita dal foglio (e senza ripetere linee già disegnate) si chiede appunto di trovare un cammino euleriano. Il teorema non solo ci dice che il familiare grafo di figura 1.31 può essere disegnato con un solo tratto di matita ma anche ci indica da dove cominciare. ∎

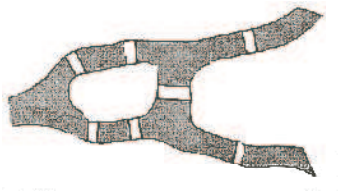

Figura 1.30

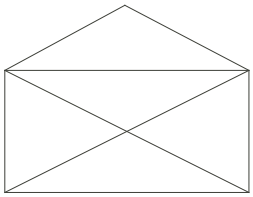

Figura 1.31

1.90 Esercizio. Dimostrare in modo costruttivo che la condizione di grado pari per tutti i nodi è sufficiente all'esistenza di un circuito euleriano. ∎

1.91 Esercizio. Si supponga di dover far disegnare ad un plotter i confini di comune di un'area molto estesa (ad esempio una regione). Tale figura è assimilabile ad un grafo connesso G, i cui nodi corrispondono ai punti dove almeno tre comuni sono confinanti e i cui archi corrispondono ai confini fra due soli comuni. Il disegno deve essere ovviamente eseguito senza che la punta scrivente passi due volte sulla stessa linea.

Quindi se G è euleriano il disegno è eseguito nel minor tempo quando la punta percorre un circuito euleriano. Tuttavia è normale che G non sia euleriano; in questo caso la punta deve sollevarsi e spostarsi su un altro punto della figura e da lì riprendere il disegno. Quindi il tempo totale di esecuzione del disegno (assimilabile alla distanza percorsa dalla punta) è pari ad una costante (linee del disegno che comunque vanno eseguite) più una parte variabile (segmenti di spostamento).

Dimostrare che il modo più efficiente di eseguire gli spostamenti consiste nel saltare fra due punti della figura corrispondenti a nodi di grado dispari. Si tratta quindi di individuare le coppie di punti su cui eseguire gli spostamenti. Far vedere come questo problema può essere risolto con un problema di accoppiamento pesato di costo minimo.

Si faccia vedere come l'approccio descritto fallisce se il disegno non è connesso. In particolare si consideri il caso estremo di un disegno consistente soltanto in punti isolati. Quale problema si deve risolvere per trovare il modo più rapido per disegnare i punti? ∎

1.92 Esercizio. Si possono disporre tutte le tessere del gioco del domino formando un'unica catena? ∎