SAPIENZA
UNIVERSITÀ DI ROMA

Corso di laurea in Ingegneria
dell'Informazione
sede di Latina

Appunti dalle lezioni di

Ricerca Operativa

Simone Sagratella

Pagine web del corso:
https://sites.google.com/diag.uniroma1.it/ricerca-operativa-latina/home

Prefazione

Queste note sono redatte in via prelimiare ad esclusivo uso degli studenti del corso di *"Ricerca Operativa"* da **6 crediti** (CFU) del Corso di Laurea in *Ingegneria dell'Informazione* della sede di Latina della SAPIENZA, Università di Roma.

Molte parti di queste note sono tratte da

1. F. Facchinei, S. Lucidi, M. Roma. *Appunti dalle lezioni di Ricerca Operativa*, SAPIENZA – Università di Roma.
2. M. Roma, *Appunti dalle lezioni di Laboratorio Ricerca Operativa*, SAPIENZA – Università di Roma.
3. L. Palagi, G. Di Pillo. *Note per il corso di OTTIMIZZAZIONE (a.a. 2007-08)*, SAPIENZA – Università di Roma.

A completamento degli argomenti svolti in queste note, è disponibile sul sito web del corso una raccolta di esercizi svolti in corrispondenza di ogni capitolo ed altro materiale didattico.

1

Introduzione

1.1 CHE COSA È LA RICERCA OPERATIVA

La Ricerca Operativa è una disciplina relativamente recente. Il termine *Ricerca Operativa* è stato coniato verso la fine degli anni '30 e deriva dal termine inglese *"Operational Research"* o *"Operations Research"* in americano.
La Ricerca Operativa è una disciplina che tratta dello sviluppo e dell'applicazione di metodi scientifici per la soluzione di problemi di decisione che si presentano in molteplici e diversi settori della vita reale. Si tratta di scegliere quali decisioni prendere per gestire nel modo più efficiente un sistema reale utilizzando strumenti matematici; quindi lo scopo della Ricerca Operativa è quello di fornire una base scientifica per cercare di analizzare e comprendere situazioni anche con strutture molto complesse e quindi utilizzare queste informazioni per predire il comportamento di un sistema e per migliorare le prestazioni del sistema stesso.
La necessità di un approccio quantitativo ai problemi di decisione è largamente riconosciuto in moltissimi settori della vita reale ed in particolare nei problemi di decisione che si presentano nella gestione dei sistemi di produzione e nella gestione d'impresa. Il semplice "buon senso", cioè l'impiego di una persona competente del settore che sulla base dell'esperienza acquisita nel corso degli anni gestisca il sistema non è piú sufficiente a far fronte alla sempre piú crescente complessità organizzativa, e quindi anche decisionale, della gran parte dei sistemi di produzione e servizio. In questo settore, come in molti altri, soprattutto negli ultimi anni, si è acquisita la consapevolezza della necessità di tecniche quantitative basate su sofisticati strumenti matematici e avanzati mezzi informatici che permettano di prendere delle decisioni operative sulla base delle informazioni disponibili.

La Ricerca Operativa, quindi, è la scienza che si occupa di fornire un contesto unitario a nozioni matematiche, informatiche e che partendo da basi teoriche arriva alla costruzione di modelli concreti e alla loro soluzione cioè ad un confronto diretto con la realtà. In questo senso, un altro termine inglese che solitamente si riferisce alla Ricerca Operativa – *Management Science* – evidenzia gli aspetti piú caratteristici della disciplina: *"management"* cioè la gestione e *"science"* a mettere in evidenza il carattere rigoroso tipico di una scienza.

1.2 BREVE STORIA DELLA RICERCA OPERATIVA

Il termine Ricerca Operativa ha origini "ufficiali" legate ad operazioni belliche della Seconda Guerra Mondiale. Tuttavia esistono esempi importanti di anticipazinoi dei metodi della Ricerca Operativa in anni piú lontani; il piú famoso risale a F. Taylor che nel 1885 elaborò uno studio sui metodi di produzione; prima ancora, nel 1776, G. Monge aveva studiato un problema di trasporti. Tuttavia la nascita della Ricerca Operativa è storicamente legata agli studi che negli anni immediatamente precedenti alla Seconda Guerra Mondiale vennero condotti in Gran Bretagna per risolvere problemi strategici e tattici in operazioni militari. Piú in particolare questi studi erano legati all'uso efficiente di un nuovo strumento di difesa: il radar. Infatti nel 1937 la Royal Air Force iniziò degli esperimenti di un sistema di controllo della difesa aerea basato sull'uso di una stazione radar situata a Bawdsey Research Station, nella costa est; già dai primi esperimenti si resero conto che era molto difficile gestire efficientemente le informazioni provenienti dal radar. Nel luglio 1938 furono compiuti altri esperimenti con l'aggiunta di quattro stazioni radar lungo la costa nella speranza che il sistema di controllo migliorasse sia in copertura sia in efficienza; invece non fu così ; dai nuovi esperimenti emersero seri problemi: c'era la necessità di coordinare e correlare le tante informazioni, spesso anche in conflitto tra di loro, che venivano ricevute dalle stazioni radar aggiunte. Nell'imminenza della Guerra si rese necessario tentare qualche nuovo approccio; perciò il sovrintendente della Bawdsey Research Station propose di sviluppare un programma di ricerca che riguardasse gli aspetti *operativi* del sistema e non piú solamente quelli prettamente tecnici che erano da considerare soddisfacenti. Il termine *"Operational Research"* – Ricerca nelle operazioni (militari) – fu coniato per descrivere questa nuova branca delle scienze applicate. Fu quindi selezionato un gruppo di scienziati di vari discipline per costituire un "OR team"; il progetto fu diretto dal comandante in capo della Royal Air Force, Air Chief Marshal Sir Hugh Dowding. Nell'estate del 1939 la Gran Bretagna effettuò l'ultima esercitazione pre-bellica dove si evidenziò un notevole miglioramento nelle operazioni di difesa aerea grazie al contributo del gruppo di scienziati. Nacque quindi una vera e propria sezione che piú tardi, nel 1941, prese il nome formale di "Operational Research Section". Durante il conflitto

mondiale ci furono importanti contributi strategici di questa sezione che permisero di salvare piloti e aerei impegnati nel conflitto. Nonostante gli scopi bellici, anche se di difesa, del progetto, per la prima volta in questa occasione si ebbe una convergenza di scienziati di diverse discipline con l'obiettivo di determinare la più efficiente utilizzazione di risorse limitate usando tecniche quantitative.
Al termine della guerra, alcuni degli scienziati coinvolti nel progetto formarono nuclei di ricercatori per lo sviluppo post bellico e la loro attività si estese a campi diversi da quello militare; in particolare, con l'espandersi delle iniziative industriali e con l'avvento dei computer che sono uno strumento essenziale per la risoluzione dei problemi, c'è stata un'espansione dell'utilizzo della Ricerca Operativa all'interno di diverse realtà applicative.
Negli anni '60 le tecniche della Ricerca Operativa avevano avuto una buona diffusione, ma comunque il loro utilizzo era limitato esclusivamente alle imprese piú grandi visti gli altissimi costi dei calcolatori elettronici dell'epoca; piú tardi, con la diffusione dei personal computer c'è stata una diffusione sempre piú ampia della Ricerca Operativa in molti ambiti della vita reale.

1.3 LA RICERCA OPERATIVA OGGI

La necessità dell'uso dei metodi della Ricerca Operativa all'interno di molteplici situazioni del mondo reale è stata col passare degli anni sempre piú riconosciuta con una sempre maggiore e rapida espansione delle aree di possibile applicazione. In particolare, gli ambiti di maggiore sviluppo dell'applicazione Ricerca Operativa riguardano *problemi manageriali*, *problemi gestionali*, *problemi di progettazione*. Alcuni esempi di problemi possono essere affrontati per mezzo della Ricerca Operativa sono i seguenti:

- *Problemi in ambito industriale:*
  - *pianificazione della produzione*;
    si tratta di determinare i livelli di produzione e/o l'utilizzazione di risorse; si hanno spesso problemi di *allocazione ottima di risorse* cioè problemi riguardanti la distribuzione di risorse limitate tra alternative concorrenti in modo da minimizzare il costo complessivo o massimizzare il guadagno totale; tali risorse possono essere materie prime, manodopera, tempi di lavoro su macchine, capitali investiti.
  - *gestione ottima delle scorte*;
    si tratta di organizzare un magazzino nella gestione di materiali grezzi, prodotti in lavorazione etc.; cioè di decidere quando e quanto, durante un processo produttivo, si devono immagazzinare prodotti in modo da rispettare le consegne minimizzando i costi, oppure se e quando con-

viene riordinare materiali in modo da ottenere il miglior compromesso tra costi di acquisto, di produzione e di immagazzinamento.

- *localizzazione e dimensionamento di impianti*;
  sono problemi in cui si deve decidere dove istallare impianti di produzione in modo da rifornire in modo ottimale aree distribuite su un territorio, oppure decidere dove costruire le stazioni base di una rete di telecomunicazioni (GSM/UMTS) per coprire il territorio e con quale potenza esse devono trasmettere.

- *Problemi di progettazione ottima:*

  - *progettazione di reti e loro gestione*;
    si tratta di definire i collegamenti e dimensionare le capacità di una rete di telecomunicazione, di trasmissione dati, di circuiti, in modo da garantire il traffico tra le varie origini e destinazioni e minimizzare il costo complessivo;

  - *progettazione strutturale*;
    si tratta di problemi che nascono nell'ingegneria civile, industriale, nella meccanica aeronautica, etc. e hanno come scopo quello di definire un progetto di un edificio, di un ponte in modo che meglio resistano a sollecitazioni derivanti da vari agenti (terremoti, venti forti) oppure del profilo di un'ala di un aereo in modo che, ad esempio, sia massimizzata la portanza;

  - *progettazione di sistemi ottici, progettazione di robot*;
    si vuole ottenere un progetto che risponda a requisiti tecnici prefissati massimizzando alcuni parametri legati, ad esempio, alla precisione o alla prestazione;

  - *allocazione ottima di componenti elettronici* (VLSI design);
    si tratta di disegnare una piastra madre in modo che, ad esempio, siano minimizzate le lunghezze dei percorsi dei segnali elettrici;

- *Problemi di economia e finanza:*

  - *scelta di investimenti*;
    si deve scegliere fra un vasto numero di possibilità di investimento quali realizzare rispettando i vincoli imposti da un budget finanziario e massimizzando il guadagno;

  - *composizione di un portafoglio*;
    è il problema di decidere quali titoli e con quali quote investire capitali in modo da massimizzare il ricavo oppure minizzando il rischio;

- *Problemi di organizzazione:*
  - *project planning*;
    si tratta di decidere come gestire le risorse e come sequenziare le molteplici attività di un progetto;
  - *determinazione dei turni del personale*;
    si tratta di coprire una serie di servizi rispettando i vincoli di contratto aziendale e minimizzando i costi, come, ad esempio, l'assegnamento di personale viaggiante ai treni o degli equipaggi ai voli in modo da minimizzare il numero dei viaggi necessari per far tornare il personale nella propria sede;
  - *manutenzione di beni*;
    cioè il problema di decidere quando e se effettuare la manutenzione di alcuni oggetti soggetti ad usura con il tempo, in modo da minimizzare il costo complessivo.
  - *instradamento di veicoli*;
    si deve decidere quali percorsi devono seguire i veicoli di un flotta (ad esempio di automezzi adibiti alla raccolta dei rifiuti o alla distribuzioni di prodotti ad una rete di negozi) in modo da minimizzare la distanza complessiva percorsa;
- *Problemi scientifici:*
  - *studi sulla struttura del DNA*;
    si tratta di problemi legati alla determinazione della sequenze di geni minimizzando la probabilità di errore;
  - *ricostruzione di immagini*;
    è il problema della visualizzazione delle informazioni provenienti, ad esempio, da un satellite oppure da una tomografia computerizzata, in modo da ottenere un'immagine della migliore qualità possibile;
- *Problemi di diagnostica medica.*
  - *interpretazione e analisi dei dati ottenibili da strumenti di analisi clinica.*
- *Problemi di controllo ottimo:*
  - *controllo di servomeccanismi e di sistemi di guida*;
  - *controllo di traiettorie.*

Tuttavia i metodi della Ricerca Operativa sono oggi utilizzati anche in settori lontani dagli ambiti piú tradizionali come le *scienze sociali*, la *biologia*, le *scienze ambientali* e moltissimi altri.

Tuttavia, soprattutto in Italia, e soprattutto nelle realtà aziendali, gli strumenti utilizzati sono stati per anni assai rudimentali e spesso non adeguati alla crescente complessità dei sistemi di produzione. C'era spesso un notevole sforzo in termini sia finanziari sia umani per dotarsi di sistemi informativi all'avanguardia, ma raramente c'era un utilizzo di queste risorse per realizzare validi sistemi di supporto alle decisioni. Con il passare degli anni la consapevolezza dell'esigenza di tecniche quantitative per la gestione d'impresa è notevolmente cresciuta anche se non c'è ancora in certi settori una totale apertura verso l'utilizzo degli strumenti della Ricerca Operativa. Tuttavia, negli anni piú recenti, l'enorme sviluppo dei mezzi di calcolo e degli strumenti metodologici hanno portato a un grande successo della Ricerca Operativa soprattutto negli Stati Uniti. Il merito di questo successo è da ricondurre alla cosapevolezza ormai acquisita che l'incremento della potenza dei mezzi di calcolo non è certo sufficiente per risolvere tutti i problemi che si possono presentare. A confermare questo asserto si riassume di seguito un esempio dovuto a G. B. Dantzig[1] che è molto significativo: si supponga di essere a capo di un'azienda che ha 70 dipendenti e deve assegnare ciascuno di essi a 70 differenti mansioni; poiché le capacità lavorative di ogni singolo dipendente sono diverse, non è indifferente per l'azienda come effettuare l'assegnamento. Naturalmente si deve fare in modo che ciascun dipendente sia assegnato ad una sola mansione e che ciascuna mansione sia svolta esattamente da un dipendente. Il problema consiste nel confrontare le 70! possibilità che ci sono per selezionare quella migliore nel senso che permetta di ottenere il maggiore utile per l'azienda. Le possibilità sono un numero molto grande, più grande di 10^{100}. Ora si supponga di disporre di un calcolatore capace di effettuare un milione di calcoli al secondo e che sia in funzione dal tempo del Big Bang, 15 milioni di anni fà; avrebbe questo calcolatore oggi nell'anno 2000 esaminato tutte le 70! combinazioni possibili ? La risposta è no. Supponiamo allora di disporre di un calcolatore che possa effettuare un bilione di assegnamenti per ogni nano secondo; la risposta sarebbe ancora no. Supponiamo allora di riempire la superficie terrestre di calcolatori di questo tipo che lavorano in parallelo; la risposta sarebbe ancora no. Se si disponesse di 10^{40} terre ciascuna ricoperta di calcolatori di questo tipo che sono in funzione dal tempo del Big Bang fino a quando il sole si raffredderà; allora, forse, la risposta potrebbe essere sì!

Da questo esempio facile da enunciare si deduce come in certe situazioni sia assolutamente impossibile esaminare tutti i casi possibili per determinare qual è il migliore. Per questo, prima dell'avvento della Ricerca Operativa, l'unica possibilità era affidarsi al buon senso di persone guidate dall'esperienza che stabilivano regole "ad hoc" di base che dovevano essere seguite per risolvere i problemi (*"ad hoc" ground-rule approach*).

[1] G. B. Dantzig, Linear Programing, *Operations Research*, vol.50, No.1, 2002, pag.42–47

A questo approccio la Ricerca Operativa contrappone un approccio assai diverso: si tratta del cosiddetto *approccio modellistico*. Esso organizza l'analisi di un problema reale in due fasi:

- la rappresentazione del problema attraverso un *modello matematico* che ne astragga gli aspetti essenziali e che schematizzi le interrelazioni esistenti tra i diversi aspetti del fenomeno che si sta studiando;
- lo sviluppo di *metodi matematici efficienti* (algoritmi di soluzione) per determinare una soluzione ottima del problema o una sua buona approssimazione.

Naturalmente per costruire correttamente un modello matematico che rappresenti un particolare fenomeno, si devono distinguere i parametri di controllo significativi da quelli non essenziali, identificando un criterio per la valutazione della qualità della soluzione. Una volta determinato il modello corretto, la Ricerca Operativa si occupa di fornire una procedura esplicita per determinare una soluzione di un problema; tale procedura può essere rappresentata da metodi matematici analitici o, come più spesso accade, da metodi numerici che determinano la soluzione del problema mediante specifici algoritmi di calcolo.
In questo contesto, il merito maggiore della Ricerca Operativa consiste nello studiare un sistema nel suo complesso; infatti, la maggior parte dei problemi reali coinvolge diverse parti di un sistema mutuamente interagenti ed è quindi essenziale studiarne l'interazione reciproca. Questa è una caratteristica distintiva della Ricerca Operativa rispetto ad altre discipline ed è quindi evidente che un aspetto caratterizzante la Ricerca Operativa sia proprio l'interdisciplinarietà; ed infatti le tecniche di cui fa uso sono numerose e provengono da diverse branche della matematica: dall'algebra lineare alla logica, dalla statistica alla teoria dei giochi, dalla teoria delle decisioni alla teoria dei sistemi. Questo ha prodotto lo sviluppo di metodologie di soluzione che rappresentano un'inusuale combinazione di tecniche e strumenti tipici di altri settori.

1.4 L'APPROCCIO MODELLISTICO

L'approccio modellistico per risolvere un problema di decisione o, più in generale, l'impiego di metodi matematici per la soluzione di problemi applicativi, viene di solito realizzato attraverso diverse fasi. Tali fasi possono essere schematizzate nel seguente modo:

- **Analisi del problema**
- **Costruzione del modello**
- **Analisi del modello**

- **Soluzione numerica**
- **Validazione del modello**

La prima fase consiste nell'*analisi della struttura del problema* per individuare i legami logico-funzionali e gli obiettivi.
Nella succesiva fase di *costruzione del modello*, chiamata anche *formulazione*, si descrivono in termini matematici le caratteristiche principali del problema; questa fase di costruzione verrà descritta in dettaglio nel seguito.
Segue l'*analisi del modello* che prevede la deduzione per via analitica, in riferimento a determinate classi di problemi, di alcune importanti proprietà; le principali sono:

- *esistenza* ed *unicità* della soluzione ottima;
- *condizioni di ottimalità*, cioè una caratterizzazione analitica della soluzione ottima;
- *stabilità* delle soluzioni al variare dei dati o di eventuali parametri presenti.

La successiva fase di *soluzione* avviene mediante opportuni algoritmi di calcolo e la soluzione numerica così ottenuta deve poi essere interpretata dal punto di vista applicativo in modo da evitare che abbia scarso rilievo pratico; in questo caso le eventuali cause di inaccettabilità devono essere inglobate nel modello stesso costruendo così un nuovo modello più completo del precedente. Tale "validazione" del modello può avvenire attraverso una *verifica sperimentale* oppure con metodi di *simulazione*. La definizione di un modello si configura quindi come un processo di raffinamento iterativo, che può essere schematizzato come rappresentato in Figura 1.4.1.

1.5 MODELLI DELLA RICERCA OPERATIVA

Il primo passo dell'approccio modellistico consiste nel rappresentare un problema reale attraverso un *modello*; è utile, pertanto, chiarire subito cosa si intende con questo termine. Il termine *modello* è di solito usato per indicare una struttura appositamente costruita per mettere in evidenza le caratteristiche principali di alcuni oggetti reali. Alcune volte possono essere concreti (come ad esempio i modelli rappresentanti prototipi di aerei o auto), ma più spesso, come nella Ricerca Operativa, si tratta di *modelli astratti* cioè *modelli matematici* che usano il simbolismo dell'algebra per mettere in evidenza le relazioni principali dell'oggetto che deve essere modellato. I modelli di cui si tratterà in seguito sono quindi modelli matematici, e sono costituiti da un insieme di relazioni che descrivono in modo semplificato, ma sempre rigoroso, uno o più fenomeni del mondo reale. La nozione di modello matematico per rappresentare il mondo reale non è certo

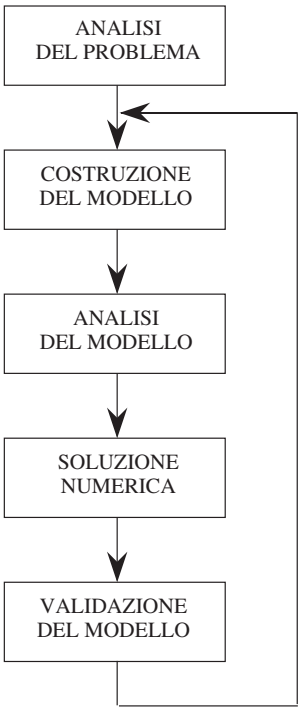


Fig. 1.4.1 Fasi dell'approccio modellistico

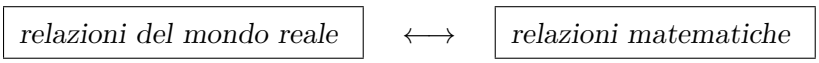
nuova: già Pitagora nel IV secolo a.C. tentava di costruire un modello matematico dell'Universo anche se sotto una luce più esoterica che scientifica. L'interesse per la modellistica matematica è notevolmente cresciuto negli anni più recenti e ai giorni nostri è sempre più viva la convinzione che ricorrendo a modelli matematici sia possibile analizzare i molteplici aspetti del mondo reale e studiare l'influenza che l'uomo può esercitare su di essi. Ciò ha portato ad un enorme sviluppo delle applicazioni della modellistica matematica anche al di fuori delle tradizionali applicazioni alle scienze fisiche. Si è così avuta di fatto una vasta utilizzazione di modelli matematici in settori lontani dagli ambiti più tradizionali come, ad esempio, le scienze sociali, la biologia, le scienze ambientali, la psicologia. Come esempi concreti, si pensi agli studi sulla dinamica della popolazione, sulla diffusione delle epidemie, sul risanamento ambientale. Questa notevole diffusione della modellistica matematica è anche dovuta al fatto che l'evoluzione di un modello matematico può essere rapidamente studiata grazie all'uso di moderni calcolatori elettronici.

È evidente come in molti casi le situazioni rappresentate da un modello sono molto complesse e alcune volte influenzate da fenomeni di natura aleatoria; per questa ragione, sono state definite diverse classi di modelli matematici: *modelli stocastici* che considerano grandezze che possono essere influenzate da fenomeni aleatori e *modelli deterministici* che considerano grandezze esatte; inoltre a seconda che le

interazioni tra le grandezze sono immediate o distribuite nel tempo, si parla di *modelli statici* e di *modelli dinamici*.
Nel seguito verranno analizzati i modelli deterministici che sono di fatto quelli più comunemente usati; in particolare si farà riferimento ai *modelli di programmazione matematica*[2] nei quali è esplicitamente definito un obiettivo da minimizzare o massimizzare ed in cui le variabili sono vincolate ad appartenere ad un insieme prefissato.

1.5.1 Costruzione di un modello matematico

L'approccio modellistico per risolvere un problema di decisione necessita come primo passo della costruzione di un adeguato modello matematico. Infatti, come già discusso in precedenza, solo un modello costruito tenendo presente tutte le caratteristiche essenziali del fenomeno che si sta studiando permette di comprendere gli aspetti più importanti e di esercitare un intervento pratico efficace.
Nella fase di costruzione del modello matematico si deve fornire una descrizione formalizzata del problema di decisione facendo uso del linguaggio formale della matematica. Si dovrà cercare, quindi, una corrispondenza tra relazioni del mondo reale (relazioni tecnologiche, leggi fisiche, vincoli di mercato, etc.) e relazioni matematiche (equazioni, disequazioni, dipendenze logiche, etc.).



La costruzione di un modello richiede, quindi, scelte e valutazioni in modo da evidenziare gli aspetti più significativi del problema reale e che meglio sono suscettibili di una formalizzazione matematica. Tale procedimento di scelta spesso non è riconducibile ad un procedimento sistematico e quindi è necessario che chi costruisce il modello abbia da un lato una conoscenza approfondita del settore applicativo per evitare che le risposte ottenute dal modello abbiano scarsa rilevanza pratica; dall'altro deve avere una notevole conoscenza dei metodi matematici disponibili per la ricerca della soluzione per evitare che la formulazione matematica porti ad un problema per il quale non esistono algoritmi risolutivi utilizzabili.
È importante ribadire che un modello è definito per mezzo delle relazioni che lo costituiscono ed è quindi necessario che tali relazioni siano il più possibile indipendenti dai dati introdotti nel modello; questo perché uno stesso modello deve poter essere usato in differenti occasioni con dati (cioè costi, disponibilità di risorse, limiti tecnologici, etc.) diversi. Lo studio di questo aspetto, come

[2]In questo contesto il termine *"programmazione"* è inteso nel senso di *"pianificazione"* e non di costruzione di programmi per il calcolatore.

già detto, rientra nella fase di analisi del modello sotto il nome di analisi della stabilità del modello rispetto ai dati introdotti.

1.5.2 Vantaggi dell'approccio modellistico

Le motivazioni che rendono molto utile la costruzione di un modello matematico sono molteplici; si riassumono di seguito le principali.

- *Possibilità di risolvere matematicamente il problema.*

  Grazie al modello è possibile analizzare matematicamente il problema ed ottenere così una soluzione che, soprattutto in riferimento a scopi di pianificazione, permette di adottare strategie che da una sola analisi strutturale del problema non apparirebbero evidenti o che a volte potrebbero essere perfino controintuitive.

- *Maggiore comprensione del problema.*

  Il modello è una rappresentazione semplificata del problema e spesso la sua costruzione consente di individuare proprietà strutturali del problema che altrimenti non sarebbero affatto evidenti.

- *Deduzione analitica di importanti proprietà.*

  Nella fase di analisi del modello è possibile dedurre per via analitica alcune importanti proprietà del problema sulla base dei risultati disponibili per la classe di problemi a cui si fa riferimento.

- *Possibilità di simulazioni.*

  Con un modello è possibile effettuare esperimenti che spesso non è possibile effettuare direttamente nella realtà; ad esempio, l'uso di un modello consente di studiare gli effetti dell'adozione di una particolare misura economica in un paese senza la necessità di sperimentarla direttamente.

1.5.3 Critiche all'approccio modellistico

Le principali critiche all'approccio modellistico e, quindi, alla costruzione di modelli per la soluzione di problemi di decisione possono essere sintetizzate nei seguenti due punti:

- Impossibilità di quantificare soddisfacentemente con opportuni valori numerici alcuni dati richiesti dal modello; questo accade, ad esempio, nel tentativo di quantificare con un costo o con un profitto alcuni valori sociali soprattutto in relazione a scopi di pianificazione.

- La qualità delle risposte che un modello produce potrebbero dipendere profondamente dall'accuratezza dei dati introdotti.

Il primo punto riguarda la possibilità di dover trattare concetti non facilmente quantificabili, ma ogni approccio scientifico può difficilmente evitare tale difficoltà; il modo migliore per superare tale problema consiste nell'incorporare tale quantificazione nel modello stesso.
La seconda critica riguarda la possibile mancanza di precisione di alcuni dei dati immessi nel modello; tale critica è meno rilevante della precedente, in quanto anche se alcuni dati introdotti sono poco accurati, è ancora possibile che la struttura del modello sia tale da garantire che la soluzione sia sufficientemente accurata.
All'estremo opposto di queste critiche si può collocare un atteggiamento di totale fiducia del modello che induca ad accettare la prima risposta prodotta dal modello senza ulteriori analisi. Tale atteggiamento, in realtà molto raro, è assai pericoloso in quanto tale risposta potrebbe rappresentare un piano operativo non accettabile nella realtà; in tal caso i motivi della non accettabilità devono essere evidenziati e incorporati in un nuovo modello modificato: si tratta, in realtà, della già citata fase di validazione del modello che quindi non può essere trascurata e che costituisce un valido mezzo per costruire modelli sempre più completi e significativi.
In conclusione, come spesso accade, l'atteggiamento corretto si colloca tra le due situazioni estreme precedentemente citate e consiste nel considerare la costruzione del modello un mezzo assai utile per affrontare un problema di decisione: rimane il fatto che la qualità delle risposte che un modello produce dipende dall'accuratezza della sua struttura e quindi non è trascurabile la fase di validazione che consente di interpretare la soluzione numerica ottenuta ed eventualmente permette di completare il modello introducendo elementi trascurati in una prima fase, in assenza dei quali la soluzione risulta non accettabile oppure di scarso rilievo dal punto di vista applicativo.

2

La Programmazione Matematica

All'interno della Ricerca Operativa, un ruolo di fondamentale importanza è svolto dalla *Programmazione Matematica* che è la disciplina che ha per oggetto lo studio dei problemi in cui si vuole minimizzare o massimizzare una funzione reale definita su $\mathbb{R}^n$ (lo spazio delle $n-$uple reali) le cui variabili sono vincolate ad appartenere ad una insieme prefissato. Si tratta quindi di *problemi di Ottimizzazione* cioè problemi nei quali si desidera minimizzare o massimizzare una quantità che è espressa attraverso una funzione.

2.1 PROBLEMI DI OTTIMIZZAZIONE

In termini generali, data una funzione $f: \mathbb{R}^n \longrightarrow \mathbb{R}$, ed $S \subseteq \mathbb{R}^n$, un *problema di Ottimizzazione* può essere formulato nella forma

$$\begin{cases} \min f(x) \\ x \in S. \end{cases} \qquad (PO)$$

Quindi un problema di Ottimizzazione consiste nel determinare, se esiste, un punto di minimo della funzione f tra i punti dell'insieme S.
Si parlerà indifferentemente di problemi di massimo o di minimo in quanto vale $\min_{x \in S} f(x) = -\max_{x \in S} (-f(x))$.
La funzione f viene chiamata *funzione obiettivo* e l'insieme S *insieme ammissibile* cioè l'insieme delle possibili soluzioni del problema. Un punto $x \in S$ si chiama *soluzione ammissibile.*
L'insieme ammissibile S è un sottoinsieme di $\mathbb{R}^n$ e quindi $x = (x_1, x_2, \ldots, x_n)^T$ è una variabile vettoriale n-dimensionale e la funzione obiettivo f è una funzione di n variabili reali $f(x_1, x_2, \ldots, x_n)$.

2.1.1 Definizioni fondamentali

Si riportano di seguito alcune definizioni fondamentali riguardanti i problemi di Ottimizzazione.

Definizione 2.1.1 *Il problema di ottimizzazione (PO) si dice* inammissibile *se* $S = \emptyset$*, cioè se non esistono soluzioni ammissibili.*

Definizione 2.1.2 *Il problema di ottimizzazione (PO) si dice* illimitato (inferiormente) *se comunque scelto un valore* $M > 0$ *esiste un punto* $x \in S$ *tale che* $f(x) < -M$

Definizione 2.1.3 *Si dice che il problema di ottimizzazione (PO)* ammette soluzione ottima (finita) *se esiste un* $x^\star \in S$ *tale che risulti* $f(x^\star) \leq f(x)$ *per ogni* $x \in S$*. Il punto* $x^\star$ *è detto* soluzione ottima *o* minimo globale *e il corrispondente valore* $f(x^\star)$ *di dice* valore ottimo*.*

Queste definizioni sono immediatamente estendibili al caso in cui un problema di Ottimizzazione è scritto in forma di massimizzazione.

2.1.2 Classificazione dei problemi di Ottimizzazione

All'interno dei problemi di Ottimizzazione, in base alla *struttura dell'insieme ammissibile* S, si possono distinguere le seguenti importanti classi di problemi:

- **Problemi di Ottimizzazione Continua.**

  Le variabili possono assumere tutti i valori reali ($x \in \mathbb{R}^n$); ed inoltre si parla di problemi di ottimizzazione continua

  - *vincolata* se $S \subset \mathbb{R}^n$
  - *non vincolata* se $S = \mathbb{R}^n$.

- **Problemi di Ottimizzazione Discreta.**

  Le variabili sono vincolate ad essere numeri interi ($x \in \mathbf{Z}^n$); si possono distinguere all'interno di questa classe di problemi altre due classi:

  - *programmazione a numeri interi* se $S \subseteq \mathbf{Z}^n$
  - *ottimizzazione booleana* se $S \subseteq \{0,1\}^n$.

- **Problemi misti.**

  Solo alcune delle variabili sono vincolate ad essere intere.

2.2 PROBLEMI DI PROGRAMMAZIONE MATEMATICA

Di solito l'insieme ammissibile S viene descritto da una numero finito di disuguaglianze del tipo $g(x) \geq b$, dove g è una funzione definita su $\mathbb{R}^n$ a valori reali e $b \in \mathbb{R}$. Cioè, formalmente, date m funzioni $g_i : \mathbb{R}^n \to \mathbb{R}$, $i = 1, \ldots, m$ ed m scalari $b_i \in \mathbb{R}$, $i = 1, \ldots, m$ si esprime S nella forma

$$S = \{x \in \mathbb{R}^n \ \mid \ g_1(x) \geq b_1, \quad g_2(x) \geq b_2, \quad \ldots \quad , \quad g_m(x) \geq b_m\}\,.$$

Ogni disuguaglianza $g_i(x) \geq b_i$ prende nome di *vincolo* e l'insieme ammissible è quindi formato da tutti quei punti $x \in \mathbb{R}^n$ che sono soluzione del sistema di disuguaglianze

$$\begin{cases} g_1(x) & \geq & b_1 \\ g_2(x) & \geq & b_2 \\ g_3(x) & \geq & b_3 \\ & \vdots & \\ g_m(x) & \geq & b_m \end{cases}$$

Osservazione 2.2.1 In questa formulazione dell'insieme S si sono utilizzati vincoli di disuguaglianza nella forma di maggiore o uguale, ma è chiaro che questa notazione include i casi in cui i vincoli sono espressi con vincoli di disuguaglianza nella forma di minore o uguale e vincoli di uguaglianza; infatti si può sempre trasformare un vincolo di minore o uguale del tipo $g(x) \leq b$ in un vincolo di maggiore o uguale semplicemente riscrivendolo nella forma $-g(x) \geq -b$. Inoltre un vincolo di uguaglianza $g(x) = b$ può essere riscritto nella forma equivalente delle due diseguaglianze $g(x) \geq b$ e $-g(x) \geq -b$.

Quindi, senza perdere di generalità, si può riscrivere il problema di ottimizzazione (PO) nella forma

$$\begin{cases} \min f(x) \\ g_i(x) \geq b_i, \quad i = 1, \ldots, m. \end{cases} \tag{2.2.1}$$

Un problema di questo tipo viene chiamato *problema di Programmazione Matematica*. I punti dell'insieme ammissibile di questo tipo di problemi sono quelli per i quali tutti i vincoli sono soddisfatti cioè tutti quei punti x tali che tutte le disuguaglianze $g_i(x) \geq b_i$, $i = 1, \ldots, m$ sono verificate.

I problemi di Programmazione Matematica si possono classificare in base alla *struttura delle funzioni che li definiscono*; in particolare si ha la seguente classificazione:

- **Problemi di Programmazione Lineare (PL)**
  La funzione obiettivo $f(x)$ e tutte le funzioni che definiscono i vincoli $g_i(x)$, $i = 1, \dots m$ sono *lineari*, cioè esprimibili nella forma $c_1x_1 + c_2x_2 + \cdots + c_nx_n$.
- **Problemi di Programmazione Non Lineare (PNL)**
  Almeno una delle funzioni che definiscono un problema di Programmazione Matematica *non è lineare*.

Si formalizzano nella definizione che segue alcuni semplici concetti riguardanti i vincoli di un problema di Programmazione Matematica.

Definizione 2.2.2 *Si consideri un vincolo di disuguaglianza del tipo $g(x) \geq b$, esso si dice:*

- violato *in un punto $\bar{x}$ se $g(\bar{x}) < b$;*
- soddisfatto *in un punto $\bar{x}$ se $g(\bar{x}) \geq b$;*
- attivo *in un punto $\bar{x}$ se $g(\bar{x}) = b$;*
- ridondante *se con la sua eliminazione l'insieme ammissibile rimane immutato.*

Alcuni esempi di problemi di Programmazione Matematica sono i seguenti:

Esempio 2.2.3 *Si consideri una funzione obiettivo di due variabili $f(x_1, x_2) = x_1 + x_2$ che si vuole minimizzare, con i vincoli $2x_1 + x_2 \geq 1, x_1 \geq 0, x_2 \geq 0$. Si ottiene il problema*

$$\begin{cases} \min x_1 + x_2 \\ 2x_1 + x_2 \geq 1 \\ x_1 \geq 0 \\ x_2 \geq 0 \end{cases}$$

che è nella forma (2.2.1) dove $g_1(x_1, x_2) = 2x_1 + x_2$, $g_2(x_1, x_2) = x_1$, $g_3(x_1, x_2) = x_2$, $b_1 = 1$, $b_2 = b_3 = 0$. L'insieme ammissibile è descritto attraverso questi tre vincoli e poiché tutte le funzioni che compaiono sono lineari nelle variabili x_1 e x_2, questo problema è un problema di Programmazione Lineare.

Esempio 2.2.4 *Si consideri una funzione obiettivo $f(x_1, x_2) = (x_1 - \frac{1}{2})^2 + (x_2 - \frac{1}{2})^2$ che si vuole massimizzare, con i vincoli $x_1 + x_2 \geq 1$, $x_1 \leq 1$, $x_2 \leq 1$. Si ottiene il problema*

$$\begin{cases} \max (x_1 - \frac{1}{2})^2 + (x_2 - \frac{1}{2})^2 \\ x_1 + x_2 \geq 1 \\ x_1 \leq 1 \\ x_2 \leq 1 \end{cases}$$

che è un problema di Programmazione Non Lineare (quadratico).

Esempio 2.2.5 *Si consideri una funzione obiettivo* $f(x_1, x_2) = 3x_1^3 + 7x_1^2 + x_2$ *che si vuole minimizzare, con vincoli* $x_1 + x_2 \leq \frac{1}{2}$, $x_1 \geq 0$, $x_2 \geq 1$. *Si ottiene il problema*

$$\begin{cases} \min 3x_1^3 + 7x_1^2 + x_2 \\ x_1 + x_2 \leq \frac{1}{2} \\ x_1 \geq 0 \\ x_2 \geq 1 \end{cases}$$

che è un problema di Programmazione Non Lineare che può essere facilmente ricondotto nella forma (2.2.1) riscrivendo il secondo vicolo nella forma $-x_1 - x_2 \geq -\frac{1}{2}$.

Esempio 2.2.6 *Si consideri una funzione obiettivo* $f(x_1, x_2) = x_1 + x_2$ *che si vuole minimizzare sulla regione ammissibile descritta dal vincolo di uguaglianza* $4x_1 - x_2 = -2$. *Il problema di Programmazione Lineare risultante è*

$$\begin{cases} \min x_1 + x_2 \\ 4x_1 - x_2 = -2 \end{cases}$$

che è un problema di Programmazione Lineare con un solo vincolo di uguaglianza.

Gli esempi appena visti, per semplicità, sono stati formulati come problemi in due variabili, in modo da permettere, fra l'altro, di comprenderne facilmente la loro struttura geometrica. Il significato geometrico di problemi di Programmazione Matematica verrà comunque trattato in dettaglio in seguito.

2.3 MODELLI DI PROGRAMMAZIONE MATEMATICA

I modelli standard più comunemente usati nella Ricerca Operativa sono i *modelli di Programmazione Matematica*, cioè modelli che possono essere rappresentati per mezzo di un problema di Programmazione Matematica. I settori applicativi all'interno dei quali sorgono problemi di questo tipo sono moltissimi: come esempi si possono citare problemi inerenti la pianificazione industriale, problemi di progettazione ottima, problemi di gestione di reti, problemi di economia e moltissimi altri.
Tuttavia, ogni lista di classi di modelli non può essere esaustiva: possono sempre presentarsi situazioni pratiche che non possono essere modellate in modo standard oppure che possono essere modellate in più di un modo standard.

La costruzione formale di un modello di Programmazione Matematica si effettua a partire da una descrizione logica e qualitativa di un problema di decisione e richiede di:

1. Associare opportune *variabili di decisione* alle grandezze reali. Tali variabili costituiscono le incognite del problema.
2. Esprimere formalmente l'*obiettivo* che si intende minimizzare o massimizzare.
3. Esprimere quantitativamente i *legami* esistenti tra le variabili e le *limitazioni* derivanti da considerazioni di carattere fisico, economico, etc. Tali legami e limitazioni definiscono i *vincoli*. L'insieme dei valori delle variabili per cui i vincoli sono soddisfatti costituisce l'*insieme ammissibile*.

A seconda della classe di problemi di Ottimizzazione entro la quale la formulazione del modello si colloca si parlerà di *modelli continui, modelli discreti, modelli misti*.

2.3.1 Esempi di modelli di Programmazione Matematica

Come primi esempi di costruzione di modelli verranno ora analizzati un semplice problema di pianificazione della produzione, un problema di pianificazione degli investimenti e un problema di progettazione industriale.

Esempio 2.3.1 *Un'industria chimica fabbrica 4 tipi di fertilizzanti,* **Tipo 1**, **Tipo 2**, **Tipo 3**, **Tipo 4**, *la cui lavorazione è affidata a due reparti dell'industria: il reparto produzione e il reparto confezionamento. Per ottenere fertilizzante pronto per la vendita è necessaria naturalmente la lavorazione in entrambi i reparti. La tabella che segue riporta, per ciascun tipo di fertilizzante i tempi (in ore) necessari di lavorazione in ciascuno dei reparti per avere una tonnellata di fertilizzante pronto per la vendita.*

	Tipo 1	**Tipo 2**	**Tipo 3**	**Tipo 4**
Reparto produzione	2	1.5	0.5	2.5
Reparto confezionamento	0.5	0.25	0.25	1

Dopo aver dedotto il costo del materiale grezzo, ciascuna tonnellata di fertilizzante dà i seguenti profitti (prezzi espressi in Euro per tonnellata)

	Tipo 1	**Tipo 2**	**Tipo 3**	**Tipo 4**
profitti netti	250	230	110	350

Determinare le quantità che si devono produrre settimanalmente di ciascun tipo di fertilizzante in modo da massimizzare il profitto complessivo, sapendo che ogni settimana, il reparto produzione e il reparto confezionamento hanno una capacità lavorativa massima rispettivamente di 100 e 50 ore.

Analisi del problema e costruzione del modello.

Si tratta di un problema di pianificazione della produzione industriale in cui le incognite, che saranno le variabili del problema, sono le quantità di fertilizzante di ciascun tipo che si devono produrre. Costruiamo un modello di Programmazione Matematica rappresentante il problema in analisi supponendo di voler pianificare la produzione settimanale.

– *Variabili di decisione.* È naturale introdurre le variabili reali x_1, x_2, x_3, x_4 rappresentanti rispettivamente le quantità di prodotto del **Tipo 1**, **Tipo 2**, **Tipo 3**, **Tipo 4** da fabbricare in una settimana.

– *Funzione Obiettivo.* Ciascuna tonnellata di fertilizzante contribuisce al profitto totale secondo la tabella data. Quindi il profitto totale sarà

$$250x_1 + 230x_2 + 110x_3 + 350x_4. \tag{2.3.1}$$

L'obiettivo dell'industria sarà quello di scegliere le variabili x_1, x_2, x_3, x_4 in modo che l'espressione (2.3.1) del profitto sia massimizzata. La (2.3.1) rappresenta la funzione obiettivo.

– *Vincoli.* Ovviamente la capacità produttiva della fabbrica limita i valori che possono assumere le variabili x_j, $j = 1, \ldots, 4$; infatti si ha una capacità massima lavorativa in ore settimanali di ciascun reparto. In particolare per il reparto produzione si hanno a disposizione al piú 100 ore settimanali e poiché ogni tonnellata di fertilizzante di **Tipo 1** utilizza il reparto produzione per 2 ore, ogni tonnellata di fertilizzante di **Tipo 2** utilizza il reparto produzione per 1.5 ore e così via per gli altri tipi di fertilizzanti si dovrà avere

$$2x_1 + 1.5x_2 + 0.5x_3 + 2.5x_4 \leq 100. \tag{2.3.2}$$

Ragionando in modo analogo per il reparto confezionamento si ottiene

$$0.5x_1 + 0.25x_2 + 0.25x_3 + x_4 \leq 50. \tag{2.3.3}$$

Le espressioni (2.3.2), (2.3.3) costituiscono i vincoli del modello. Si devono inoltre esplicitare vincoli dovuti al fatto che le variabili x_j, $j = 1, \ldots 4$ rappresentando quantità di prodotto non possono essere negative e quindi vanno aggiunti i vincoli di non negatività

$$x_1 \geq 0, \quad x_2 \geq 0, \quad x_3 \geq 0, \quad x_4 \geq 0.$$

Posto $x = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix}$, l'insieme ammissibile S sarà quindi così definito:

$$S = \left\{ x \in \mathbb{R}^4 \;\middle|\; \begin{array}{l} 2x_1 + 1.5x_2 + 0.5x_3 + 2.5x_4 \leq 100, \\ 0.5x_1 + 0.25x_2 + 0.25x_3 + x_4 \leq 50, \\ x_1 \geq 0, \quad x_2 \geq 0, \quad x_3 \geq 0, \quad x_4 \geq 0 \end{array} \right\}$$

La formulazione finale quindi può essere scritta in questa forma

$$\begin{cases} \max\left(250x_1 + 230x_2 + 110x_3 + 350x_4\right) \\ 2x_1 + 1.5x_2 + 0.5x_3 + 2.5x_4 \leq 100 \\ 0.5x_1 + 0.25x_2 + 0.25x_3 + x_4 \leq 50 \\ x_1 \geq 0, x_2 \geq 0, x_3 \geq 0, x_4 \geq 0. \end{cases}$$

□

Esempio 2.3.2 – CAPITAL BUDGETING. *Supponiamo di dover investire £1000 sul mercato finanziario. Supponiamo inoltre che il mercato offra tre tipi diversi di investimenti* $\mathbf{A}, \mathbf{B}, \mathbf{C}$ *ciascuno caratterizzato da un prezzo d'acquisto e da un rendimento netto, che sono riassunti nella seguente tabella:*

	A	**B**	**C**
costo	*750*	*200*	*800*
rendimento	*20*	*5*	*10*

Si vuole decidere quali degli investimenti effettuare per massimizzare il rendimento sapendo che gli investimenti **A**, **B**, **C** *non si possono effettuare in modo parziale cioè non sono frazionabili.*

Analisi del problema e costruzione del modello.

Si tratta di un problema di pianificazione degli investimenti. Si devono definire formalmente le variabili di decisione, l'insieme delle soluzioni ammissibili e la funzione obiettivo.

– *Variabili di decisione.* Si tratta quindi di esprimere matematicamente la scelta elementare: effettuare o non effettuare l'investimento. Una scelta naturale delle variabili di decisione è la seguente:

$$x_i = \begin{cases} 0 & \text{non si effettua l'investimento } i\text{−esimo} \\ 1 & \text{si effettua l'investimento } i\text{−esimo} \end{cases} \qquad i = \mathbf{A}, \mathbf{B}, \mathbf{C} \qquad (2.3.4)$$

– *Insieme ammissibile.* In base alla definizione delle variabili, le possibili scelte compatibili con il nostro budget sono:

(0) non si effettuano investimenti $x_A = x_B = x_C = 0$

(1) si effettua l'investimento **A**; $x_A = 1, x_B = x_C = 0$

(2) si effettua l'investimento **B**; $x_A = 0, x_B = 1, x_C = 0$

(3) si effettua l'investimento **C**; $x_A = x_B = 0, x_C = 1$

(4) si effettuano gli investimenti **A** e **B**; $x_A = x_B = 1, x_C = 0$

(5) si effettuano gli investimenti **B** e **C**; $x_A = 0, x_B = x_C = 1$.

Notiamo che le possibilità **A**, **C** e **A**, **B**, **C** non sono ammissibili in quanto il costo supera la nostra disponibilità. L'insieme ammissibile, ovvero l'insieme delle possibili scelte $(0) - (5)$ è dato da:

$$S = \left\{ \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}, \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix} \right\}$$

Si tratta quindi di un sottoinsieme dei vettori di $\mathbb{R}^3$ a componenti 0–1 ovvero $S \subseteq \{0,1\}^3$.

– *Funzione obiettivo.* L'obiettivo che ci proponiamo è la massimizzazione del rendimento totale. Quindi dobbiamo esprimere la funzione obiettivo che corrisponde al rendimento netto relativo alla scelta di $x = \begin{pmatrix} x_A \\ x_B \\ x_C \end{pmatrix}$ in S, cioè $f(x) = 20x_A + 5x_B + 10x_C$.

È possibile ottenere la soluzione ottima valutando esaustivamente la funzione obiettivo per ogni elemento di S, ottenendo in relazione alle possibili scelte:

(0) $f_0 = 0$

(1) $f_1 = 20$

(2) $f_2 = 5$

(3) $f_3 = 10$

(4) $f_4 = 25$

(5) $f_5 = 15$.

La soluzione ottima è ovviamente quella corrispondente alla scelta (4), cioè all'effettuare gli investimenti **A** e **B**, con valore della funzione obiettivo pari a £25.

Questo *non è un modello corretto* per due motivi:

1. *L'insieme ammissibile S è rappresentato in modo estensivo*, cioè elencando tutte le soluzioni ammissibili. In questo caso la cardinalità dell'insieme ammissibile è al più quella di $\{0,1\}^3$ cioè 2^3, ma in generale, se la dimensione del problema fosse piú grande sarebbe impossibile valutare esaustivamente le soluzioni del problema. Se, ad esempio, il numero degli investimenti fosse stato 100 (che dal punto di vista delle applicazioni reali è del tutto verosimile) la cardinalità dell'insieme ammissibile sarebbe stata 2^{100} e per la valutazione di 2^{100} possibilità anche supponendo di utilizzare un calcolatore che effettui 10^{10} valutazioni al secondo (velocità superiore a quella raggiungibile dai calcolatori attuali) occorrerebbero 10^{20} secondi, cioè 3000 miliardi di anni !

2. *Il modello non è indipendente dai dati del problema*, cioè cambiando i dati del problema (prezzi e/o rendimenti) sarebbe necessario cambiare completamente il modello.

In generale, in un modello corretto, si cerca di dare una *rappresentazione intensiva* dell'insieme ammissibile S, cioè individuare le proprietà $P(x)$ che consentono di distinguere le soluzioni ammissibili dagli elementi dell'insieme $\{0,1\}^3$ che non lo sono. Si vuole quindi scrivere l'insieme S in una forma del tipo:

$$S = \left\{ x \in \{0,1\}^3 : \text{ vale la proprietà } \ P(x) \right\}.$$

Nell'esempio, la proprietà distintiva degli elementi di S è il costo complessivo che non deve essere superiore a £1000. Possiamo esprimere matematicamente questa relazione come:

$$P(x) : \ 750x_A + 200x_B + 800x_C \leq 1000$$

e quindi l'insieme ammissibile si può scrivere

$$S = \left\{ x = \begin{pmatrix} x_A \\ x_B \\ x_C \end{pmatrix} \in \{0,1\}^3 \ \middle| \ 750x_A + 200x_B + 800x_C \leq 1000 \right\}.$$

In conclusione, il modello matematico corretto per il problema di decisione in esame è:

$$\begin{cases} \max\left(20x_A + 5x_B + 10x_C\right) \\ 750x_A + 200x_B + 800x_C \leq 1000 \\ x_i \in \{0,1\} \quad i = \mathbf{A}, \mathbf{B}, \mathbf{C}. \end{cases}$$

□

Esempio 2.3.3 *Un'industria deve costruire un silos di forma cilindrica per contenere grandi quantitativi di un liquido che verrà poi distribuito in piccole confezioni pronte per la vendita al minuto. Tale silos deve essere posto in un magazzino appoggiato su una delle basi. Tale magazzino è a pianta rettangolare di dimensioni metri* 20×10 *ed ha un tetto spiovente lungo il lato di 10 metri, che ha altezza massima di metri* 5 *e altezza minima di metri* 3*. Per costruire questo silos deve essere usato del materiale plastico sottile flessibile che può essere tagliato, modellato e incollato saldamente. Sapendo che si dispone di non più di* 200 m^2 *di tale materiale plastico si costruisca un modello che permetta di determinare le dimensioni del silos (raggio di base ed altezza) in modo da massimizzare la quantità di liquido che può esservi contenuto.*

Analisi del problema e costruzione del modello.

Si tratta di determinare il dimensionamento ottimale di un contenitore cilindrico per uso industriale cercando di massimizzare il suo volume tenendo presente che deve essere contenuto in un magazzino di dimensioni fissate. Si devono innanzitutto definire formalmente le variabili di decisione, l'insieme delle soluzioni ammissibili e la funzione obiettivo.

– *Variabili di decisione.* È immediato introdurre due variabili x_1 e x_2 che rappresentano rispettivamente la lunghezza (in metri) del raggio di base e dell'altezza del contenitore cilindrico.

– *Funzione obiettivo.* La funzione obiettivo è rappresentata dal volume del contenitore cilindrico ed è data da

$$\pi x_1^2 x_2.$$

– *Vincoli.* Il diametro della base non può superare le dimensioni del magazzino e quindi deve essere

$$2x_1 \leq 10.$$

La limitazione dell'altezza del contenitore varia al variare del diametro di base in quanto il tetto è spiovente. Dato che la pendenza del tetto è del 20%, dovrà risultare

$$x_2 \leq 5 - 0.2 \cdot 2x_1.$$

Inoltre disponendo solo di una quantità limitata di materiale plastico la superficie totale del contenitore cilindrico non può superare $200m^2$ e quindi deve risultare

$$2\pi x_1^2 + 2\pi x_1 x_2 \leq 200.$$

Si devono infine esplicitare i vincoli di non negatività $x_1 \geq 0$, $x_2 \geq 0$. Quindi l'insieme ammissibile è

$$S = \left\{ \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \in \mathbb{R}^2 \;\middle|\; \begin{array}{c} x_1 \leq 5, \quad x_2 \leq 5 - 0.2 \cdot 2x_1, \quad 2\pi x_1^2 + 2\pi x_1 x_2 \leq 200, \\ x_1 \geq 0, \quad x_2 \geq 0 \end{array} \right\}$$

La formulazione complessiva risulta quindi

$$\begin{cases} \max \ \pi x_1^2 x_2 \\ x_1 \leq 5 \\ x_2 \leq 5 - 0.2 \cdot 2x_1 \\ 2\pi x_1^2 + 2\pi x_1 x_2 \leq 200 \\ x_1 \geq 0, \quad x_2 \geq 0. \end{cases}$$

□

Osservazione 2.3.4 Negli Esempi 2.3.1 e 2.3.2 ora analizzati, sia la funzione obiettivo sia i vincoli sono rappresentati attraverso espressioni *lineari* nelle variabili di decisione. Quindi questi modelli hanno una forma particolare che, in generale prende nome di *Modello di Programmazione Lineare*, (PL). Questa classe di modelli è molto importante e sarà la classe di problemi che tratteremo nel seguito.

Osservazione 2.3.5 Nell'Esempio 2.3.1 abbiamo assunto che le variabili di decisione potessero assumere valori reali e quindi, in particolare, frazionari. Tale

assunzione potrebbe essere vera nel caso in cui per quantità di prodotto si intenda una misura, ad esempio in litri, quintali, o altra quantità frazionabile di prodotto. Altrimenti se tale quantità rappresenta, ad esempio il numero di motori per automobile, allora le variabili x_j che danno la misura di questa quantità devono assumere *valori interi*. In tal caso, sempre nell'ipotesi che il modello sia lineare, si parla di *Modello di Programmazione Lineare Intera* (PLI). Questo è anche il caso del modello dell'Esempio 2.3.2.

Osservazione 2.3.6 A differenza degli Esempi 2.3.1 e 2.3.2, nell'Esempio 2.3.3 sia la funzione obiettivo, sia uno dei vincoli sono rappresentati attraverso espressioni *non lineari* nelle variabili di decisione. In questo caso si parla di *Modello di Programmazione Non Lineare* (PNL). La presenza di espressioni non lineari in un modello di programmazione matematica è piuttosto frequente: si pensi, ad esempio, ad una generica situazione in cui il profitto unitario che si ricava dalla vendita di un prodotto varia al variare della quantità dei prodotti venduti fino a quel momento; nella realtà, in accordo ad elementari leggi di mercato, accade molto spesso che il prezzo unitario di un prodotto possa aumentare se cresce la richiesta e quindi se una variabile x rappresenta la quantità di prodotto venduto e $p(x)$ il prezzo di vendita (dipendente da x), il profitto che si ricava dalla vendita di x prodotti sarà $p(x)x$; il temine $p(x)$ introduce una non linearità nella funzione obiettivo. Come esempio di ciò, riferendoci all'Esempio 2.3.1, se avessimo supposto che il prezzo unitario di vendita del prodotto **P1** fosse $250 + 3x_1$ cioè fosse dipendente dalla quantità di prodotto venduto x_1 il contributo al profitto complessivo dato dalla vendita di x_1 prodotti **P1** sarebbe stato $(250 + 3x_1)x_1$. Verrebbe così introdotta una non linearità data dal termine $3x_1^2$. Anche in questo caso in cui la sola funzione obiettivo è non lineare ed i vincoli continuano ad essere lineari, si parla di modelli di Programmazione Non Lineare. Tuttavia i modelli non lineari sono di solito molto più difficili da risolvere e quindi molto spesso si cerca di approssimarli con modelli lineari.

3

Modelli di Programmazione Lineare

3.1 GENERALITÀ

Come già detto nel capitolo precedente, è possibile classificare i modelli di Programmazione Matematica in base alla struttura particolare che possono avere la funzione obiettivo e i vincoli. Riprendiamo qui, espandendola, la definizione di *problemi di Programmazione Lineare* nei quali sia la funzione obiettivo, sia i vincoli sono rappresentati mediante funzioni lineari nelle variabili di decisione.

Preliminarmente, richiamiamo il concetto di *funzione lineare*.

Definizione 3.1.1 *Una funzione reale di n variabili reali* $f: \mathbb{R}^n \longrightarrow \mathbb{R}$ *si dice* lineare *se valgono le seguenti condizioni:*

i) *per ogni* $x, y \in \mathbb{R}^n$ *si ha* $\qquad f(x+y) = f(x) + f(y)$*;*

ii) *per ogni* $x \in \mathbb{R}^n$ *e* $\lambda \in \mathbb{R}$ *risulta* $\qquad f(\lambda x) = \lambda f(x)$*.*

Una immediata conseguenza di questa definizione è che una funzione è lineare se e solo se può essere scritta nella forma

$$c_1 x_1 + c_2 x_2 + \ldots + c_n x_n \tag{3.1.1}$$

con $c_1, \ldots, c_n$ costanti reali. Infatti è immediato verificare che una funzione della forma (3.1.1) soddisfa la Definizione 3.1.1; d'altra parte, se una funzione $f(x)$ è lineare cioè se soddisfa la Definizione 3.1.1, allora si può scrivere nella forma (3.1.1); infatti se indichiamo con $\{e_1, e_2, \ldots, e_n\}$ la base canonica di $\mathbb{R}^n$ allora

risulta $x = \sum_{i=1}^{n} x_i e_i$ dove le x_i sono le componenti del vettore x. Quindi utilizzando la linearità si ha

$$\begin{aligned} f(x) &= f(x_1e_1 + x_2e_2 + \cdots + x_ne_n) = f(x_1e_1) + f(x_2e_2) + \cdots + f(x_ne_n) = \\ &= x_1f(e_1) + x_2f(e_2) + \cdots x_nf(e_n) = c_1x_1 + c_2x_2 + \cdots + c_nx_n \end{aligned}$$

dove $c_i = f(e_i)$ per $i = 1, \ldots, n$. □

Quindi

$$x_1 + 4x_2 - 3.5x_3$$
$$-2x_1 + (\sin 4)x_2 + \pi x_3 - 4x_5,$$

sono funzioni lineari, mentre

$$(x_1)^2 + 4x_2 - 3.5x_3$$
$$x_1 + 4x_2 - 3.5e^{x_3}$$
$$-2x_1 + \sin x_2 + \pi x_3 - 4x_5,$$

non sono funzioni lineari.

3.2 STRUTTURA DI UN MODELLO DI PROGRAMMAZIONE LINEARE

Esaminiamo ora la struttura di un generico modello di Programmazione Lineare. Un modello di Programmazione Lineare è caratterizzato da

- una singola *funzione obiettivo lineare* da minimizzare o massimizzare che può essere quindi scritta nella forma

$$f(x) = c_1x_1 + \ldots + c_nx_n = \sum_{j=1}^{n} c_jx_j.$$

- un numero finito di *vincoli lineari* che, supponendo siano m, possono essere scritti nella forma

$$\begin{array}{cccc} a_{11}x_1+ & \ldots & +a_{1n}x_n & \geq b_1 \\ a_{21}x_1+ & \ldots & +a_{2n}x_n & \geq b_2 \\ \vdots & \ldots & \vdots & \vdots \\ a_{m1}x_1+ & \ldots & +a_{mn}x_n & \geq b_m. \end{array}$$

Introducendo il vettore $c \in \mathbb{R}^n$, definito $c = (c_1, \ldots, c_n)^T$ e $x \in \mathbb{R}^n$ definito $x = (x_1, \ldots, x_n)^T$ la funzione obiettivo può essere scritta in notazione vettoriale

$$c^T x.$$

Inoltre, introducendo la matrice $(m \times n)$

$$A = \begin{pmatrix} a_{11} & \dots & a_{1n} \\ \vdots & & \vdots \\ a_{m1} & \dots & a_{mn} \end{pmatrix}$$

e il vettore $b = (b_1, \dots, b_m)^T$ la formulazione completa di un generico problema di Programmazione Lineare può essere scritta nella forma

$$\begin{cases} \min c^T x \\ Ax \geq b. \end{cases}$$

Osservazione 3.2.1 Come già osservato in relazione ad un generico problema di Programmazione Matematica, (cfr. Osservazione 2.2.1) non si perde di generalità formulando un generico problema di Programmazione Lineare con vincoli di sola diseguaglianza nella forma di maggiore o uguale. Infatti, ogni vincolo di disuguaglianza nella forma di minore o uguale e ogni vincolo di uguaglianza può essere ricondotto a questa forma con semplici operazioni algebriche.

Per esempio,

$$\begin{array}{lll} \max & x_1 + x_2 & \\ & x_1 + x_2 & \geq \ 1 \\ & x_1 + x_2 & \leq \ 3 \\ & x_1 \geq 0, \ \ x_2 \geq 0, & \end{array}$$

e

$$\begin{array}{lll} \min & 2x_1 - x_2 + x_3 + 3x_4 & \\ & x_1 + x_2 - x_4 & = \ 1 \\ & x_1 + 2x_2 - x_3 + 2x_4 & \leq \ 3 \\ & x_1 \geq 0, \ \ x_2 \geq 0, \ \ x_4 \geq 0, & \end{array}$$

sono problemi di PL.

Le applicazioni della Ricerca Operativa che possono essere formulate mediante l'uso di modelli di Programmazione Lineare sono molto frequenti e importanti. In riferimento alle applicazioni di tipo economico la funzione obiettivo ha di solito il significato di profitto (da massimizzare) oppure di costo (da minimizzare). Profitti e costi sono ottenuti come somma dei *profitti* e *costi marginali* cioè di quelli relativi a ciascuna unità di prodotto. Quando è richiesta la massimizzazione di un profitto, il modello contiene, di solito, vincoli che esprimono limitazioni superiori sulle risorse (vincoli di capacità produttiva, disponibilità di materie prime); se invece è richiesta la minimizzazione di un costo sono di solito presenti vincoli sulla domanda (richieste di mercato) che impongono limitazioni inferiori alle variabili.

È possibile la presenza di *vincoli di continuità* che esprimono conservazione o riduzione di masse o volumi ed hanno spesso la forma di vincoli di uguaglianza.

I modelli di Programmazione Lineare hanno un impiego molto generale non limitato ad applicazioni economiche o progettuali; ad esempio, essi sono usati come elementi base di procedimenti di soluzione di problemi piú complessi: è il caso di alcuni algoritmi di ottimizzazione discreta che sono basati sulla soluzione di una successione di problemi di Programmazione Lineare.

3.3 CONSIDERAZIONI GENERALI SUI MODELLI DI PROGRAMMAZIONE LINEARE

Mettiamo ora in evidenza le caratteristiche che un problema reale deve possedere per poter essere formulato come modello di Programmazione Lineare ed i pregi dei modelli di Programmazione Lineare.
Innanzitutto, chiariamo che le ipotesi che vengono assunte nel formulare un problema come modello di Programmazione Lineare sono le seguenti:

- *proporzionalità*: il contributo di una variabile di decisione alla funzione obiettivo e ai vincoli è proporzionale secondo una costante moltiplicativa alla quantità rappresentata dalla variabile stessa;
- *additività*: il contributo delle variabili di decisione alla funzione obiettivo e ai vincoli è dato dalla somma dei contributi di ogni singola variabile.
- *continuità*: ogni variabile di decisione può assumere tutti i valori reali nell'intervallo di ammissibilità, e quindi le variabili possono assumere valori frazionari.

In relazione ad applicazioni reali queste ipotesi non rappresentano una grossa restrizione nel senso che sono molti gli ambiti e i problemi che sono ben rappresentati da un modello di Programmazione Lineare; si tenga comunque presente che esistono casi significativi in cui queste ipotesi non sono soddisfatte e quindi in questi casi è necessario considerare Modelli di Programmazione Non Lineare.

La particolare attenzione dedicata ai modelli di Programmazione Lineare deriva, comunque, dai numerosi vantaggi che essa presenta e che possono essere così sintetizzati:

1. *Generalità e flessibilità.*

   I modelli di Programmazione Lineare possono descrivere moltissime situazioni reali anche assai diverse tra loro e quindi hanno un carattere di universalità e di adattabilità alle diverse realtà applicative e anche quando l'ipotesi di linearità non è accettabile, il modello lineare costituisce una buona base di partenza per successive generalizzazioni.

2. *Semplicità.*

   I modelli di Programmazione Lineare sono espressi attraverso il linguaggio dell'algebra lineare e quindi sono facilmente comprensibili anche in assenza di conoscenze matematiche piú elevate.

3. *Efficienza degli algoritmi risolutivi.*

   Come accennato in precedenza i modelli reali hanno dimensioni molto elevate ed è quindi indispensabile l'uso del calcolatore che con opportuni programmi di calcolo possa rapidamente fornire una soluzione numerica. Relativamente ai modelli di Programmazione Lineare esistono programmi molto efficienti e largamente diffusi che sono in grado di risolvere rapidamente problemi con migliaia di vincoli e centinaia di migliaia di variabili.

4. *Possibilità di analisi qualitative.*

   I modelli di Programmazione Lineare permettono di ottenere, oltre la soluzione numerica del problema, anche ulteriori informazioni relative alla dipendenza della soluzione da eventuali parametri presenti, che possono avere significative interpretazioni economiche.

3.4 CLASSI DI MODELLI DI PROGRAMMAZIONE LINEARE

Lo scopo di questo paragrafo è quello di illustrare alcune classi di problemi di Programmazione Lineare tipici che si incontrano frequentemente nelle applicazioni reali. Questa divisione in classi ha uno scopo esclusivamente didattico al fine di fornire una esposizione sistematica di esempi di modelli di Programmazione Lineare di tipo generale. Nella realtà, nella maggior parte dei casi, i problemi che si presentano non sono riconducibili ad una classe specifica, ma possono essere costituiti da molteplici elementi. Tuttavia, la trattazione per grandi classi di problemi dovrebbe fornire strumenti utili per la modellizzazione di problemi reali. Tenendo presente questa osservazione, nel seguito esaminreremo tre grandi classi di modelli di Programmazione Lineare che rappresentano situazioni molto diffuse del mondo reale; si tratta dei

- *modelli di allocazione ottima di risorse*,
- *modelli di miscelazione*,
- *modelli di trasporto.*

Per ciascuna classe di modelli verranno presentati alcuni esempi e una formulazione generale.
Consigliamo lo studente, al fine di acquisire una sufficiente abilità nella formulazione di problemi di Programmazione Lineare, di provare da solo a formulare i problemi descritti prima di leggere la formulazione fornita.

3.4.1 Modelli di allocazione ottima di risorse

Si tratta di modelli che considerano il problema di come dividere (allocare) risorse limitate tra varie esigenze in competizione fra di loro. Il generico termine *"risorse"* può rappresentare, ad esempio, disponibilità di macchinari, materie prime, mano d'opera, energia, tempi macchina, capitali, etc.

Esempio 3.4.1 *Un colorificio produce due tipi di coloranti* **C1** *e* **C2** *utilizzando 3 preparati base in polvere* **P1**, **P2**, **P3** *che vengono sciolti in acqua. La differente concentrazione dei preparati base dà origine ai due diversi tipi di coloranti. Le quantità (in ettogrammi) di preparati base necessarie per produrre un litro di colorante di ciascuno dei due tipi è riportato nella seguente tabella*

	C1	**C2**
P1	*1*	*1*
P2	*1*	*2*
P3	-	*1*

Ogni giorno la quantità di ciascuno dei preparati base (in ettogrammi) della quale il colorificio può disporre è la seguente

P1	**P2**	**P3**
750	*1000*	*400*

Il prezzo di vendita del colorante **C1** *è di 7 Euro al litro, mentre il colorante* **C2** *viene venduto a 10 Euro al litro. Determinare la strategia ottimale di produzione giornaliera in modo da massimizzare i ricavi ottenuti dalla vendita dei due coloranti.*

Formulazione.
Si vuole costruire il modello di Programmazione Lineare che rappresenti il problema in analisi considerando le limitazioni date dalle produzioni effettivamente realizzabili.
È immediato associare le variabili di decisione ai quantitativi di coloranti prodotti. Siano, quindi, rispettivamente x_1 e x_2 i quantitativi (in litri) da produrre giornalmente dei due coloranti.
Nel formulare il modello di Programmazione Lineare si deve verificare che siano soddisfatte le ipotesi fondamentali:

- *Proporzionalità.*

  I consumi dei preparati base e i ricavi ottenibili sono proporzionali ai quantitativi di coloranti prodotti. Ad esempio, per produrre una quantità x_2 di colorante **C2** si consumano $2x_2$ ettogrammi di **P2** e dalla vendita di x_2 litri

di **C2** si ricavano $10x_2$ Euro indipendentemente dalla quantità prodotta e venduta dell'altro tipo di colorante.

- *Additività.*

  I consumi dei preparati base e i ricavi rispettivamente associati alla produzione dei due coloranti sono additivi, nel senso che per produrre x_1 litri di colorante **C1** e x_2 di **C2** si consumano $x_1 + 2x_2$ ettogrammi di preparato di base **P2** e si ricavano $7x_1 + 10x_2$ Euro.

- *Continuità.*

  Ogni variabile introdotta nel modello può assumere tutti i valori reali nell'intervallo di ammissibilità.

– *Variabili.* Come già detto, prendiamo come variabili di decisione x_1 e x_2, rispettivamente i quantitativi (in litri) di colorante **C1** e **C2** da produrre giornalmente.
– *Funzione obiettivo.* È rappresentata dal profitto totale che per le ipotesi fatte è dato (in Euro) da $7x_1 + 10x_2$.
– *Vincoli.* Poiché il consumo di preparati base non può essere superiore alla disponibilità si deve avere

$$\begin{aligned} x_1 + x_2 &\leq 750 \\ x_1 + 2x_2 &\leq 1000 \\ x_2 &\leq 400. \end{aligned}$$

Inoltre si deve esplicitare il vincolo di non negatività sulle variabili

$$x_1 \geq 0, x_2 \geq 0.$$

Quindi la formulazione finale è

$$\begin{cases} \max\left(7x_1 + 10x_2\right) \\ x_1 + x_2 \leq 750 \\ x_1 + 2x_2 \leq 1000 \\ x_2 \leq 400 \\ x_1 \geq 0, x_2 \geq 0. \end{cases}$$

□

Esempio 3.4.2 *Una azienda automobilistica produce tre diversi modelli di autovettura: un modello economico, uno normale ed uno di lusso. Ogni autovettura viene lavorata da tre robot:* **A**, **B** *e* **C**. *I tempi necessari alla lavorazione sono riportati, in minuti, nella tabella seguente insieme al profitto netto realizzato per autovettura*

	Economica	**Normale**	**Lusso**
A	*20*	*30*	*62*
B	*31*	*42*	*51*
C	*16*	*81*	*10*
Prezzo	*1000*	*1500*	*2200*

I robot **A** *e* **B** *sono disponibili per 8 ore al giorno mentre il robot* **C** *è disponibile per 5 ore al giorno. Il numero di autovetture di lusso prodotte non deve superare il 20% del totale mentre il numero di autovetture economiche deve costituire almeno il 40% della produzione complessiva. Supponendo che tutte le autovetture prodotte vengano vendute, formulare un problema di Programmazione Lineare che permetta di decidere le quantità giornaliere (non necessariamente intere) da produrre per ciascun modello in modo tale da massimizzare i profitti rispettando i vincoli di produzione.*

Formulazione.
È un problema di allocazione ottima di risorse e può essere formulato in termini di Programmazione Lineare nel seguente modo.

– *Variabili.* Indichiamo con x_1, x_2, x_3, rispettivamente il numero di autovetture (assunte non necessariamente intere) del modello economico, normale e di lusso da produrre giornalmente.

– *Funzione obiettivo.* La funzione obiettivo è data dal profitto globale ottenuto dalla vendita delle automobili e quindi può essere scritta

$$1000x_1 + 1500x_2 + 2200x_3.$$

– *Vincoli.* Ci sono due tipologie di vincoli da considerare:

- i vincoli sulla capacità produttiva; poiché il robot **A** è disponibile giornalmente per 8 ore, cioè per 480 minuti si ha il vincolo

$$20x_1 + 30x_2 + 62x_3 \leq 480.$$

Ragionando in modo analogo si ottengono i vincoli relativi alla disponibilità dei robot **B** e **C**, e quindi si ottengono i seguenti vincoli:

$$31x_1 + 42x_2 + 51x_3 \leq 480$$

$$16x_1 + 81x_2 + 10x_3 \leq 300.$$

- i vincoli sul numero totale dei singoli tipi di autovetture da fabbricate giornalmente che possono essere scritti nella forma

$$x_3 \leq 0.2\,(x_1 + x_2 + x_3)$$

$$x_1 \geq 0.4\,(x_1 + x_2 + x_3)\,.$$

Si devono inoltre esplicitare i vincoli di non negatività

$$x_1 \geq 0 \quad x_2 \geq 0 \quad x_3 \geq 0.$$

Quindi la formulazione completa può essere scritta

$$\begin{cases} \max\,(1000x_1 + 1500x_2 + 2200x_3) \\ 20x_1 + 30x_2 + 62x_3 \leq 480 \\ 31x_1 + 42x_2 + 51x_3 \leq 480 \\ 16x_1 + 81x_2 + 10x_3 \leq 300 \\ x_3 \leq 0.2\,(x_1 + x_2 + x_3) \\ x_1 \geq 0.4\,(x_1 + x_2 + x_3) \\ x_1 \geq 0 \quad x_2 \geq 0 \quad x_3 \geq 0. \end{cases}$$

□

Osservazione 3.4.3 Nel modello precedente sono state utilizzate variabili di decisione *continue* associate a quantità che possono essere considerate indivisibili (autovetture). Questa ipotesi potrebbe risultare impropria, tuttavia permette di formulare il problema come Problema di Programmazione Lineare (e non di Programmazione Lineare Intera, cioè come un problema piú "trattabile"). D'altra parte, in generale, tale ipotesi può non far perdere validità al modello soprattutto se i valori assunti dalle variabili di decisione sono relativamente molto grandi. Ogni approssimazione a valori interi del valore ottimo delle variabili, ovviamente, fa perdere l'ottimalità della soluzione così ottenuta, ma in molti casi tale soluzione approssimata può essere efficacemente utilizzata nella pratica.

Esempio 3.4.4 *Si consideri la stessa azienda dell'esempio precedente con la sola differenza che, questa volta, i tre modelli di autovetture possono essere prodotte utilizzando uno qualsiasi dei tre robot senza richiedere quindi che per avere un'autovettura finita sia necessaria la lavorazione di tutti i tre robot.*

Formulazione.

– *Variabili.* Indichiamo con x_{ij} , con $i = 1, 2, 3$ e $j = 1, 2, 3$, il numero di autovetture del modello j-esimo da produrre giornalmente con il robot i-esimo.

– *Funzione obiettivo.* La funzione obiettivo diventa:

$$1000(x_{11} + x_{21} + x_{31}) + 1500(x_{12} + x_{22} + x_{32}) + 2200(x_{13} + x_{23} + x_{33})$$

– *Vincoli.*

- I vincoli sulla capacità produttiva si esprimono:

$$20x_{11} + 30x_{12} + 62x_{13} \leq 480.$$

$$31x_{21} + 42x_{22} + 51x_{23} \leq 480$$

$$16x_{31} + 81x_{32} + 10x_{33} \leq 300.$$

- i vincoli sul numero totale dei singoli tipi di autovetture da fabbricare assumono la forma:

$$x_{13} + x_{23} + x_{33} \leq 0.2 \sum_{i=1}^{3} \sum_{j=1}^{3} x_{ij}$$

$$x_{11} + x_{21} + x_{31} \geq 0.4 \sum_{i=1}^{3} \sum_{j=1}^{3} x_{ij}.$$

Si devono inoltre esplicitare i vincoli di non negatività

$$x_{ij} \geq 0 \qquad i = 1, 2, 3, \qquad j = 1, 2, 3.$$

Quindi la formulazione finale è la seguente:

$$\begin{cases} \max\left(1000(x_{11} + x_{21} + x_{31}) + 1500(x_{12} + x_{22} + x_{32}) + 2200(x_{13} + x_{23} + x_{33})\right) \\ 20x_{11} + 30x_{12} + 62x_{13} \leq 480 \\ 31x_{21} + 42x_{22} + 51x_{23} \leq 480 \\ 16x_{31} + 81x_{32} + 10x_{33} \leq 300 \\ x_{13} + x_{23} + x_{33} \leq 0.2\left(x_{11} + x_{12} + x_{13} + x_{21} + x_{22} + x_{23} + x_{31} + x_{32} + x_{33}\right) \\ x_{11} + x_{21} + x_{31} \geq 0.4\left(x_{11} + x_{12} + x_{13} + x_{21} + x_{22} + x_{23} + x_{31} + x_{32} + x_{33}\right) \\ x_{ij} \geq 0 \qquad i = 1, 2, 3, \qquad j = 1, 2, 3. \end{cases}$$

□

Formulazione generale di un problema di allocazione ottima di risorse

Per costruire uno schema generale di formulazione per questo tipo di problemi si assuma di disporre di m risorse $\mathbf{R_1}, \mathbf{R_2}, \ \ldots \ , \mathbf{R_m}$ e di voler fabbricare n diversi prodotti $\mathbf{P_1}, \mathbf{P_2}, \ \ldots \ , \mathbf{P_n}$.

Le risorse possono essere sia umane (mano d'opera) sia materiali (disponibilità di macchinari o di materie prime). Il problema della pianificazione delle risorse consiste nel determinare le quantità da fabbricare di ciascun prodotto $\mathbf{P_1}, \ldots, \mathbf{P_n}$ in modo da massimizzare il profitto rispettando i vincoli sulle risorse disponibili o sui livelli di produzione richiesti.

Si indichi con $a_{ij}, i = 1, \ldots, m, j = 1, \ldots, n$ la quantità della risorsa $\mathbf{R_i}$ necessaria per fabbricare una unità del prodotto $\mathbf{P_j}$. Si può così costruire la seguente tabella

	$\mathbf{P_1}$	$\cdots$	$\mathbf{P_j}$	$\cdots$	$\mathbf{P_n}$
$\mathbf{R_1}$	a_{11}	$\cdots$	a_{1j}	$\cdots$	a_{1n}
$\vdots$	$\vdots$		$\vdots$		$\vdots$
$\mathbf{R_i}$	a_{i1}	$\cdots$	a_{ij}	$\cdots$	a_{in}
$\vdots$	$\vdots$		$\vdots$		$\vdots$
$\mathbf{R_m}$	a_{m1}	$\cdots$	a_{mj}	$\cdots$	a_{mn}

Supponiamo che ciascuna risorsa $\mathbf{R_i}$ non possa superare un valore prefissato $b_i, \quad i = 1, \ldots, m$

$\mathbf{R_1}$	$\mathbf{R_2}$	$\cdots$	$\mathbf{R_m}$
b_1	b_2	$\cdots$	b_m

e che nella vendita di ciascuna unità di prodotto $\mathbf{P_j}$ si ricavi un profitto netto $c_j, \quad j = 1, \ldots, n$

$\mathbf{P_1}$	$\mathbf{P_2}$	$\cdots$	$\mathbf{P_n}$
c_1	c_2	$\cdots$	c_n.

È utile ribadire le ipotesi già esposte in precedenza le quali devono valere in generale per la costruzione di modelli di Programmazione Lineare: *proporzionalità, additività, continuità* cioè i consumi delle risorse e i ricavi ottenibili sono proporzionali ai quantitativi di prodotto fabbricati; i consumi globali di risorse e i ricavi totali si ottengono come somma dei consumi e dei ricavi marginali; le variabili possono assumere valori frazionari.

Formulazione 1: risorse concorrenti.

Esaminiamo prima la situazione in cui il bene fabbricato per essere finito e pronto per la vendita deve utilizzare tutte le risorse, anche se in misura diversa.

– *Variabili di decisione.* Si introducono le variabili di decisione $x_1, x_2, \ldots, x_n$ rappresentanti (in un'opportuna unità di misura) la quantità di ciascun prodotto $\mathbf{P_1}, \mathbf{P_2}, \ \ldots \ , \mathbf{P_n}$. Queste saranno le incognite del problema. Tali variabili di

decisione sono i cosiddetti *livelli di attività*. Introducendo come spazio delle variabili lo spazio delle $n-$uple reali $\mathbb{R}^n$ si può considerare un $x \in \mathbb{R}^n$ definendo $x = (x_1, \ldots, x_n)^T$.

– *Funzione obiettivo.* Per le ipotesi fatte la funzione obiettivo (da massimizzare) può essere scritta

$$z = c_1 x_1 + \ldots + c_n x_n = \sum_{j=1}^{n} c_j x_j.$$

Introducendo $c \in \mathbb{R}^n$, definito $c = (c_1, \ldots, c_n)^T$ la funzione obiettivo può essere scritta in notazione vettoriale

$$z = c^T x.$$

– *Vincoli.* Si devono introdurre i seguenti vincoli:

- Vincoli di capacità produttiva:

  tenendo conto delle limitazioni delle risorse si hanno i seguenti m vincoli

$$\begin{array}{ccccc} a_{11}x_1+ & \ldots & +a_{1n}x_n & \leq b_1 \\ a_{21}x_1+ & \ldots & +a_{2n}x_n & \leq b_2 \\ \vdots & \ldots & \vdots & \vdots \\ a_{m1}x_1+ & \ldots & +a_{mn}x_n & \leq b_m. \end{array}$$

- Vincoli di non negatività:

  le variabili devono essere non negative in quanto esse rappresentano livelli di produzione e quindi si hanno i vincoli

$$x_i \geq 0, \quad i = 1, \ldots, n.$$

Introducendo la matrice $(m \times n)$

$$A = \begin{pmatrix} a_{11} & \ldots & a_{1n} \\ \vdots & & \vdots \\ a_{m1} & \ldots & a_{mn} \end{pmatrix}$$

e il vettore $b = (b_1, \ldots, b_m)^T$ la formulazione completa del problema può essere scritta nella forma

$$\begin{cases} \max c^T x \\ Ax \leq b \\ x \geq 0. \end{cases}$$

È una formulazione generale (con solo vincoli di disuguaglianza e vincoli di non negatività) in cui si può porre un generico problema di allocazione ottima di risorse.

Nella pratica, potrebbe essere necessario imporre ulteriori vincoli:

- Vincoli di domanda
  - limitazioni inferiori sulle variabili x_i cioè
    $$x_i \geq l_i \qquad i = 1, \ldots, n$$
    con $l_i \geq 0$ per assicurare che i prodotti siano fabbricati in quantità significative. In questo caso, per ogni indice i per il quale $l_i > 0$ il vincolo di non negatività $x_i \geq 0$ è ridondante.
  - limitazioni superiori sulle variabili, cioè
    $$x_i \leq u_i \qquad i = 1, \ldots, n$$
    dovute ad eventuali possibilità limitate di assorbimento dei prodotti da parte del mercato.

  Introducendo le notazioni vettoriali $l = (l_1, \ldots, l_n)^T$ e $u = (u_1, \ldots, u_n)^T$ questi vincoli possono essere scritti nella forma $l \leq x \leq u$, $x \in \mathbb{R}^n$.

- Vincoli di interezza.

  Se inoltre non ha senso considerare i prodotti quantità divisibili allora si deve definire un modello di programmazione a numeri interi. Cioè nel caso in cui non si possa supporre che i livelli di attività siano frazionari (ad es. se i prodotti sono quantità indivisibili come motori, lavatrici etc.), allora si deve aggiungere il vincolo che le quantità x_i siano intere.

Formulazione 2: risorse alternative.

Si consideri ora invece la situazione in cui il bene fabbricato per essere finito e pronto per la vendita necessita esclusivamente di una risorsa. Nella pratica questo può accadere se, ad esempio, ciascun reparto in cui può essere suddivisa un'industria è in grado di produrre autonomamente ciascuno dei prodotti, ovvero la lavorazione di un prodotto avviene esclusivamente in uno dei reparti disponibili.

– *Variabili di decisione.* Si introducono le variabili di decisione x_{ij} rappresentanti la quantità di prodotto $\mathbf{P_j}$ da fabbricare utilizzando la risorsa $\mathbf{R_i}$.

– *Funzione obiettivo.* Per le ipotesi fatte la funzione obiettivo (da massimizzare) può essere scritta

$$c_1 \sum_{i=1}^{m} x_{i1} + c_2 \sum_{i=1}^{m} x_{i2} + \ldots + c_n \sum_{i=1}^{m} x_{in} = \sum_{j=1}^{n} c_j \sum_{i=1}^{m} x_{ij}.$$

– *Vincoli.* I vincoli di capacità produttiva sono della forma

$$\begin{array}{cccc} a_{11}x_{11}+ & \ldots & +a_{1n}x_{1n} & \leq b_1 \\ a_{21}x_{21}+ & \ldots & +a_{2n}x_{2n} & \leq b_2 \\ \vdots & \ldots & \vdots & \vdots \\ a_{m1}x_{m1}+ & \ldots & +a_{mn}x_{mn} & \leq b_m. \end{array}$$

Infine si devono esplicitare i vincoli di non negatività della variabili cioè $x_{ij} \geq 0$, $i = 1, \ldots, m$, $j = 1, \ldots, n$.

Come si può facilmente osservare la matrice A dei coefficienti delle disequazioni lineari che descrivono i vincoli è rimasta immutata rispetto alla matrice considerata nella formulazione del caso delle risorse concorrenti già vista, ma c'è una sostanziale differenza nelle variabili.

Modelli multi–plant

Si tratta di problemi di pianificazione della produzione in cui modelli di grandi dimensioni sono ottenuti come combinazione di modelli piú piccoli. Tali modelli combinati sono sicuramente piú efficaci dei sottomodelli dai quali essi sono costituiti. Esaminiamo un esempio di questa situazione.

Esempio 3.4.5 *Un'industria manifatturiera possiede due impianti di produzione e fabbrica due tipi di prodotti* $\mathbf{P_1}$ *e* $\mathbf{P_2}$ *utilizzando due macchine utensili: una per la levigatura e una per la pulitura. Per avere un prodotto finito è necessaria l'utilizzazione di entrambe le macchine. Il primo impianto ha una disponibilità massima settimanale di 80 ore della macchina per la levigatura e di 60 ore della macchina per la pulitura. Le disponibilità massime orarie delle due macchine nel secondo impianto sono rispettivamente di 60 e 75 ore settimanali. La tabella che segue riporta, per ciascun prodotto, il numero di ore di lavorazione necessarie su ciascuna macchina per ottenere un prodotto finito (poiché le macchine possedute dal secondo impianto sono piú vecchie, i tempi di utilizzo sono maggiori)*

	Impianto 1		Impianto 2	
	$\mathbf{P_1}$	$\mathbf{P_2}$	$\mathbf{P_1}$	$\mathbf{P_2}$
levigatura	*4*	*2*	*5*	*3*
pulitura	*2*	*5*	*5*	*6*

Inoltre ciascuna unità di prodotto utilizza 4 Kg di materiale grezzo. Il profitto netto ottenuto dalla vendita di una unità di prodotto $\mathbf{P_1}$ *e* $\mathbf{P_2}$ *è rispettivamente di 10\$ e 15\$.*

(a) *Costruire un modello lineare che permetta di massimizzare il profitto complessivo ottenuto dalla vendita dei prodotti in ciascun impianto sapendo che settimanalmente l'industria dispone di 75 Kg di materiale grezzo nel primo impianto e di 45 Kg di materiale grezzo nel secondo impianto.*

(b) *Costruire un modello lineare che permetta di massimizzare il profitto complessivo ottenuto dalla vendita dei prodotti supponendo che l'industria non allochi a priori 75 Kg di materiale grezzo nel primo impianto e di 45 Kg di materiale grezzo nel secondo impianto, ma lasci al modello la decisione di come ripartire tra i due impianti 120 Kg complessivi disponibili di questo materiale grezzo.*

Formulazione
– *Variabili.* Si introducono le variabili x_1 e x_2 associate alla quantità di prodotto $\mathbf{P_1}$ e $\mathbf{P_2}$ fabbricato settimanalmente dal primo impianto e le variabili x_3 e x_4 associate alla quantità di prodotto $\mathbf{P_1}$ e $\mathbf{P_2}$ fabbricato settimanalmente dal secondo impianto.

Formulazione del caso (a)
Questo caso, nella pratica, corrisponde a costruire due modelli indipendenti: uno riferito al primo impianto, uno riferito al secondo impianto. Una "risorsa" (il materiale grezzo) è già allocata a priori.

Impianto 1: La formulazione relativa al primo impianto è:

$$\begin{cases} \max(10x_1 + 15x_2) \\ 4x_1 + 4x_2 \leq 75 \\ 4x_1 + 2x_2 \leq 80 \\ 2x_1 + 5x_2 \leq 60 \\ x_1 \geq 0, x_2 \geq 0 \end{cases}$$

Impianto 2: La formulazione relativa al secondo impianto è:

$$\begin{cases} \max(10x_3 + 15x_4) \\ 4x_3 + 4x_4 \leq 45 \\ 5x_3 + 3x_4 \leq 60 \\ 5x_3 + 6x_4 \leq 75 \\ x_3 \geq 0, x_4 \geq 0 \end{cases}$$

Formulazione del caso (b)
Questo caso corrisponde a costruire un unico modello comprendente entrambi gli impianti. L'allocazione della "risorsa" data dal materiale grezzo è lasciata al modello stesso.

La formulazione relativa a questo caso è:

$$\begin{cases} \max\ (10x_1 + 15x_2 + 10x_3 + 15x_4) \\ \quad 4x_1 + 4x_2 + 4x_3 + 4x_4 \leq 120 \\ \quad 4x_1 + 2x_2 \leq 80 \\ \quad 2x_1 + 5x_2 \leq 60 \\ \quad 5x_3 + 3x_4 \leq 60 \\ \quad 5x_3 + 6x_4 \leq 75 \\ \quad x_1 \geq 0, \quad x_2 \geq 0, \quad x_3 \geq 0, \quad x_4 \geq 0 \end{cases}$$

□

Osservazione 3.4.6 Nel caso (b) si richiede al modello di ripartire i 120 Kg di materiale grezzo piuttosto che effettuare un'allocazione arbitraria a priori, quindi

ci si può aspettare una maggiore efficienza nell'allocazione di queste risorse nel caso (b). Un confronto delle soluzioni ottime di questi problemi conferma questa intuizione: infatti nel caso (a), ottimizzando la produzione dell'impianto 1 e quella dell'impianto 2, si ottiene un guadagno complessivo di $225\$ + 168.75\$ = 393.75\$$, mentre nel caso (b) si ottiene un guadagno di 404.15\$.

Osservazione 3.4.7 Si osservi la particolare struttura della matrice dei coefficienti dei vincoli che è tipica dei problemi di questo tipo

$$\begin{pmatrix} 4 & 2 & 0 & 0 \\ 2 & 5 & 0 & 0 \\ 0 & 0 & 5 & 3 \\ 0 & 0 & 5 & 6 \end{pmatrix}$$

Una matrice con questa struttura si chiama *matrice a blocchi*. Una siffatta struttura permette di utilizzare metodi particolari per la soluzione del problema. Infatti possono essere utilizzate *tecniche di decomposizione* che consentono di risolvere efficientemente anche problemi di questo tipo anche di dimensioni molto elevate. Si osservi che le tecniche di decomposizione non consistono nella suddivisione del problema in sottoproblemi, ma piuttosto con tale termine ci si riferisce a procedure computazionali specifiche che pur considerando il problema complessivo sfruttano la sua particolare struttura. L'importanza della decomposizione non è soltanto computazionale ma ha anche una significativa interpretazione economica; infatti essa corrisponde a considerare una pianificazione decentralizzata.

Modelli multiperiodo

Si tratta di problemi di allocazione ottima di risorse limitate analoghi a quelli già trattati, ma dove la pianificazione è effettuata su un orizzonte temporale composto da piú periodi elementari; si richiede, cioè, di estendere la programmazione mensile della produzione di un'azienda in modo da ottenere un piano di produzione semestrale con possibilità di giacenze al termine di ciascun mese. L'esempio che segue riporta una semplice situazione di questo tipo.

Esempio 3.4.8 *Si consideri l'industria manifatturiera vista nel precedente Esempio 3.4.5 nel caso in cui abbia solamente il primo impianto di produzione. In questo caso si deve programmare la produzione dei due prodotti* $\mathbf{P_1}$ *e* $\mathbf{P_2}$ *nelle due successive settimane sapendo che nella prima settimana si potranno vendere al piú 12 prodotti* $\mathbf{P_1}$ *e 4 prodotti* $\mathbf{P_2}$*, mentre nella seconda si potranno vendere al piú 8 prodotti* $\mathbf{P_1}$ *e 12 prodotti* $\mathbf{P_2}$*. Inoltre nella prima settimana c'è la possibiltà di produrre piú prodotti rispetto a quelli che si possono vendere, immagazzinando i prodotti in eccesso prevedendo un loro utilizzo nella settimana successiva.*

Costruire un modello lineare che permetta di massimizzare il profitto complessivo ottenuto dalla vendita dei prodotti nelle due settimane sapendo che settimanalmente l'industria dispone di 75 Kg di materiale grezzo e tenendo conto che il costo di immagazzinamento di un prodotto (sia di tipo $\mathbf{P_1}$ *sia di tipo* $\mathbf{P_2}$*) è di 2 \$. Si ricorda che il profitto netto ottenuto dalla vendita di 1 unità di prodotto* $\mathbf{P_1}$ *e* $\mathbf{P_2}$ *è rispettivamente di 10\$ e 15\$.*

Formulazione

– *Variabili.* Si introducono le variabili x_1 e x_2 associate alla quantità di prodotti $\mathbf{P_1}$ e $\mathbf{P_2}$ fabbricati nella prima settimana, le variabili x_3 e x_4 associate alla quantità di prodotti $\mathbf{P_1}$ e $\mathbf{P_2}$ fabbricati nella seconda settimana e le variabili y_1 e y_2 che indicano le quantità di prodotti $\mathbf{P_1}$ e $\mathbf{P_2}$ fabbricati nella prima settimana ed immagazzinati per venderli nella seconda.

– *Funzione obiettivo.* Nella prima settimana saranno vendute le quantità $(x_1 - y_1)$ di prodotto $\mathbf{P_1}$ e $(x_2 - y_2)$ di prodotto $\mathbf{P_2}$, nella seconda le quantità $(x_3 + y_1)$ di prodotto $\mathbf{P_1}$ e $(x_4 + y_2)$ di prodotto $\mathbf{P_2}$. Tenendo conto dei costi di immagazzinamento si ottiene la seguente funzione obiettivo:

$$10(x_1 - y_1) + 15(x_2 - y_2) + 10(x_3 + y_1) + 15(x_4 + y_2) - 2(y_1 + y_2) = \\ 10(x_1 + x_3) + 15(x_2 + x_4) - 2(y_1 + y_2).$$

– *Vincoli.* In questo problema si hanno nuovamente quattro tipologie di vincoli:

- i vincoli sulle capacità produttive nelle due settimane:

$$\begin{array}{rcrrcrcl} 4x_1 & + & 4x_2 & & & & \leq & 75 \\ 4x_1 & + & 2x_2 & & & & \leq & 80 \\ 2x_1 & + & 5x_2 & & & & \leq & 60 \\ & & & 4x_3 & + & 4x_4 & \leq & 75 \\ & & & 4x_3 & + & 2x_4 & \leq & 80 \\ & & & 2x_3 & + & 5x_4 & \leq & 60 \end{array}$$

- vincoli che rappresentano il fatto che, alla fine della prima settimana, una parte dei prodotti può essere immagazzinata

$$x_1 - y_1 \leq 12$$

$$x_2 - y_2 \leq 4$$

- vincoli che rappresentano il fatto che il numero dei prodotti disponibili nella seconda settimana non deve superare le richieste del mercato

$$y_1 + x_3 \leq 8$$

$$y_2 + x_4 \leq 12$$

- vincoli che rappresentano la non negatività delle variabili

$$x_1 \geq 0, \quad x_2 \geq 0, \quad x_3 \geq 0, \quad x_4 \geq 0, \quad y_1 \geq 0, \quad y_2 \geq 0.$$

La formulazione relativa a questo problema è:

$$\begin{cases} \max \left(10(x_1 + x_2) + 15(x_3 + x_4) - 2(y_1 + y_2) \right) \\ \\ \begin{array}{rcrcrcrcrcl} 4x_1 & + & 4x_2 & & & & & & & \leq & 75 \\ 4x_1 & + & 2x_2 & & & & & & & \leq & 80 \\ 2x_1 & + & 5x_2 & & & & & & & \leq & 60 \\ & & & 4x_3 & + & 4x_4 & & & & \leq & 75 \\ & & & 4x_3 & + & 2x_4 & & & & \leq & 80 \\ & & & 2x_3 & + & 5x_4 & & & & \leq & 60 \\ x_1 & & & & & & - & y_1 & & \leq & 12 \\ & & x_2 & & & & & & - \; y_2 & \leq & 4 \\ & & & x_3 & & & + & y_1 & & \leq & 8 \\ & & & & & x_4 & & & + \; y_2 & \leq & 12 \end{array} \\ \\ x_1 \geq 0, \; x_2 \geq 0, \; x_3 \geq 0, \; x_4 \geq 0, \; y_1 \geq 0, \; y_2 \geq 0. \end{cases}$$

Osservazione 3.4.9 Se non si fosse prevista la possibilità di poter immagazzinare dei prodotti non venduti, si sarebbe dovuto massimizzare separatamente i profitti ottenuti dalla vendita dei prodotti fabbricati nella prima e nella seconda settimana risolvendo i seguenti problemi:

$$\begin{cases} \max(10x_1 + 15x_2) \\ 4x_1 + 4x_2 \leq 75 \\ 4x_1 + 2x_2 \leq 80 \\ 2x_1 + 5x_2 \leq 60 \\ 0 \leq x_1 \leq 12 \\ 0 \leq x_2 \leq 4, \end{cases} \qquad \begin{cases} \max(10x_1 + 15x_2) \\ 4x_1 + 4x_2 \leq 75 \\ 4x_1 + 2x_2 \leq 80 \\ 2x_1 + 5x_2 \leq 60 \\ 0 \leq x_1 \leq 8 \\ 0 \leq x_2 \leq 12. \end{cases}$$

In questo caso si sarebbe ottenuto un guadagno complessivo di $180\$ + 212\$ = 392\$$. Mentre la soluzione ottima del modello di Programmazione Lineare, descritto precedentemente e che prevedeva anche la possibilità di poter immagazzinare i prodotti non venduti, porta ad un guadagno di 429.1\$. Questo mette in evidenza la convenienza di effettuare una programmazione complessiva sulle due settimane, prevedendo la possibilità di produrre nella prima settimana di piú di quanto si possa vendere e considerando anche le spese relative all'immagazzinamento dei prodotti non venduti.

Osservazione 3.4.10 Si osservi che i primi sei vincoli del precedente modello multiperiodo presentano una struttura particolare. Infatti possono essere rappresentati da una matrice *a blocchi* (in particolare nell'esempio considerato tutti i blocchi sono uguali). Il fatto di avere la maggior parte dei vincoli con una struttura a blocchi è una caratteristica di tutti i modelli multiperiodo. Come detto per i modelli multi-plan, questa particolare struttura può essere sfruttata attraverso l'uso di tecniche di decomposizione in modo da risolvere efficientemente anche problemi di questo tipo di grosse dimensioni.

3.4.2 Modelli di miscelazione

Nei modelli di allocazione ottima le risorse devono essere ripartite mentre nei modelli di miscelazione le risorse devono essere combinate tra di loro. I modelli di miscelazione decidono come combinare (miscelare) tali risorse in maniera da soddisfare al meglio determinati obiettivi rispettando opportune richieste.

Esempio 3.4.11 *Un'industria conserviera deve produrre succhi di frutta mescolando polpa di frutta e dolcificante ottenendo un prodotto finale che deve soddisfare alcuni requisiti riguardanti il contenuto di vitamina C, di sali minerali e di zucchero. La polpa di frutta e il dolcificante vengono acquistati al costo rispettivamente di 4 Euro e 6 Euro ogni ettogrammo. Inoltre dalle etichette si ricava che 100 grammi di polpa di frutta contengono 140 mg di vitamina C, 20 mg di sali minerali e 25 grammi di zucchero, mentre 100 grammi di dolcificante contengono 10 mg di sali minerali, 50 grammi di zucchero e non contengono vitamina C. I requisiti che il prodotto finale (cioè il succo di frutta pronto per la vendita) deve avere sono i seguenti: il succo di frutta deve contenere almeno 70 mg di vitamina C, almeno 30 mg di sali minerali e almeno 75 grammi di zucchero. Si devono determinare le quantità di polpa di frutta e di dolcificante da utilizzare nella produzione del succo di frutta in modo da minimizzare il costo complessivo dell'acquisto dei due componenti base.*

Formulazione.
Si vuole costruire un modello di Programmazione Lineare che rappresenti il problema in analisi tenendo presente i requisiti di qualità richiesti. Si verifica facilmente che le ipotesi fondamentali di un modello di Programmazione Lineare sono soddisfatte.

– *Variabili.* È naturale associare la variabili di decisione alle quantità di polpa di frutta e di dolcificante da utilizzare per la produzione del succo di frutta. Quindi siano x_1 e x_2 rispettivamente le quantità espresse in ettogrammi di polpa di frutta e di dolcificante che devono essere utilizzate.

– *Funzione obiettivo.* È rappresentata dal costo complessivo dell'acquisto dei due componenti base e quindi è data (in centesimi di Euro) da $400x_1 + 600x_2$. Questa espressione naturalmente deve essere minimizzata.

– *Vincoli.* Poiché un ettogrammo di polpa contiene 140 mg di vitamina C e il dolcificante non contiene vitamina C, il primo vincolo da considerare riguardante il contenuto di vitamina C del succo di frutta si può scrivere nella forma

$$140x_1 \geq 70.$$

Analogamente per rispettare il requisito sul contenuto di sali minerali del succo di frutta si dovrà imporre il vincolo

$$20x_1 + 10x_2 \geq 30.$$

Infine il vincolo sul contenuto di zucchero del succo di frutta si può esprimere nella forma

$$25x_1 + 50x_2 \geq 75.$$

Infine si deve esplicitare il vincolo di non negatività sulle variabili cioè

$$x_1 \geq 0, x_2 \geq 0.$$

Quindi la formulazione finale è

$$\begin{cases} \min(400x_1 + 600x_2) \\ 140x_1 \geq 70 \\ 20x_1 + 10x_2 \geq 30 \\ 25x_1 + 50x_2 \geq 75 \\ x_1 \geq 0, x_2 \geq 0 \end{cases}$$

□

Esempio 3.4.12 – Il Problema della Dieta
Una dieta prescrive che giornalmente devono essere assimilate quantità predeterminate di calorie, proteine e calcio, intese come fabbisogni minimi giornalieri, disponendo di cinque alimenti base (pane, latte, uova, carne, dolce). Tali fabbisogni minimi sono di 2000 calorie, 50 g. di proteine, 700 mg. di calcio. Dalle tabelle dietetiche si ricavano i seguenti contenuti di calorie (in cal.), proteine (in g.), calcio (in mg.) per ogni singola porzione di ciascun alimento, intendendo come porzione una quantità espressa in grammi e quindi frazionabile.

	Pane	**Latte**	**Uova**	**Carne**	**Dolce**
calorie	*110*	*160*	*180*	*260*	*420*
proteine	*4*	*8*	*13*	*14*	*4*
calcio	*2*	*285*	*54*	*80*	*22*

I costi (in Euro) e il numero massimo di porzioni tollerate giornalmente sono i seguenti

	Pane	**Latte**	**Uova**	**Carne**	**Dolce**
costo	*2*	*3*	*4*	*19*	*20*
porz.	*4*	*8*	*3*	*2*	*2*

Determinare una dieta a costo minimo che soddisfi le prescrizioni richieste.

Formulazione.
Poiché si è supposto che le porzioni siano frazionabili ed inoltre valgono le ipotesi di linearità, si può costruire un modello di Programmazione Lineare per rappresentare il problema in analisi.

– *Variabili.* È ovvio introdurre le variabili x_1, x_2, x_3, x_4, x_5 indicanti le quantità di porzioni dei singoli alimenti da includere giornalmente nella dieta.

– *Funzione obiettivo.* È rappresentata dal costo complessivo ed è quindi data da

$$2x_1 + 3x_2 + 4x_3 + 19x_4 + 20x_5.$$

– *Vincoli.* Poiché sono prescritti i fabbisogni minimi giornalieri, si avranno i seguenti vincoli:

$$\begin{aligned} \text{calorie} &\longrightarrow 110x_1 + 160x_2 + 180x_3 + 260x_4 + 420x_5 \geq 2000 \\ \text{proteine} &\longrightarrow 4x_1 + 8x_2 + 13x_3 + 14x_4 + 4x_5 \geq 50 \\ \text{calcio} &\longrightarrow 2x_1 + 285x_2 + 54x_3 + 80x_4 + 22x_5 \geq 700 \end{aligned}$$

Inoltre i vincoli sul numero massimo di porzioni giornaliere di ciascun alimento e di non negatività

$$0 \leq x_1 \leq 4,\ 0 \leq x_2 \leq 8,\ 0 \leq x_3 \leq 3,\ 0 \leq x_4 \leq 2,\ 0 \leq x_5 \leq 2.$$

La formulazione completa sarà quindi

$$\begin{cases} \min\left(2x_1 + 3x_2 + 4x_3 + 19x_4 + 20x_5\right) \\ 110x_1 + 160x_2 + 180x_3 + 260x_4 + 420x_5 \geq 2000 \\ 4x_1 + 8x_2 + 13x_3 + 14x_4 + 4x_5 \geq 50 \\ 2x_1 + 285x_2 + 54x_3 + 80x_4 + 22x_5 \geq 700 \\ 0 \leq x_1 \leq 4, 0 \leq x_2 \leq 8, 0 \leq x_3 \leq 3, 0 \leq x_4 \leq 2, 0 \leq x_5 \leq 2. \end{cases}$$

Se inoltre si vuole supporre, ad esempio, che nella dieta sia presente almeno una porzione di dolce e due di latte si dovranno imporre i vincoli $x_5 \geq 1$ e $x_2 \geq 2$ da aggiungere alla precedente formulazione. In questo caso, i vincoli già presenti $x_5 \geq 0$ e $x_2 \geq 0$ sono ridondanti. □

Formulazione generale di un problema di miscelazione

Formalmente, supponiamo di disporre di n sostanze diverse che indichiamo con $\mathbf{S_1}, \mathbf{S_2}, \ \ldots\ , \mathbf{S_n}$ ciascuna delle quali contenga una certa quantità di ciascuno degli m componenti utili che indichiamo con $\mathbf{C_1}, \mathbf{C_2}, \ \ldots\ , \mathbf{C_m}$. Supponendo che ogni sostanza $\mathbf{S_j}$ abbia costo unitario c_j, $j = 1, \ldots, n$

$\mathbf{S_1}$	$\mathbf{S_2}$	$\cdots$	$\mathbf{S_n}$
c_1	c_2	$\cdots$	c_n

si desidera ottenere la miscela piú economica che soddisfi alcuni requisiti qualitativi, cioè contenga una quantità non inferiore a b_i di ciascun $\mathbf{C_i}$, $i = 1, \ldots, m$

$\mathbf{C_1}$	$\mathbf{C_2}$	$\cdots$	$\mathbf{C_m}$
b_1	b_2	$\cdots$	b_m.

Si indichi con $a_{ij}, i = 1, \ldots, m, j = 1, \ldots, n$ la quantità di componente $\mathbf{C_i}$ presente nella sostanza $\mathbf{S_j}$. Si può così costruire la seguente tabella

	$\mathbf{S_1}$	$\cdots$	$\mathbf{S_j}$	$\cdots$	$\mathbf{S_n}$
$\mathbf{C_1}$	a_{11}	$\cdots$	a_{1j}	$\cdots$	a_{1n}
$\vdots$	$\vdots$		$\vdots$		$\vdots$
$\mathbf{C_i}$	a_{i1}	$\cdots$	a_{ij}	$\cdots$	a_{in}
$\vdots$	$\vdots$		$\vdots$		$\vdots$
$\mathbf{C_m}$	a_{m1}	$\cdots$	a_{mj}	$\cdots$	a_{mn}

Formulazione.
Supponendo che valgano le ipotesi di proporzionalità, additività ed inoltre assumendo che le quantità di sostanze da utilizzare siano frazionabili, si può formulare questo problema in termini di un problema di Programmazione Lineare.

– *Variabili.* È naturale introdurre le variabili di decisione $x_1, x_2, \ldots, x_n$ rappresentanti la quantità di ciascuna sostanza $\mathbf{S_1}, \mathbf{S_2}, \ \ldots \ , \mathbf{S_n}$ da utilizzare nella miscela. Queste saranno le incognite del problema. Introducendo come spazio delle variabili lo spazio delle $n-$uple reali $\mathbb{R}^n$ si può considerare un $x \in \mathbb{R}^n$ definendo $x = (x_1, \ldots, x_n)^T$.

– *Funzione obiettivo.* Per le ipotesi fatte, la funzione obiettivo può essere scritta

$$z = c_1 x_1 + \ldots + c_n x_n = \sum_{j=1}^{n} c_j x_j.$$

Introducendo $c \in \mathbb{R}^n$, definito $c = (c_1, \ldots, c_n)^T$, la funzione obiettivo può essere scritta in notazione vettoriale

$$z = c^T x.$$

– *Vincoli.* Si devono introdurre i seguenti vincoli:

- Vincoli di qualità.

  Tenendo conto del fatto che la miscela deve contenere una quantità non inferiore a b_i di ciascun componente $\mathbf{C_i}$ si dovrà avere

  $$\sum_{j=1}^{n} a_{ij} x_j \geq b_i, \quad i = 1, \ldots, m.$$

- Vincoli di non negatività.

  Si devono infatti considerare i vincoli di non negatività sulle variabili cioè $x_j \geq 0$, $j = 1, \ldots, n$.

Introducendo la matrice $(m \times n)$

$$A = \begin{pmatrix} a_{11} & \dots & a_{1n} \\ \vdots & & \vdots \\ a_{m1} & \dots & a_{mn} \end{pmatrix}$$

e il vettore $b = (b_1, \dots, b_m)^T$ la formulazione completa del problema può essere scritta nella forma

$$\begin{cases} \min c^T x \\ Ax \geq b \\ x \geq 0. \end{cases}$$

Nella pratica, potrebbe essere necessario introdurre ulteriori vincoli:

- possono essere presenti limitazioni superiori o inferiori sulle variabili cioè $x_j \geq L$, $x_j \leq M$, $j = 1, \dots, n$;
- se è richiesto anche che la miscela contenga una quantità non superiore ad un valore d_i di ciascun componente $\mathbf{C_i}$ si dovrà aggiungere alla formulazione un altro vincolo di qualità:

$$\sum_{j=1}^{n} a_{ij} x_j \leq d_i, \quad i = 1, \dots, m;$$

- in alcuni casi si richiede che una certa sostanza appartenga alla miscela solo se un'altra sostanza vi appartiene (o non vi appartiene). Questi vincoli richiedono l'uso di variabili booleane come descritto in seguito.

□

Esempio 3.4.13 *Il prodotto finale di una fabbrica è ottenuto raffinando materie prime grezze e miscelandole insieme. Queste materie prime possono essere di due categorie: naturali e sintetizzate. In particolare, sono disponibili tre materie prime naturali (***N1***, **N2**, **N3***) e due materie prime sintetizzate (***S1***, **S2***). Le materie prime naturali e quelle sintetizzate richiedono differenti linee di produzione. Ogni settimana è possibile raffinare non più di 500 quintali di materie prime naturali e non più di 300 quintali di materie prime sintetizzate. Si assume che non ci sia perdita di peso durante la raffinazione e che si possa trascurare il costo di raffinazione. Inoltre esiste una restrizione tecnologica sulla gradazione del prodotto finale: nell'unità di misura in cui questa gradazione è misurata, essa deve essere tra 2 e 7; si assume che tale gradazione nella miscela finale dipenda linearmente dalle singole gradazioni delle materie prime componenti. Nella tabella che segue è riportato il costo (in euro) per quintale e la gradazione delle materie prime grezze.*

	N1	**N2**	**N3**	**S1**	**S2**
costo	*300*	*190*	*250*	*200*	*230*
grad.	*6.0*	*1.9*	*8.5*	*5.0*	*3.5*

Il prodotto finale viene venduto a 350 euro per quintale. Determinare come va pianificata la produzione settimanale per massimizzare il profitto netto.

Formulazione.

– *Variabili.* Introduciamo le variabili di decisione x_1, x_2, x_3, x_4, x_5 rappresentanti le quantità (in quintali) di **N1**, **N2**, **N3**, **S1**, **S2** che devono essere comprate e raffinate in una settimana. Inoltre introduciamo una ulteriore variabile y che indica la quantità di prodotto finale che deve essere fabbricato.

– *Funzione obiettivo.* La funzione obiettivo da massimizzare sarà data dal profitto netto cioè da

$$350y - 300x_1 - 190x_2 - 250x_3 - 200x_4 - 230x_5.$$

– *Vincoli.* Sono presenti tre tipi di vincoli

· capacità di raffinamento

$$x_1 + x_2 + x_3 \leq 500$$

$$x_4 + x_5 \leq 300;$$

· limitazioni sulla gradazione

$$6.0x_1 + 1.9x_2 + 8.5x_3 + 5.0x_4 + 3.5x_5 \leq 7y$$

$$6.0x_1 + 1.9x_2 + 8.5x_3 + 5.0x_4 + 3.5x_5 \geq 2y;$$

· vincolo di continuità

$$x_1 + x_2 + x_3 + x_4 + x_5 = y.$$

Questo vincolo di continuità esprime il fatto che il peso finale del prodotto deve essere uguale alla somma dei pesi degli ingredienti.

Inoltre si devono esplicitare i vincoli di non negatività delle variabili.
La formulazione finale risulta quindi

$$\begin{cases} \max(-300x_1 - 190x_2 - 250x_3 - 200x_4 - 230x_5 + 350y) \\ x_1 + x_2 + x_3 \leq 500 \\ x_4 + x_5 \leq 300 \\ 6.0x_1 + 1.9x_2 + 8.5x_3 + 5.0x_4 + 3.5x_5 - 7y \leq 0 \\ 6.0x_1 + 1.9x_2 + 8.5x_3 + 5.0x_4 + 3.5x_5 - 2y \geq 0 \\ x_1 + x_2 + x_3 + x_4 + x_5 - y = 0 \\ x_1 \geq 0, x_2 \geq 0, x_3 \geq 0, x_4 \geq 0, x_5 \geq 0, y \geq 0 \end{cases}$$

□

Osservazione 3.4.14 Un errore comune è quello di scrivere i vincoli sulla gradazione

$$6.0x_1 + 1.9x_2 + 8.5x_3 + 5.0x_4 + 3.5x_5 \leq 7$$

$$6.0x_1 + 1.9x_2 + 8.5x_3 + 5.0x_4 + 3.5x_5 \geq 2.$$

Queste relazioni sono evidentemente dimensionalmente errate: il primo membro ha le dimensioni di *gradazione* × *quantità* mentre il secondo membro ha le dimensioni della *gradazione*. Tuttavia, invece delle variabili x_i in queste due disuguaglianze si potevano usare le variabili x_i/y per rappresentare le *proporzioni degli ingredienti*, piuttosto che le *quantità assolute* x_i; ovviamente, in questo caso si dovevano modificare anche le altre espressioni. Comunque, l'uso delle variabili x_i/y è ovviamente possibile solo nel caso in cui la quantità di prodotto fabbricato è non nulla, cioè $y \neq 0$.

Modelli di input–output

I modelli di miscelazione possono essere visti come modelli piú generali in cui le sostanze $\mathbf{S_j}$ e i componenti utili $\mathbf{C_i}$ sono genericamente definiti come *"input"* e *"output"*; per ogni input j si deve decidere la quantità x_j da utilizzare incorrendo in un costo $c_j x_j$ e creando $a_{ij}x_j$ unità di output i. Lo scopo è quello di determinare la combinazione a più basso costo di input che fornisce, per ogni output i, una quantità di unità di output compresa tra valori prefissati. Nei modelli di miscelazione analizzati fino ad ora, gli input sono dati dalle sostanze che devono essere mescolate, gli output sono dati dalle qualità della miscela risultante.

Un esempio di questa generalizzazione è dato dai problemi di *assegnazione di personale a turni* che rappresentano problemi di fondamentale importanza in diversi settori applicativi; in questo caso gli output possono corrispondere alle ore lavorate in un certo giorno i e, per ogni turno lavorativo j, a_{ij} rappresenta il numero di ore che una persona assegnata al turno j lavorerà il giorno i (ponendo $a_{ij} = 0$ se la persona assegnata al turno j non lavora il giorno i); le c_j rappresentano il salario di una persona assegnata al turno j e x_j il numero di persone assegnate a quel turno. In questo contesto, la funzione obiettivo diventa il costo totale dei salari mensile, mentre i vincoli diventano quelli dovuti al fatto che ogni giorno i, il numero totale di ore lavorative fornite dalle persone che lavorano quel giorno deve essere pari ad almeno un valore prefissato b_i. Supponendo di voler considerare n giorni e m possibili turni, un modello di Programmazione Lineare che rappresenti questa situazione è dato da

$$\begin{array}{lllll} \min & c_1x_1 + \ldots + c_nx_n & & & \\ a_{11}x_1 + & \ldots & +a_{1n}x_n & \geq b_1 \\ a_{21}x_1 + & \ldots & +a_{2n}x_n & \geq b_2 \\ \vdots & \ldots & \vdots & \vdots \\ a_{m1}x_1 + & \ldots & +a_{mn}x_n & \geq b_m \\ x_j \geq 0, & j = 1, \ldots, n. & & \end{array}$$

In questo caso però, a differenza degli altri casi di miscelazione visti fino ad ora, l'assunzione di continuità delle variabili non è molto plausibile e potrebbe risultare necessario introdurre il vincolo di interezza sulle variabili.
Il concetto di modello di "input–output" fu una delle prime applicazioni della Programmazione Lineare nelle analisi economiche.

Si riporta, di seguito, un semplice esempio di assegnamento di personale a turni di lavoro.

Esempio 3.4.15 *Un catena di ristoranti opera sette giorni alla settimana e richiede il seguente numero minimo di camerieri:*

Lun.	**Mar.**	**Mer.**	**Giov.**	**Ven.**	**Sab.**	**Dom.**
52	*50*	*47*	*55*	*70*	*40*	*40*

Ciascun cameriere lavora seguendo turni cosí definiti: cinque giorni lavorativi ogni settimana e due di riposo; inoltre sono possibili al piú quattro giorni consecutivi di lavoro seguiti da uno di riposo; inoltre uno solo dei due giorni del fine settimana (sabato o domenica) deve far parte del turno di lavoro. I turni risultanti sono sei e sono schematizzati nella tabella che segue (dove "L" indica giornata lavorativa e "R" riposo):

TURNI:	1^o	2^o	3^o	4^o	5^o	6^o	7^o	8^o
Lun.	L	R	L	L	L	L	L	L
Mar.	L	L	R	L	L	R	L	L
Mer.	L	L	L	R	L	L	R	L
Giov.	L	L	L	L	R	L	L	R
Ven.	R	L	L	L	L	L	L	L
Sab.	L	R	L	R	L	R	L	R
Dom.	R	L	R	L	R	L	R	L

Il salario settimanale di un cameriere è pari a 250 Euro se assegnato ad un turno che non comprende la domenica, mentre è pari a 270 Euro se il turno comprende anche la domenica. Il gestore di questa catena di ristoranti vuole minimizzare il costo che deve sostenere per retribuire i camerieri in modo da soddisfare le richieste giornaliere.

Formulazione.

–*Variabili.* Si associano le variabili di decisione x_j al numero di camerieri assegnati al turno j, $j = 1, \ldots, 8$.

–*Funzione obiettivo.* È data dal salario complessivo dei camerieri e quindi può essere espressa nella forma

$$250x_1 + 270x_2 + 250x_3 + 270x_4 + 250x_5 + 270x_6 + 250x_7 + 270x_8.$$

–*Vincoli.* I vincoli sono dovuti al fatto che ogni giorno c'è una richiesta minima di camerieri. Osservando ogni giorno quale turno prevede il lavoro o il riposo si ottengono i seguenti vincoli

$$\begin{aligned}
x_1 + x_3 + x_4 + x_5 + x_6 + x_7 + x_8 &\geq 52 \\
x_1 + x_2 + x_4 + x_5 + + x_7 + x_8 &\geq 50 \\
x_1 + x_2 + x_3 + x_5 + x_6 + x_8 &\geq 47 \\
x_1 + x_2 + x_3 + x_4 + x_6 + x_7 &\geq 55 \\
x_2 + x_3 + x_4 + x_5 + x_6 + x_7 + x_8 &\geq 70 \\
x_1 + x_3 + x_5 + x_7 &\geq 40 \\
x_2 + x_4 + x_6 + x_8 &\geq 40
\end{aligned}$$

Si deve inoltre esplicitare la non negatività delle variabili $x_j \geq 0, \quad j = 1, \ldots, 8$ e l'interezza $x_j \in \mathbf{Z}$, $j = 1, \ldots, 8$.

La formulazione completa sarà quindi

$$\begin{cases} \min\left(250x_1 + 270x_2 + 250x_3 + 270x_4 + 250x_5 + 270x_6 + 250x_7 + 270x_8\right) \\ x_1 + x_3 + x_4 + x_5 + x_6 + x_7 + x_8 \geq 52 \\ x_1 + x_2 + x_4 + x_5 + x_7 + x_8 \geq 50 \\ x_1 + x_2 + x_3 + x_5 + x_6 + x_8 \geq 47 \\ x_1 + x_2 + x_3 + x_4 + x_6 + x_7 \geq 55 \\ x_2 + x_3 + x_4 + x_5 + x_6 + x_7 + x_8 \geq 70 \\ x_1 + x_3 + x_5 + x_7 \geq 40 \\ x_2 + x_4 + x_6 + x_8 \geq 40 \\ x_j \geq 0, x_j \in \mathbf{Z}, \quad i = 1, \ldots, 8 \end{cases}$$

□

È chiaramente riconoscibile questa formulazione come un modello di miscelazione; è sufficiente, infatti, introdurre la matrice A che definisce i vincoli di un problema di miscelazione nel seguente modo:

$$a_{ij} = \begin{cases} 1 & \text{se nel posto } (i,j) \text{ della tabella c'è la lettera "L"} \\ 0 & \text{se nel posto } (i,j) \text{ della tabella c'è la lettera "R".} \end{cases}$$

3.4.3 Modelli di trasporto

Si tratta di problemi in cui si hanno un certo numero di località (origini) ciascuna delle quali ha una quantità fissata di merce disponibile e un certo numero di clienti residenti in altre località (destinazioni) i quali richiedono quantitativi precisi di merce. Quindi conoscendo il costo unitario del trasporto della merce da ciascuna località origine a ciascuna località destinazione è necessario pianificare i trasporti, cioè la quantità di merce che deve essere trasportata da ciascuna località origine a ciascuna località destinazione in modo da soddisfare l'ordine dei clienti minimizzando il costo complessivo derivante dai trasporti.

Esempio 3.4.16 *Un'industria dell'acciaio dispone di due miniere* $\mathbf{M_1}$ *e* $\mathbf{M_2}$ *e di tre impianti di produzione* $\mathbf{P_1}$ $\mathbf{P_2}$ $\mathbf{P_3}$*. Il minerale estratto deve essere giornalmente trasportato agli impianti di produzione soddisfacendo le rispettive richieste. Le miniere* $\mathbf{M_1}$ *e* $\mathbf{M_2}$ *producono giornalmente rispettivamente 130 e 200 tonnellate di minerale. Gli impianti richiedono giornalmente le seguenti quantità (in tonnellate) di minerale*

$\mathbf{P_1}$	$\mathbf{P_2}$	$\mathbf{P_3}$
80	*100*	*150*

Il costo (in euro) del trasporto da ciascuna miniera a ciascun impianto di produzione di una tonnellata di minerale è riportato nella seguente tabella

	$\mathbf{P_1}$	$\mathbf{P_2}$	$\mathbf{P_3}$
$\mathbf{M_1}$	*10*	*8*	*21*
$\mathbf{M_2}$	*12*	*20*	*14*

Formulare un modello che descriva il trasporto dalle miniere agli impianti di produzione in modo da minimizzare il costo globale del trasporto.

Analisi del problema.
È un problema di trasporti con 2 origini ($\mathbf{M_1}$, $\mathbf{M_2}$) e 3 destinazioni ($\mathbf{P_1}$ $\mathbf{P_2}$ $\mathbf{P_3}$). Si noti che risulta $130 + 200 = 330$ e $80 + 100 + 150 = 330$, ovvero la somma delle disponiblità uguaglia la somma delle richieste.

Formulazione.

– *Variabili.* Associamo le variabili di decisione alle quantità di minerale che deve essere trasportato; indichiamo con $x_{ij} \quad i = 1, 2, \quad j = 1, 2, 3,$ le quantità (in tonnellate) di minerale da trasportare giornalmente da ciascuna miniera $\mathbf{M_i}$ a ciascun impianto di produzione $\mathbf{P_j}$.

– *Funzione obiettivo.* La funzione obiettivo da minimizzare è data dalla somma dei costi dei trasporti cioè da

$$z = 10x_{11} + 8x_{12} + 21x_{13} + 12x_{21} + 20x_{22} + 14x_{23}.$$

– *Vincoli.* I vincoli di origine esprimono il fatto che la somma della quantità di minerale trasportato dalla miniera $\mathbf{M_i}$ deve essere uguale alla disponibilità giornaliera della miniera stessa:

$$\begin{aligned} x_{11} + x_{12} + x_{13} &= 130 \\ x_{21} + x_{22} + x_{23} &= 200. \end{aligned}$$

I vincoli di destinazione esprimono il fatto che la somma delle quantità di minerale trasportato all'impianto di produzione $\mathbf{P_j}$ deve essere pari alla richiesta giornaliera di tale impianto:

$$\begin{aligned} x_{11} + x_{21} &= 80 \\ x_{12} + x_{22} &= 100 \\ x_{13} + x_{23} &= 150. \end{aligned}$$

Infine si devono considerare i vincoli di non negatività $x_{ij} \geq 0, \quad i = 1, 2, \quad j = 1, 2, 3.$
La formulazione completa è quindi

$$\begin{cases} \min\left(10x_{11} + 8x_{12} + 21x_{13} + 12x_{21} + 20x_{22} + 14x_{23}\right) \\ x_{11} + x_{12} + x_{13} = 130 \\ x_{21} + x_{22} + x_{23} = 200 \\ x_{11} + x_{21} = 80 \\ x_{12} + x_{22} = 100 \\ x_{13} + x_{23} = 150 \\ x_{ij} \geq 0, \quad i = 1, 2, \quad j = 1, 2, 3. \end{cases}$$

□

Formulazione generale di un problema di trasporti
Sono definite m località *origini* indicate con $\mathbf{O_1}, \ldots, \mathbf{O_m}$, e n località *destinazioni* indicate con $\mathbf{D_1}, \ldots, \mathbf{D_n}$. Ogni origine $\mathbf{O_i}$, $(i = 1, \ldots, m)$ può fornire una certa disponibilità $a_i \geq 0$ di merce che deve essere trasferita dalle origini alle destinazioni

$\mathbf{O_1}$	$\cdots$	$\mathbf{O_m}$
a_1	$\cdots$	a_m.

Ad ogni destinazione $\mathbf{D_j}$, $(j = 1, \ldots, n)$ è richiesta una quantità $b_j \geq 0$ di merce.

$\mathbf{D_1}$	$\cdots$	$\mathbf{D_n}$
b_1	$\cdots$	b_n.

Supponiamo che il costo del trasporto di una unità di merce da $\mathbf{O_i}$ a $\mathbf{D_j}$ sia pari a c_{ij}. Tali costi nella realtà sono spesso collegati alle distanze tra origini e destinazioni.
Il problema consiste nel pianificare i trasporti in modo da soddisfare le richieste delle destinazioni minimizzando il costo del trasporto complessivo nella seguente ipotesi:

- la disponibilità complessiva uguaglia la richiesta complessiva, cioè

$$\sum_{i=1}^{m} a_i = \sum_{j=1}^{n} b_j; \tag{3.4.1}$$

si escludono possibilità di giacenze nelle origini, cioè tutta la merce prodotta in una origine deve essere trasportata in una delle destinazioni; si escludono possibilità di giacenze nelle destinazioni, cioè la quantità totale che arriva in una destinazione $\mathbf{D_j}$ deve uguagliare la richiesta b_j.

Formulazione.
Si vuole dare una formulazione del problema in esame in termini di un problema di programmazione lineare supponendo quindi che siano verificate le ipotesi di linearità e continuità.

– *Variabili.* Per ogni coppia di origine e destinazione $\mathbf{O_i}$, $\mathbf{D_j}$ si introducono le variabili di decisione x_{ij} rappresentanti la quantità di merce da trasportare da $\mathbf{O_i}$, a $\mathbf{D_j}$. Si tratta di mn variabili

	$\mathbf{D_1}$	$\cdots$	$\mathbf{D_j}$	$\cdots$	$\mathbf{D_n}$
$\mathbf{O_1}$	x_{11}	$\cdots$	x_{1j}	$\cdots$	x_{1n}
$\vdots$	$\vdots$		$\vdots$		$\vdots$
$\mathbf{O_i}$	x_{i1}	$\cdots$	x_{ij}	$\cdots$	x_{in}
$\vdots$	$\vdots$		$\vdots$		$\vdots$
$\mathbf{O_m}$	x_{m1}	$\cdots$	x_{mj}	$\cdots$	x_{mn}

– *Funzione obiettivo.* La funzione obiettivo da minimizzare sarà data da costo totale del trasporto e quindi da

$$z = \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij} x_{ij}.$$

– *Vincoli.* Per le ipotesi fatte, si avranno due tipi di vincoli:

- vincoli di origine

$$\sum_{j=1}^{n} x_{ij} = a_i \qquad i = 1, \ldots, m; \tag{3.4.2}$$

impongono che tutta la merce prodotta in una origine sia trasportata alle destinazioni; si tratta di m vincoli;

- vincoli di destinazione

$$\sum_{i=1}^{m} x_{ij} = b_j \qquad j = 1, \ldots, n; \tag{3.4.3}$$

impongono che la quantità totale di merce che arriva in ciascuna delle destinazioni uguaglia la richiesta; si tratta si n vincoli.

Si devono infine considerare i vincoli di non negatività delle variabili

$$x_{ij} \geq 0 \qquad i = 1, \ldots, n; \quad j = 1, \ldots, m.$$

Si è così ottenuta una formulazione del problema dei trasporti con mn variabili e $m + n + mn$ vincoli:

$$\begin{cases} \min \left(\displaystyle\sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij} x_{ij} \right) \\ \displaystyle\sum_{j=1}^{n} x_{ij} = a_i \qquad i = 1, \ldots, m \\ \displaystyle\sum_{i=1}^{m} x_{ij} = b_j \qquad j = 1, \ldots, n \\ x_{ij} \geq 0 \qquad i = 1, \ldots, n; \quad j = 1, \ldots, m. \end{cases} \tag{3.4.4}$$

Osservazione 3.4.17 È chiaro che per le ipotesi fatte dovrà risultare

$$\sum_{i=1}^{m} \sum_{j=1}^{n} x_{ij} = \sum_{j=1}^{n} \sum_{i=1}^{m} x_{ij} = \sum_{i=1}^{m} a_i = \sum_{j=1}^{n} b_j.$$

Esaminiamo, ora, un risultato che è una condizione necessaria e sufficiente affinché un generico problema dei trasporti scritto nella forma (3.4.4) con $a_i \geq 0$ e $b_j \geq 0$ abbia soluzione; tale risultato chiarisce perché nella formulazione classica del problema dei trasporti si adotta l'ipotesi (3.4.1) cioè che la disponibilità complessiva uguagli la richiesta complessiva.

Teorema 3.4.1 *Condizione necessaria e sufficiente affinché esista una soluzione ammissibile per problema (3.4.4), è che risulti*

$$\sum_{i=1}^{m} a_i = \sum_{j=1}^{n} b_j. \tag{3.4.5}$$

Dimostrazione: Dimostriamo innanzitutto la necessità, cioè che se esiste una soluzione ammissibile che denotiamo con $\bar{x}_{ij}$, $i = 1, \ldots, m$, $j = 1, \ldots, n$, allora la condizione (3.4.5) deve essere verificata; poiché $\bar{x}_{ij}$ deve soddisfare i vincoli, dalle equazioni dei vincoli nella (3.4.4) si ottiene

$$\sum_{i=1}^{m} \sum_{j=1}^{n} \bar{x}_{ij} = \sum_{i=1}^{m} a_i$$

$$\sum_{j=1}^{n} \sum_{i=1}^{m} \bar{x}_{ij} = \sum_{j=1}^{n} b_j,$$

e sottraendo membro a membro si ha

$$\sum_{i=1}^{m} a_i - \sum_{j=1}^{n} b_j = 0$$

che è la (3.4.5).
Dimostriamo ora la sufficienza; supponiamo quindi che valga la (3.4.5) e poniamo

$$\sum_{i=1}^{m} a_i = \sum_{j=1}^{n} b_j = A.$$

Si vuole allora dimostrare che esiste una soluzione ammissibile; infatti, sia $\bar{x}_{ij} := \dfrac{a_i b_j}{A}$, $i = 1, \ldots, m$, $j = 1, \ldots, n$; allora $\bar{x}_{ij}$ ora definito è una soluzione ammissibile per il problema dei trasporti. Infatti risulta innanzitutto $\bar{x}_{ij} \geq 0$ per ogni $i = 1, \ldots, m$ e $j = 1, \ldots, n$ per la non negatività degli a_i e dei b_j; inoltre

$$\sum_{j=1}^{n} \bar{x}_{ij} = \sum_{j=1}^{n} \frac{a_i b_j}{A} = \frac{a_i \sum_{j=1}^{n} b_j}{A} = a_i$$

$$\sum_{i=1}^{m} \bar{x}_{ij} = \sum_{i=1}^{m} \frac{a_i b_j}{A} = \frac{b_j \sum_{i=1}^{m} a_i}{A} = b_j$$

e quindi $\bar{x}_{ij}$ soddisfando i vincoli del problema è una soluzione ammissibile. □

Il teorema appena dimostrato garantisce quindi che, se è soddisfatta l'ipotesi (3.4.1) allora il problema dei trasporti ammette sempre soluzione.

Osservazione 3.4.18 La soluzione ammissibile del teorema, ovviamente, *non è l'unica soluzione* ammissibile del problema.

Riportiamo di seguito, senza dimostrazione, un altro risultato di fondamentale importanza nella trattazione del problema dei trasporti.

Teorema 3.4.2 *Se nel problema dei trasporti le* a_i, $i = 1, \ldots, m$ *e le* b_j, $j = 1, \ldots, n$ *sono intere e se il problema ammette soluzione ottima, allora ha una soluzione ottima intera.*

Passiamo, ora, ad analizzare alcune varianti della formulazione classica del problema dei trasporti; può infatti accadere che non tutte le rotte di trasporto siano disponibli: se non è possibile il trasporto da una certa origine $\mathbf{O_i}$ ad una destinazione $\mathbf{D_j}$ si pone, per convenzione, $c_{ij} = \infty$. Oppure possono esistere rotte di trasporto in cui vi sono limitazioni sulle quantità massima di merci trasportabili. Infine, si può supporre che la disponibilità complessiva possa essere superiore alla domanda cioè

$$\sum_{i=1}^{m} a_i \geq \sum_{j=1}^{n} b_j. \tag{3.4.6}$$

In tal caso, possono essere ammesse giacenze nelle origini e/o nelle destinazioni; se si accetta di avere giacenze nelle origini, allora i vincoli di origine diventano

$$\sum_{j=1}^{n} x_{ij} \leq a_i \qquad i = 1, \ldots, m;$$

se si accetta di avere giacenze nelle destinazioni, allora i vincoli di destinazione diventano

$$\sum_{i=1}^{m} x_{ij} \geq b_j \qquad j = 1, \ldots, n.$$

nel caso in cui vale la (3.4.6), per porre il problema dei trasporti nella sua formulazione classica, cioè con vincoli di uguaglianza, si può introdurre una destinazione fittizia che abbia una richiesta pari a

$$\sum_{i=1}^{m} a_i - \sum_{j=1}^{n} b_j$$

ponendo uguale a zero il costo per raggiungere questa destinazione fittizia da qualsiasi origine.

4

Il linguaggio di modellizzazione algebrica AMPL

Come ampiamente discusso, l'*approccio modellistico* rappresenta un potente "strumento" per la soluzione di un problema di decisione. In particolare, rappresentare un problema attraverso un modello di Programmazione Matematica permette di utilizzare i numerosi solutori disponibili che sono in grado di risolvere efficientemente problemi anche di dimensioni molto elevate. Tuttavia pur disponendo di efficienti algoritmi di soluzione, sarà necessario trasferire al solutore il problema rappresentato dal modello di Programmazione Matematica. Ovvero è necessario implementare il modello in una qualche forma riconoscibile dai vari solutori. Gli strumenti che svolgono questa funzione vengono chiamati *generatori algebrici di modelli* e, ormai da diversi anni, ne sono stati messi a punto alcuni che hanno avuto una vasta diffusione. Uno di questi è AMPL (acronimo di *A Mathematical Programming Language*) ed è, ad oggi, uno dei più largamente utilizzati per la formulazione di problemi di Programmazione Lineare, Programmazione Non Lineare e Programmazione Lineare Intera e mista. Esso ha la funzione di interfaccia tra il modello algebrico ed il solutore numerico e rappresenta un potente linguaggio di modellazione algebrica, ovvero un linguaggio che contiene diverse primitive per esprimere le notazioni normalmente utilizzate per formulare un problema di Programmazione Matematica: sommatorie, funzioni matematiche elementari, operazioni tra insiemi, etc. Utilizza un'architettura molto avanzata, fornendo un'ottima flessibilità d'uso; infatti, la naturale notazione algebrica utilizzata permette di implementare modelli anche complessi e di dimensioni molto elevate in una forma assai concisa e facilmente comprensibile. È di fatto un linguaggio ad alto livello e non fa uso di particolari strutture dati perché utilizza solamente file di testo sia per l'implementazione del modello sia per la memorizzazione dei dati. In aggiunta, per affermare l'esigenza già ricordata di rendere

un modello indipendente dai dati, permette di tenere distinta la struttura del modello dai dati numerici che sono memorizzati separatamente. Inoltre, permette di richiamare al suo interno diversi tra i più efficienti solutori per problemi di Programmazione Matematica di cui attualmente si dispone. È disponibile per i sistemi operativi Windows e Linux e tutte le informazioni sono disponibili sul sito `http://www.ampl.com` (nel seguito, si farà esplicito riferimento a sistemi Windows). Gli studenti possono scaricare la "student version" di AMPL dal seguente link `http://www.ampl.com/DOWNLOADS`. Tale versione è limitata nel numero delle variabili e dei vincoli.

Il testo di riferimento è:

> Robert Fourer, David M. Gay, Brian W. Kernighan. AMPL *A Modeling Language For Mathematical Programming*, Second Edition, Duxbury Thomson, 2003

Ulteriore materiale riguardante AMPL e i solutori supportati è disponibile al link `http://www.ampl.com/REFS`.

4.1 INSTALLAZIONE E AVVIO DI AMPL

Come già accennato, gli studenti possono scaricare la "student version" di AMPL dal link `http://www.ampl.com/DOWNLOADS`. Sono presenti la versione per Windows e per Unix/Linux. Si tratta di una versione limitata nel numero delle variabili e dei vincoli (sul sito AMPL è anche disponibile una versione di prova completa con una "trial license" valida 30 giorni). Per quanto riguarda la versione Windows, andrà scaricato il file `amplcml.zip` che è un archivio contenente tutti i file necessari. Una volta estratto l'archivio in una directory, saranno creati due directory (MODELS e TABLES) che contengono numerosi utili esempi e dei file eseguibili. In particolare il file `ampl.exe` è l'eseguibile di AMPL che chiamato dal prompt di comandi avvierà il programma. Gli altri file eseguibili `cplex.exe`, `gurobi.exe`, `minos.exe` e `lpsolve.exe` sono i solutori disponibili, rispettivamente CPLEX 12.5, Gurobi 2.0, MINOS 5.5 e LP_SOLVE 4.0. Si tratta di una versione a linea di comando e quindi deve essere utilizzata dalla finestra del *prompt di comandi* (finestra nera). Quindi aprire tale finestra, portarsi nella directory dove è stato estratto il file `amplcml.zip` (oppure aggiungere questa directory nel `PATH` di Windows) ed avviare il programma digitando il comando `ampl`. Apparirà così il prompt dei comandi di AMPL:

```
ampl:
```

A questo punto si è nell'ambiente AMPL, ovvero possono essere digitati i comandi di AMPL. Alternativamente, per avviare AMPL si può cliccare due volte su sw.exe (*scrolling-window utility*), digitando poi `ampl` sul prompt che appare.

4.2 UN PRIMO ESEMPIO

In linea generale (anche se assai poco pratico) un modello potrebbe essere inserito dal prompt di comandi di AMPL, ovvero digitando, ad esempio, sulla riga di comando i seguenti comandi in successione:

```
ampl: var x1;
ampl: var x2;
ampl: maximize funzione_obiettivo: x1 + x2;
ampl: subject to vincolo1: x1 + x2 <= 1;
ampl: subject to vincolo2: x1 - x2 <= 2;
ampl: subject to x1_non_neg: x1 >= 0;
ampl: subject to x2_non_neg: x2 >= 0;
```

Vedremo più avanti come sia possibile scrivere tali comandi in un file, ma per il momento soffermiamoci sull'analisi dei comandi AMPL utilizzati nell'esempio e sulle parole chiave del linguaggio. Inanzitutto osserviamo che

- ogni comando termina con un "`;`"
- sono presenti due variabili (x_1 e x_2) e queste sono dichiarate con i comandi `var x1;` e `var x2;`
- la funzione obiettivo è introdotta dalla parola chiave `maximize` in quanto trattasi di un problema di massimizzazione (per problemi di minimizzazione la parola chiave sarà `minimize`). Tale parola chiave è seguita da una etichetta proposta dall'utente (nel caso dell'esempio è `funzione_obiettivo`) seguita da "`:`" che introduco all'espressione della funzione obiettivo
- i vincoli sono elencati di seguito: ciascun vincolo è introdotto dalla parola chiave `subject to` (che può anche esser abbreviata in `s.t.`), seguita dall'etichetta che si vuole dare al vincolo e da "`:`" dopo il quale è riportata l'espressione del vincolo.

Questo semplice esempio ci ha permesso di introdurre la struttura tipica di un modello nel linguaggio AMPL, struttura che ricalca fedelmente quella standard di un modello di Programmazione Matematica come riportato nella Tabella **??**. Ovvero, dopo aver introdotto le *variabili di decisione*, deve essere formalizzata la *funzione obiettivo* e i *vincoli*.
Ora che abbiamo il modello implementato in AMPL, naturalmente viene spontaneo domandarsi come può essere determinata la soluzione del problema di ottimizzazione rappresentato dal modello (che nel caso dell'esempio è un problema

di Programmazione Lineare). Come già detto da AMPL sono disponibili alcuni solutori. Nella directory dove risiede AMPL oltre al file `ampl.exe` sono presenti altri file eseguibili che rappresentano i solutori; essi sono CPLEX,GUROBI, MINOS e LPSOLVE. Ciascuno di essi risolve alcune classe di problemi di Programmazione Matematica. Nel seguito riporteremo nel dettaglio le tipologie di problemi risolte da ciascun solutore. Ora vogliamo introdurre il comando per selezionare il solutore da utilizzare, ovvero

`option solver` *nome_solutore*;

Quindi digitando:

```
ampl: option solver cplex;
```

stiamo comunicando ad AMPL di voler utilizzare il solutore CPLEX. Quest'ultimo è un solutore per problemi di Programmazione Lineare (e non solo) e può quindi essere utilizzato per risolvere il problema riportato nell'esempio.
A questo punto è sufficiente dare il comando per risolvere il problema che è

`solve;`

Digitando questo comando al promtp dei comandi, ovvero

```
ampl: solve;
```

si ottiene il seguente messaggio:

```
CPLEX 12.5.1.0: optimal solution; objective 1;
0 dual simplex iterations (0 in phase I)
```

con il quale AMPL ci comunica che il solutore ha determinato una soluzione ottima con valore della funzione obiettivo pari a 1. Per vedere il valore delle variabili all'ottimo è necessario un altro comando, ovvero

`display` *nome_variabile*;

Quindi digitando, ad esempio,

```
ampl: display x1;
```

otteniamo il valore di x_1^*. Analogamente per l'altra variable x_2^*. Abbiamo così ottenuto il punto di massimo che stavamo cercando.

Scrivere il modello utilizzando la linea di comando, ovviamente, è piuttosto scomodo, soprattutto perché in questo modo, una volta usciti da AMPL il modello è completamente perso. Comviene, pertanto, scrivere il modello in un file testo che deve avere estensione `.mod`. Quindi utilizzando un qualsiasi editor di testo possiamo riscrivere il modello nel seguente modo:

esempio.mod

```
var x1;
var x2;

maximize funzione-obiettivo: x1 + x2;

subject to vincolo1: x1 + x2 <= 1;
subject to vincolo2: x1 - x2 <= 2;
subject to x1_non_neg: x1 >= 0;
subject to x2_non_neg: x2 >= 0;
```

A questo punto, dal prompt di AMPL è sufficiente scrivere il seguente comando per leggere il file del modello:

`model` *<PATH>\nome_file.mod*

Quindi, assumendo che la directory dove risiede il file del modello `esempio.mod` sia *C:\MODELLI*, digitando,

```
ampl: model C:\MODELLI\esempio.mod;
```

carichiamo il modello. A questo punto, si procede come descritto in precedenza per risolvere il problema e stampare i risultati ottenuti.

Si può inoltre creare un file contenente i comandi da dare (in modo da non doverli digitare ogni volta). Tale file deve avere estensione `.run`. Un esempio di questo file relativamente all'esempio fino ad ora esaminato è il seguente:

esempio.run

```
reset;
model C:\MODELLI\esempio.mod;
option solver cplex;
solve;
display x1;
display x2;
display funzione_obiettivo;
```

Si noti che nel file è stato aggiunto il comando `reset;` che ha la funzione di eliminare da AMPL dati relativi ad un modello precedentemente risolto. È conveniente introdurlo all'inzio di ogni file `.run`.

Per lanciare il file `.run` *nell'ambiente* AMPL si utilizza il comando `include esempio.run;`. Alternativamente, *fuori dell'ambiente* AMPL è sufficiente avere tale file nella directory che contiene AMPL (oppure in un'altra directory se si è introdotto nel *PATH* il percorso dove risiede AMPL), aprire un terminale del prompt dei comandi in tale directory e dare il comando `ampl esempio.run`.

Infine, per uscire dall'ambiente AMPL è sufficiente il comando `quit;`.

4.3 I SOLUTORI

Abbiamo già menzionato il fatto che alcuni solutori sono disponibili nell'istallazione AMPL "student version". Essi sono:

- CPLEX
- GUROBI
- LPSOLVE
- MINOS.

CPLEX.
Risolve problemi di Programmazione Lineare e problemi di Programmazione Quadratica convessi utilizzando il metodo del simplesso e metodi a punti interni. Inoltre risolve anche problemi di Programmazione Lineare e problemi di Programmazione Quadratica convessi con variabili intere utilizzando procedure di tipo Branch-and-Bound. La versione distribuita con la "student version" di AMPL è limitata a 500 variabili e 500 vincoli.

GUROBI.
Risolve problemi di Programmazione Lineare con il metodo del simplesso e con metodi a punti interni. Risolve inoltre problemi di Programmazione Lineare Misti Interi utilizzanzo procedure di tipo Branch-and-Bound. Risolve inoltre anche problemi di Programmazione Programmazione Quadratica e problemi di Programmazione Quadratica Misti Interi. Utilizzando la "student version" di AMPL, GUROBI limita la dimensione dei problemi a 500 variabili e 500 vincoli.

LPSOLVE.
Risolve problemi di Programmazione Lineare e Programmazione Lineare Intera di dimensioni moderate.

MINOS.
Risolve problemi di Programmazione Lineare attraverso il metodo del simplesso e problemi di Programmazione Non Lineare utilizzando metodi di tipo gradiente ridotto.

Esistono altri solutori dei quali è stato previsto l'uso con AMPL, ma non tutti sono disponibili gratuitamente.

4.4 ALCUNI ESEMPI DI MODELLI DI PROGRAMMAZIONE LINEARE

Esaminiamo ora alcuni semplici modelli di Programmazione Lineare costruendo prima la formulazione algebrica e poi realizzandone un'implementazione in AMPL. Iniziamo con il modello dell'Esempio 2.3.1 già visto nel paragrafo 2.3.1.

Si riporta la formulazione finale ottenuta

$$\begin{cases} \max\left(250x_1 + 230x_2 + 110x_3 + 350x_4\right) \\ 2x_1 + 1.5x_2 + 0.5x_3 + 2.5x_4 \leq 100 \\ 0.5x_1 + 0.25x_2 + 0.25x_3 + x_4 \leq 50 \\ x_1 \geq 0, x_2 \geq 0, x_3 \geq 0, x_4 \geq 0 \end{cases}$$

e di seguito il file `.mod` che implementa questo modello.

fertilizzanti.mod

```
var x1;
var x2;
var x3;
var x4;

maximize profitto: 250*x1+230*x2+110*x3+350*x4;

subject to vincolo1: 2*x1+1.5*x2+0.5*x3+2.5*x4 <= 100;
s.t. vincolo2: 0.5*x1+0.25*x2+0.25*x3+x4<= 50;
s.t. x1_nonneg: x1 >=0;
s.t. x2_nonneg: x2 >=0;
s.t. x3_nonneg: x3 >=0;
s.t. x4_nonneg: x4 >=0;
```

Esaminiamo ora un altro modello già visto, ed in particolare quello riguardante la produzione multi-plant dell'Esempio 3.4.5

Si riportano sinteticamente le formulazioni nei due casi.

Formulazione del caso (a)
Questo caso, nella pratica, corrisponde a costruire due modelli indipendenti: uno riferito al primo impianto, uno riferito al secondo impianto. Una "risorsa" (il materiale grezzo) è già allocata a priori.

Impianto 1: La formulazione relativa al primo impianto è:

$$\begin{cases} \max(10x_1 + 15x_2) \\ 4x_1 + 4x_2 \leq 75 \\ 4x_1 + 2x_2 \leq 80 \\ 2x_1 + 5x_2 \leq 60 \\ x_1 \geq 0, x_2 \geq 0 \end{cases}$$

A questo punto, prima di passare all'impianto 2, vediamo come fare per scrivere in AMPL il precedente problema di PL. Per fare questo è sufficiente creare il seguente file di modello (un semplice file di testo ma con estensione `.mod`) ad esempio `impianto1.mod`.

`impianto1.mod`

```
var x1;
var x2;

maximize profitto: 10*x1 + 15*x2;

subject to   m_grezzo: 4*x1 + 4*x2 <= 75;
subject to levigatura: 4*x1 + 2*x2 <= 80;
s.t.         pulitura: 2*x1 + 5*x2 <= 60;
s.t.         x1_non_neg: x1 >= 0;
s.t.         x2_non_neg: x2 >= 0;
```

Risolvendo il problema, si ottiene che all'ottimo l'impianto 1 ha un profitto di 225 Euro, ottenuto fabbricando $11,25$ unità di $\mathbf{P_1}$ e $7,5$ di $\mathbf{P_2}$.

Impianto 2: La formulazione relativa al secondo impianto è:

$$\begin{cases} \max(10x_3 + 15x_4) \\ 4x_3 + 4x_4 \leq 45 \\ 5x_3 + 3x_4 \leq 60 \\ 5x_3 + 6x_4 \leq 75 \\ x_3 \geq 0, x_4 \geq 0 \end{cases}$$

Il modello in AMPL per l'impianto 2 è:

impianto2.mod

```
var x3;
var x4;

maximize profitto: 10*x3 + 15*x4;

subject to   m_grezzo: 4*x3 + 4*x4 <= 45;
subject to levigatura: 5*x3 + 3*x4 <= 60;
s.t.         pulitura: 5*x3 + 6*x4 <= 75;
s.t.         x3_non_neg: x3 >= 0;
s.t.         x4_non_neg: x4 >= 0;
```

Risolvendo il problema si ha che all'ottimo l'impianto 2 ha un profitto di $168,75$ Euro ottenuto fabbricando $11,25$ unità di $\mathbf{P_2}$ e nessuna unità di $\mathbf{P_1}$.

Formulazione del caso (b)
Questo caso corrisponde a costruire un unico modello comprendente entrambi gli impianti. L'allocazione della "risorsa" data dal materiale grezzo è lasciata al modello stesso.

La formulazione relativa a questo caso è:

$$\begin{cases} \max & (10x_1 & + & 15x_2 & + & 10x_3 & + & 15x_4) & & \\ & 4x_1 & + & 4x_2 & + & 4x_3 & + & 4x_4 & \leq & 120 \\ & 4x_1 & + & 2x_2 & & & & & \leq & 80 \\ & 2x_1 & + & 5x_2 & & & & & \leq & 60 \\ & & & & & 5x_3 & + & 3x_4 & \leq & 60 \\ & & & & & 5x_3 & + & 6x_4 & \leq & 75 \\ & x_1 \geq 0, & & x_2 \geq 0, & & x_3 \geq 0, & & x_4 \geq 0 & & \end{cases}$$

e in AMPL si avrà:

impianto1e2.mod

```
var x1;
var x2;
var x3;
var x4;

maximize profitto: 10*(x1+x3) + 15*(x2+x4);
```

```
s.t. m_grezzoTOT:   4*(x1+x2+x3+x4) <= 120;
s.t. levigatura1:   4*x1 + 2*x2 <= 80;
s.t. pulitura1:     2*x1 + 5*x2 <= 60;
s.t. levigatura2:   5*x3 + 3*x4 <= 60;
s.t. pulitura2:     5*x3 + 6*x4 <= 75;

s.t. x1_non_neg:        x1 >= 0;
s.t. x2_non_neg:        x2 >= 0;
s.t. x3_non_neg:        x3 >= 0;
s.t. x4_non_neg:        x4 >= 0;
```

Risolvere questo problema otteniamo un profitto complessivo di $404,17$ Euro ottenuto fabbricando nell'impianto 1: $9,17$ unità di $\mathbf{P_1}$ e $8,33$ unità di $\mathbf{P_2}$ e fabbricando, nell'impianto 2, solo $12,5$ unità di $\mathbf{P_2}$. Come abbiamo già visto, è importante notare che il profitto totale in questo caso ($404,17$ Euro) è superiore alla somma dei profitti ottenuti considerando separatamente i due modelli e che l'impianto 1 ora usa 70Kg di materiale grezzo (contro i 75Kg del caso precedente) mentre l'impianto 2 ne usa 50Kg (contro i 45Kg del caso precedente).

4.5 GLI INSIEMI E I PARAMETRI IN AMPL

AMPL permette di scrivere il modello in forma parametrica. Questo, nella sostanza, significa scrivere il modello nel file `.mod` senza specificare i dati che vengono invece scritti separatamente in un file `.dat`. Praticamente nel file dei dati si effettuano le *dichiarazioni*, mentre nel file dei dati si effettuano le *assegnazioni*. Fatto questo, dal prompt di AMPL oltre al comando

`model` $<PATH> \setminus nome\_file.mod$

utilizzato per caricare il modello, sarà necessario un comando per caricare il file dei dati. Tale comando è:

`data` $<PATH> \setminus nome\_file.dat$

Quindi, assumendo che la directory dove risiedono i file del modello (`esempio.mod`) e dei dati (`esempio.dat`) sia $C:\setminus MODELLI$, digitando anche

```
ampl: data C:\MODELLI\esempio.dat;
```

carichiamo anche i dati del modello. A questo punto, si procede come descritto in precedenza per risolvere il problema e stampare i risultati ottenuti.

Per introdurre l'uso di insiemi e parametri, consideriamo di nuovo i modelli presentati nel paragrafo precedente riguardanti i due impianti di produzione. Infatti, AMPL consente di scrivere il problema in modo diverso, utilizzando un concetto molto utilizzato in AMPL: gli *insiemi*. Grazie ad esso, è possibile tenere separati il modello dai dati. Il modello relativo all'impianto 1 può essere infatti riscritto come segue:

impianto.mod

```
set Prodotti;
set Risorse;

param q_max{Risorse} >=0;
param richiesta{Risorse,Prodotti} >=0;
param prezzo{Prodotti} >=0;

var x{Prodotti} >=0;

maximize profitto: sum{i in Prodotti} prezzo[i]*x[i];
```

```
s.t. vincolo_risorsa {j in Risorse}:
sum{i in Prodotti} richiesta[j,i]*x[i] <= q_max[j];
```

Ora analiziamo le istruzione del file `impianto.mod`. Anzitutto notiamo le istruzioni

```
set Prodotti;
set Risorse;
```

con le quali si dichiarano `Prodotti` e `Risorse` come insiemi di un numero imprecisato di elementi (prodotti e risorse). Subito dopo abbiamo tre istruzioni che ci servono per definire altrettanti parametri del modello. La prima di queste

```
param q_max{Risorse}>=0;
```

definisce un vettore di parametri con tante componenti quanti sono gli elementi dell'insieme `Risorse` e definisce le quantità massime disponibili di ciascuna risorsa. Il $>= 0$ specifica che i valori immessi devono essere non negativi, AMPL eseguirà automaticamente il controllo sui dati.
L'istruzione successiva ovvero

```
param richiesta{Risorse,Prodotti}>=0;
```

definisce invece una matrice di parametri con tante righe quanti sono le risorse e tante colonne quante sono i prodotti, specificando per ogni prodotto la quantità di risorsa necessaria alla sua realizzazione.

L' istruzione

```
param prezzo{Prodotti} >=0;
```

definisce un vettore di parametri con tante componenti quanti sono gli elementi dell'insieme `Prodotti` specificando il prezzo di ogni prodotto.

La funzione obiettivo `profitto` è definita dalla istruzione

```
maximize profitto: sum{i in Prodotti} prezzo[i]*x[i];
```

I vincoli sono definiti atttraverso l'istruzione che segue:

```
s.t. vincolo_risorsa {j in Risorse}:
     sum{i in Prodotti} richiesta[j,i]*x[i] <= q_max[j];
```

Con essa si definiscono tanti vincoli quanti sono gli elementi dell'insieme `Risorse`.

A questo punto, completato il file della definizione del modello (`.mod`) è necessario definire un file con estensione `.dat` dove vengono specificati i valori per gli insiemi e i parametri del modello. Per quanto riguarda l'impianto 1 si avrà:

impianto1.dat

```
set Prodotti := P1 P2;
set Risorse  := m_grezzo levigatura pulitura;

param         q_max :=
m_grezzo        75
levigatura      80
pulitura        60;

param  richiesta: P1   P2 :=
m_grezzo          4     4
levigatura        4     2
pulitura          2     5;

param  prezzo :=
P1       10
P2       15;
```

Facciamo notare che in questo modo, avendo cioè a disposizione il file `impianto.mod` contenente il modello separato dai dati del problema (contenuti invece nel file `impianto1.dat`), possiamo risolvere differenti problemi di massimizzazione del profitto semplicemente cambiando i dati contenuti nel file con estensione `.dat`. Sfruttando questo fatto è banale risolvere il problema relativo all'impianto 2: si utilizza lo stesso file `.mod` e un diverso file `.dat`, che è il seguente:

impianto2.dat

```
set Prodotti := P1 P2;
set Risorse  := m_grezzo levigatura pulitura;

param         q_max :=
m_grezzo        45
levigatura      60
pulitura        75;


param  richiesta: P1   P2 :=
m_grezzo          4     4
levigatura        5     3
pulitura          5     6;
```

```
param  prezzo :=
P1       10
P2       15;
```

Nello stesso modo, cioè specificando il solo file dei dati, è possibile risolvere il problema completo:

impianto.dat

```
set Prodotti :=
       P1_impianto1
       P2_impianto1
       P1_impianto2
       P2_impianto2;

set Risorse :=
       m_grezzo
       levigatura1
       pulitura1
       levigatura2
       pulitura2;

param         q_max :=
      m_grezzo         120
      levigatura1      80
      pulitura1        60
      levigatura2      60
      pulitura2        75;


param  richiesta: P1_impianto1   P2_impianto1  P1_impianto2  P2_impianto2:=
 m_grezzo          4                4             4               4
 levigatura1       4                2             0               0
 pulitura1         2                5             0               0
 levigatura2       0                0             5               3
 pulitura2         0                0             5               6;
```

```
param          prezzo :=
        P1_impianto1       10
        P2_impianto1       15
        P1_impianto2       10
        P2_impianto2       15;
```

Si può ovviamente modificare il file `.run` scritto in precedenza aggiungendo l'istruzione necessaria per caricare il file dei dati.

È importante ribadire che nel file `.mod` vengono effettuate le *dichiarazioni* di insiemi e parametri, mentre nel file `.dat` vengono effettuate le *assegnazioni*.

Esaminiamo nel dettaglio nel paragrafi che seguono gli elementi base della sintassi di AMPL riguardante *insiemi*, *parametri*, *variabili*, *funzione obiettivo* e *vincoli*.

4.5.1 Gli insiemi

Gli insiemi definiscono gli elementi di base con i quali si possono indicizzare variabili, parametri e vincoli del modello.

La dichiarazione di un *insieme generico* (nel file `.mod`) si effettua come segue:

```
set NomeInsieme;
```

L'assegnazione di un *insieme generico* (nel file `.dat`) si effettua come segue:

```
set NomeInsieme := e1 e2 e3 e4 e5 ;
```

Questa istruzione specifica che l'insieme `NomeInsieme` è composto dagli elementi $\{e_1, e_2, e_3, e_4, e_5\}$. Gli insiemi così definiti corrispondono agli insiemi generici. Esiste la possibilità di definire altri tipi di insiemi:

- *insiemi ordinati*

  ```
  set NomeInsieme ordered;
  ```

- *insiemi numerici*

  ```
  set NomeInsieme := 1 .. N;
  set NomeInsieme := 1 .. N by p;
  ```

- *insiemi ordinati e ciclici*

  ```
  set NomeInsieme circular;
  ```

Per quanto riguarda gli insiemi numerici la notazione `NomeInsieme := 1 .. N` definisce tutti i numeri interi tra 1 e N, mentre `NomeInsieme := 1 .. N by p` definisice tutti i numeri interi tra 1 e N distanti fra di loro di p numeri. Si osservi, quindi, che per quanto riguarda gli insiemi numerici la dichiarazione di fatto è un'assegnazione e quindi l'assegnazione di un insieme numerico *non* deve essere ripetuta nel file `.dat`.

Riportiamo ora gli operatori e le funzioni più comuni tra due **insiemi generici**:

Operatore/Funzione	**Significato**
`A union B`	insieme di elementi che stanno in A o B
`A inter B`	insieme di elementi che stanno sia in A che in B
`A within B`	A sottoinsieme di B
`A diff B`	insieme di elementi che stanno in A ma non in B
`A symdiff B`	insieme di elementi che stanno in A o in B ma non in entrambi
`card(A)`	numero di elementi che stanno in A

Di seguito gli operatori e le funzioni più comuni tra due **insiemi ordinati**:

Funzione	Significato
`first(A)`	primo elemento di A
`last(A)`	ultimo elemento di A
`next(a,A)`	elemento di A dopo a
`prev(a,A)`	elemento di A prima di a
`next(a,A,k)`	k-esimo elemento di A dopo a
`prev(a,A,k)`	k-esimo elemento di A prima di a
`ord(a,A)`	posizione di a in A
`ord0(a,A)`	come `ord(a,A)` ma restituisce 0 se a non è in A
`member(k,A)`	elemento di A in k-esima posizione

È inoltre possibile definire insiemi in modo implicito, ovvero senza specificarne il nome, ma solo indicando la loro espressione. Alcuni esempi sono i seguenti: dati A e B due insiemi,

```
{A,B}
{a in A, b in B}
```

sono due espressioni equivalenti per definire il prodotto cartesiano $A \times B$, ovvero l'insieme di tutte le coppie ordinate (a, b) con $a \in A$ e $b \in B$. Un altro esempio è dato dall'espressione

```
{a in A : prezzo[a]>=5}
```

che sta ad indicare tutti gli elementi a dell'insieme A tali che il `prezzo` di a sia maggiore o uguale a 5, dove, come vedremo più avanti, `prezzo` è un parametro indicizzato sull'insieme A.

Gli insiemi multidimensionali

In AMPL è possibile definire insiemi a più dimensioni. Tale dimesione deve essere specificata nella dichiarazione di insieme (file `.mod`). La dicharazione

```
set INSIEME dimension p;
```

si usa per indicare che l'insieme `INSIEME` è costituito da $p-$uple *ordinate*, ovvero i suoi elementi sono della forma $(a_1, a_2, \ldots, a_p)$. Un esempio potrebbe essere:

```
set TRIPLE dimension 3;
```

che indica che l'insieme `TRIPLE` è formato da tutte triple ordinate di valori che verranno poi specificate nel file `.dat`, ad esempio, nel seguente modo:

```
set TRIPLE := (a,b,c) (d,e,f) (g,h,i) (l,m.n);
```

oppure

```
set TRIPLE :=
a b c
d e f
g h i
l m n;
```

Un modo alternativo di ottenere insiemi multidimensionali di dimesnione p è l'utilizzazione del prodotto cartesiano di p insiemi. Il prodotto cartesiano in AMPL si indica con `cross`. Quindi se, ad esempio, A, B e C sono stati già dichiarati come insiemi, l'istruzione

```
set TRIPLE := A cross B cross C;
```

indica l'insieme delle triple ordinate (a, b, c) con $a \in A$, $b \in B$ e $c \in C$.

Partendo da un insieme multidimensionale è possibile ottenere gli insiemi componenti l'insieme multidimensionale effettuando un procedimento inverso al precedente. Con riferimento all'esempio precedente, dall'insieme `TRIPLE` si possono ottenere i tre insiemi di partenza nel seguente modo con l'uso di `setof`:

```
set A := setof{(i,j,k) in TRIPLE} i;
set B := setof{(i,j,k) in TRIPLE} j;
set C := setof{(i,j,k) in TRIPLE} k;
```

4.5.2 I parametri

I parametri rappresentano i dati di un problema. Una volta che essi vengono assegnati (nel file `.dat`), essi non vengono in nessun caso modificati dal solutore. Un parametro deve essere *dichiarato* nel file `.mod` e *assegnato* nel file `.dat`. L'istruzione

```
param t;
```

utilizzata nel file `.mod` effettua la dichiarazione del parametro t. È possibile anche dichiarare vettori e matrici di parametri. Quindi se `PROD` e `ZONA` sono due insiemi le istruzioni

```
set PROD;
set ZONA;
param T;
param costi{PROD};
param prezzo{PROD,ZONA};
param domanda{PROD,ZONA,1..T};
```

oltre che dichiarare gli insiemi, definiscono:

- il parametro `T`;

- il parametro `costi` indicizzato dall'insieme `PROD`, ovvero `costi` è un vettore che ha tante componenti quanti sono gli elementi di `PROD`;
- il parametro a due dimensioni `prezzo`, indicizzato dagli insiemi `PROD` e `ZONA`;
- il parametro a tre dimensioni `domanda`, indicizzato dagli `PROD`, `ZONA` e dall'insieme dei numeri interi che vanno da 1 a T.

L'assegnazione dei parametri avviene nel file `.dat`. Un esempio di assegnazione dei parametri prima introdotti è il seguente:

```
set PROD := p1 p2;
set ZONA := z1 z2;
param T := 2;

param costi :=
   p1 5
   p2 4;

param prezzo: z1    z2  :=
   p1          2     7
   p2          5     9;

param domanda:=
  [*,*,1] :   z1    z2:=
     p1       10    15
     p2       13    22

  [*,*,2] :   z1    z2:=
     p1       32    25
     p2       18    15;
```

Si osservi innanzitutto, che la scelta di scrivere i dati incolonnati ha il solo scopo di rendere il file più leggibile: nessuna formattazione particolare è richiesta da AMPL. Inoltre è opportuno soffermarci sull'uso dei ":"; infatti i ":" sono obbligatori se si assegnano valori a due o più vettori di parametri monodimensionali indicizzati sullo stesso insieme, come nel caso di `prezzo` e `domanda`.

Il parametro `domanda` può essere alternativamente assegnato nel seguente modo:

```
param : domanda :=
   p1  z1  1  10
   p1  z1  2  32
   p1  z2  1  15
   p1  z2  2  25
```

```
p2  z1  1  13
p2  z1  2  18
p2  z2  1  22
p2  z2  2  15;
```

Nella dichiarazione dei parametri (file `.mod`) si possono anche includere controlli o restrizioni sui parametri stessi. Ad esempio, le istruzioni

```
param T > 0;
param N integer, <= T;
```

controllano che il parametro T sia positivo e che il parametro N sia un numero intero minore o uguale a T. Esiste anche l'istruzione `check` per effettuare controlli contenenti espressioni logiche. Un esempio potrebbe essere il seguente:

```
set PROD;
param offertatot >0;
param offerta{PROD} >0;
check: sum{p in PROD} offerta[p]=offertatot;
```

Esiste, infine, la possibilità di dichiarare parametri "calcolati" come nel seguente esempio:

```
set PROD;
param offerta{PROD};
param offertatot:=sum{p in PROD} offerta[p];
```

4.5.3 Le variabili

Le variabili rappresentano le incognite del problema e il loro valore è calcolato dal solutore. Una volta che la soluzione è stata determinta, il valore delle variabili all'ottimo rappresenta la soluzione del problema Si osservi, quindi, la differenza con i parametri: questi ultimi vengono assegnati dall'utente e rimangono costanti; il valore delle variabili cambia durante le iterazioni del solutore. Alle variabili l'utente può eventualmemte (ma è del tutto opzionale) assegnare dei valori inziali che sono poi modificati dal solutore.

La dichiarazione delle variabili è obbligatoria. Per default, una variabile è considerata *reale*, oppure può essere specificata *intera* o *binaria* (a valori in $\{0, 1\}$). L'esempio che segue riporta la dichiarazione di variabili con la specifica del tipo di variabili:

```
var x;
var n integer;
```

```
var d binary;
```

Analogamente a quanto avviene per i parametri, anche le variabili possono essere indicizzate da insiemi come riportato nel seguente esempio:

```
set PROD;
set OPERAI;
param dom{PROD};
var num{PROD} integer;
var assegnamento{PROD,OPERAI} binary;
```

Anche nel caso delle variabili si possono introdurre dei controlli contestualmente alla loro dichiarazione come riportato nell'esempio seguente:

```
set PROD;
param dom{PROD};
var x >=0;
var quantita{p in PROD} >=0, <= dom[p];
```

È possibile *fissare* una variabile ad un valore mediante l'istruzione `fix` che può essere utilizzata nel file `.dat` o nel file `.run` e NON nel file `.mod`. Quindi data la variabile x, l'istruzione

```
fix x :=4;
```

assegna il valore 4 ad x. In questo caso il solutore *non cambierà il valore di tale variabile* che verrà considerata fissata al valore assegnato. Esiste poi il comando opposto `unfix` che sblocca una variabile precedentemente fissata. Nel caso dell'esempio si avrebbe

```
unfix x;
```

Infine, c'è la possibilità di inizializzare una variabile ad un determinato valore con il comando `let` da utilizzare nel file `.dat` o nel file `.run` e NON nel file `.mod`. Se scriviamo

```
let x:= 10;
```

stiamo inizializzando la variabile x al valore 10. Questo vuole dire che l'algoritmo risolutivo assegnerà 10 come valore iniziale della variabile x, valore che sarà cambiato dal solutore nel corso delle sue iterazioni. Si osservi, quindi, la differenza fondamentale tra il "fixing" di una variabile che non permette di modificare il valore assegnato a quella variabile e l'assegnazione di un valore iniziale ad una variabile.

4.5.4 La funzione obiettivo e i vincoli

La funzione obiettivo è sempre presente in un modello di Programmazione Matematica e rappresenta ciò che vogliamo massimizzare o minimizzare. Essa deve

essere specificata nel file del modello (`.mod`). La parola chiave del linguaggo AMPL per introdurre la funzione obiettivo è `minimize` o `maximize` a seconda che ci trovi di fronte ad un problema di minimizzazione o massimizzazione. La sintassi è

```
maximize nome_funzione_obiettivo :  espressione_aritmetica ;
```

oppure

```
minimize nome_funzione_obiettivo :  espressione_aritmetica ;
```

Quindi un esempio potrebbe essere:

```
maximize profitto_totale : sum{i in PROD} prezzo[i]*quantita[i];
```

I vincoli sono parte integrante di un modello di Programmazione Matematica e sono specificati anch'essi nel file del modello (`.mod`). La parola chiave è `subject to` che può essere abbreviata in `s.t.`. La sintassi è

```
subject to nome_vincolo :  espressione_aritmetica e/o logica;
```

Se x è una variabile reale, la più semplice espressione di un vincolo potrebbe essere

```
subject to vincolo : x >=0;
```

In realtà, questo tipo di vincoli semplici viene spesso inserito come restrzione nella dichiarazione della variabile x. Anche i vincoli possono essere indicizzati e quindi invece di scrivere tanti vincoli, in una sola espressione si possono esplicitare più vincoli. Un esempio di ciò è il seguente:

```
set PROD;
set REPARTI;

param orelavoro{PROD,REPARTI};
param maxore{REPARTI};
param prezzo{PROD};

var x{PROD};

maximize ricavo : sum{in PROD} prezzo[i]*x[i];

subject to vincoli{j in REPARTI} sum{i in PROD}
                        orelavoro[i,j]*x[i] <= maxore[j];
```

In questo esempio, con un unico vincolo si sono scritti tanti vincoli quanti sono gli elementi dell'insieme `RISORSE` al posto dei vincoli

```
subject to vincolo_1 : sum{i in PROD} orelavoro[i,R1]*x[i]
                            <= maxore[R1];
subject to vincolo_2 : sum{i in PROD} orelavoro[i,R2]*x[i]
                            <= maxore[R2];
   .
   .
   .
   .
subject to vincolo_m : sum{i in PROD} orelavoro[i,Rm]*x[i]
                            <= maxore[Rm];
```

avendo supposto che l'insieme sia composto dagli elementi $\{R_1, R_2, \ldots, R_m\}$. Quest'ultima scrittura non darebbe luogo a messaggi di errore in AMPL, ma oltre che essere molto lunga presenta l'ovvio inconveniente di dover conoscere già nel file del modello (`.mod`) quali sono gli elementi dell'insieme `REPARTI`. Questo contravviene al fatto che un *modello deve essere indipendente dai dati*; infatti, non utilizzando la scrittura indicizzata dei vincoli, un cambio degli elementi dell'insieme `REPARTI` (che ricordiamo è assegnato nel file dei dati (`.dat`)) non permetterebbe più al modello implementato di essere corretto.

4.5.5 Le espressioni

Nella costruzione della funzione obiettivo e dei vincoli, così come nell'imporre condizioni sui parametri e sulle variabili si utilizzano espressioni aritmetiche, funzioni e operatori di diverso tipo. Abbiamo già riportato in precedenza i principali operatori e funzioni sugli insiemi. Le principali funzioni e operatori aritmetici e logici sono riportati nelle tabelle che seguono (per una elenco completo si fa riferimento al testo di AMPL già citato.

Funzione	Significato
`abs(x)`	valore assoluto di x
`sin(x)`	$\sin(x)$
`cos(x)`	$\cos(x)$
`tan(x)`	$\tan(x)$
`asin(x)`	$\arcsin(x)$
`acos(x)`	$\arccos(x)$
`atan(x)`	$\arctan(x)$
`exp(x)`	$\exp(x)$
`sqrt(x)`	radice quadrata di x, $\sqrt{x}$
`log(x)`	logaritmo naturale di x, $\ln(x)$
`log10(x)`	logaritmo in base 10 di x, $\log(x)$
`ceil(x)`	parte intera superiore di x, $\lceil x \rceil$
`floor(x)`	parte intera inferiore di x, $\lfloor x \rfloor$

Operatori aritmetici	Significato
`^`	potenza
`+`	somma
`-`	sottrazione
`*`	prodotto
`/`	divisione
`div`	divisione intera
`mod`	modulo
`sum`	sommatoria
`prod`	produttoria
`min`	minimo
`max`	massimo
$>$, $>=$	maggiore, maggiore o uguale
$<$, $<=$	minore, minore o uguale
`=`	$=$
`<>`, $!=$	diverso

Operatori logici	Significato
`not`	negazione logica
`or`	"or" logico
`and`	"and" logico
`exists`	quantificatore esistenziale logico
`forall`	quantificatore universale logico
`if then else`	espressione condizionale

4.5.6 Due esempi di modelli di Programmazione Lineare

Un problema di pianificazione dei trasporti

Esempio 4.5.1 *Si devono pianificare i trasporti di un tipo di merce da cinque città, Asti, Bergamo, Como, Domodossola, Empoli (città origini) ad altre quattro città, Mantova, Napoli, Olbia, Palermo (città destinazioni). La tabella che segue riporta il costo unitario del traporto della merce da ciascuna città origine a ciascuna città destinazione, insieme alla domanda di merce da soddisfare esattamente di ogni città destinazione e alla quantità massima di merce disponibile presso ciascuna città origine*

	Mantova	Napoli	Olbia	Palermo	*disponibilità max*
Asti	1	3.5	4	4.5	100
Bergamo	0.1	3	4.5	4.8	110
Como	0.3	2.8	4.7	4.9	130
Domodossola	1	2.2	3.9	4	85
Empoli	1.7	2.2	5	5.3	120
domanda	30	18	45	56	

Come si può osservare, la somma dei quantitativi di merce disponibili nelle città origini è maggiore alla somma delle domande delle città destinazione e questo perché nelle città orgini sono disponibili dei depositi dove immagazzinare la merce. Si devono pianificare i trasporti dalle città origini alle città destinazioni in modo da minizzare il costo totale dei trasporti.

Formulazione

Introdotte le varibili di decisione x_{ij}, $i = 1, 2, 3, 4, 5$, $j = 1, 2, 3, 4$, associate alla quantità di merce da trasportare dalla città origine i-esima alla città destinazione j-esima, indicando con c_{ij} il costo unitario del trasporto dalla città origine i-esima alla città destinazione j-esima e indicando rispettivamente con a_i, $i = 1, 2, 3, 4, 5$ e b_j, $j = 1, 2, 3, 4$ la disponibilità massima di merce nelle origini e la domanda di merce nelle destinazioni, un modello lineare che rappresenta il problema in analisi è il seguente:

$$\begin{cases} \min \left(\displaystyle\sum_{i=1}^{5} \sum_{j=1}^{4} c_{ij} x_{ij} \right) \\ \displaystyle\sum_{j=1}^{4} x_{ij} \leq a_i \qquad i = 1, \ldots, 5 \\ \displaystyle\sum_{i=1}^{5} x_{ij} = b_j \qquad j = 1, \ldots, 4 \\ x_{ij} \geq 0 \end{cases}$$

Segue il file `trasporto1.mod` che realizza un'implementazione in AMPL del modello ora costruito.

trasporto1.mod

```
####    SEZIONE PER LA DICHIARAZIONE DEGLI INSIEMI    ####

set ORIGINI;         # introduce l'insieme delle origini
set DESTINAZIONI;    # introduce l'insieme delle destinazioni


####    SEZIONE PER LA DICHIARAZIONE DEI PARAMETRI    ####

param  costo_origdest {ORIGINI,DESTINAZIONI} >= 0
                           # vettore di parametri a 2
                           # indici (matrice). Rappresenta
                           # i costi unitari di trasporto
                           # (non negativi) dalle origini
                           # alle destinazioni.

param  max_uscita {ORIGINI} >= 0;
                           # vettore di parametri ad un
                           # indice. Rappresenta la quantita'
                           # (non negativa) massima di merce
                           # che puo' essere trasportata da
                           # ciascuna origine.

param  domanda {DESTINAZIONI} >=0;
                           # vettore di parametri ad un
                           # indice. Rappresenta la quantita'
                           # (non negativa) di merce che deve
                           # essere trasportata a ciascuna
                           # destinazione.

####   SEZIONE PER LA DICHIARAZIONE DELLE VARIABILI  ####

var x {i in ORIGINI,j in DESTINAZIONI} >= 0;
                     # x[i,j] e' l'elemento di una matrice di
                     # variabili (2 indici) e rappresenta il
                     # quantitativo di merce trasportato dalla
                     # origine 'i' alla destinazione 'j'.
```

```
####  SEZIONE PER LA DICHIARAZIONE DELLA FUNZIONE ####
####              OBIETTIVO E DEI VINCOLI          ####

minimize cost_trasporto:  sum{i in ORIGINI,j in DESTINAZIONI}
                          costo_origdest[i,j]*x[i,j];

                  # a ciascun trasporto origine/destinazione
                  # e' associato un costo, i costi poi vengono
                  # sommati

s.t. origini {i in ORIGINI}:
     sum{j in DESTINAZIONI} x[i,j] <= max_uscita[i] ;
              # da ciascuna origine non e' possibile inviare
              # piu' di un determinato quantitativo di merce.


s.t. destinazioni {j in DESTINAZIONI}:
     sum{i in ORIGINI} x[i,j] = domanda[j] ;
          # a ciascuna destinazione deve arrivare
          # esattamente una quantita' determinata di merce.
```

Il file `trasporto1.dat` che permette di specificare i dati da introdurre nel modello precedente è il seguente

trasporto1.dat

```
set ORIGINI := asti bergamo como domodossola empoli;
                      # l'insieme ORIGINI ha 5 elementi
set DESTINAZIONI := mantova napoli olbia palermo;
                      # l'insieme DESTINAZIONI ha 4 elementi

param:            max_uscita :=
     asti              100
     bergamo           110
     como              130
     domodossola        85
     empoli            120       ;
                      # il vettore di parametri 'max_uscita'
                      # ha 5 elementi (tanti quante le origini).
```

```
param               domanda   :=
     mantova           30
     napoli            18
     olbia             45
     palermo           56        ;
                    # il vettore di parametri 'domanda'
                    # ha 4 elementi (tanti quante le  destina-
                    # zioni). Si noti che quando si assegnano
                    # gli elementi di un solo vettore non sono
                    # necessari i ':' dopo la parola chiave
                    # 'param'


param costo_origdest : mantova  napoli  olbia  palermo :=
          asti            1       3.5     4        4.5
          bergamo          .1     3       4.5      4.8
          como             .3     2.8     4.7      4.9
          domodossola     1       2.2     3.9      4.0
          empoli          1.7     2.2     5.0      5.3   ;
           # la matrice di parametri 'costo_origdest' contiene
           # 4*5 = 20 elementi, uno per ogni coppia origine/de-
           # stinazione; non e' necessario specificare
           # lo '0' (zero) davanti alla virgola.
```

Supponiamo ora che nelle origini ci sia un costo di prelevamento della merce di cui tenere conto nel calcolo del costo totale, aggiuntivo al costo dei trasporti. Nella tabella che segue si riporta i costi unitari di prelevamento da ciascuna città origine.

	Asti	Bergamo	Como	Domodossola	Empoli
costo prelevamento	2	3	4	7	6

Modificare il file `.mod` e il file `.dat` in modo da tener conto di questi costi aggiuntivi.

Chiamati c_i, $i = 1, 2, 3, 4, 5$, i costi unitari di prelevamento nella i-esima città origine, la funzione obiettivo che rappresenta il costo complessivo diventa

$$\sum_{i=1}^{5}\sum_{j=1}^{4} c_{ij}x_{ij} + \sum_{i=1}^{5} c_i \sum_{j=1}^{4} x_{ij}$$

Il file `trasporto1es.mod` che segue riporta il file del modello modificato in accordo con quanto richiesto dall'esercizio

trasporto1es.mod

```
set ORIGINI;
set DESTINAZIONI;

param  costo_origdest {ORIGINI,DESTINAZIONI} >= 0;

param  costo_origine {ORIGINI} >= 0;

param  max_uscita {ORIGINI} >= 0;

param  domanda {DESTINAZIONI} >=0;

var x {i in ORIGINI,j in DESTINAZIONI} >= 0;

minimize cost_trasporto: sum{i in ORIGINI,j in DESTINAZIONI}
           costo_origdest[i,j]*x[i,j] +  sum{i in ORIGINI}
           costo_origine[i]* sum{j in DESTINAZIONI} x[i,j];


 s.t. origini {i in ORIGINI}:
     sum{j in DESTINAZIONI} x[i,j] <= max_uscita[i] ;

 s.t. destinazioni {j in DESTINAZIONI}:
     sum{i in ORIGINI} x[i,j] = domanda[j] ;
```

Come si può vedere è stato aggiunta l'istruzione

```
param costo_origine {ORIGINI}
```

per rappresentare i costi di prelevemento da ciascuna città origine. Inoltre è stata modificata la funzione obiettivo.

Il file `trasporto1es.dat` che segue riporta il nuovo file di dati per il modello modificato come richiesto dall'esercizio.

trasporto1es.dat

```
set ORIGINI := asti bergamo como domodossola empoli;

set DESTINAZIONI := mantova napoli olbia palermo;

param:           costo_origine  max_uscita :=
     asti              2          100
     bergamo           3          110
     como              4          130
     domodossola       7           85
     empoli            6          120       ;

param               domanda   :=
     mantova           30
     napoli            18
     olbia             45
     palermo           56        ;

param costo_origdest : mantova  napoli  olbia  palermo :=
          asti            1       3.5     4        4.5
          bergamo          .1     3       4.5      4.8
          como             .3     2.8     4.7      4.9
          domodossola     1       2.2     3.9      4.0
          empoli          1.7     2.2     5.0      5.3   ;
```

Introduciamo ora ulteriori modifiche del modello dei trasporti considerato, prendendo in considerazione elementi aggiuntivi che sono di solito presenti nei problemi di trasporto. Supponiamo quindi che ci sia

1. un costo aggiuntivo da imputarsi a tasse portuali ad ogni unità di merce trasportata alla città di Olbia pari a 0.1;
2. un vincolo che impone che almeno i 4/5 della merce trasportata provenga da città origine del nord Italia;

3. un vincolo che impone che almeno i 2/3 della merce trasportata raggiunga città destinazione del sud Italia e delle isole.

4. un ulteriore costo di trasporto per ogni unità di merce trasportata per qualche coppia di città origine e città destinazione da imputarsi, ad esempio, a pedaggi autostradali; la tabella che segue riporta per quali coppia di città esiste questo costo aggiuntivo e a quanto ammonta

Bergamo—Napoli	2	Domodossola—Mantova	0.5	Empoli—Olbia	3.5
Bergamo—Olbia	3.5	Domodossola—Palermo	3.8	Empoli—Palermo	3.1

Per tener conto di queste esigenze aggiuntive, sarà necessario introdurre nuovi insiemi nel modello in AMPL: l'insieme `NORD` delle città del nord, l'insieme `SUD` delle città del sud e l'insieme `ISOLE` delle città che si trovano nelle isole. Inoltre si dovrà fare uso degli operatori tra insiemi di *intersezione* e *unione* (`inter` e `union`). L'uso di questi operatori è molto semplice e permette di definire gli insiemi intersezione e unione, ad esempio, nel seguente modo:

```
set ORIGINI;            # insieme delle citta' origini
set DESTINAZIONI;       # insieme delle citta' destinazioni
set NORD;               # insieme di alcune citta' del nord
set SUD;                # insieme di alcune citta' del sud
set ISOLE;              # insieme di alcune citta' delle isole

set ORI_NORD := ORIGINI inter NORD;
                        # insieme intersezione degli
                        # insiemi ORIGINI e NORD

set DEST_SUDISOLE := DESTINAZIONI inter {SUD union ISOLE};
                        # insieme intersezione dell'insieme
                        # DESTINAZIONI con l'insieme
                        # (SUD unione ISOLE)
```

Il file `trasporto2.mod` che segue riporta il nuovo modello di trasporti che tiene conto delle nuove esigenze sia nella funzione obiettivo sia per la presenza di nuovi vincoli.

`trasporto2.mod`

```
####    SEZIONE PER LA DICHIARAZIONE DEGLI INSIEMI    ####

set ORIGINI;
```

```
set DESTINAZIONI;
set NORD;
set SUD;
set ISOLE;

set ORI_NORD := ORIGINI inter NORD;

set DEST_SUDISOLE := DESTINAZIONI inter {SUD union ISOLE};
                        # introduce l'insieme intersezione
                        # dell'insieme DESTINAZIONI con l'insieme
                        # (SUD unione ISOLE)

      # le due dichiarazioni che seguono definiscono l'insieme COPPIE
      # formato da coppie di elementi: uno appartenente ad ORIGINI e
      # l'altro a DESTINAZIONI e possono essere utilizzate entrambi

#set COPPIE := {ORIGINI,DESTINAZIONI};
set COPPIE within {ORIGINI,DESTINAZIONI};

param  costo_origdest {ORIGINI,DESTINAZIONI} >= 0;

param  costo_origine {ORIGINI} >= 0;

param  max_uscita {ORIGINI} >= 0;

param  domanda {DESTINAZIONI} >=0;

param costo_coppie {COPPIE} >=0;
             # per gli elementi  appartenenti all'insieme
             # COPPIE vi e' un ulteriore costo
             # di trasporto (pedaggio autostradale)

param costo_portuale;
             # costo unitario da imputarsi ad ogni unita' di
             # prodotto trasportato alla destinazione "olbia"

var x {i in ORIGINI,j in DESTINAZIONI} >= 0;

minimize cost_trasporto:  sum{i in ORIGINI,j in DESTINAZIONI}
       costo_origdest[i,j]*x[i,j] +  sum{i in ORIGINI}
       costo_origine[i]* sum{j in DESTINAZIONI} x[i,j] +
```

```
        sum{(i,j) in COPPIE} costo_coppie[i,j] * x[i,j] +
        sum{(i,"olbia") in {ORIGINI,DESTINAZIONI}}
                          costo_portuale * x[i,"olbia"];

                # a ciascun trasporto origine/destinazione
                # e' associato un costo, i costi poi vengono
                # sommati; agli elementi appartenenti allo
                # insieme COPPIE e' associato un costo in piu'.

s.t. origini {i in ORIGINI}:
     sum{j in DESTINAZIONI} x[i,j] <= max_uscita[i] ;

s.t. destinazioni {j in DESTINAZIONI}:
     sum{i in ORIGINI} x[i,j] = domanda[j] ;

s.t. origini_nord: sum {i in ORI_NORD,j in DESTINAZIONI}
      x[i,j] >= 4/5*sum {i in ORIGINI,j in DESTINAZIONI} x[i,j] ;
         # dalle citta' di origine del nord vengono effettuati
         # almeno i 4/5 dei trasporti totali.

s.t. destinazioni_isole:
      sum {i in ORIGINI,j in DESTINAZIONI : j in {SUD union ISOLE}}
      x[i,j] >= 2/3 * sum {i in ORIGINI,j in DESTINAZIONI} x[i,j] ;
         # verso le citta' di destinazione del sud e delle
         # isole vengono effettuati almeno i 2/3 dei trasporti
         # totali.
```

Il file di dati `trasporto2.dat` che segue permette di introdurre i dati assegnati nel modello.

`trasporto2.dat`

```
set ORIGINI := asti bergamo como domodossola empoli;

set DESTINAZIONI := mantova napoli olbia palermo;

set NORD := asti como domodossola empoli mantova;

set SUD := bari napoli;
```

```
set ISOLE := olbia palermo;

param:           costo_origine  max_uscita :=
     asti              2          100
     bergamo           3          110
     como              4          130
     domodossola       7           85
     empoli            6          120       ;

param               domanda   :=
     mantova           30
     napoli            18
     olbia             45
     palermo           56        ;

param costo_origdest : mantova  napoli  olbia  palermo :=
          asti            1       3.5     6.0      4.5
          bergamo          .1     3       4.5      4.8
          como             .3     2.8     4.7      4.9
          domodossola     1       2.2     3.9      4.0
          empoli          1.7     2.2     5.0      5.3   ;

param: COPPIE :                   costo_coppie   :=
        bergamo napoli               2
        bergamo olbia                3.5
        domodossola mantova          0.5
        domodossola palermo          3.8
        empoli olbia                 3.5
        empoli palermo               3.1          ;
           # le possibili coppie origine/destinazione sono
           # 5*4 = 20 ma l'insieme COPPIE ne contiene solo 6

param costo_portuale := 0.1 ;
   # per ogni unita' di merce che ha come destinazione "olbia"
   # vi e' questo costo aggiuntivo;
```

Si osservi che in questo file dei dati non è stato assegnato esplicitamente l'insime `COPPIE` con un'istruzione del tipo

```
set COPPIE :=
   (bergamo , napoli) (bergamo , olbia) (domodossola , mantova)
   (domodossola , palermo)  (empoli , olbia)  (empoli , palermo);
```

oppure

```
set COPPIE :=
   bergamo napoli
   bergamo olbia
   domodossola mantova
   domodossola palermo
   empoli olbia
   empoli palermo;
```

Infatti, AMPL, in questo caso prenderà come elementi dell'insieme `COPPIE` quelli specificati nell'assegnazione del parametro `COPPIE`.

Un problema di produzione industriale multiperiodo

Esaminiamo ora un modello per la produzione industriale *multiperiodo*. In particolare, consideriamo l'Esercizio 3.1.1 riportato nel secondo volume di queste dispense (*Esercizi svolti di Ricerca Operativa*) che implementeremo in AMPL utilizzando alcune delle strutture del linguaggio che abbiamo introdotto nel paragrafo precedente. In particolare, vedremo l'uso oltre che degli insiemi generici, anche degli insiemi numerici, ordinati e ciclici. Si tratta pianificare la produzione trimestrale di pneumatici la cui formulazione è la seguente:

$$
\begin{cases}
\min\Big(3100(A_{L1}^{ott}+A_{L1}^{nov}+A_{L1}^{dic})+3220(A_{L2}^{ott}+A_{L2}^{nov}+A_{L2}^{dic})+ \\
\quad +4720(B_{L1}^{ott}+B_{L1}^{nov}+B_{L1}^{dic})+5080(B_{L2}^{ott}+B_{L2}^{nov}+B_{L2}^{dic})+ \\
\quad +350(A_{im}^{ott}+A_{im}^{nov}+B_{im}^{ott}+B_{im}^{nov})\Big) \\
0.10A_{L1}^{ott}+0.12B_{L1}^{ott}\leq 2000 \\
0.10A_{L1}^{nov}+0.12B_{L1}^{nov}\leq 400 \\
0.10A_{L1}^{dic}+0.12B_{L1}^{dic}\leq 200 \\
0.12A_{L2}^{ott}+0.18B_{L2}^{ott}\leq 3000 \\
0.12A_{L2}^{nov}+0.18B_{L2}^{nov}\leq 800 \\
0.12A_{L2}^{dic}+0.18B_{L2}^{dic}\leq 1000 \\
A_{L1}^{ott}+A_{L2}^{ott}=16000+A_{im}^{ott} \\
A_{L1}^{nov}+A_{L2}^{nov}+A_{im}^{ott}=7000+A_{im}^{nov} \\
A_{L1}^{dic}+A_{L2}^{dic}+A_{im}^{nov}=4000 \\
B_{L1}^{ott}+B_{L2}^{ott}=14000+B_{im}^{ott} \\
B_{L1}^{nov}+B_{L2}^{nov}+B_{im}^{ott}=4000+B_{im}^{nov} \\
B_{L1}^{dic}+B_{L2}^{dic}+B_{im}^{nov}=6000 \\
A_{Li}^{ott}\geq 0, A_{Li}^{nov}\geq 0, A_{Li}^{dic}\geq 0,\ i=1,2 \\
B_{Li}^{ott}\geq 0, B_{Li}^{nov}\geq 0, B_{Li}^{dic}\geq 0,\ i=1,2.
\end{cases}
$$

Decidiamo di definire un insieme *generico* che chiameremo `tipi` per quanto riguarda il tipo di pneumatico (tipo A e tipo B). Per quanto riguarda le linee, pensando all'attribuzione a ciascuna linea di un numero intero, definiamo l'insieme delle linee (`linee`) come un insieme *numerico* formato dai numeri interi da 1 a `numlinee`, dove `numlinee` è un parametro che corrisponde al numero di linee presenti e che verrà assegnato nel file dei dati. Naturalmente, la dichiarazione del parametro `numlinee` dovrà precedere quella dell'insieme `linee`. Si ribadisce che in questo modo il modello definito nel file dei dati è indipendente dai dati che saranno introdotti nel file `.dat`. Infine, si sceglie di utilizzare un insieme (ordinato) *ciclico* per quanto riguarda l'insieme dei mesi. Questa scelta è dettata dai vincoli che dovranno essere implementati e sarà esaminata nel dettaglio nel seguito.

Per quanto riguarda le variabili, sono state introdotte le variabili con tre indici `x{tipi,linee,mesi}` e le variabili `x_im{tipi,mesi}` relative alle quantità

immagazzinate. Poiché nell'ultimo prediodo non è previsto l'immagazzinamento, sarà necessario imporre che il quantitativo di pneumatici immagazzinato nell'ultimo periodo sia pari a 0.

Un file `.mod` che rappresenta bene il modello in esame è riportato di seguito.

pneumatici.mod

```
set tipi;   # insieme generico

param numlinee integer;   # parametro numlinee con
                          # controllo dell'interezza

set linee:= 1..numlinee;    # insieme numerico

set mesi circular;   # insieme ordinato ciclico

param ordine{mesi,tipi}>=0;
param disponibilita{mesi,linee}>=0;
param tempi{tipi,linee}>=0;

param costo_orario>=0;
param costo_immagazzinamento>=0;

param costo_materiale_grezzo{tipi}>=0;

var x{tipi,linee,mesi}>=0;  # quantita' di pneumatici di
                            # ciascun tipo prodotti da
                            # ciascuna linea in ciascun mese

var x_im{tipi,mesi}>=0; # quantita' di pneumatici di ciascun
                        # tipo immagazzinato in ciascun mese
                        # ove questo e' possibile
                        # (sara' quindi necessario porre
                        # x_im[i,last(mesi)]=0
                        # al variare di {i in tipi})


minimize costo_totale:
 sum{i in tipi, j in linee, k in mesi}
                          costo_orario*tempi[i,j]*x[i,j,k] +
 sum{i in tipi, j in linee, k in mesi}
                          costo_materiale_grezzo[i]*x[i,j,k]+
```

```
 sum{i in tipi, l in mesi} costo_immagazzinamento*x_im[i,l];

# vincoli disponibilita' delle macchine

s.t. vincoli_disponibilita_linee{k in mesi, j in linee}:
      sum{i in tipi} tempi[i,j]*x[i,j,k] <= disponibilita[k,j];



# vincoli dovuti alla richiesta e all'immagazzinamento

s.t. vincoli_richiesta{k in mesi, i in tipi}:
        x_im[i,prev(k,mesi)]+sum{j in linee} x[i,j,k] =
                                  ordine[k,i]+x_im[i,k];
```

Vale la pena soffermarci sulla definizione dei vincoli. Iniziamo dai vincoli sulla disponibilità delle macchine. Si tratta di esprimere un vincolo per ogni linea di produzione. Quindi, in prima analisi, potevamo scrivere i seguenti vincoii:

```
s.t. vincoli_disponibilita_linea1{k in mesi}:
      sum{i in tipi} tempi[i,1]*x[i,1,k] <= disponibilita[k,1];

s.t. vincoli_disponibilita_linea2{k in mesi}:
      sum{i in tipi} tempi[i,2]*x[i,2,k] <= disponibilita[k,2];
    .
    .
    .
```

Ma questa scrittura, oltre che essere lunga, prevede di conoscere già nel file del modello quante sono le linee di produzione, dato che invece deve essere un parametro assegnato nel file dei dati.

Per quanto riguarda i vincoli dovuti alla richiesta e all'immagazzinamento, essendo l'insieme `mesi` un insieme ciclico, possiamo evitare di tenere separati i vincoli riguardanti il primo e l'ultimo mese purché si imponga che il quantitativo di pneumatici immagazzinato nell'ultimo periodo sia pari a 0. Anche in questo caso, per ragioni analoghe, si è evitato di scrivere i vincoli nella forma

```
s.t. vincoli_richiesta_tipoA{k in mesi}:
            x_im["tipoA",prev(k,mesi)]+
```

```
            sum{j in linee} x["tipoA",j,k] =
            ordine[k,"tipoA"]+x_im["tipoA",k];

s.t. vincoli_richiesta_tipoB{k in mesi}:
            x_im["tipoB",prev(k,mesi)]+
            sum{j in linee} x["tipoB",j,k] =
            ordine[k,"tipoB"]+x_im["tipoB",k];
    .
    .
    .
```

L'assegnazione dei parametri nel file `.dat` è molto standard. Come abbiamo già detto dobbiamo *fissare* a 0 le variabili che identificano le quantità di pneumatici immagazzinati nell'ultimo mese. Il fixing delle variabili si può effettuare o nel file `.dat` o nel file `.run`. In questo caso, trattandosi di una richiesta parte integrante del modello, lo effettuiamo nel file `.dat`. Si tratterebbe di scrivere le istruzioni

```
fix x_im["tipoA",last(mesi)]:=0;
fix x_im["tipoB",last(mesi)]:=0;
  .
  .
  .
```

Anche in questo caso, sia per brevità di scrittura, ma soprattutto per rendere il modello valido anche nel caso in cui si cambino i dati, scegliamo di fissare le variabili lasciando variare il *tipo* all'interno dell'insieme `tipi`. In questo caso si può procedere utilizzando una sintassi del linguaggio AMPL non ancora introdotta: si tratta dell'istruzione `for`. Il suo uso verrà specificato meglio nel seguito, ma la sintassi è abbastanza standard. In questo caso è

```
for {i in tipi}{fix x_im[i,last(mesi)]:=0;}
```

Quindi il file del dati per il modello in esame può essere scritto come segue:

pneumatici.dat

```
set mesi := ott nov dic;
set tipi := tipoA tipoB;

param numlinee := 2;

param ordine:   tipoA tipoB :=
       ott     16000  14000
```

```
        nov      7000   4000
        dic      4000   6000;

param disponibilita:       1     2 :=
                ott      2000  3000
                nov       400   800
                dic       200  1000;

param tempi:                 1     2 :=
                tipoA      0.10  0.12
                tipoB      0.12  0.18;

param costo_orario := 6000;

param costo_immagazzinamento := 350;

param costo_materiale_grezzo :=
                        tipoA 2500
                        tipoB 4000;

for {i in tipi}{fix x_im[i,last(mesi)]:=0;}
```

4.6 I PRINCIPALI COMANDI AMPL

Abbiamo già utilizzato alcuni comandi AMPL per poter determinare la soluzione ottima di alcuni esempi di modelli. Ad esempio, abbiamo già utilizzato il comando `solve` per invocare il solutore, oppure il comando `option solver cplex` per definire il solutore da utilizzare ed anche il comando `display` per visualizzare il risultato ottenuto. Vogliamo ora dare un quadro più generale riguardante l'uso di questi comandi.
Innanzitutto, i comandi vengono dati su riga di comando al prompt di AMPL, oppure sono inseriti in un file `.run`. Essi possono essere di fatto utilizzati per scrivere dei veri e propri programmi in AMPL.

4.6.1 Il comando `option`

Il comando `option` serve per visualizzare o cambiare il valore delle *opzioni*. Le *opzioni* sono *variabili di stato* dell'ambiente AMPL. Ciascuna di esse ha un nome ed un valore che può essere un numero o una stringa di caratteri. Per aver un'idea di quali sono le variabili di stato in AMPL digitare sul prompt dei comandi di AMPL il comando

```
option;
```

Verrà visualizzato un lungo elenco di tutte le variabili di stato di AMPL e il loro valore corrente. Il comando `option` senza ulteriore specificazione serve infatti per visualizzare il valore delle varibili di stato. Il comando `option` accetta una "wild card" che è rappresentata dal carattere "*" ed è utilizzato per rappresentare qualsiasi stringa. Quindi, ad esempio, con il comando `option presolve*` si otterrà la lista di tutte le opzioni il cui nome inizia per `presolve` e il loro valore corrente.

Per visualizzare il valore corrente di un'opzione specifica si dovrà specificare il `nomeopzione`. Quindi per visualizzare il valore dell'opzione `nomeopzione` sarà sufficiente specificare

```
option nomeopzione;
```

Per modificare il valore dell'opzione `nomeopzione` sarà sufficiente il comando che indichi questo nuovo valore:

```
option nomeopzione nuovovalore;
```

Abbiamo già visto un esempio di questo comando quando abbiamo selezionato il solutore da utilizzare con il comando `option solver cplex`. In questo caso la variabile di stato è `solver` che viene impostata al valore `cplex`.

Prestare molta attenzione al fatto che il comando `option` non controlla subito che il valore assegnato abbia senso o meno; un messaggio di errore si avrà solo in fase di esecuzione.

Infine, per riportare tutte le opzioni al loro valore di default si utilizza il comando

```
reset options;
```

Per un elenco completo di tutte le opzioni si rimanda al testo di AMPL già citato. Ne riportiamo di seguito solamente tre di uso frequente:

- `solver` specifica il solutore. Per default il suo valore è `cplex` e può essere cambiato utililzzando il nome degli altri solutori.
- `presolve` specifica le opzioni del preprocessamento. Il preprocessamento è un'operazione che AMPL può effettuare allo scopo di ridurre il problema, ad esempio, eliminando vincoli ridondanti, fissando valori di alcune variabili, etc. Tale procedimento è molto utile (e a volte indispensabile) nella risoluzione di problemi a grandi dimensioni. Il valore di default è 10. Per inibire il preprocessamento è sufficiente assegnare valore 0 a `presolve`.
- `show_stats` specifica il livello di dettaglio delle informazioni sul problema e sulla soluzione che devono essere visualizzate. Il valore di default è 0, in corrispondenza del quale vengono visualizzate informazione minime. Assegnando il valore 1 o superiori a `show_stats` aumenta il livello di dettaglio delle informazioni visualizzate.

Attraverso il comando `option` si possono inoltre specificare le opzioni relative al solutore utilizzato.

4.6.2 Il comando `display`

Il comando `display` si utilizza per visualizzare oggetti presenti nel modello come, ad esempio, gli elementi di un insieme, il valore delle variabili, dei parametri, della funzione obiettivo, dei vincoli. Nella sua versione più semplice consente di visualizzare il valore di un oggetto denominato `nomeoggetto` tramite il comando

```
display nomeoggetto;
```

Dopo il comando `display` posso essere anche elencati un certo numero i oggetti da visualizzare separati dalla virgola. Con il comando `display` possono essere anche utilizzate espressioni algebriche o logiche come riportato negli esempi che seguono (facendo riferimento agli oggetti utilizzati nell'esempio precedente:

```
display mesi;
display x;
```

```
display costo_totale;
display {i in tipi} x[i,1,"nov"];

display sum{i in tipi, j in linee, k in mesi}
                    costo_materiale_grezzo[i]*x[i,j,k];

display {i in tipi, k in mesi : x[i,1,k] > 100};
```

Non forniamo spiegazioni dettagliate di queste istruzioni perché sono molto intuitive. Ci soffermiamo solamente sull'uso dei ":" che può essere introdotto nei costrutti logici, come nell'ultimo comando `display` dell'esempio, con il significato di "*tale che*".

Le opzioni del comando `display` riguardano la formattazione delle informazione da visualizzare e l'approssimazione utilizzata nell'arrotondamento dei valori numerici da visualizzare. Per esse si fa riferimento al Capitolo 12 del testo di AMPL già citato ed in particolare alle Tabelle 12-1 e 12-2.

4.6.3 Reindirizzamento dell'output dei comandi

È molto utile poter reindirizzare l'output dei comandi in un file nel quale conservare tale output. Questo vale per tutti i comandi, ma in particolare per il comando `display`. Infatti, in questo modo si può facilmente memorizzare la soluzione ottima e altre informazioni. Per creare un file testo di output `nomefile.out` nel quale reindirizzare l'output del comando `display` è sufficiente il comando

```
display oggetto > nomefile.out;
```

In questo modo viene creato o eventualmente sovrascritto (se già esistente) il file `nomefile.out` nel quale verrà scritto l'output del comando. Se si vuole "appendere", ovvero aggiungere alla fine del file, altri output è sufficiente il comando

```
display oggetto2 >> nomefile.out;
```

In questo modo, nel file `nomefile.out`, dopo il/i valore/i di `oggetto` compariranno il/i valore/i di `oggetto2`.

Quindi, sempre in riferimento all'esempio precedente, è possibile aggiungere nel file `.run` (ovviamente dopo il comando `solve`) i comandi

```
display x > risultati.out;
display x_im >> risultati.out;
display costo_totale >> risultati.out;
```

per creare il file `risultati.out` contenente i valori della variabili all'ottimo e il valore ottimo della funzione obiettivo.

4.6.4 Il comando `display` per visualizzare altre grandezze relative alle variabili all'ottimo

Nella risoluzione di problemi di Programmazione Lineare, AMPL oltre a fornire (ove esista) la soluzione ottima del problema, permette di visualizzare anche altri elementi del problema come i prezzi ombra i costi ridotti associati alla soluzione ottima. Per visualizzare questi elementi è sufficiente aggiungere dei suffissi al nome della variable. In particolare, se x è un variabile del problema, possiamo utilizzare il comando

```
display x.lb, x.ub;
```

per visualizzare il lower bound e l'upper bound della variabile x. Quindi, ad esempio, se x è una variabile definita non negativa, il comando fornirà il valore `0` per il lower bound e il valore `Infinity` per l'upper bound. Il comando

```
display x.slack;
```

visualizza la differenza tra il valore della variabile e il più vicino bound.

Il concetto di "bound" e di "slack" ha un'interpretazione analoga anche per i vincoli di un modello. Ovvero si pensa al vincolo scritto nella forma

$$\textit{lower bound} \quad \leq \quad \textit{body} \quad \leq \quad \textit{upper bound}$$

e quindi, se `vinc` è l'etichetta data ad un vincolo, il comando

```
display vinc.lb, vinc.body, vinc.ub;
```

visualizza il lower bound del vincolo, il valore della parte variabile del vincolo e l'upper bound del vincolo. Il comando

```
display vinc.slack;
```

visualizza la differenza tra il valore del vincolo e il più vicino bound.

Il comando `display` si può utilizzare anche per avere informzioni sulle quantità *duali* associate al problema. Come sarà esaminato nel Capitolo **??**, a ciascun vincolo di un problema di Programmazione Lineare si può associare una variabile duale e il valore ottimo di tale variabile viene chiamato *prezzo ombra* o *valore marginale*. Con il comando

```
display vincolo;
```

si visualizza tale valore, ovviamente senza la necessità di dover costruire esplicitamente il problema duale. Similmente il comando

```
display x.rc;
```

visualizza il *costo ridotto* associato alla variabile x.

L'uso delle quantità duali e la loro interpretazione verrà trattata nel dettaglio nel prossimo Capitolo **??** nel quale verrà affrontata *l'analisi di sensitività* della soluzione ottima rispetto a parametri di un problema di Programmazione Lineare.

4.6.5 Comandi per aggiornare il modello: `reset`, `drop` e `restore`

Sono disponibili comandi per modificare anche solo parzialmente un modello. Il comando `reset`, già utilizzato, cancella completamente il modello e i dati. Equivale ad uscire (con il comando `quit`) da AMPL e rientrare. Esistono poi comandi per far in modo che alcuni vincoli siano ignorati. Il comando è `drop`. Quindi utilizzando i comandi

```
drop vincolo1;
drop vincoli_risorse{i in RISORSE};
drop vincolo_budget["periodo1"];
```

si ottiene che i vincoli corrispondenti vengano ignorati. Con il comando `restore` si ripristinano vincoli che fossero stati eventualmente in precedenza "ignorati".

4.6.6 Altri utili comandi: `show`, `xref`, `expand`

Sono comandi che servono per visualizzare componenti del modello.

- Il comando `show` visualizza tutte le componenti del modello, ovvero parametri, insiemi, variabili, vincoli e funzione obiettivo.
- Il comando `xref` visualizza tutte le componenti del modello che dipendono da una specifica componente.
- Il comando `expand` applicato ad un vincolo genera tutti i vincoli derivanti da un vincolo scritto in forma indicizzata su un insieme. Applicato ad una variabile, visualizza tutti i coefficienti non nulli di questa variabile nei termini lineari della funzione obiettivo e dei vincoli. Se inoltre la variabile compare anche in espressioni non lineari allora viene aggiunto all'output l'espressione `+ nonlinear`.

4.6.7 Nomi generici per variabili, vincoli, e funzioni obiettivo

AMPL rende disponibili parametri che forniscono il *numero* delle variabili, dei vincoli e delle funzioni obiettivo del problema:

- `_nvars`: numero delle variabili del problema
- `_ncons`: numero dei vincoli del problema

- `_nobjs`: numero delle funzioni obiettivo del problema.

Sono disponibili inoltre parametri che forniscono i *nomi* delle componenti del problema:

- `_varname`: nomi delle variabili del problema
- `_conname`: nomi dei vincoli del problema
- `_objname`: nomi delle funzioni obiettivo del problema.

Infine, sono disponibili sinoninmi per le componenti del problema:

- `_var`: sinonimo per le variabili del problema
- `_con`: sinonimo per i vincoli del problema
- `_obj`: sinonimo per le funzioni obiettivo del problema

Utilizzando questi *sinonimi* è possibile scrivere un file `.run` che può essere utilizzato per la soluzione di problemi diversi senza dover indicare volta per volta il nome specifico delle variabili e della funzione obiettivo nel comando `display`. Un esempio di un tale file `.run` è il seguente:

```
reset;
model modello.mod;
data modello.dat;

option solver cplex;
solve;

display _varname, _var;
display _objname, _obj;
```

5

La Programmazione Lineare

In questo capitolo esaminiamo in modo più dettagliato la Programmazione Lineare illustrando una tecnica risolutiva per il caso di due sole variabili che aiuta a comprendere alcune delle caratteristiche più importanti dei problemi di Programmazione Lineare. Queste caratteristiche sono poi oggetto di una formalizzazione e generalizzazione, che suggeriscono anche i rudimenti per un metodo di soluzione generale.

5.1 INTRODUZIONE

La Programmazione Lineare è indubbiamente l'argomento centrale dell'Ottimizzazione e fra i vari modelli della Ricerca Operativa, la Programmazione Lineare è quello che viene più ampiamente utilizzato. Infatti, la PL non solo si applica a numerosi problemi reali che hanno di per sé una struttura lineare, ma è anche un importante strumento di supporto nell'analisi e nella risoluzione di problemi di programmazione matematica più complessi.
Il padre della PL viene comunemente, e giustamente, indicato in George Dantzig che per primo ne ideò, nel 1947, un algoritmo risolutivo (il metodo del Simplesso, che verrà esaminato nel seguito). Tuttavia, alcuni dei concetti fondamentali della programmazione lineare possono essere fatti risalire molto più indietro nel tempo. Già Fourier, nel 1827, aveva studiato come trovare soluzioni ammissibili di un sistema di disuguaglianze lineari; un metodo di calcolo destinato a minimizzare gli errori d'osservazione e dovuto a Vallée Poussin (1910) presenta lati simili al metodo del simplesso; infine lavori di von Neumann degli anni venti e trenta, sulla teoria dei giochi e su alcuni modelli economici, sono antecedenti diretti del lavoro

di Dantzig. Nel 1939, poi, il matematico sovietico Kantorovich aveva pubblicato (in russo) una monografia sulla programmazione della produzione che anticipa, sotto molti aspetti importanti, i temi trattati da Dantzig. Purtroppo questo lavoro fu a lungo ignorato in Occidente essendo riscoperto solo venti anni dopo, quando la PL aveva avuto un grande sviluppo.
Ancora prima della pubblicazione dello studio di Kantorovich, Leontief (1932) aveva presentato il suo lavoro fondamentale, in cui si proponeva una struttura matriciale (*Interindustry Input-Output model*) per lo studio dell'economia americana. Per questo modello Leontief vinse il Premio Nobel per l'Economia nel 1976.
La caratteristica di tutti i lavori antecedenti quelli di Dantzig era uno scarso interesse verso l'applicabilità pratica, dovuta principlamente all'impossibilità di effettuare i calcoli necessari. Il metodo del Simplesso proposto da Dantzig si rivelò invece efficiente in pratica e questo, unitamente al simultaneo avvento dei calcolatori elettronici, decretò il successo della Programmazione Lineare e, con esso, l'inizio dello sviluppo rigoglioso della Ricerca Operativa.

5.2 STRUTTURA DI UN PROBLEMA DI PROGRAMMAZIONE LINEARE

Come abbiamo già visto nel paragrafo 3.1 un problema di Programmazione Lineare è caratterizzato da una funzione obiettivo lineare (da minimizzare o massimizzare) della forma

$$f(x) = c_1x_1 + \ldots + c_nx_n = \sum_{j=1}^{n} c_jx_j$$

e da un numero finito m di vincoli lineari della forma

$$\begin{array}{ccccc} a_{11}x_1+ & \ldots & +a_{1n}x_n & \geq b_1 \\ a_{21}x_1+ & \ldots & +a_{2n}x_n & \geq b_2 \\ \vdots & \ldots & \vdots & \vdots \\ a_{m1}x_1+ & \ldots & +a_{mn}x_n & \geq b_m. \end{array} \tag{5.2.1}$$

Ricordiamo inoltre che, introducendo il vettore $c \in \mathbb{R}^n$, definito $c = (c_1, \ldots, c_n)^T$, $x \in \mathbb{R}^n$ definito $x = (x_1, \ldots, x_n)^T$, il vettore $b = (b_1, \ldots, b_m)^T$ e la matrice $(m \times n)$

$$A = \begin{pmatrix} a_{11} & \ldots & a_{1n} \\ \vdots & & \vdots \\ a_{m1} & \ldots & a_{mn} \end{pmatrix}$$

un generico problema di Programmazione Lineare può essere scritto nella forma

$$\begin{cases} \min c^T x \\ Ax \geq b. \end{cases}$$

Si osservi che, indicando con a_i^T, $i = 1, \ldots, m$, le righe della matrice A, ciascun vincolo del problema, ovvero ciascuna disuguaglianza della (5.2.1) può essere scritto nella forma $a_i^T x \geq b_i$, $i = 1, \ldots, m$.

5.3 INTERPRETAZIONE GEOMETRICA DI UN PROBLEMA DI PROGRAMMAZIONE LINEARE

In questo paragrafo si vuole fornire una interpretazione geometrica di un problema di Programmazione Lineare. In particolare, quando un problema di Programmazione Lineare contiene solamente due variabili, si può rappresentare efficacemente il problema sul piano cartesiano e si può determinare una sua soluzione in maniera elementare con semplici deduzioni geometriche. Le situazioni che verranno presentate nel seguito vogliono rappresentare un punto di partenza intuitivo per la trattazione di problemi di Programmazione Lineare in n variabili; i risultati che verranno dedotti per via elementare nel caso bidimensionale trovano, infatti, una generalizzazione consistente nel caso di un generico problema di Programmazione Lineare.
A questo scopo verranno considerati due esempi di problemi di Programmazione Lineare già ottenuti come formulazione di un semplice problema di allocazione ottima di risorse (Esempio 3.4.1) e di un semplice problema di miscelazione (Esempio 3.4.11).

5.3.1 Rappresentazione di vincoli lineari

Preliminarmente, si richiama il fatto che sul piano cartesiano Ox_1x_2 l'equazione

$$a_1x_1 + a_2x_2 = c \tag{5.3.1}$$

rappresenta una retta che partiziona il piano in due semipiani. Ciascun semipiano è caratterizzato da punti $P(x_1, x_2)$ che soddisfano la disequazione $ax_1 + bx_2 \geq c$ oppure la disequazione $ax_1 + bx_2 \leq c$. Quindi ogni disequazione del tipo

$$ax_1 + bx_2 \geq c \qquad \text{oppure} \qquad ax_1 + bx_2 \leq c$$

individua univocamente un semipiano. Si riporta, ora, un semplice risultato geometrico che verrà utilizzato nel seguito.

Lemma 5.3.1 *Si considera una famiglia di rette parallele*

$$a_1x_1 + a_2x_2 = c \tag{5.3.2}$$

con $a_1, a_2 \in \mathbb{R}$ *fissati e con* $c \in \mathbb{R}$. *Il vettore* $a = \begin{pmatrix} a_1 \\ a_2 \end{pmatrix}$ *individua una direzione ortogonale alle rette della famiglia (5.3.2) ed è orientato dalla parte in cui sono le rette della famiglia ottenute per valori crescenti della c, cioè verso il semipiano in cui risulta* $a_1x_1 + a_2x_2 \geq c$.

Nella pratica, per determinare quale dei due semipiani è individuato dalla disuguaglianza lineare $a_1x_1 + a_2x_2 \geq c$ si può procedere semplicemente in questo modo: dopo aver rappresentato la retta $a_1x_1 + a_2x_2 = c$ per individuare qual è il semipiano di interesse, si può scegliere un punto P del piano (l'origine degli assi è il piú semplice) e valutare l'espressione $a_1x_1 + a_2x_2$ in questo punto; se il valore così ottenuto è maggiore o uguale di c allora il semipiano individuato dalla disuguaglianza lineare $a_1x_1 + a_2x_2 \geq c$ è quello contenente il punto P; in caso contrario è quello opposto.

5.3.2 Rappresentazione di funzioni obiettivo lineari

Quanto esposto nel paragrafo precedente è utile anche per esaminare la variazione di una funzione lineare che rappresenta la funzione obiettivo di un problema di Programmazione Lineare. In due variabili, la funzione obiettivo di un problema di Programazione Lineare è un'espressione del tipo $c_1x_1+c_2x_2$ da massimizzare o da minimizzare. Per rappresentare questa funzione obiettivo su un piano cartesiano Ox_1x_2 si considera la famiglia di rette parallele

$$c_1x_1 + c_2x_2 = C \tag{5.3.3}$$

ottenuta al variare di C, che rappresentano le *curve di livello* della funzione $f(x_1, x_2) = c_1x_1 + c_2x_2$ che ovviamente in questo caso sono rette. Se il problema è di minimizzazione, si cercherà di ottenere un valore piú basso possibile per la C in corrispondenza di valori ammissibili per x_1 e x_2; viceversa, se il problema è di massimizzazione, si cercherà ottenere un valore piú alto possibile per la C. Sulla base di quanto esposto nel paragrafo precedente, valori superiori della C si determinano traslando le rette nel verso individuato dal vettore $\begin{pmatrix} c_1 \\ c_2 \end{pmatrix}$ che rappresenta, quindi, una *direzione di crescita* per la funzione $c_1x_1+c_2x_2$. Ovviamente, la direzione opposta sarà una *direzione di decrescita*.
Quindi, geometricamente, un problema di massimizzazione consisterà nel considerare la traslazione nel verso della direzione di crescita della funzione obiettivo, mentre in un problema di minimizzazione si considera la traslazione nel verso opposto.

5.3.3 Esempi di risoluzione grafica

Esempio 5.3.2
Si consideri ora il problema di allocazione ottima di risorse dell'Esempio 3.4.1 che è rappresentato dal seguente problema di Programazione Lineare:

$$\begin{cases} \max(7x_1 + 10x_2) \\ x_1 + x_2 \leq 750 \\ x_1 + 2x_2 \leq 1000 \\ x_2 \leq 400 \\ x_1 \geq 0, x_2 \geq 0. \end{cases}$$

Sul piano cartesiano Ox_1x_2 ciascun vincolo individua un semipiano. Ovviamente i vincoli di non negatività delle variabili $x_1 \geq 0$ e $x_2 \geq 0$ rappresentano rispettivamente il semipiano delle ascisse non negative e il semipiano delle ordinate non negative.
L'insieme ammissibile del problema di Programmazione Lineare che stiamo esaminando è dato quindi dall'intersezione di tali semipiani e si può indicare con

$$S = \Big\{(x_1, x_2) \in \mathbb{R}^2 \ \mid \ x_1 + x_2 \leq 750, x_1 + 2x_2 \leq 1000, x_2 \leq 400, x_1 \geq 0, x_2 \geq 0\Big\}.$$

Tale regione di piano prende nome di *regione ammissibile*, è un insieme convesso. Tutti i punti $P(x_1, x_2)$ appartenenti a questa regione sono punti dell'insieme ammissibile del problema e quindi tutti i punti di questa regione costituiscono soluzioni ammissibili del problema.
Si consideri ora la funzione obiettivo $7x_1 + 10x_2$ e si consideri la famiglia di rette

$$7x_1 + 10x_2 = C$$

ottenute al variare del parametro C. Esse costituiscono le curve di livello della funzione in due variabili $f(x_1, x_2) = 7x_1 + 10x_2$ che sono ovviamente delle rette. In riferimento al problema di allocazione ottima rappresentato dal problema di Programmazione Lineare che stiamo esaminando, il parametro C rappresenta il profitto totale che deve essere massimizzato. Per $C = 0$ si ottiene la linea di livello passante per l'origine del piano Ox_1x_2. Ovviamente, scegliendo $x_1 = 0$ e $x_2 = 0$ (che è un punto ammissibile in quanto $(0, 0) \in S$) si ottiene il profitto totale nullo. All'aumentare del valore di tale profitto, cioè all'aumentare del valore della costante C, graficamente si ottengono rette parallele alla retta di livello passante per l'origine traslate nella direzione di crescita della funzione $7x_1 + 10x_2$ data dal vettore $\begin{pmatrix} 7 \\ 10 \end{pmatrix}$. Poiché si desidera massimizzare la funzione obiettivo, si cercherà di raggiungere il valore piú alto per la C ottenuto scegliendo per x_1 e x_2 valori ammissibili cioè tali che $(x_1, x_2) \in S$. Osservando la rappresentazione grafica della regione ammissibile S si deduce che all'aumentare di C, le rette

di livello della funzione obiettivo intersecano la regione ammissibile finché $C \leq 6000$. Tale valore è ottenuto per $x_1 = 500$ e $x_2 = 250$ e non esistono altri punti della regione ammissibile in cui la funzione obiettivo assume valori maggiori. Il valore 6000 è, quindi, il massimo valore che la funzione obiettivo può raggiungere soddisfacendo i vincoli. Tale soluzione ottima è raggiunta in corrispondenza del punto P di coordinate $(x_1, x_2) = (500, 250)$; tale punto non è un punto qualsiasi, ma costituisce quello che nella geometria piana viene detto vertice del poligono convesso che delimita la regione ammissibile. Il fatto che l'ottimo del problema è raggiunto in corrispondenza di un vertice della regione ammissibile non è casuale, ma come si vedrà in seguito, è una caratteristica fondamentale di un generico problema di Programmazione Lineare. Si osservi fin d'ora che la frontiera della regione ammissibile è definita dalle rette

$$x_1 + x_2 = 750, \quad x_1 + 2x_2 = 1000, \quad x_2 = 400, \quad x_1 = 0, \quad x_2 = 0$$

e che ogni intersezione di due di queste rette è un vertice della regione ammissibile; il numero di queste possibili intersezioni (non tutte necessariamente appartenenti alla regione ammissibile) è ovviamente pari al piú a 10. Si osservi, infine, che nel punto di ottimo sono attivi i vincoli $x_1 + x_2 \leq 750$ e $x_1 + 2x_2 \leq 1000$ mentre non è attivo il vincolo $x_2 \leq 400$.
Nel caso particolare che abbiamo esaminando, la soluzione ottima determinata è unica, ma in generale può accadere che le rette di livello della funzione obiettivo siano parallele ad un segmento del perimetro del poligono che delimita la regione ammissibile; in questo caso potrebbe accadere che esistano piú punti ammissibili in cui la funzione assume lo stesso valore ottimo e quindi la soluzione non sarebbe piú unica; nel problema in esame, ciò accadrebbe, ad esempio, se la funzione obiettivo fosse $cx_1 + 2cx_2$ con c costante reale positiva; infatti, tutti i punti del segmento $\overline{AB}$ rappresentano soluzioni ottime. Tuttavia, anche in questo caso si può sempre trovare un vertice che costituisce una soluzione ottima.

Esempio 5.3.3
Consideriamo ora il problema di miscelazione dell'Esempio 3.4.11 che è rappresentato dal seguente problema di Programmazione Lineare:

$$\begin{cases} \min(400x_1 + 600x_2) \\ 140x_1 \geq 70 \\ 20x_1 + 10x_2 \geq 30 \\ 25x_1 + 50x_2 \geq 75 \\ x_1 \geq 0, x_2 \geq 0 \end{cases}$$

L'insieme ammissibile del problema di Programmazione Lineare che stiamo esaminando è dato quindi dall'intersezione di semipiani e si può indicare con

$$\widetilde{S} = \Big\{(x_1, x_2) \in \mathbb{R}^2 \mid 140x_1 \geq 70, 20x_1 + 10x_2 \geq 30, 25x_1 + 50x_2 \geq 75, x_1 \geq 0, x_2 \geq 0\Big\}.$$

Tutti i punti $P(x_1, x_2)$ appartenenti a questa regione sono punti dell'insieme ammissibile del problema e quindi tutti i punti di questa regione costituiscono soluzioni ammissibili del problema. Si osservi che, a differenza della regione ammissibile del problema considerato nell'esempio precedente, la regione ammissibile $\widetilde{S}$ è illimitata.
Ora, tracciando le curve di livello della funzione obiettivo $400x_1+600x_2$ si ottiene la famiglia di rette

$$400x_1 + 600x_2 = C.$$

Trattandosi di un problema di minimizzazione si vuole determinare il valore piú basso di C ottenuto scegliendo per x_1 e x_2 valori ammissibili cioè tali che $(x_1, x_2) \in \widetilde{S}$. Osservando la rappresentazione grafica della regione ammissibile $\widetilde{S}$ e osservando che la direzione di decrescita è quella opposta al vettore $\begin{pmatrix} 400 \\ 600 \end{pmatrix}$, si deduce che al diminuire di C, le rette di livello della funzione obiettivo intersecano la regione ammissibile finché $C \geq 1000$. Tale valore è ottenuto per $x_1 = 1$ e $x_2 = 1$ e non esistono altri punti della regione ammissibile in cui la funzione obiettivo assume valori minori. Il valore 1000 è, quindi, il minimo valore che la funzione obiettivo può raggiungere soddisfacendo i vincoli. Tale soluzione ottima è raggiunta in corrispondenza del punto P di coordinate $(x_1, x_2) = (1, 1)$; si osservi che anche in questo caso tale punto è un punto particolare della regione ammissibile. Si osservi, infine che in questo punto sono attivi i vincoli $20x_1 + 10x_2 \geq 30$ e $25x_1 + 50x_2 \geq 75$ mentre risulta non attivo il vincolo $140x_1 \geq 70$.

Abbiamo esaminato due esempi di interpretazione geometrica e soluzione grafica di problemi di Programmazione Lineare in due variabili. In entrambe i problemi è stato possibile determinare una soluzione ottima. Tuttavia è facile dedurre, sempre per via geometrica, che un problema di Programmazione Lineare può non ammettere soluzione ottima. Ad esempio, se nell'Esempio 5.3.2 sostituiamo il vincolo $x_2 \leq 400$ con il vincolo $x_2 \geq 1000$, la regione ammissibile sarebbe vuota nel senso che non esisterebbe nessun punto del piano che soddisfa tutti i vincoli. In questo caso il problema risulterebbe inammissibile e questo indipendentemente dalla funzione obiettivo e dal fatto che il problema è in forma di minimizzazione o massimizzazione.
Un altro esempio di problema di Programmazione Lineare che non ammette soluzione ottima si può ottenere considerando il problema dell'Esempio 5.3.3 e supponendo che la funzione obiettivo debba essere massimizzata anziché minimizzata. In questo caso nella regione ammissibile (che è illimitata) la funzione obiettivo può assumere valori arbitrariamente grandi cioè tali che comunque scelto un valore $M > 0$ esiste un punto in cui la funzione obiettivo assume valore maggiore di M; questo significa che il problema è illimitato superiormente e quindi non può esistere una soluzione ottima.

Sulla base di queste considerazioni sulla geometria di un problema di Programmazione Lineare in due variabili si può intuire che le situazioni che si possono verificare sono le seguenti:

- *il problema ammette soluzione ottima* (che può essere o non essere unica) in un vertice del poligono convesso che delimita la regione ammissibile;
- *il problema non ammette soluzione ottima* perché
  - la regione ammissibile è vuota
  - la regione ammissibile è illimitata e la funzione obiettivo è illimitata superiormente (se il problema è di massimizzazione) o illimitata inferiormente (se il problema è di minimizzazione).

5.4 ELEMENTI DI GEOMETRIA IN $\mathbb{R}^N$

5.4.1 Rette, semirette, segmenti

Definizione 5.4.1 *Siano dati un punto* $\bar{x} \in \mathbb{R}^n$ *ed una direzione* $d \in \mathbb{R}^n$. *L'insieme dei punti di* $\mathbb{R}^n$

$$\{x \in \mathbb{R}^n \mid x = \bar{x} + \lambda d,\ \lambda \in \mathbb{R}\}$$

è una retta *passante per* $\bar{x} \in \mathbb{R}^n$ *e avente come direzione* $d \in \mathbb{R}^n$. *L'insieme dei punti di* $\mathbb{R}^n$

$$\{x \in \mathbb{R}^n \mid x = \bar{x} + \lambda d,\ \lambda \geq 0\}$$

è una semiretta *avente origine in* $\bar{x} \in \mathbb{R}^n$ *e direzione* $d \in \mathbb{R}^n$.

Definizione 5.4.2 *Siano* x *e* y *due punti in* $\mathbb{R}^n$. *L'insieme dei punti di* $\mathbb{R}^n$ *ottenuti come*

$$\{z \in \mathbb{R}^n \mid z = (1-\lambda)x + \lambda y,\ 0 \leq \lambda \leq 1\}$$

è un segmento (chiuso) di estremi x e y *e viene sinteticamente indicato con la notazione* $[x, y]$.

Esempio 5.4.3 Nella Figura 5.4.1 è rappresentato il segmento in $\mathbb{R}^2$ avente per estremi i punti $x = (1,1)^T$ e $y = (8,5)^T$. Rappresentando i punti di questo segmento nella forma $z = (1-\beta)x + \beta y$, $\beta \in [0,1]$, per $\beta = 0$ ritroviamo il punto x, mentre per $\beta = 1$ ritroviamo il punto y; i punti segnati nella figura come x_a, x_b e x_c corrispondono rispettivamente a valori di β pari a 0.25, 0.5 e 0.75.

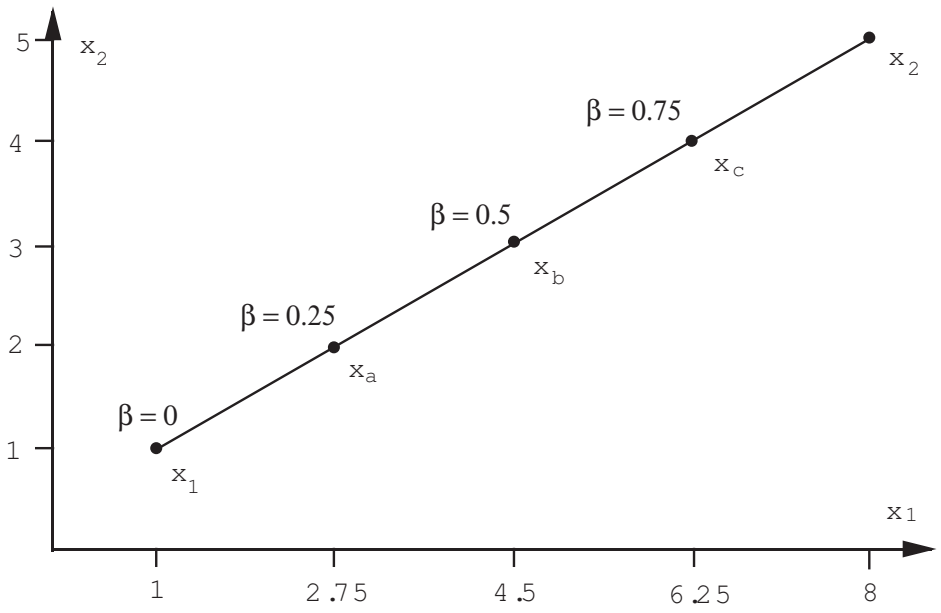


Fig. 5.4.1 Esempio di segmento.

Dalla Figura 5.4.1 risulta ovvio che il concetto di segmento è la generalizzazione, al caso di $\mathbb{R}^n$ del usuale concetto di segmento valido nel piano.
Notiamo anche come, nel caso in cui gli estremi appartengano ad $\mathbb{R}$, e sono quindi due numeri (scalari), diciamo a e b, il concetto di segmento (chiuso) di estremi a e b coincida con quello di intervallo $[a,b]$, fatto che giustifica la notazione $[x,y]$ impiegata per indicare il segmento.

5.4.2 Insiemi Convessi

Definizione 5.4.4 *Un insieme $X \subseteq \mathbb{R}^n$ è* convesso *se per ogni coppia di punti appartenenti all'insieme, appartiene all'insieme anche tutto il segmento che li congiunge.*

Utilizzando il concetto di segmento chiuso, la definizione di insieme convesso può essere riformulata nel modo seguente:

Un insieme X è convesso se per ogni coppia di vettori $x, y \in X$ si ha $[x, y] \subseteq X$.

Dalla definizione segue che l'insieme vuoto e l'insieme costituito da un solo vettore sono insiemi convessi (banali). Il più semplice insieme convesso non banale è il *segmento* di estremi $x, y \in \mathbb{R}^n$.

Esempio 5.4.5 In $\mathbb{R}^2$ gli insiemi (a) , (b) della Figura 5.4.2 sono convessi, mentre gli insiemi (c), (d) della stessa figura non lo sono. Infatti agli insiemi (c),(d) appartengono coppie di punti, quali quelle segnate nella figura, tali che il segmento che li congiunge presenta dei punti non appartenenti all'insieme; ciò non avviene invece comunque si prendano coppie di punti negli insiemi (a) e (b).

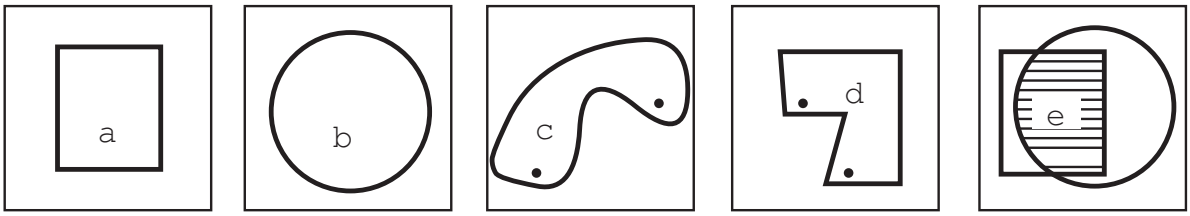


Fig. 5.4.2 Insiemi convessi e non convessi.

Una importante proprietà degli insiemi convessi è espressa dal seguente teorema.

Teorema 5.4.1 *L'intersezione di due insiemi convessi è un insieme convesso.*

Dimostrazione: Siano $X_1, X_2 \subseteq \mathbb{R}^n$ due insiemi convessi e sia $X = X_1 \cap X_2$ la loro intersezione. Siano x ed y due vettori in X, allora $x, y \in X_1$ ed $x, y \in X_2$. Poiché X_1 ed X_2 sono insiemi convessi abbiamo che $[x, y] \subseteq X_1$ e che $[x, y] \subseteq X_2$. Ma allora $[x, y] \subseteq X$ e l'insieme X è convesso □

Esempio 5.4.6 L'insieme (e) della Figura 5.4.2 è dato dall'intersezione di due insiemi convessi ed è convesso

Dal Teorema 5.4.1 si può derivare, con un semplice ragionamento induttivo, il seguente corollario.

Corollario 5.4.7 *L'intersezione di un numero finito di insiemi convessi è un insieme convesso.*

Passiamo ora a considerare dei particolari insiemi convessi che rivestono un ruolo importante nella teoria della programmazione lineare.

Definizione 5.4.8 *Sia a un vettore di $\mathbb{R}^n$ e b un numero reale. L'insieme*

$$H = \{x \in \mathbb{R}^n : a^T x = b\}$$

è detto iperpiano definito dall'equazione $a^T x = b$. Gli insiemi

$$\begin{aligned} S^{\leq} &= \{x \in \mathbb{R}^n : a^T x \leq b\} \\ S^{\geq} &= \{x \in \mathbb{R}^n : a^T x \geq b\} \end{aligned}$$

sono detti semispazi chiusi definiti dalle disequazioni $a^T x \leq b$ e $a^T x \geq b$.

Nel caso dello spazio $\mathbb{R}^2$ il concetto di iperpiano coincide con quello di retta, mentre nel caso dello spazio $\mathbb{R}^3$ il concetto di iperpiano coincide con quello di piano. In maniera intuitiva, i semispazi possono essere pensati come l' insieme dei punti che "giacciono" da una stessa parte rispetto all'iperpiano.

Esempio 5.4.9 Con riferimento alla Figura 5.4.3, l'iperpiano (= retta) $10x_1 + 5x_2 = 25$ divide lo spazio (= piano) in due semispazi: $S^{\geq} = \{x \in \mathbb{R}^2 : 10x_1 + 5x_2 \geq 25\}$, indicato in grigio nella figura, e $S^{\leq} = \{x \in \mathbb{R}^2 : 10x_1 + 5x_2 \leq 25\}$, indicato in bianco nella figura.

Notiamo che l'iperpiano H fa parte di tutti e due i semispazi e che l'intersezione dei due semispazi coincide con l'iperpiano. In termini insiemistici abbiamo che

$$H \subset S^{\geq}, \qquad H \subset S^{\leq}, \qquad S^{\geq} \cap S^{\leq} = H.$$

I semispazi e gli iperpiani sono insiemi convessi.

Teorema 5.4.2 *Un semispazio chiuso è un insieme convesso.*

Dimostrazione: Dimostreremo il teorema per un semispazio $S^{\leq} = \{x \in \mathbb{R}^n : a^T x \leq b\}$, la dimostrazione per il semispazio $S^{\geq}$ ottenuto invertendo il verso della disequazione è analoga. Consideriamo due generici vettori x ed y appartenenti ad $S^{\leq}$, vogliamo dimostrare che ogni vettore $z \in [x, y]$ appartiene ad $S^{\leq}$, ovvero soddisfa la relazione $a^T z \leq b$.
Sia $z = \beta x + (1-\beta)y$ con $0 \leq \beta \leq 1$. Poiché x ed y appartengono ad $S^{\leq}$ abbiamo che $a^T x \leq b$ e $a^T y \leq b$. Inoltre, poiché β ed $1-\beta$ sono reali non negativi abbiamo che

$$a^T(\beta x + (1-\beta)y) = \beta a^T x + (1-\beta)a^T y \leq \beta b + (1-\beta)b = b$$

e quindi che $a^T z \leq b$ □

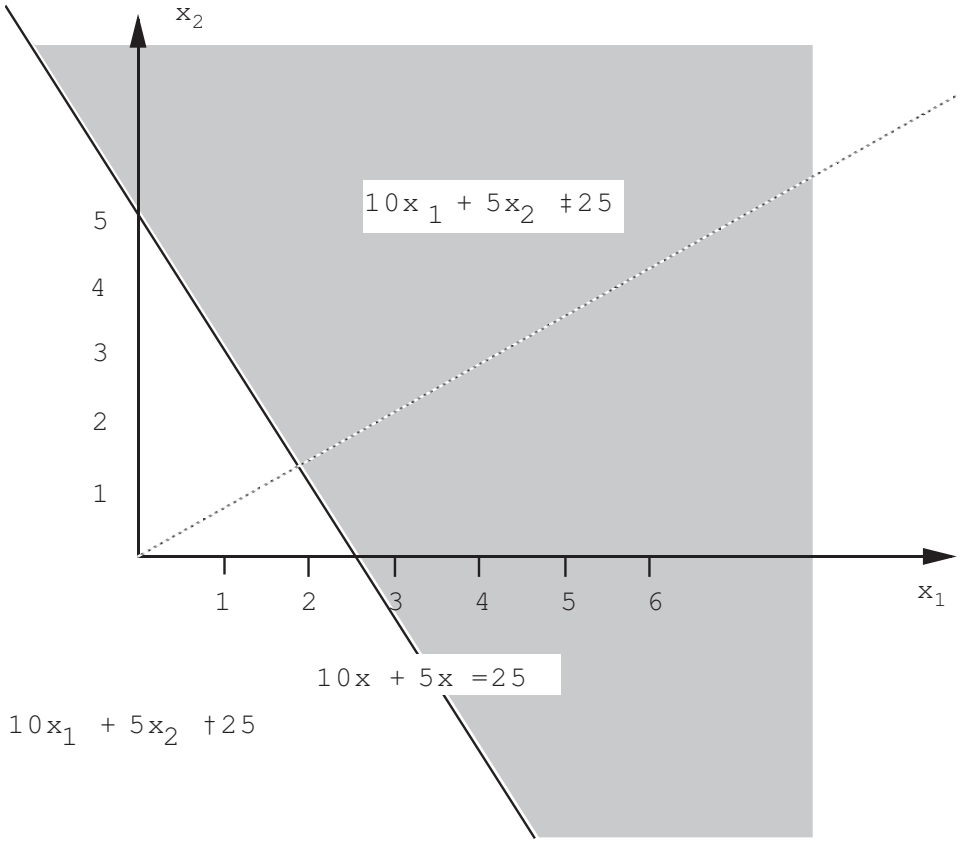


Fig. 5.4.3 Retta e semipiani individuati da un'equazione lineare.

Utilizzando il Teorema 5.4.2 e il Teorema 5.4.1 e ora facile dimostrare che anche un iperpiano è un insieme convesso.

Corollario 5.4.10 *Un iperpiano è un insieme convesso.*

Dimostrazione: Un iperpiano è l'intersezione di due semispazi chiusi ($S^{\leq}$ e $S^{\geq}$). Per il Teorema (5.4.2) un semispazio chiuso è un insieme convesso mentre, per il Teorema (5.4.1), l'intersezione di due insiemi convessi è un insieme convesso. □

Notiamo ora che l'insieme ammissibile di un problema di Programmazione Lineare è definito come l'insieme di punti che soddisfa i vincoli, cioè un insieme di equazioni e disequazioni lineari. Usando la terminologia appena introdotta, possiamo anche dire che l'insieme dei punti ammissibili di un problema di PL è dato dall'intersezione di un numero finito di semispazi (disequazioni lineari) e

iperpiani (equazioni lineari). Quindi, applicando il Teorema 5.4.2, il Corollario 5.4.10 e il Teorema 5.4.1 abbiamo il seguente risultato.

Teorema 5.4.3 *L'insieme ammissibile di un problema di programmazione lineare è un insieme convesso.*

In particolare è usuale introdurre la seguente definizione (dove si farà uso della nozione di insieme limitato[1]):

Definizione 5.4.11 *Un insieme $P \subseteq \mathbb{R}^n$ è un* poliedro *se è l'intersezione di un numero finito di semispazi chiusi e iperpiani. Un poliedro limitato viene detto* polìtopo.

Usando un punto di vista più algebrico possiamo parafrasare la precedente definizione e dire che un poliedro è l'insieme di soluzioni di un qualunque sistema di equazioni e disequazioni lineari. In particolare, notiamo che l'insieme vuoto è un poliedro (è l'insieme di soluzioni di un sistema di equazioni inconsistente) e che anche $\mathbb{R}^n$ è un poliedro ($\mathbb{R}^n$ è, per esempio, l'insieme di soluzioni dell'equazione lineare $0x_1 + 0x_2 + \cdots + 0x_n = 0$). Naturalmente, poiché un poliedro $P \subseteq \mathbb{R}^n$ può essere sempre descritto nella forma $P = \{x \in \mathbb{R}^n \mid Ax \geq b\}$ dove A è una matrice $m \times n$ e $b \in \mathbb{R}^m$, una caratterizzazione alternativa (algebrica) di poliedro è la seguente:

un insieme $P \subseteq \mathbb{R}^n$ si dice poliedro se esiste una matrice A, $m \times n$ e $b \in \mathbb{R}^m$ tale che risulti

$$P = \{x \in \mathbb{R}^n \mid Ax \geq b\}.$$

Osserviamo che risulta evidente che l'insieme ammissibile di un problema di Programmazione Lineare è un poliedro.

5.4.3 Vertici

In questa sezione formalizziamo il concetto intuitivo di *vertice*. Questo concetto riveste un ruolo fondamentale nella teoria della Programmazione Lineare[2].

[1] Un insieme $P \subset \mathbb{R}^n$ si dice *limitato* se esiste una costante $M > 0$ tale che, per ogni punto x appartenente a P risulti $|x_i| \leq M$ per ogni $i = 1, \ldots, n$

[2] Per precisione notiamo che nella letteratura la Definizione 5.4.12 che segue è la definizione di *punto estremo*, mentre viene normalmente indicato con vertice un punto che soddisfa una proprietà più complessa, che qui non riportiamo. Nel caso però di poliedri, che saranno gli unici insiemi convessi che

Definizione 5.4.12 *Un vettore x appartenente ad un insieme convesso C è detto* vertice *di C se non esistono due punti distinti $x_1, x_2 \in C$ tali che $x \neq x_1$, $x \neq x_2$ ed $x \in [x_1, x_2]$.*

Nell'insieme di Figura 5.4.4 il punto A non è un vertice, in quanto è interno al segmento che congiunge i punti B e C, anch'essi appartenenti all'insieme; lo stesso vale per il punto D, interno al segmento [E,F]. Sono invece vertici dell'insieme i punti E, F, G, H.

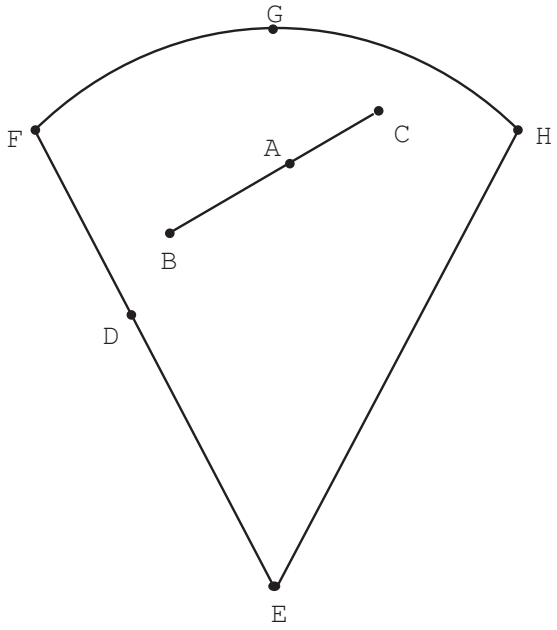


Fig. 5.4.4 Vertici di un insieme.

Il problema che ci proponiamo di affrontare ora è quello di *caratterizzare* i vertici dell'insieme dei punti ammissibili di un problema di PL. Una risposta è fornita dal teorema che segue che mette in relazione l'esistenza di un vertice con l'esistenza di n vincoli attivi linearmente indipendenti. Si consideri quindi un generico problema di Programmazione Lineare scritto nella forma

$$\begin{cases} \min c^T x \\ Ax \geq b \end{cases}$$

prenderemo in considerazione in questo corso, le due definizioni coincidono, cioé un punto appartenente a un poliedro è un vertice del poliedro stesso se e solo se è un suo punto estremo.

dove $c \in \mathbb{R}^n$, $x \in \mathbb{R}^n$, $b \in \mathbb{R}^m$ e $A \in \mathbb{R}^{m \times n}$. Denotando con a_i^T, $i = 1, \ldots, m$ le righe della matrice A possiamo introdurre la seguente definizione:

Definizione 5.4.13 VINCOLI ATTIVI
Se un vettore $\bar{x} \in \mathbb{R}^n$ soddisfa $a_i^T \bar{x} = b_i$ per qualche $i \in \{1, \ldots, m\}$ si dice che il corrispondente vincolo è attivo *in $\bar{x}$. Inoltre, dato $\bar{x} \in \mathbb{R}^n$ si indica con $I(\bar{x})$ l'insieme degli indici dei vincoli attivi, cioè:*

$$I(\bar{x}) = \left\{ i \in \{1, \ldots, m\} \;\mid\; a_i^T \bar{x} = b_i \right\}.$$

Per brevità, nel seguito, chiameremo spesso *vincoli linearmente indipendenti* quei vincoli per i quali risultano linearmente indipendenti i vettori a_i^T corrispondenti.

Teorema 5.4.4 *Siano dati un poliedro $P = \{x \in \mathbb{R}^n \mid Ax \geq b\}$ e un punto $\bar{x} \in P$. Il punto $\bar{x}$ è un vertice di P se e solo se esistono n righe a_i^T della matrice A con $i \in I(\bar{x})$ che sono linearmente indipendenti.*

Dimostrazione: (Facoltativa) Dimostriamo innanzitutto la condizione necessaria, cioè che se esiste un vertice del poliedro allora esistono n vincoli attivi nel vertice linearmente indipendenti. Supponiamo che $\bar{x}$ sia un vertice del poliedro P e che, per assurdo, il numero dei vincoli attivi in $\bar{x}$ linearmente indipendenti sia $k < n$. Allora esiste un vettore $d \in \mathbb{R}^n$ non nullo tale che

$$a_i^T d = 0, \qquad \text{per ogni} \quad i \in I(\bar{x}). \tag{5.4.1}$$

Poiché per ogni vincolo non attivo in $\bar{x}$, cioè per ogni $i \notin I(\bar{x})$ si ha

$$a_i^T \bar{x} > b_i,$$

allora esiste $\epsilon > 0$ sufficientemente piccolo tale che i vettori

$$\begin{aligned} y &= \bar{x} - \epsilon d \\ z &= \bar{x} + \epsilon d \end{aligned}$$

soddisfano $a_i^T y \geq b_i$, $a_i^T z \geq b_i$ per ogni $i \notin I(\bar{x})$. Inoltre per la (5.4.1), per ogni $i \in I(\bar{x})$ si ha

$$\begin{aligned} a_i^T y &= a_i^T \bar{x} - \epsilon a_i^T d = b_i \\ a_i^T z &= a_i^T \bar{x} + \epsilon a_i^T d = b_i \end{aligned}$$

e quindi i vettori y e z soddisfano tutti i vincoli $a_i^T x \geq b_i$, $i = 1, \ldots, m$ e quindi appartengono al poliedro P. Ora poiché risulta

$$\bar{x} = \frac{1}{2}y + \frac{1}{2}z,$$

con y e z vettori di P entrambi diversi da $\bar{x}$, allora $\bar{x}$ non è un vertice e questa è una contraddizione.
Dimostriamo ora la condizione sufficiente, cioè che se esistono n vincoli attivi in uno stesso punto linearmente indipendenti allora tale punto è un vertice di P. Supponiamo quindi che esistano n righe a_i^T con $i \in I(\bar{x})$ linearmente indipendenti e che per assurdo $\bar{x}$ non sia vertice di P. Innanzitutto osserviamo che se $\bar{x}$ non è un vertice, allora necessariamente $P \supset \{\bar{x}\}$ (cioè $\bar{x}$ non è l'unico punto di P) ed inoltre esistono due vettori y e z entrambi diversi da $\bar{x}$ appartenenti a P, cioè che soddisfano

$$a_i^T y \geq b_i, \qquad a_i^T z \geq b_i, \qquad i = 1, \ldots, m,$$

tali che

$$\bar{x} = \lambda y + (1 - \lambda)z \qquad \text{con} \quad \lambda \in (0, 1).$$

Ora, se per qualche $i \in I(\bar{x})$ risultasse $a_i^T y > b_i$ oppure $a_i^T z > b_i$ allora si avrebbe

$$a_i^T \bar{x} = \lambda a_i^T y + (1 - \lambda) a_i^T z > \lambda b_i + (1 - \lambda) b_i = b_i$$

e questo contraddice il fatto che $i \in I(\bar{x})$. Allora deve necessariamente essere

$$a_i^T y = b_i, \qquad a_i^T z = b_i, \qquad \text{per ogni} \quad i \in I(\bar{x})$$

ma questo implica che il sistema

$$a_i^T x = b_i, \qquad i \in I(\bar{x})$$

ammette piú di una soluzione (cioè $\bar{x}$, y e z) contraddicendo l'ipotesi che esistano n righe a_i^T con $i \in I(\bar{x})$ linearmente indipendenti nel qual caso, come è noto, la soluzione è unica. □

Seguono dei corollari che discendono in maniera immediata dal teorema appena dimostrato.

Corollario 5.4.14 *Sia dato un poliedro $P = \{x \in \mathbb{R}^n \mid Ax \geq b\}$. Se la matrice $A \in \mathbb{R}^{m \times n}$ ha un numero di righe linearmente indipendenti minore di n, allora P non ha vertici. In particolare se $m < n$ allora P non ha vertici.*

Corollario 5.4.15 *Siano dati un poliedro* $P = \{x \in \mathbb{R}^n \mid Ax \geq b\}$ *e un punto* $\bar{x} \in P$. *Il punto* $\bar{x}$ *è un vertice di* P *se e solo se è soluzione unica del sistema*

$$a_i^T x = b_i \qquad i \in I(\bar{x}).$$

Corollario 5.4.16 *Un poliedro* $P = \{x \in \mathbb{R}^n \mid Ax \geq b\}$ *ha al piú un numero finito di vertici.*

Dimostrazione: Se $m < n$ il poliedro ovviamente non ha vertici. Se $m \geq n$, per il Corollario 5.4.15 ogni vertice del poliedro corrisponde ad un sottoinsieme di n righe linearmente indipendenti della matrice A. Ora poiché la matrice A ha al piú $\binom{m}{n} = \frac{m!}{n!(m-n)!}$ sottoinsiemi distinti di n righe, allora il poliedro ha al piú $\frac{m!}{n!(m-n)!}$ vertici. □

Esempio 5.4.17 *Determinare i vertici del poliedro descritto dalle disuguaglianze*

$$\begin{cases} 3x_1 - 2x_2 \geq -30 \\ 2x_1 - x_2 \geq -12 \\ x_1 \geq 0 \\ x_2 \geq 0 \end{cases}$$

e rappresentarlo geometricamente su un sistema di assi cartesiani Ox_1x_2.

Si osservi innanzitutto che in questo esempio la dimensione n è pari a 2 e il numero dei vincoli m pari a 4. Si devono determinare tutte le possibili intersezioni delle rette $3x_1 - 2x_2 = -30$, $2x_1 - x_2 = -12$, $x_1 = 0$, $x_2 = 0$ che costituiscono il poliedro; si osservi che tali intersezioni sono $\binom{4}{2} = 6$. Per ogni punto così ottenuto si deve verificare innanzitutto l'appartenenza del punto al poliedro, e poi, affinché sia un vertice, l'indipendenza lineare dei vincoli attivi in quel punto.

1. Il sistema $\begin{cases} 3x_1 - 2x_2 = -30 \\ 2x_1 - x_2 = -12 \end{cases}$ corrispondente al primo e al secondo vincolo ha come unica soluzione il punto $P_1 = (6, 24)$ che si verifica immediatamente appartenere al poliedro; in questo punto ovviamente risultano attivi il primo e il secondo vincolo e quindi $I(P_1) = \{1, 2\}$ e poiché i vettori $a_1^T = (3, -2)$ e $a_2^T = (2, -1)$ corrispondenti a questi due vincoli sono linearmente indipendenti, allora il punto P_1 è un vertice del poliedro.

2. Il sistema $\begin{cases} 3x_1 - 2x_2 = -30 \\ x_1 = 0 \end{cases}$ corrispondente al primo e al terzo vincolo ha come unica soluzione il punto $P_2 = (0, 15)$ che non appartiene al poliedro.

3. Il sistema $\begin{cases} x_1 - 2x_2 = -30 \\ x_2 = 0 \end{cases}$ corrispondente al primo e al quarto vincolo ha come unica soluzione il punto $P_3 = (-10, 0)$ che non appartiene al poliedro.

4. Il sistema $\begin{cases} 2x_1 - x_2 = -12 \\ x_1 = 0 \end{cases}$ corrispondente al secondo e al terzo vincolo ha come unica soluzione il punto $P_4 = (0, 12)$ che si verifica immediatamente appartenere al poliedro; in questo punto ovviamente risultano attivi il secondo e il terzo vincolo e quindi $I(P_4) = \{2, 3\}$ e poiché i vettori $a_2^T = (2, \ -1)$ e $a_3^T = (1, \ 0)$ corrispondenti a questi due vincoli sono linearmente indipendenti, allora il punto P_4 è un vertice del poliedro.

5. Il sistema $\begin{cases} 2x_1 - x_2 = -12 \\ x_2 = 0 \end{cases}$ corrispondente al secondo e al quarto vincolo ha come unica soluzione il punto $P_5 = (-6, 0)$ che non appartiene al poliedro.

6. Il sistema $\begin{cases} x_1 = 0 \\ x_2 = 0 \end{cases}$ corrispondente al terzo e al quarto vincolo ha come unica soluzione il punto $P_6 = (0, 0)$ che si verifica immediatamente essere appartenente al poliedro; in questo punto ovviamente risultano attivi il terzo e il quarto vincolo e quindi $I(P_6) = \{3, 4\}$ e poiché i vettori $a_3^T = (1, \ 0)$ e $a_4^T = (0, \ 1)$ corrispondenti a questi due vincoli sono linearmente indipendenti, allora il punto P_6 è un vertice del poliedro.

□

Esempio 5.4.18 *Dato il poliedro descritto dalle seguenti disuguaglianze*

$$\begin{cases} x_1 + 2x_2 + 2x_3 \leq 2 \\ x_1 + 4x_2 + 2x_3 \leq 3 \\ x_1 \geq 0 \\ x_2 \geq 0 \\ x_3 \geq 0 \end{cases}$$

verificare se i punti $P_1 = (0, 0, 0)$, $P_2 = (0, 0, 1/2)$ *e* $P_3 = (0, 0, 1)$ *sono vertici del poliedro.*

In questo esempio la dimensione n è pari a 3 e il numero dei vincoli m è pari a 5. Riscrivendo i primi due vincoli nella forma di disuguaglianza di maggiore o uguale, la matrice A dei coefficienti delle disuguaglianze che descrivono il poliedro

è

$$A = \begin{pmatrix} -1 & -2 & -2 \\ -1 & -4 & -2 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Per ogni punto dato, dopo aver verificato l'appartenenza del punto al poliedro, si deve verificare se esistono tre vincoli attivi in quel punto linearmente indipendenti.
Nel punto $P_1 = (0, 0, 0)$ (che appartiene al poliedro) sono attivi il terzo, il quarto e il quinto vincolo e quindi $I(P_1) = \{3, 4, 5\}$ e poiché le righe a_3^T, a_4^T e a_5^T della matrice A sono linearmente indipendenti, il punto P_1 è vertice del poliedro.
Nel punto $P_2 = (0, 0, 1/2)$ (che appartiene al poliedro) sono attivi solamente due vincoli (il terzo e il quarto) e quindi il punto P_2 non può essere un vertice del poliedro.
Nel punto $P_3 = (0, 0, 1)$ (che appartiene al poliedro) si hanno tre vincoli attivi; in particolare risulta $I(P_3) = \{1, 3, 4\}$ e le corrispondenti righe a_1^T, a_3^T e a_4^T della matrice A sono linearmente indipendenti e quindi il punto P_3 è un vertice del poliedro. □

Esempio 5.4.19 *Determinare i vertici del poliedro descritto dalle seguenti disuguaglianze*

$$\begin{cases} x_1 + 2x_2 + x_3 \leq 3 \\ 3x_1 - x_2 + x_3 \leq 2 \\ 2x_1 + x_2 + x_3 \leq 3 \\ 4x_1 + x_2 + 2x_3 \leq 4. \end{cases}$$

In questo caso si ha $n = 3$ e $m = 4$ e quindi si devono determinare punti del poliedro in cui sono attivi tre vincoli linearmente indipendenti. Si devono quindi considerare $\binom{4}{3} = 4$ sistemi di equazioni in tre variabili:

1. il sistema ottenuto dai primi tre vincoli ha come unica soluzione il punto $P_1(1, 1, 0)$ che non è ammissibile;

2. si consideri ora il sistema ottenuto dal primo, dal secondo e dal quarto vincolo; la matrice dei coefficienti di questo sistema ha rango 2 in quanto i tre vincoli considerati (il primo, il secondo e il quarto) non sono linearmente indipendenti (il vettore corrispondente al quarto vincolo si può ottenere come somma dei vettori corrispondenti al primo e al secondo vincolo). Quindi non si può avere un vertice.

3. il sistema ottenuto dal primo, dal terzo e dal quarto vincolo ha come unica soluzione il punto $P_2 = (2, 2, -3)$ che appartiene al poliedro e poiché i tre vincoli attivi in P_3 sono linearmente indipendenti, P_2 è un vertice del poliedro.

4. il sistema ottenuto dal secondo, dal terzo e dal quarto vincolo ha come unica soluzione il punto $P_3 = (3, 2, -5)$ che appartiene al poliedro e poiché i tre vincoli attivi in P_3 sono linearmente indipendenti, P_3 è un vertice del poliedro.

□

Osservazione 5.4.20 Se tra vincoli che descrivono un poliedro è presente un vincolo di uguaglianza, nella determinazione dei vertici ci si può limitare a considerare solo i sistemi che contengono questo vincolo di uguaglianza, facendo diminuire considerevolmente il numero dei sistemi da prendere in considerazione. L'esempio che segue mostra una situazione di questo tipo.

Esempio 5.4.21 *Calcolare tutti i vertici del seguente poliedro:*

$$\begin{array}{rcrcrcl} 2x_1 & - & x_2 & + & x_3 & \leq & 4 \\ x_1 & & & - & x_3 & = & 1 \\ x & \geq & 0. & & & & \end{array}$$

Bisogna analizzare tutti i possibili sistemi di tre equazioni "estraibili" dal sistema dato, che ha cinque vincoli. Riportiamo il sistema per esteso:

$$\begin{array}{rcrcrcl} 2x_1 & - & x_2 & + & x_3 & \leq & 4 \\ x_1 & & & - & x_3 & = & 1 \\ x_1 & & & & & \geq & 0 \\ & & x_2 & & & \geq & 0 \\ & & & & x_3 & \geq & 0. \end{array}$$

Siccome è presente un vincolo di uguaglianza, ci si può limitare ad analizzare solo i sistemi che contengono il vincolo di uguaglianza.
I vertici sono 2:

$$v_1 = (5/3, 0, 2/3)^T,$$

corrispondente al sistema formato dal primo, secondo e quarto vincolo e

$$v_2 = (1, 0, 0)^T,$$

corrispondente al sistema formato dal secondo, quarto e quinto vincolo.

Per quanto riguarda gli altri sistemi "estraibili" risulta che per il sistema formato dai vincoli

- primo, secondo e terzo: la soluzione corrispondente non è ammissibile;
- primo, secondo e quinto: la soluzione corrispondente non è ammissibile;
- secondo, terzo e quarto: la soluzione corrispondente non è ammissibile;
- secondo, terzo e quinto: il rango è minore di tre. □

Come è facile osservare, non tutti i poliedri hanno almeno un vertice. Un controesempio banale di poliedro che non ha vertici è un semispazio in $\mathbb{R}^n$ con $n > 1$. Se la matrice A ha un numero di righe strettamente minore di n, allora il poliedro $P = \{x \in \mathbb{R}^n \mid Ax \geq b\}$ non ha vertici perché è ovvio che in questo caso non è possibile trovare n vincoli attivi né, tantomeno, n vincoli attivi linearmente indipendenti (vedi il Corollario 5.4.14).

Il fatto che un poliedro abbia o non abbia vertici è basato sulla possibilità di un poliedro di contenere o meno rette. Questo concetto verrà ora formalizzato introducendo innanzitutto la seguente definizione.

Definizione 5.4.22 *Si dice che un poliedro P* contiene una retta *se esiste un punto $\tilde{x} \in P$ e un vettore non nullo $d \in \mathbb{R}^n$ tale che $\tilde{x} + \lambda d \in P$ per ogni $\lambda \in \mathbb{R}$. Si dice che un poliedro P* contiene una semiretta *se esiste un punto $\tilde{x} \in P$ e un vettore non nullo $d \in \mathbb{R}^n$ tale che $\tilde{x} + \lambda d \in P$ per ogni $\lambda \geq 0$, $\lambda \in \mathbb{R}$.*

Riportiamo quindi, senza dimostrazione, il seguente risultato, che non verrà utilizzato nel seguito, ma che aiuta a capire la relazione tra vertici e poliedri che non contengono rette.

Teorema 5.4.5 *Sia P un poliedro non vuoto. P possiede almeno un vertice se e solo se P non contiene rette.*

5.5 IL TEOREMA FONDAMENTALE DELLA PROGRAMMAZIONE LINEARE

Quanto fino ad ora esaminato permette di enunciare e dimostrare un risultato di fondamentale importanza che caratterizza i problemi di Programmazione Lineare.

Teorema 5.5.1 – Teorema Fondamentale della Programmazione Lineare
Si consideri il problema di Programmazione Lineare

$$\begin{cases} \min c^T x \\ Ax \geq b. \end{cases} \tag{PL}$$

Supponiamo che il poliedro $P = \{x \in \mathbb{R}^n \mid Ax \geq b\}$ *non contenga rette. Allora una e una sola delle seguenti tre affermazioni è vera:*

1. *Il problema* (PL) *è inammissibile, ovvero il poliedro* P *è vuoto;*
2. *Il problema* (PL) *è illimitato inferiormente;*
3. *Il problema* (PL) *ammette soluzioni ottime e almeno una di queste soluzioni è un vertice del poliedro* P*.*

Prima di dimostrare questo teorema enunciamo e dimostriamo un lemma che sarà alla base della dimostrazione del Teorema fondamentale.

Lemma 5.5.1 *Si consideri il problema di Programmazione Lineare*

$$\begin{cases} \min c^T x \\ Ax \geq b. \end{cases}$$

Supponiamo che il poliedro $P = \{x \in \mathbb{R}^n \mid Ax \geq b\}$ *sia non vuoto e non contenga rette. Supponiamo inoltre che il problema non sia illimitato inferiormente. Allora se* $\tilde{x}$ *è un punto di* P *che non è un vertice di* P*, è possibile trovare un punto* $\hat{x}$ *appartenente a* P *tale che* $c^T\hat{x} \leq c^T\tilde{x}$ *e il numero di vincoli attivi linearmente indipendenti in* $\hat{x}$ *è maggiore che in* $\tilde{x}$*.*

Dimostrazione: (Facoltativa) Sia $\tilde{x}$ un punto qualunque di P e indichiamo con k il numero dei vincoli attivi in $\tilde{x}$ che sono linearmente indipendenti. Siccome $\tilde{x}$ non è un vertice, ne segue che $k < n$. Sia $I(\tilde{x})$ l'insieme degli indici dei vincoli attivi in $\tilde{x}$; poiché $k < n$, si può trovare un vettore $d \in \mathbb{R}^n$ non nullo tale che $a_i^T d = 0$ per ogni $i \in I(\tilde{x})$. Inoltre, si può assumere che $c^T d \leq 0$; infatti se

questo non si verificasse (cioè se fosse $c^T d > 0$) sarebbe sufficiente prendere $-d$ e ottenere comunque $c^T d \leq 0$.
Ora possono verificarsi due casi: $c^T d < 0$ e $c^T d = 0$.
• Primo caso: $c^T d < 0$. Consideriamo la semiretta $x(\lambda) = \tilde{x} + \lambda d$ con $\lambda \geq 0$. Per ogni punto di tale semiretta, ovvero per ogni $\lambda \geq 0$, e per $i \in I(\tilde{x})$ si ha

$$a_i^T x(\lambda) = a_i^T \tilde{x} + \lambda a_i^T d = a_i^T \tilde{x} = b_i. \tag{5.5.1}$$

Quindi tutti i vincoli che erano attivi in $\tilde{x}$ rimangono attivi in tutti i punti della semiretta. Ora, se l'intera semiretta è contenuta nel poliedro P si può far tendere λ a $+\infty$ e poiché $c^T d < 0$ si ha

$$\lim_{\lambda \to +\infty} c^T(\tilde{x} + \lambda d) = c^T \tilde{x} + \lim_{\lambda \to +\infty} \lambda c^T d = -\infty.$$

Il problema sarebbe quindi illimitato inferiormente, ma ciò è stato escluso per ipotesi e quindi la semiretta non è interamente contenuta in P. Se la semiretta $x(\lambda)$ non è interamente contenuta in P, devono esistere valori di λ per i quali i punti $x(\lambda)$ non appartengono al poliedro P, ovvero deve esistere almeno un indice $j \notin I(\tilde{x})$ tale che, per tali valori di λ, il $j-$esimo vincolo è violato, cioè risulta $a_j^T x(\lambda) < b_j$. Tra questi indici j deve esistere un indice j_0 tale che possa essere scelto un $\hat{\lambda} > 0$ in modo che risulti

$$\begin{aligned} a_{j_0}^T x(\hat{\lambda}) &= b_{j_0} \\ a_j^T x(\hat{\lambda}) &\geq b_j, \qquad \text{per ogni} \qquad j \notin I(\tilde{x}). \end{aligned} \tag{5.5.2}$$

Poiché per la (5.5.1) per ogni $\lambda \geq 0$ risulta $a_j^T x(\lambda) = b_j$ quando $j \in I(\tilde{x})$, il punto $\hat{x} = x(\hat{\lambda})$ appartiene al poliedro P. Dalla (5.5.2), ricordando che $j_0 \notin I(\tilde{x})$, si ha che il vincolo j_0-esimo non era attivo in $\tilde{x}$ ed è attivo in $\hat{x} = x(\hat{\lambda})$ che è un punto del poliedro. Quindi, spostandosi da $\tilde{x}$ a $\tilde{x} + \hat{\lambda} d$, il numero dei vincoli attivi aumenta di almeno uno.
Dobbiamo ora dimostrare che $a_{j_0}^T$ (ovvero la riga della matrice A corrispondente al vincolo che è divenuto attivo passando da $\tilde{x}$ a $\tilde{x} + \hat{\lambda} d$) non può essere ottenuta come combinazione lineare delle righe a_i^T, $i \in I(\tilde{x})$ (ovvero delle righe corrispondenti ai vincoli attivi in $\tilde{x}$). Infatti, se per assurdo fosse

$$a_{j_0} = \sum_{i \in I(\tilde{x})} \mu_i a_i \qquad \text{con} \quad \mu_i \in \mathbb{R}, \qquad \mu_i \text{ non tutti nulli}, \tag{5.5.3}$$

moltiplicando scalarmente per il vettore d entrambe i membri della (5.5.3) e tenendo conto che $a_i^T d = 0$ per ogni $i \in I(\tilde{x})$, si avrebbe

$$a_{j_0}^T d = \sum_{i \in I(\tilde{x})} \mu_i a_i^T d = 0 \tag{5.5.4}$$

e questo è assurdo perché $j_0 \notin I(\tilde{x})$ ed invece dalla (5.5.2) risulterebbe

$$b_{j_0} = a_{j_0}^T x(\hat{\lambda}) = a_{j_0}^T(\tilde{x} + \hat{\lambda} d) = a_{j_0}^T \tilde{x}.$$

Perciò, spostandosi da $\tilde{x}$ a $\tilde{x} + \hat{\lambda}d$, il numero dei vincoli attivi linearmente indipendenti è almeno pari a $k+1$. Inoltre, ricordando che $c^T d < 0$ e $\hat{\lambda} > 0$, si ha che $c^T\hat{x} = c^T\tilde{x} + \hat{\lambda}c^T d < c^T\tilde{x}$. Possiamo quindi concludere che nel caso $c^T d < 0$ l'affermazione del lemma è verificata.
• Secondo caso: $c^T d = 0$. Consideriamo la retta $x(\lambda) = \tilde{x} + \lambda d$ con $\lambda \in \mathbb{R}$. Poiché si è supposto che il poliedro P non contenga rette, ragionando nello stesso modo del caso precedente, ci si può spostare da $\tilde{x}$ lungo la direzione d e determinare un punto $\hat{x}$ in cui il numero dei vincoli attivi linearmente indipendenti è maggiore del numero dei vincoli attivi linearmente indipendenti in $\tilde{x}$. Inoltre, poiché $c^T d = 0$ si ha $c^T\hat{x} = c^T\tilde{x} + \hat{\lambda}c^T d = c^T\tilde{x}$. Quindi, anche in questo caso, l'affermazione del lemma risulta verificata.

□

Passiamo ora alla **dimostrazione del Teorema Fondamentale della Programmazione Lineare**.

Dimostrazione: Le tre affermazioni 1, 2 e 3 sono ovviamente mutuamente escludentesi (cioè al più una di esse può essere vera). Per dimostrare il teorema è allora sufficiente far vedere che almeno una di esse è vera. A questo fine basta mostrare che se né l'affermazione 1 né quella 2 sono verificate allora l'affermazione 3 è verificata. Supponiamo dunque che P sia non vuoto e che il problema di Programmazione Lineare (PL) non sia illimitato inferiormente.
Se P è costituito da un solo punto $\bar{x}$, cioè $P = \{\bar{x}\}$, allora $\bar{x}$ è un vertice ed è anche, ovviamente, una soluzione ottima del problema; il teorema è quindi vero in questo caso.
Consideriamo allora il caso non banale in cui P è costituito da infiniti punti[3]. Per dimostrare che l'affermazione 3 è vera dimostriamo utilizziamo il risultato del Lemma 5.5.1, ovvero che se $\tilde{x}$ è un punto di P che non è un vertice, è possibile trovare un punto $\hat{x}$ appartenente a P tale che $c^T\hat{x} \leq c^T\tilde{x}$ e il numero di vincoli attivi linearmente indipendenti in $\hat{x}$ è maggiore che in $\tilde{x}$. Il punto $\hat{x}$ appartiene a P e sono possibili allora due casi: o $\hat{x}$ è un vertice di P o è possibile applicare di nuovo il risultato del Lemma 5.5.1 e concludere che esiste un ulteriore punto $\check{x}$ in P in cui il numero di vincoli attivi linearmente indipendenti è strettamente maggiore del numero di vincoli attivi linearmente indipendenti in $\hat{x}$ e $c^T\check{x} \leq c^T\hat{x}$. Iterando questo procedimento, e tenendo conto che il numero di vincoli attivi linearmente indipendenti in un punto può essere al più n, un semplice ragionamento induttivo mostra che dal Lemma 5.5.1 possiamo dedurre che

[3]Se il poliedro P contiene almeno due punti distinti deve contenere, in quanto insieme convesso, tutto il segmento che congiunge questi due punti. Siccome questo segmento contiene infiniti punti possiamo concludere che un poliedro non vuoto o contiene un singolo punto o ne contiene infiniti

$$\begin{array}{c}\text{Se } \tilde{x} \text{ è un punto di } P \text{ che non è un vertice,}\\ \text{allora è possibile trovare un vertice } \hat{v} \text{ di } P \text{ tale che}\\ c^T\hat{v} \leq c^T\tilde{x}.\end{array} \tag{5.5.5}$$

Siano, ora, $\{v_1, \ldots, v_p\}$ i vertici di P (ricordiamo che i vertici sono sicuramente in numero finito, si veda il Corollario 5.4.16); indichiamo con v^* uno di questi vertici per cui $c^T v^* \leq c^T v_h$ per ogni $h = 1, \ldots, p$. Dalla definizione di v^* e dalla (5.5.5) segue immediatamente che per ogni punto $\tilde{x} \in P$ possiamo scrivere, per un qualche vertice $\hat{v}$:

$$c^T v^* \leq c^T\hat{v} \leq c^T\tilde{x}.$$

Questo mostra che il vertice v^* è una soluzione ottima del problema di Programmazione Lineare (PL) e che l'affermazione 3 è vera. □

Un'immediata conseguenza del Teorema Fondamentale della Programmazione Lineare è che se il poliedro è un politopo non vuoto, allora il problema ammette soluzione ottima in un vertice del politopo. Questo risultato è formalizzato nel seguente corollario.

Corollario 5.5.2 *Sia dato il problema di Programmazione Lineare*

$$\begin{cases}\min c^T x\\ Ax \geq b.\end{cases}$$

Se il poliedro $P = \{x \in \mathbb{R}^n \mid Ax \geq b\}$ *è un politopo non vuoto, allora il problema di Programmazione Lineare ammette soluzione ottima (finita) in un vertice del poliedro* P.

Osservazione 5.5.3 La struttura lineare di un problema di Programmazione Lineare è l'elemento chiave che permette di ottenere un risultato così forte [4] circa

[4] In effetti è possibile ottenere risultati ancora più forti di quelli fin qui elencati. Usando il Teorema 5.4.5 possiamo sostituire l'ipotesi che il poliedro "non contenga rette" con quella che "possieda almeno un vertice". È possibile mostrare che in effetti questa ipotesi (che il poliedro "non contenga rette" o, equivalentemente "possieda almeno un vertice") è necessaria solo per dimostrare che nel caso 3 del Teorema 5.5.1 se esistono vertici allora c'è almeno una soluzione ottima che cade su un vertice. La dimostrazione di questo risultato più forte richiede però strumenti analitici più complessi di quelli usati in questo corso. Vogliamo comunque riportare, per completezza, questa versione del Teorema 5.5.1.

la possibile soluzione di un problema di ottimizzazione. Infatti, come controesempio si consideri il problema in una variabile reale

$$\begin{cases} \min \dfrac{1}{x} \\ \\ x \geq 1. \end{cases}$$

Questo problema non ammette soluzione ottima pur non essendo illimitato inferiormente. L'alternativa espressa dal Teorema Fondamentale della Programmazione Lineare in questo caso non vale proprio perché viene meno l'ipotesi fondamentale di linearità della funzione obiettivo.

Osservazione 5.5.4 Se un problema di Programmazione Lineare, come accade spesso nei problemi provenienti da modelli reali, presenta limitazioni inferiori e superiori sulle variabili cioè è del tipo

$$\begin{cases} \min c^T x \\ Ax \geq b \\ l \leq x \leq u \end{cases}$$

dove $l \in \mathbb{R}^n$ e $u \in \mathbb{R}^n$ sono rispettivamente una limitazione inferiore e superiore delle variabili, allora il poliedro che descrive l'insieme ammissibile è un politopo e quindi vale la caratterizzazione delle soluzioni data dal Corollario 5.5.2.

È interessante approfondire un poco la natura dell'insieme delle soluzioni di un problema di PL. Nello studio della risoluzione grafica di problemi di PL (cfr. paragrafo 5.3) si sarà notato che sembra essere vero che se un problema di PL ha più di una soluzione ottima, allora ne ammette infinite. Ci proponiamo qui di precisare questa affermazione.

Teorema 5.5.2 *Si consideri il problema di Programmazione Lineare*

$$\begin{cases} \min c^T x \\ Ax \geq b. \end{cases} \tag{PL}$$

Una e una sola delle seguenti tre affermazioni è vera:

1. *Il problema* (PL) *è inammissibile, ovvero il poliedro P è vuoto;*
2. *Il problema* (PL) *è illimitato inferiormente;*
3. *Il problema* (PL) *ammette soluzioni ottime.*

Nel caso in cui il problema ammetta soluzioni ottime e se P ammette almeno un vertice, allora almeno una soluzione ottima cade su un vertice.

Sia dato un poliedro $P \subseteq \mathbb{R}^n$ e un corrispondente problema di PL:

$$\begin{cases} \min \ c^T x \\ x \in P. \end{cases}$$

Supponiamo che questo problema abbia (almeno) una soluzione ottima x^*. Indichiamo con $z^* = c^T x^*$ il *valore ottimo*, cioè il valore assunto dalla funzione obiettivo all'ottimo. È evidente che se $\hat{x}^*$ è una qualunque altra soluzione ottima, risulta $z^* = c^T \hat{x}^*$. Vice versa, se un punto x è ammissibile, cioè se $x \in P$ e risulta $c^T x = z^*$, allora x è una soluzione ottima per definizione. Riassumendo possiamo affermare che l'insieme delle soluzioni ottime del problema di PL dato è

$$P \cap \{x \in \mathbb{R}^n \ : \ c^T x = z^*\}.$$

Questo mostra immediatamente che l'insieme delle soluzioni ottime di un problema di Programmazione Lineare è un poliedro contenuto in P, in quanto intersezione di P, definito da un insieme di equazioni e disequazioni lineari con l'iperpiano

$$\{x \in \mathbb{R}^n \ : \ c^T x = z^*\}.$$

Quindi vale il seguente teorema.

Teorema 5.5.3 *Sia dato un problema di PL*

$$\begin{cases} \min \ c^T x \\ x \in P. \end{cases}$$

L'insieme delle soluzioni ottime di questo problema è un poliedro contenuto in P.

6

Modelli di Programmazione Lineare Intera

Come è stato già osservato in precedenza, quando tutte le variabili di un problema di Programmazione Lineare sono vincolate ad assumere valori interi, si parla di Programmazione Lineare Intera. Moltissimi problemi reali possono essere rappresentati da modelli di Programmazione Lineare Intera; tipicamente si tratta di problemi in cui le variabili di decisione rappresentano quantità indivisibili (come il numero di automobili, di persone addette a certe mansioni, etc.) oppure sono problemi caratterizzati dalla necessità di scegliere tra un numero finito di alternative diverse. In quest'ultimo caso, in particolare, si avranno problemi di Programmazione Lineare 0–1, cioè problemi in cui le variabili sono binarie e assumono valore 0 oppure 1.

6.1 VARIABILI INTERE PER RAPPRESENTARE QUANTITÀ INDIVISIBILI

Un numero molto elevato di problemi reali è caratterizzato dalla indivisibilità del bene da produrre o della risorsa da utilizzare. Di qui la necessità di rappresentare tali problemi attraverso modelli di Programmazione Lineare con variabili intere. Questo tipo di problemi riguardano molte applicazioni reali: dai problemi in ambito industriale come la distribuzione dei beni e il sequenziamento delle attività produttive, ai problemi economici come la gestione ottima di un portafoglio titoli; dai problemi di progettazione ottima ai problemi inerenti la biologia e la fisica delle alte energie.
Esempi di modelli di Programmazione Lineare Intera caratterizzati da varibili di decisione associate a quantità indivisibili sono già stati presi in esame all'interno della trattazione dei modelli di Programmazione Lineare. Una situazione tipica

è data dall'Esempio 3.4.2 in cui il bene da produrre è rappresentato da autovetture che sono ovviamente indivisibili; quindi la formulazione di Programmazione Lineare già fornita per questo esempio in realtà, per essere aderente alla situazione reale, deve essere integrata con la presenza del vincolo di interezza sulle varibili che rappresentano i livelli di produzioni delle autovetture. Analogamente l'introduzione del vincolo di interezza sulle variabili è indispensabile quando viene meno una delle ipotesi fondamentali della Programmazione Lineare cioè la continuità delle variabili; in questo caso i modelli di Programmazione Lineare Intera sono uno strumento essenziale per rappresentare situazioni del mondo reale di questo tipo.

6.2 VARIABILI BINARIE PER RAPPRESENTARE SCELTE ALTERNATIVE

Si supponga di dover modellare il fatto che un certo evento possa verificarsi oppure no. La natura binaria del problema suggerisce l'idea di modellare questa dicotomia per mezzo di un variabile binaria $\delta \in \{0, 1\}$; si porrà $\delta = 1$ se l'evento si verifica e $\delta = 0$ altrimenti.

6.2.1 Problemi di assegnamento

Un generico problema di assegnamento consiste nel determinare il modo ottimale di assegnare lavori a persone o, più in generale, di assegnare *mezzi* (persone, macchine, etc.) ad *attività*.
Supponiamo che n persone $\mathbf{P_1}$,...,$\mathbf{P_n}$, debbano svolgere n lavori $\mathbf{L_1}$,..., $\mathbf{L_n}$; ciascun lavoro deve essere svolto esattamente da una persona e ciascuna persona deve svolgere esattamente un lavoro. Naturalmente le persone hanno diversi livelli di esperienza, competenza ed abilità e quindi si può introdurre un costo dell'assegnamento della persona i al lavoro j; indichiamo tale costo con c_{ij} e supponiamo che sia noto. Questo costo può, ad esempio, essere interpretato come tempo medio impiegato dalla persona i ad eseguire il lavoro j.
Il problema consiste, quindi, nell'assegnare i lavori alle persone minimizzando il costo totale di realizzazione di tutti i lavori.
Questo tipo di problemi sorge in molte situazioni pratiche: esempi tipici sono i problemi di assegnamento del personale all'interno di una azienda e i problemi di assegnare determinati mezzi di trasporto ad alcune particolari linee. Un esempio di problema di assegnamento è stato già considerato nell'Introduzione (pagina 6) quando si è brevemente analizzato il caso dell'assegnamento di 70 dipendenti a 70 mansioni diverse.

Esaminiamo, ora, alcuni esempi.

Esempi

Esempio 6.2.1 *Una compagnia finanziaria necessita di ricoprire tre lavori* **LAV1**, **LAV2**, **LAV3**, *che richiedono differenti abilità ed esperienza. Sono disponibili tre candidati* **C1**, **C2**, **C3**, *che possono essere assunti con il medesimo salario. A causa delle loro differenti capacità, il costo di assegnazione di ciascun candidato che la compagnia deve sostenere dipende dal tipo di lavoro al quale è assegnato. La stima di tale costo riferito a ciascun candidato se fosse assegnato a ciascuno dei tre lavori è riportato nella tabella seguente*

	LAV1	**LAV2**	**LAV3**
C1	*5*	*4*	*7*
C2	*6*	*7*	*3*
C3	*8*	*11*	*2*

Si desidera assegnare ogni candidato esattamente ad un lavoro in modo da minimizzare il costo complessivo che la compagnia deve sostenere.

Formulazione.
L'esempio in esame è di piccole dimensioni: infatti ci sono solamente $3! = 6$ possibili assegnazioni.

– *Variabili.* Per ogni lavoro e per ogni persona, introduciamo le variabili binarie

$$x_{ij} = \begin{cases} 1 & \text{se il candidato } i \text{ è assegnato al lavoro } j \\ 0 & \text{altrimenti.} \end{cases}$$

– *Funzione obiettivo.* La funzione obiettivo da minimizzare sarà

$$5x_{11} + 4x_{12} + 7x_{13} + 6x_{21} + 7x_{22} + 3x_{23} + 8x_{31} + 11x_{32} + 2x_{33}.$$

– *Vincoli.* Come già osservato nel caso generale, si devono considerare i seguenti vincoli

$$\sum_{i=1}^{3} x_{ij} = 1 \qquad j = 1, \ldots, 3.$$

$$\sum_{j=1}^{3} x_{ij} = 1 \qquad i = 1, \ldots, 3.$$

La formulazione completa si può scrivere

$$\begin{cases} \min(5x_{11}+4x_{12}+7x_{13}+6x_{21}+7x_{22}+3x_{23}+ \\ \quad +8x_{31}+11x_{32}+2x_{33}) \\ x_{11}+x_{21}+x_{31}=1 \\ x_{12}+x_{22}+x_{32}=1 \\ x_{13}+x_{23}+x_{33}=1 \\ x_{11}+x_{12}+x_{13}=1 \\ x_{21}+x_{22}+x_{23}=1 \\ x_{31}+x_{32}+x_{33}=1 \\ x_{ij}\in\{0,1\} \qquad i,j=1,\ldots,3. \end{cases}$$

Formulazione generale di un problema di assegnamento

Esaminiamo, ora, una formulazione in termini di programmazione lineare per un generico problema di assegnamento. Supponiamo che n persone $\mathbf{P_1},\ldots,\mathbf{P_n}$, debbano svolgere n lavori $\mathbf{L_1},\ldots,\mathbf{L_n}$ e che ciascun lavoro deve essere svolto esattamente da una persona e ciascuna persona deve svolgere esattamente un lavoro. Sia c_{ij} il costo dell'assegnamento della persona i al lavoro j; si devono assegnare i lavori alle persone minimizzando il costo totale di realizzazione di tutti i lavori.

Formulazione.

– *Variabili.* Per ogni lavoro i e per ogni persona j, $(i,j=1,\ldots,n)$ introduciamo le seguenti variabili binarie

$$x_{ij}=\begin{cases} 1 & \text{se la persona } i \text{ è assegnata al lavoro } j \\ 0 & \text{altrimenti.} \end{cases}$$

Si tratta di n^2 variabili:

	$\mathbf{L_1}$	$\cdots$	$\mathbf{L_j}$	$\cdots$	$\mathbf{L_n}$
$\mathbf{P_1}$	x_{11}	$\cdots$	x_{1j}	$\cdots$	x_{1n}
$\vdots$	$\vdots$		$\vdots$		$\vdots$
$\mathbf{P_i}$	x_{i1}	$\cdots$	x_{ij}	$\cdots$	x_{in}
$\vdots$	$\vdots$		$\vdots$		$\vdots$
$\mathbf{P_n}$	x_{n1}	$\cdots$	x_{nj}	$\cdots$	x_{nn}

– *Funzione obiettivo.* La funzione obiettivo da minimizzare sarà data dal costo totale cioè da

$$\sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}x_{ij}.$$

Naturalmente, se le c_{ij} anziché essere dei costi fossero i valori utili ricavati dall'assegnamento della persona i al lavoro j, allora la funzione obiettivo andrebbe massimizzata.

– *Vincoli.* (Vincoli di assegnamento.) Poiché esattamente una persona deve essere assegnata al lavoro j, allora si avranno i seguenti n vincoli

$$\sum_{i=1}^{n} x_{ij} = 1 \qquad j = 1, \ldots, n.$$

Inoltre, poiché ciascuna persona deve essere assegnata ad una sola attività, si avranno altri n vincoli

$$\sum_{j=1}^{n} x_{ij} = 1 \qquad i = 1, \ldots, n.$$

La formulazione completa sarà, quindi, data da

$$\begin{cases} \min \displaystyle\sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}x_{ij} \\ \displaystyle\sum_{i=1}^{n} x_{ij} = 1 \\ \displaystyle\sum_{j=1}^{n} x_{ij} = 1 \\ x_{ij} \in \{0,1\}, \qquad i,j = 1, \ldots, n. \end{cases}$$

Osservazione 6.2.2 Osservando la formulazione ottenuta si può facilmente dedurre che la struttura di un problema di assegnamento è del tutto simile a quella del problema dei trasporti (cfr. Paragrafo 3.4.3) in cui si ha lo stesso numero di origini e di destinazioni. La differenza sostanziale sta nel fatto che in un problema di assegnamento le variabili sono binarie, $x_{ij} \in \{0,1\}$ mentre in un problema dei trasporti le variabili sono reali non negative $x_{ij} \geq 0$. D'altra parte, per il Teorema 3.4.2, se in un problema dei trasporti i termini noti dei vincoli sono interi, se esiste soluzione ottima allora esiste soluzione ottima intera del problema dei trasporti. Quindi, poiché in un problema di assegnamento tali termini noti sono pari ad 1, i vincoli $x_{ij} \in \{0,1\}$, $i,j = 1, \ldots, n$ possono essere riscritti nella forma $0 \leq x_{ij} \leq 1$, $i,j = 1, \ldots, n$. Inoltre, poiché i vincoli $x_{ij} \leq 1$ sono implicati dai vincoli di assegnamento, si possono scrivere semplicemente i vincoli $x_{ij} \geq 0$, $i,j = 1, \ldots, n$ e comunque avere la garanzia che se esiste una soluzione ottima allora esiste una soluzione ottima intera 0–1. Quindi un problema di assegnamento

può essere considerato equivalente al problema

$$\begin{cases} \min \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij} \\ \sum_{i=1}^{n} x_{ij} = 1 \\ \sum_{j=1}^{n} x_{ij} = 1 \\ x_{ij} \geq 0, \qquad i,j = 1,\ldots,n \end{cases}$$

cioè può essere riscritto come un particolare problema di Programmazione Lineare avente la struttura medesima del problema dei trasporti.

È possibile effettuare una generalizzazione del problema dell'assegnamento per categorie di lavori. Infatti, frequentemente ci sono molti lavori identici che richiedono la stessa qualifica; tali lavori possono essere raggruppati in categorie di attività. Assumiamo quindi che esistano n categorie di attività e denotiamo con b_j il numero di lavori raggruppati nella $j-$esima categoria. Anche le persone possono essere raggruppate in categorie di persone aventi lo stesso valore; assumiamo che esistano m di queste categorie di persone e sia a_i il numero di persone poste nella $i-$esima categoria. Denotiamo con c_{ij} il valore utile ottenuto assegnando una persona della categoria i ad un lavoro della categoria j. Assumiamo che $\sum_{i=1}^{m} a_i = \sum_{j=1}^{n} b_j$. Introducendo le variabili x_{ij} rappresentanti il numero di persone della stessa categoria i assegnate ad un lavoro della categoria j, questo generale problema di assegnamento può essere formulato in termini di un problema di programmazione lineare nel seguente modo:

$$\begin{cases} \min \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij} x_{ij} \\ \sum_{j=1}^{n} x_{ij} = a_i \qquad i = 1,\ldots,m \\ \sum_{i=1}^{m} x_{ij} = b_j \qquad j = 1,\ldots,n \\ x_{ij} \geq 0 \quad i = 1,\ldots,m \quad j = 1,\ldots,n. \end{cases}$$

Esempio 6.2.3 *Una compagnia aerea cerca di pianificare le proprie linee aeree creando un aereoporto centrale e cercando di avere un elevato numero di voli in arrivo in questo aereoporto in una certa fascia oraria ed un elevato numero di partenze nella fascia oraria immediatamente successiva. Questo permette ai passeggeri di avere un elevato numero di combinazioni tra città di partenza e città di destinazione con una sola coincidenza e al più un cambio di aereo nell'aereoporto centrale. Il fine è quello di creare una tale struttura in modo da minimizzare i cambi di aerei e quindi il movimento di bagagli nell'aereoporto centrale. Supponiamo che la compagnia aerea abbia cinque voli che arrivano tra le 8 e le 8.30 nell'aereoporto centrale e che poi gli stessi aerei partono per altre diverse destinazioni tra le 8.40 e le 9.20. La tabella che segue riporta il numero medio di passeggeri che arrivano con uno dei voli in arrivo* **A1**, **A2**, **A3**, **A4**, **A5** *e che ripartono con i voli in partenza* **P1**, **P2**, **P3**, **P4**, **P5**, *ovvero passeggeri che non cambiano aereo*

	P1	**P2**	**P3**	**P4**	**P5**
A1	*15*	*20*	*8*	*16*	*12*
A2	*17*	*9*	*15*	*25*	*12*
A3	*12*	*32*	*16*	*9*	*20*
A4	-	*15*	*9*	*7*	*30*
A5	-	-	*35*	*10*	*18*

Il volo **A4** *arriva troppo tardi e non permette di prendere il volo in partenza* **P1**; *analogamente il volo* **A5** *non permette coincidenze con i voli in partenza* **P1** *e* **P2**. *Supponendo che tutti gli aerei sono identici, il problema consiste nell'assegnare ciascun aereo in arrivo ad uno dei voli in partenza in modo da minimizzare il numero delle persone che devono cambiare aereo.*

Formulazione.
Il problema in analisi può essere formulato come problema di assegnamento.

– *Variabili.* Introduciamo le variabili di decisione x_{ij} definite come segue

$$x_{ij} = \begin{cases} 1 & \text{se l'aereo del volo } \mathbf{Ai} \text{ è assegnato al volo } \mathbf{Pj} \\ 0 & \text{altrimenti.} \end{cases}$$

– *Funzione obiettivo.* Definiamo come funzione obiettivo il numero di passeggeri che non devono cambiare aereo:

$$\begin{aligned} & 15x_{11} + 20x_{12} + 8x_{13} + 16x_{14} + 12x_{15} + \\ + \; & 17x_{21} + 9x_{22} + 15x_{23} + 25x_{24} + 12x_{25} + \\ + \; & 12x_{31} + 32x_{32} + 16x_{33} + 9x_{34} + 20x_{35} + \\ + \; & 15x_{42} + 9x_{43} + 7x_{44} + 30x_{45} + \\ + \; & 35x_{53} + 10x_{54} + 18x_{55}. \end{aligned}$$

Tale funzione deve naturalmente essere massimizzata.

– *Vincoli.* I vincoli saranno

$$\begin{aligned}
x_{11}+x_{12}+x_{13}+x_{14}+x_{15} &= 1\\
x_{21}+x_{22}+x_{23}+x_{24}+x_{25} &= 1\\
x_{31}+x_{32}+x_{33}+x_{34}+x_{35} &= 1\\
x_{42}+x_{43}+x_{44}+x_{45} &= 1\\
x_{53}+x_{54}+x_{55} &= 1\\
x_{11}+x_{21}+x_{31} &= 1\\
x_{12}+x_{22}+x_{32}+x_{42} &= 1\\
x_{13}+x_{23}+x_{33}+x_{43}+x_{53} &= 1\\
x_{14}+x_{24}+x_{34}+x_{44}+x_{54} &= 1\\
x_{15}+x_{25}+x_{35}+x_{45}+x_{55} &= 1.
\end{aligned}$$

Quindi la formulazione completa sarà

$$\begin{cases}
\max\Big(15x_{21}+20x_{22}+8x_{23}+16x_{24}+12x_{25}+\\
\quad +17x_{11}+9x_{12}+15x_{13}+25x_{14}+12x_{15}+\\
\quad +12x_{31}+32x_{32}+16x_{33}+9x_{34}+20x_{35}+\\
\quad +15x_{42}+9x_{43}+7x_{44}+30x_{45}+\\
\quad +35x_{53}+10x_{54}+18x_{55}\Big)\\
x_{11}+x_{12}+x_{13}+x_{14}+x_{15}=1\\
x_{21}+x_{22}+x_{23}+x_{24}+x_{25}=1\\
x_{31}+x_{32}+x_{33}+x_{34}+x_{35}=1\\
x_{42}+x_{43}+x_{44}+x_{45}=1\\
x_{53}+x_{54}+x_{55}=1\\
x_{11}+x_{21}+x_{31}=1\\
x_{12}+x_{22}+x_{32}+x_{42}=1\\
x_{13}+x_{23}+x_{33}+x_{43}+x_{53}=1\\
x_{14}+x_{24}+x_{34}+x_{44}+x_{54}=1\\
x_{15}+x_{25}+x_{35}+x_{45}+x_{55}=1\\
x_{ij}\geq 0 \quad i,j=1,\dots,5.
\end{cases}$$

Riportiamo di seguito il file `aeroporto.mod` e `aeroporto.dat` che realizzano un'implementazione in AMPL del modello ora formulato.

`aeroporto.mod`

```
set ARRIVI;
set PARTENZE;

param passeggeri{ARRIVI,PARTENZE}>=0;
```

```
var x{ARRIVI,PARTENZE} binary;
#var x{ARRIVI,PARTENZE} >=0, <= 1;

maximize passeggeri_che_non_cambiano_aereo :
    sum{i in ARRIVI, j in PARTENZE} passeggeri[i,j]*x[i,j];

s.t. vincoliA{i in ARRIVI} : sum{j in PARTENZE} x[i,j] = 1;
s.t. vincoliP{j in PARTENZE}  : sum{i in ARRIVI} x[i,j] = 1;
```

aeroporto.dat

```
set ARRIVI := A1, A2, A3, A4, A5;
set PARTENZE := P1, P2, P3, P4, P5;

param passeggeri : P1   P2   P3   P4   P5 :=
                A1  15   20   8    16   12
                A2  17   9    15   25   12
                A3  12   32   16   9    20
                A4  0    15   9    7    30
                A5  0    0    35   10   18;

# fissare le 3 variabili a zero

fix x["A4","P1"]:=0;
fix x["A5","P1"]:=0;
fix x["A5","P2"]:=0;
```

Una soluzione ottima di questo problema è la seguente:

$$x_{11} = 1, x_{24} = 1, x_{32} = 1, x_{45} = 1, x_{53} = 1$$

e le altre variabili nulle; il valore ottimo della funzione obiettivo è 137.
Osservando questa soluzione si può notare come ciascun aereo in arrivo sia stato assegnato al volo in partenza che permette di mantenere sullo stesso aereo il maggior numero di passeggeri eccetto che per il volo in arrivo **A1**: infatti l'aereo in arrivo con il volo **A1** è stato assegnato al volo in partenza **P1** e quindi il numero dei passeggeri che non devono cambiare aereo è 15 contro ad esempio un numero di 20 o 16 passeggeri che sarebbero rimasti sullo stesso aereo se questo fosse stato assegnato rispettivamente al volo in partenza **P2** o **P4**.

6.2.2 Problemi di Knapsack binario

Il problema del "knapsack", nella sua versione originaria, può essere descritto come segue: dato un insieme di n oggetti di dimensioni diverse e differenti valori, si vuole determinare un sottoinsieme di questi oggetti da inserire in una "bisaccia" (knapsack) di capacità limitata in modo da massimizzare il valore trasportato.
In questo caso l'evento da modellare è l'inserimento dell'oggetto nella "bisaccia"; è quindi intuitivo introdurre una variabile binaria $x_i \in \{0,1\}$, $i = 1,\dots,n$ che assuma valore 1 se l'i-esimo oggetto è inserito nella "bisaccia", 0 se invece non è inserito.
Più in generale, supponiamo di avere n progetti e un budget disponibile per la loro realizzazione. Il problema consiste nello scegliere un sottoinsieme dei progetti in modo da massimizzare la somma dei valori senza superare il limite imposto dal budget nell'ipotesi che ciascun progetto scelto deve essere realizzato completamente e non è accettata una realizzazione parziale del progetto.

Esempio 6.2.4 *Si supponga di disporre di un capitale di 18 mila euro e di poterle investire in 4 progetti diversi. Nel primo progetto si debbono investire 8 euro per ricavarne 40, nel secondo si debbono investire 6 euro per ricavarne 24, nel terzo progetto si debbono investire 5 euro per ricavarne 15, infine nel quarto progetto si debbono investire 4 euro per ricavarne 8. Formulare il problema di PLI che consente di scegliere l'insieme di progetti che massimizza il profitto rispettando i vincoli di disponibilità di capitale.*

Formulazione.

– *Variabili.* Le variabili di decisione sono definite, per $i = 1,2,3,4$ come segue

$$x_i = \begin{cases} 1 & \text{se si sceglie il progetto } i \\ 0 & \text{altrimenti.} \end{cases}$$

– *Funzione obiettivo.* La funzione obiettivo da massimizzare è

$$40x_1 + 24x_2 + 15x_3 + 8x_4.$$

– *Vincoli.* I vincoli esprimono il fatto che il costo degli investimenti non può superare il budget disponibile, cioè

$$8x_1 + 6x_2 + 5x_3 + 4x_4 \leq 18.$$

Complessivamente il problema si scrive:

$$\begin{cases} \max 40x_1 + 24x_2 + 15x_3 + 8x_4 \\ 8x_1 + 6x_2 + 5x_3 + 4x_4 \leq 18 \\ x_i \in \{0,1\} \qquad i = 1,\dots,4 \end{cases}$$

□

Formulazione generale di un problema di knapsack binario.
In generale, supponiamo di avere n progetti tali che l'i-esimo progetto ha costo di realizzazione a_i ed un valore pari c_i, $i = 1, \ldots, n$; supponiamo inoltre che esista un budget b disponibile per la realizzazione dei progetti. Il problema consiste nello scegliere un sottoinsieme dei progetti in modo da massimizzare la somma dei valori senza superare il limite imposto dal budget.

L'evento da modellare, in questo caso, è la realizzazione del singolo progetto. Ciò può essere effettuato introducendo n variabili binarie nel seguente modo.

– *Variabili.* Introduciamo le variabili $x_i \in \{0, 1\}, \ \ i = 1, \ldots, n$, tali che

$$x_i = \begin{cases} 1 & \text{se l'}i\text{-esimo progetto è realizzato} \\ 0 & \text{se l'}i\text{-esimo progetto non è realizzato.} \end{cases}$$

– *Funzione obiettivo.* È data dal valore complessivo cioè da

$$\sum_{i=1}^{n} c_i x_i.$$

– *Vincoli.* Non si deve superare il budget disponibile e quindi si deve imporre

$$\sum_{i=1}^{n} a_i x_i \leq b.$$

La formulazione complessiva può essere quindi scritta

$$\begin{cases} \max \sum_{i=1}^{n} c_i x_i \\ \sum_{i=1}^{n} a_i x_i \leq b \\ x \in \{0, 1\}^n. \end{cases}$$

Tuttavia, in generale, questi problemi di scelta tra sottoprogetti possono avere più vincoli: si pensi al caso in cui il budget sia mensile e si voglia pianificare per più di un mese; in questo caso il problema è detto *knapsack multidimensionale.*

6.2.3 Problemi di "Capital Budgeting" (pianificazione degli investimenti)

I problemi di pianificazione degli investimenti rappresentano una delle problematiche di maggiore importanza all'interno delle problematiche finanziarie. Anche all'interno delle realtà aziendali, la politica degli investimenti è strettamente legata alla pianificazione finanziaria e ai processi di spesa. Di solito gli investimenti sono valutati attraverso il cosiddetto *indice di redditività* di ciascun investimento e una strategia di scelta degli investimenti dettata dal buon senso e

quella di ordinare gli investimenti in base a tali indici e scegliendo gli investimenti nell'ordine stabilito cercano di non violare il vincolo sul budget disponibile ed eventuali altri vincoli. Ovviamente una formulazione di un modello di Programmazione Lineare 0–1 che rappresenti il problema permette invece di ottenere una soluzione ottima del problema

Il modello di Programmazione Lineare Intera che descrive il problema della pianificazione degli investimenti viene denominato *modello di "Capital Budgeting"* ed è stato proposto alla fine degli anni '50 dagli economisti Manne e Markowitz; quest'ultimo fu poi insignito del premio Nobel per l'Economia.

In sintesi il problema della pianificazione degli investimenti può essere così descritto: siano dati n progetti di investimento da realizzare o meno. Si fissa un orizzonte temporale T entro il quale si vuole effettuare l'analisi (ad esempio, $T = 1$ anno). T si suddivide in t periodi $T = \{1, \ldots, t\}$ (ad esempio, un anno può essere diviso in quattro trimestri, ovvero $t = 4$). Ciascun progetto di investimento i-esimo, $i = 1, \ldots, n$ è caratterizzato da un vettore $a_i = (a_{i1}, \ldots, a_{it})^T$ del *flusso di cassa*, ovvero a_{ij} rappresenta il flusso di cassa (positivo o negativo) generato dall'i-esimo progetto nel periodo j-esimo. Si assume che il flusso di cassa positivo corrisponda ad una spesa, mentre un flusso di cassa nagativo corrisponda ad un guadagno. Quindi se, ad esempio, il flusso di cassa relativo ad un certo progetto su un orizzonte temporale di quattro periodi è pari a $(4,\ 3,\ -2,\ -7)$, allora la realizzazione del progetto richiede spese di 4 e 3 nei primi due periodi e poi fornisce un guadagno di 2 e 7 rispettivamente nel terzo e quarto periodo. Fra l'altro, questa struttura è tipica dei flussi di cassa. La spesa totale sarebbe $4+3-2-7 = -2$, ovvero un guadagno complessivo di 2. Ora, per ogni $i = 1, \ldots, n$ denotiamo con

$$c_i = -\sum_{j=1}^{t} a_{ij}$$

l'indice di redditività del progetto i-esimo, dove il segno meno di far corrispondere c_i al valore del guadagno. Inoltre, per ogni periodo $j = 1, \ldots, t$ c'è un budget limitato denotato con b_j. Il problema consiste nel determinare un sottoinsieme di progetti da realizzare in modo da avere guadagno massimo. Si assume inoltre che i progetti non sono frazionabili, cioè non possono essere realizzati parzialmente.

Formulazione.

– *Variabili.* Introduciamo le variabili $x_i \in \{0, 1\}$ $j = 1, \ldots, n$, così definite:

$$x_i = \begin{cases} 1 & \text{se l'}i\text{-esimo progetto è realizzato} \\ 0 & \text{se il } i\text{-esimo progetto non è realizzato.} \end{cases}$$

– *Funzione obiettivo.* È data dal valore complessivo cioè da

$$\sum_{i=1}^{n} c_i x_i.$$

– *Vincoli.* Per ogni periodo $j \in T$ non si deve superare il budget b_j disponibile e quindi per ogni $j = 1, \ldots, t$ si deve imporre

$$\sum_{i=1}^{n} a_{ij} x_i \leq b_j.$$

Si ha quindi un vincolo di "knapsack" per ogni periodo $j = 1, \ldots, t$.

La formulazione complessiva può essere quindi scritta

$$\begin{cases} \max \sum_{j=i}^{n} c_i x_i & \\ \sum_{i=1}^{n} a_{ij} x_i \leq b_j & j = 1, \ldots, t \\ x \in \{0,1\}^n. & \end{cases}$$

Osservazione 6.2.5 Se fossero possibili realizzazioni parziali di alcuni progetti, le variabili corrispondenti non sarebbero vincolate ad appartenere all'insieme $\{0, 1\}$, ma sarebbero appartenenti all'intervallo $[0, 1]$ e rappresenterebbero il livello di realizzazione del progetto; in questo caso si avrebbe un problema di Programmazione Lineare Mista

Osservazione 6.2.6 Altri vincoli che realizzano delle condizioni logiche sono facilmente introducibili nella formulazione. Ad esempio:

- la realizzazione di un particolare progetto (ad esempio il 5^o)

$$x_5 = 1;$$

- la realizzazione di *esattamente* uno dei progetti 1^o, 3^o e 11^o:

$$x_1 + x_3 + x_{11} = 1;$$

- la realizzazzione di *almeno* due dei progetti 1^o, 3^o e 11^o:

$$x_1 + x_3 + x_{11} \geq 2;$$

- la realizzazzione di *al piú* due dei progetti 1^o, 3^o e 11^o:

$$x_1 + x_3 + x_{11} \leq 2.$$

6.3 VARIABILI BINARIE COME VARIABILI INDICATRICI

Un altro classico uso di variabili $0-1$, consiste nell'indicare le relazioni di dipendenza tra alcune grandezze di un problema; cioè, in questo caso, le variabili binarie vengono utilizzate come *variabili indicatrici*.
Supponiamo che la variabile $x_i \geq 0$ rappresenti una grandezza del problema e di conoscere un limite superiore di tale variabile, cioè un valore M positivo maggiore del più grande valore che può assumere la x_i. Allora, può essere necessario imporre la condizione:

$$x_i > 0 \Rightarrow \delta = 1 \tag{6.3.1}$$

oppure la condizione equivalente $\delta = 0 \Rightarrow x_i = 0$ (si ricordi che era assunto che $x_i \geq 0$). L'implicazione (6.3.1) può essere modellata con il vincolo

$$x_i - M\delta \leq 0.$$

Tuttavia, in altri casi, può essere necessario imporre la condizione

$$\delta = 1 \Rightarrow x_i > 0 \tag{6.3.2}$$

(che è equivalente a $x_i = 0 \Rightarrow \delta = 0$, poiché, per ipotesi, $x_i \geq 0$). La condizione logica (6.3.2) non si può facilmente rappresentare con un vincolo. Supponiamo, ad esempio, che in un problema di miscelazione una variabile x_i rappresenti la quantità di un ingrediente da includere nella miscela e quindi si ha $x_i \geq 0$; si può usare una variabile indicatrice $\delta \in \{0, 1\}$ per distinguere tra il caso in cui $x_i = 0$ e $x_i > 0$. La condizione logica (6.3.2) afferma che se $\delta = 1$ allora l'ingrediente rappresentato da x deve apparire nella miscela, ma non fornisce nessuna indicazione sulla quantità dell'ingrediente. In realtà, è più verosimile imporre una condizione logica del tipo

$$\delta = 1 \Rightarrow x_i \geq \varepsilon > 0 \tag{6.3.3}$$

cioè se $\delta = 1$ allora la variabile x_i assume un valore almeno pari ad ε.
La (6.3.3) è rappresentabile dal vincolo

$$x - \varepsilon\delta \geq 0. \tag{6.3.4}$$

Riepilogando possiamo considerare il seguente schema: se x_i è una variabile non negativa e $\delta \in \{0, 1\}$ ed inoltre $x_i < M$ e $\varepsilon > 0$, allora

$$x_i - M\delta \leq 0 \quad \Leftrightarrow \quad \begin{cases} x_i > 0 \Rightarrow \delta = 1 \\ \delta = 0 \Rightarrow x_i = 0 \end{cases}$$

$$x_i - \varepsilon\delta \geq 0 \quad \Leftrightarrow \quad \begin{cases} \delta = 1 \Rightarrow x_i \geq \varepsilon \\ x_i = 0 \Rightarrow \delta = 0. \end{cases}$$

Analizziamo, ora, un esempio di miscelazione in cui applichiamo quanto appena esposto.

Esempio 6.3.1 *Sia data la seguente tavola di valori nutrizionali che riporta il tipo di alimento, il costo unitario, le unità di sostanze (proteine, carboidrati, grassi, vitamine, calcio) per unità di alimento*

	costo	**prot.**	**carb.**	**grassi**	**vitam.**	**calcio**
1	*0.15*	*0*	*7*	*1*	*1*	*0*
2	*0.23*	*1*	*0*	*3*	*1*	*4*
3	*0.79*	*5*	*0*	*4*	*0*	*1*
4	*0.47*	*2*	*2*	*1*	*3*	*0*
5	*0.52*	*0*	*3*	*0*	*2*	*1*

Formulare un problema di PLI che permetta di trovare una dieta di costo minimo sapendo che si devono assumere almeno 3 unità di proteine, 10 unità di carboidrati, 2 unità di grasso, 3 unità di vitamine e 2 unità di calcio e sapendo che se è presente l'alimento 1 la dieta non può contenere l'alimento 5.

Formulazione.

È un classico problema di miscelazione; le quantità di alimenti presenti nella dieta si suppongono frazionabili. A causa della presenza di una condizione logica, è necessario utilizzare, in aggiunta alle variabili del problema, una variabile $0-1$ per modellarla cioè per esprimere con un vincolo il legame tra la presenza nella dieta dell'alimento 1 e dell'alimento 5.

– *Variabili di decisione.* Introduciamo come variabili del problema le unità di alimenti presenti nella dieta, x_i con $i = 1, \ldots, 5$. Inoltre, introduciamo la variabile booleana $\delta \in \{0, 1\}$.

– *Vincoli.* Si hanno i seguenti vincoli:

- Vincoli di qualità: la dieta deve contenere alcuni valori minimi di sostanze nutrizionali; dalla tabella si ottiene che deve essere

$$x_2 + 5x_3 + 2x_4 \geq 3$$

$$7x_1 + 2x_4 + 3x_5 \geq 10$$

$$x_1 + 3x_2 + 4x_3 + x_4 \geq 2$$

$$x_1 + x_2 + 3x_4 + 2x_5 \geq 3$$

$$4x_2 + x_3 + x_5 \geq 2$$

- Vincolo logico: se nella dieta è presente l'alimento 1 allora non deve esserci l'alimento 5. Vogliamo quindi definire dei vincoli che consentano di esprimere le seguenti condizioni logiche

$$\begin{aligned} x_1 > 0 \quad &\Rightarrow \quad \delta = 1 \\ \delta = 1 \quad &\Rightarrow \quad x_5 = 0 \end{aligned}$$

  Secondo quanto descritto, ciò può essere modellato introducendo i vincoli

$$x_1 - M\delta \leq 0$$

$$x_5 - M(1-\delta) \leq 0$$

  dove M è un numero positivo maggiore del più grande valore che possono assumere le variabili.

- Vincoli di non negatività: Si tratta di quantità di alimenti, e quindi deve essere

$$x_i \geq 0 \qquad i = 1, \ldots, 5.$$

– *Funzione obiettivo.* È il costo da minimizzare ed è data da

$$0.15x_1 + 0.23x_2 + 0.79x_3 + 0.47x_4 + 0.52x_5.$$

Complessivamente la formulazione di PLI per questo problema può essere scritta

$$\begin{cases} \min\left(0.15x_1 + 0.23x_2 + 0.79x_3 + 0.47x_4 + 0.52x_5\right) \\ x_2 + 5x_3 + 2x_4 \geq 3 \\ 7x_1 + 2x_4 + 3x_5 \geq 10 \\ x_1 + 3x_2 + 4x_3 + x_4 \geq 2 \\ x_1 + x_2 + 3x_4 + 2x_5 \geq 3 \\ 4x_2 + x_3 + x_5 \geq 2 \\ x_1 - M\delta \leq 0 \\ x_5 - M(1-\delta) \leq 0 \\ x_i \geq 0 \quad i = 1, \ldots, 5 \\ \delta \in \{0, 1\}. \end{cases}$$

□

Si riportano di seguito i file `dieta.mod` e `dieta.dat` che realizzano un implementazione AMPL di questo problema.

dieta.mod

```
set ALIMENTI;
set SOSTANZE;

param contenuti{ALIMENTI,SOSTANZE}>=0;
param costi{ALIMENTI}>=0;
param contenuti_min{SOSTANZE};
param BigM>0;

var x{ALIMENTI}>=0;
var d binary;

minimize costo : sum{i in ALIMENTI} costi[i]*x[i];

s.t. vincoli_qualita{j in SOSTANZE} :
   sum{i in ALIMENTI} contenuti[i,j]*x[i] >= contenuti_min[j];

s.t. vincolo_logico1 : x["A1"] - BigM * d <=0;
s.t. vincolo_logico2 : x["A4"] - (1-d)*BigM <=0;
```

dieta.dat

```
set ALIMENTI:= A1 A2 A3 A4 A5;
set SOSTANZE:= prot carb grassi vitam calcio;

param BigM := 10000;

param costi :=
A1  0.15
A2  0.23
A3  0.79
A4  0.47
A5  0.52;

param contenuti_min :=
prot 3
carb 10
grassi 2
vitam 3
calcio 2;
```

```
param contenuti : prot carb grassi vitam calcio :=
A1  0 7 1 1 0
A2  1 0 3 1 4
A3  5 0 4 0 1
A4  2 2 1 3 0
A5  0 3 0 2 1;
```

6.3.1 Problema del costo fisso

Esaminiamo un altro esempio di applicazione di variabili indicatrici: *il problema del costo fisso*. Nei modelli di PL la funzione obiettivo è una funzione lineare nelle variabili di decisione che, di solito, rappresentano livelli di attività. Questa ipotesi, in molti problemi pratici, non è verosimile: può infatti accadere che il costo di un'attività abbia un costo iniziale (set-up), ad esempio l'acquisto di un macchinario, che esiste solo se quell'attività è svolta a livello non nullo.
In riferimento ad un'applicazione industriale, indichiamo con c il costo della manifattura per unità di prodotto, con $f \geq 0$ il costo di set-up (costo fisso) e con $x \geq 0$ la quantità di prodotto da fabbricare.
Quindi se $x = 0$ il costo totale è ovviamente nullo; se $x > 0$ allora il costo totale è dato da $cx + f$. Quindi la funzione obiettivo è data dall'espressione

$$f(x) = \begin{cases} cx + f & \text{se} \quad x > 0 \\ 0 & \text{se} \quad x = 0. \end{cases}$$

Tale funzione ha una discontinuità nell'origine e quindi non è lineare.
Per formulare questo problema in termini di programmazione lineare, introduciamo una variabile indicatrice $\delta \in \{0, 1\}$ tale che, se il prodotto rappresentato dalla x è fabbricato in una qualsiasi quantità allora $\delta = 1$; se il prodotto non è fabbricato allora $\delta = 0$. Dobbiamo, quindi modellare con un vincolo le condizioni logiche

$$x > 0 \Rightarrow \delta = 1 \tag{6.3.5}$$

$$x = 0 \Rightarrow \delta = 0. \tag{6.3.6}$$

L'implicazione (6.3.5) si realizza introducendo il vincolo

$$x - M\delta \leq 0$$

dove M è un numero positivo maggiore del più grande valore che può assumere la x. Per realizzare l'implicazione (6.3.6), si dovrebbe introdurre un vincolo del

tipo $x - \varepsilon\delta \geq 0$ con $\varepsilon > 0$; in realtà, ciò non è necessario perché, come vedremo, la condizione (6.3.6) discende direttamente dal fatto che ci troviamo in un problema di minimizzazione. Infatti, il problema può essere formulato come

$$\min (cx + f\delta)$$

con vincolo aggiuntivo

$$x - M\delta \leq 0$$

con $x \geq 0$ e $\delta \in \{0, 1\}$.
Dalla struttura della funzione discende immediatamente che se $x = 0$ allora, poiché si tratta di un problema di minimo, all'ottimo deve essere $\delta = 0$, essendo $f \geq 0$. Quindi non è necessario introdurre nella formulazione la condizione logica (6.3.6).

Si può facilmente generalizzare il problema del costo fisso al caso di n attività. Supponiamo che $x_i,\ \ i = 1, \ldots, n$ rappresenti il livello al quale viene svolta ciascuna attività. Supponiamo che il costo della i-esima attività sia dato da

$$\begin{cases} c_i x_i + f_i & \text{se} \quad x_i > 0 \\ 0 & \text{se} \quad x_i = 0 \end{cases} \qquad i = 1, \ldots, n$$

dove $f_i \geq 0$ è il costo fisso dell'attività i-esima e deve essere pagato solo se l'attività i viene svolta ad un livello non nullo.
Il corrispondente problema di ottimizzazione è:

$$\min\ z(x) = \sum_{i=1}^{n} c_i x_i + \sum_{i \in I(x)} f_i$$

dove $I(x) = \{i \in \{1, \ldots, n\} : x_i > 0\}$ e quindi è una funzione discontinua nell'origine, non lineare. Per formularlo come problema di PLI, si introduce per ogni $i = 1, \ldots, n$ una variabile $\delta_i \in \{0, 1\}$ tale che

$$\delta_i = \begin{cases} 1 & \text{se l'attività } i \text{ è svolta a livello non nullo} \\ 0 & \text{se l'}i\text{-esima attività non è svolta.} \end{cases}$$

Si vuole quindi che siano verificate le seguenti condizioni logiche

$$x_i > 0 \quad \Rightarrow \quad \delta_i = 1, \qquad x_i = 0 \quad \Rightarrow \quad \delta_i = 0.$$

Analogamente al caso precedente, il problema può essere formulato

$$\min \left(\sum_{i=1}^{n} c_i x_i + \sum_{i=1}^{n} \delta_i f_i \right)$$

con vincoli aggiuntivi

$$x_i - M\delta_i \leq 0 \qquad i = 1, \ldots, n$$

e con

$$x_i \geq 0, \quad \delta_i \in \{0,1\} \quad i = 1, \ldots, n.$$

È chiaro che se $x_i = 0$, allora all'ottimo $\delta_i = 0$ perché $f_i \geq 0$ e quindi la condizione logica $x_i = 0 \Rightarrow \delta_i = 0$ è automaticamente verificata. Inoltre, se $x_i > 0$ allora $\delta_i = 1$ e quindi il suo costo fisso si aggiungerà al valore della funzione costo nella funzione obiettivo. È quindi evidente che una soluzione ottima del problema iniziale è anche ottima per il problema trasformato.

Esempio 6.3.2 *In una centrale elettrica sono a disposizione tre generatori e ogni giorno si deve decidere quali usare di giorno e quali di notte per assicurare una produzione di almeno 4000 megawatts di giorno e di almeno 2800 megawatts di notte. L'uso di un generatore comporta la presenza di personale tecnico che sorvegli il suo funzionamento; tale personale viene retribuito in maniera diversa tra il giorno e la notte e a seconda del tipo di generatore; tali costi di attivazione sono riportati nella tabella che segue (in euro) insieme al costo (in euro) per ogni megawatt prodotta e alla massima capacità di produzione in megawatts per ogni singolo periodo (giorno/notte).*

	Costo attivazione		**Costo per**	**Capacità**
	giorno	**notte**	**megawatt**	**max**
Generatore A	*750*	*1000*	*3*	*2000*
Generatore B	*600*	*900*	*5*	*1700*
Generatore C	*800*	*1100*	*6*	*2500*

Formulare un modello di PLI che permetta di rappresentare il problema in analisi.

Formulazione.
È un problema di costo fisso e può essere formulato in termini di Programmazione Lineare Intera come appena descritto in generale. Per brevità di notazione, chiameremo 1^o periodo il giorno e 2^o periodo la notte.

– *Variabili.* Indichiamo con x_{A_i}, x_{B_i} e x_{C_i}, $i = 1, 2$, i megawatts generati rispettivamente dai generatori A, B e C nel periodo i. Inoltre, per ottenere una formulazione lineare, è necessario introdurre sei variabili $0-1$, δ_{A_i}, δ_{B_i} e δ_{C_i}, $i = 1, 2$, definite come segue :

$$\delta_{A_i} = \begin{cases} 1 & \text{se il generatore A è attivato nell'}i\text{-esimo periodo} \\ 0 & \text{se nell'}i\text{-esimo periodo il generatore A non è attivato} \end{cases} \qquad i = 1, 2.$$

Analoga è la definizione per le altre variabili δ_{B_i} e δ_{C_i}, $i = 1, 2$.

– *Funzione obiettivo.* La funzione obiettivo da minimizzare può esser scritta

$$3x_{A_1} + 3x_{A_2} + 5x_{B_1} + 5x_{B_2} + 6x_{C_1} + 6x_{C_2} + 750\delta_{A_1} +$$

$$+1000\delta_{A_2} + 600\delta_{B_1} + 900\delta_{B_2} + 800\delta_{C_1} + 1100\delta_{C_2}.$$

– *Vincoli.* Si devono considerare i vincoli sulla richiesta cioè

$$x_{A_1} + x_{B_1} + x_{C_1} \geq 4000$$

$$x_{A_2} + x_{B_2} + x_{C_2} \geq 2800.$$

Inoltre, per quanto esposto nel caso generale si devono considerare i vincoli

$$x_{A_i} - 2000\delta_{A_i} \leq 0 \qquad i = 1, 2$$

$$x_{B_i} - 1700\delta_{B_i} \leq 0 \qquad i = 1, 2$$

$$x_{C_i} - 2500\delta_{C_i} \leq 0 \qquad i = 1, 2.$$

Quindi la formulazione complessiva può essere scritta

$$\begin{cases} \min\Big(3x_{A_1} + 3x_{A_2} + 5x_{B_1} + 5x_{B_2} + 6x_{C_1} + 6x_{C_2} + \\ \qquad +750\delta_{A_1} + 1000\delta_{A_2} + 600\delta_{B_1} + 900\delta_{B_2} + 800\delta_{C_1} + 1100\delta_{C_2}\Big) \\ x_{A_1} + x_{B_1} + x_{C_1} \geq 4000 \\ x_{A_2} + x_{B_2} + x_{C_2} \geq 2800 \\ x_{A_1} - 2000\delta_{A_1} \leq 0 \\ x_{B_1} - 1700\delta_{B_1} \leq 0 \\ x_{C_1} - 2500\delta_{C_1} \leq 0 \\ x_{A_2} - 2000\delta_{A_2} \leq 0 \\ x_{B_2} - 1700\delta_{B_2} \leq 0 \\ x_{C_2} - 2500\delta_{C_2} \leq 0 \\ x_{A_i} \geq 0, x_{B_i} \geq 0, x_{C_i} \geq 0, \qquad i = 1, 2 \\ \delta_{A_i} \in \{0, 1\}, \delta_{B_i} \in \{0, 1\}, \delta_{C_i} \in \{0, 1\} \qquad i = 1, 2. \end{cases}$$

□

Si riportano di seguito i file `centrale.mod` e `centrale.dat` che rapprentano una implementazione AMPL del problema in esame.

`centrale.mod`

```
set GENERATORI;
set PERIODI;

param costo_fisso{GENERATORI,PERIODI}>=0;
param costo_unitario{GENERATORI}>=0;
param capacita_max{GENERATORI};
param produzione_min{PERIODI};
param BigM = 10^6;
```

```
var x{GENERATORI,PERIODI} >=0;
var d{GENERATORI,PERIODI} binary;

minimize costo_totale : sum{i in GENERATORI, j in PERIODI}
        (costo_unitario[i]*x[i,j]+costo_fisso[i,j]*d[i,j]);

s.t. produzione_minima{j in PERIODI} : sum{i in GENERATORI}
           x[i,j]>= produzione_min[j];

s.t. vincoli_logici{i in GENERATORI, j in PERIODI} :
     x[i,j] - capacita_max[i]*d[i,j] <=0;
```

centrale.dat

```
set GENERATORI := A, B, C;
set PERIODI := giorno notte;

param : costo_unitario    capacita_max :=
  A       3                   2000
  B       5                   1700
  C       6                   2500;

param produzione_min :=
   giorno     4000
   notte      2800;

param costo_fisso : giorno    notte :=
           A          750      1000
           B          600       900
           C          800      1100;
```

6.3.2 Problemi di "lot sizing" (gestione della scorte)

I modelli multiperiodo esaminati nel paragrafo 3.4.1 rientrano in una classe di modelli per la programmazione della produzione che va sotto il nome di *Modelli per la gestione della scorte ("lot sizing")* che anche da un punto di vista storico costituiscono un argomento centrale della Ricerca Operativa

Attualmente negli USA alcune indagini hanno evidenziato che il 50% delle aziende americane di produzione utilizzano strumenti matematici per la gestione ottima delle scorte. C'è la necessita di integrare la fase produttiva con quella della gestione delle scorte. L'utilizzazione di scorte nei processi produzione ha numerosi vantaggi:

- *economia di scala* che si possono conseguire aumentando i volumi produttivi minimizzando l'incidenza dei costi fissi;
- *flessibiltà della produzione*: si riesce a far fronte con le scorte all'eventuale andata fuori servizio di qualche linea di produzione;
- *equipartizione dei carichi di lavori* sull'intero orizzonte produttivo.

Un problema di "lot sizing" si può formalizzare nel seguente modo: si tratta di pianificare la fabbricazione di un bene in assegnato un orizzonte temporale costituito da un insieme finito di periodi di controllo $T = \{1, \dots, t\}$. Per ogni periodo $i \in \{1, \dots, t\}$ è nota la richiesta di questo bene (che deve essere soddisfatta esattamente) che indichiamo con d_i. Sono noti i costi unitari c_i, $i = 1, \dots, t$ di produzione del bene in ciascun periodo ed inoltre in ogni periodo, ad eccezione dell'ultimo, è possibile immagazzinare quantità di questo bene che andrà a fare parte della quantità di bene disponibile nel periodo successivo. Anche il costo di stockaggio unitario è assegnato ed è pari a b_i. La novità rispetto ai modelli multiperiodo consiste nella presenza di costi di setup corrispondenti all'avviamento della produzione in ciascun periodo; si tratta di costi fissi che non dipendono dalle quantità prodotte e vengono sostenuti solamente se si produce qualcosa nel periodo; indichiamo con f_i questi costi fissi.
Il problema consiste nel determinare le quantità di bene da produrre in ciascun periodo e le quantià da immagazzinare in modo da soddisfare le richieste minizzando il costo complessivo dato dalla somma dei costi di produzione e di stockaggio tenendo conto che all'inzio del primo periodo non c'è nessuna scorta disponibile e che nell'ultimo periodo non si può effettuare alcuno stockaggio.

Formulazione.

– *Variabili.* Indichiamo con x_i, $i = 1, \dots, t$ il livello di produzione nel periodo i-esimo, cioè le quantità del bene da produrre in quel periodo. Indichiamo inoltre con s_i, $i = 1, \dots, t-1$ le quantità di bene che vengono immagazzinate nel periodo i. Inoltre, per $i = 1, \dots, t$ introduciamo le seguenti variabili $0-1$:

$$\delta_i = \begin{cases} 1 & \text{se nell'}i\text{−esimo periodo c'è produzione} \\ 0 & \text{altrimenti;} \end{cases}$$

Il problema può essere efficacemente rappresentato come in Figura 6.3.1

Fig. 6.3.1 Un problema di "Lot sizing"

– *Funzione obiettivo.* La funzione obiettivo sarà data dalla somme dei costi di produzione e dei costi di stockaggio e quindi può essere scritta nella forma

$$\sum_{i=1}^{t} c_i x_i + \sum_{i=1}^{t-1} b_i s_i + \sum_{i=1}^{t} f_i \delta_t$$

– *Vincoli.* I vincoli del problema sono i seguenti già esaminati nel caso di modelli multiperiodo:

$$\begin{aligned} x_1 &= d_1 + s_1 \\ s_{i-1} + x_i &= d_i + s_i, \qquad i = 2, \ldots, t-1 \\ s_{t-1} + x_t &= d_t \\ x_1 \geq 0, x_2 \geq 0, &\ \ldots\ , x_t \geq 0, \\ s_1 \geq 0, s_2 \geq 0, &\ \ldots\ , s_{t-1} \geq 0 \end{aligned}$$

Inoltre si devono considerare i vincoli relativi alla presenza dei costi fissi, ovvero i vincoli

$$x_i - M\delta_i \leq 0 \qquad i = 1, \ldots, t$$

dove M, ad esempio, può essere scelta pari a $\sum_{i=1}^{t} d_i$, cioè pari a quanto viene richiesto durante l'intero orizzonte temporale.

Quindi la formulazione complessiva di un problema di "lot sizing" si può scrivere come

$$\begin{cases} \min\left(\sum_{t=1}^{t} c_i x_i + \sum_{i=1}^{i-1} b_i s_i + \sum_{i=1}^{t} f_i \delta_i\right) \\ x_1 - s_1 = d_1 \\ s_{i-1} + x_i - s_i = d_i \qquad i = 2, \ldots, t-1 \\ y_{t-1} + x_t = d_t, \\ \\ x_i - M\delta_i \leq 0 \qquad i = 1, \ldots, t \\ \\ x_1 \geq 0, \ldots, \quad x_t \geq 0, \\ s_1 \geq 0, \ldots, \quad s_{t-1} \geq 0 \\ \delta_1 \in \{0,1\}, \quad \ldots, \quad \delta_i \in \{0,1\}. \end{cases}$$

□

6.3.3 Problemi di localizzazione di impianti

Si tratta di problemi che nascono nell'ambito della pianificazione industriale che possono essere schematizzati nel seguente modo: sono date n aree $\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_n$, distribuite in un territorio. In ciascuna di esse è possibile costruire una fabbrica che produce merce. Per ciascuna area $\mathbf{A_i}$, $i = 1, \ldots n$ è nota la massima capacità produttiva p_i, $i = 1, \ldots, n$ che una fabbrica avrebbe se fosse localizzata in $\mathbf{A_i}$. Sia inoltre f_i il costo fisso di costruzione della fabbrica nell'area $\mathbf{A_i}$. Sono inoltre dati m siti $\mathbf{C}_1, \mathbf{C}_2, \ldots, \mathbf{C}_m$, ove risiedono clienti ai quali deve essere trasportate la merce prodotta. Per ciascun sito $\mathbf{C}_j$ è assegnato un quantitativo r_j, $j = 1, \ldots, m$, di una data merce richiesta presso il sito $\mathbf{C}_j$. Tale richiesta deve essere soddisfatta esattamente. Per soddisfare questa richiesta possono essere costruite $q \leq n$ fabbriche che producono la merce. Esistono altri costi fissi dovuti alla eventuale costruzione di una strada dall'area $\mathbf{A}_i$ al sito $\mathbf{C}_j$, per ogni $i = 1, \ldots, n$ e $j = 1, \ldots m$; indicheremo questi costi fissi con f_{ij}. Siano inoltre c_{ij} il costo necessario per trasportare una unità di merce dalla fabbrica costruita nell'area $\mathbf{A}_i$ al sito $\mathbf{C}_j$ e M_{ij} il quantitativo massimo di merce trasportabile. Il problema consiste nel determinare quante fabbriche e su quali aree costruirle, insieme a quali strade di collegamento costruire, in modo da soddisfare le richieste di i siti minimizzando i costi di costruzione delle fabbriche, delle strade di collegamento e il costo del trasporto della merce una volta che le costruzioni delle fabbriche sono state ultimate determinando al tempo stesso il piano per il trasporto della merce per soddisfare tutte le richieste.
Questo problema può essere formulato come problema di Programmazione Lineare Intera nel seguente modo: si introducono le seguenti variabili

$$\delta_i = \begin{cases} 1 & \text{se una fabbrica è costruita sull'area } \mathbf{A}_i \\ 0 & \text{altrimenti;} \end{cases}$$

$$y_{ij} = \begin{cases} 1 & \text{se una strada è costruita da } \mathbf{A}_i \text{ a } \mathbf{C}_j \\ 0 & \text{altrimenti.} \end{cases}$$

Si introducono inoltre le variabili x_{ij} che rappresentano la quantità di merce trasportata dalla fabbrica costruita nell'area $\mathbf{A}_i$ al sito $\mathbf{C}_j$.
I vincoli sono innanzitutto i vincoli di richiesta

$$\sum_{i=1}^{n} x_{ij} = r_j \quad \text{per ogni} \quad j = 1, \ldots, m.$$

Inoltre per ogni $i = 1, \ldots, m$, si vuole che se $\sum_{j=1}^{m} x_{ij} > 0$ allora $\delta_i = 1$. Questa implicazione si realizza con i vincoli

$$\sum_{j=1}^{m} x_{ij} - p_i \delta_i \leq 0 \qquad i = 1, \ldots, n.$$

Ragionando analogamente si ottengono i vincoli

$$x_{ij} - M_{ij} y_{ij} \leq 0 \qquad i = 1, \ldots n, \quad j = 1, \ldots m.$$

Infine dovrà essere

$$\sum_{i=1}^{n} \delta_i \leq q.$$

Si devono poi esplicitare i vincoli $x_{ij} \geq 0$ e $\delta_i \in \{0, 1\}$, $y_{ij} \in \{0, 1\}$, $i = 1, \ldots n$, $j = 1, \ldots m$.
La funzione obiettivo si può quindi scrivere

$$\sum_{i=1}^{n} \sum_{j=1}^{m} c_{ij} x_{ij} + \sum_{i=1}^{n} f_i \delta_i + \sum_{i=1}^{n} \sum_{j=1}^{m} f_{ij} y_{ij}.$$

Esaminiamo, ora, un esempio molto semplice di problema di localizzazione di impianti.

Esempio 6.3.3 *Una compagnia di distribuzione deve rifornire i suoi clienti* $\mathbf{C_1}$, $\mathbf{C_2}$, $\mathbf{C_3}$, $\mathbf{C_4}$ *e* $\mathbf{C_5}$ *che sono dislocati in località diverse di una regione. Per ottimizzare il rifornimento la compagnia vuole costruire un numero di depositi non superiore a due disponendo di tre possibili zone dove costruirli. A seconda della zona in cui vengono costruiti, i tre possibili depositi hanno un costo di costruzione e una capacità massima diversi. La tabella che segue riporta questi costi in migliaia di euro e queste capacità in tonnellate.*

	Costo costruzione	**Capacità massima**
Deposito 1	*10000*	*180*
Deposito 2	*15000*	*230*
Deposito 3	*13000*	*500*

Il quantitativo di merce (in tonnellate) richiesto da ciascun cliente è riportato nella tabella che segue insieme ai costi (in migliaia di euro) del trasporto di una unità di merce da ciascuno dei possibili depositi a ciascun cliente.

	$\mathbf{C_1}$	$\mathbf{C_2}$	$\mathbf{C_3}$	$\mathbf{C_4}$	$\mathbf{C_5}$
Richiesta	*91*	*170*	*135*	*153*	*110*
Deposito 1	*15*	*13*	*27*	*9*	*7*
Deposito 2	*12*	*21*	*34*	*21*	*3*
Deposito 3	*7*	*10*	*2*	*17*	*12*

Costruire un modello lineare che rappresenti il problema in analisi per soddisfare esattamente la richiesta minimizzando il costo complessivo trascurando la possibilità di costruire ulteriori collegamenti rispetto a quelli esistenti e supponendo che non ci siano limitazioni sulle quantità massime di merci trasportabili.

Formulazione.
È un problema che rientra nello schema generale di un problema di localizzazione di impianti e quindi può essere formulato in termini di Programmazione Lineare Intera come appena descritto nel caso generale.

– *Variabili.* È sufficiente introdurre le variabili binarie

$$\delta_i = \begin{cases} 1 & \text{se è costruito l'}i\text{−esimo deposito} \\ 0 & \text{altrimenti} \end{cases}$$

e le variabili x_{ij} che rappresentano la quantità di merce da trasportare dal deposito $i-$esimo alla zona $j-$esima.

– *Funzione obiettivo.* La funzione obiettivo da minimizzare sarà

$$15x_{11} + 13x_{12} + 27x_{13} + 9x_{14} + 7x_{15} + 12x_{21} + 21x_{22} + 34x_{23} + 21x_{24} + 3x_{25} \\ +7x_{31} + 10x_{32} + 2x_{33} + 17x_{34} + 12x_{35} + 10000\delta_1 + 15000\delta_2 + 13000\delta_3.$$

– *Vincoli.* I vincoli da considerare sono innanzitutto i vincoli di richiesta

$$\sum_{i=1}^{3} x_{i1} = 91, \quad \sum_{i=1}^{3} x_{i2} = 170, \quad \sum_{i=1}^{3} x_{i3} = 135, \quad \sum_{i=1}^{3} x_{i4} = 153, \quad \sum_{i=1}^{3} x_{i5} = 110.$$

Inoltre

$$\sum_{j=1}^{5} x_{1j} - 180\delta_1 \leq 0,$$

$$\sum_{j=1}^{5} x_{2j} - 230\delta_2 \leq 0,$$

$$\sum_{j=1}^{5} x_{3j} - 500\delta_3 \leq 0.$$

Poiché non si possono costruire più di due depositi, si deve poi imporre che

$$\delta_1 + \delta_2 + \delta_3 \leq 2.$$

Naturalmente devono essere anche esplicitati i vincoli

$$x_{ij} \geq 0 \quad \delta_i \in \{0,1\} \qquad i = 1,2,3 \quad j = 1,2,3,4,5.$$

Quindi la formulazione complessiva è:

$$\begin{cases} \min\Big(15x_{11} + 13x_{12} + 27x_{13} + 9x_{14} + 7x_{15} + 12x_{21} + 21x_{22} + 34x_{23} \\ \quad +21x_{24} + 3x_{25} + 7x_{31} + 10x_{32} + 2x_{33} + 17x_{34} + 12x_{35} \\ \quad +10000\delta_1 + 15000\delta_2 + 13000\delta_3\Big) \\ \sum_{i=1}^{3} x_{i1} = 91 \\ \sum_{i=1}^{3} x_{i2} = 170 \\ \sum_{i=1}^{3} x_{i3} = 135 \\ \sum_{i=1}^{3} x_{i4} = 153 \\ \sum_{i=1}^{3} x_{i5} = 110 \\ \sum_{j=1}^{5} x_{1j} - 180\delta_1 \leq 0 \\ \sum_{j=1}^{5} x_{2j} - 230\delta_2 \leq 0 \\ \sum_{j=1}^{5} x_{3j} - 500\delta_3 \leq 0 \\ \delta_1 + \delta_2 + \delta_3 \leq 2 \\ x_{ij} \geq 0 \quad \delta_i \in \{0,1\} \qquad i = 1,2,3 \quad j = 1,2,3,4,5 \end{cases}$$

□

Anche in questo caso riportiamo di seguito i file del modello e dei dati che realizzano una implementazione AMPL del problema in analisi.

localizzazione.mod

```
set CLIENTI;
set ZONE;

param costo_costruzione{ZONE}>=0;
param capacita_max{ZONE}>=0;
param richieste{CLIENTI}>=0;
param costi_trasp{ZONE,CLIENTI}>=0;

var x{ZONE,CLIENTI}>=0;
var d{ZONE} binary;

minimize costo_totale : sum{i in ZONE, j in CLIENTI}
```

```
 costi_trasp[i,j]*x[i,j]+sum{i in ZONE}costo_costruzione[i]*d[i];

s.t. vincoli_domanda{j in CLIENTI} :
                     sum{i in ZONE} x[i,j]=richieste[j];
s.t. vincoli_logici{i in ZONE} : sum{j in CLIENTI}
                     x[i,j] - capacita_max[i]*d[i]<=0;
s.t. max_depositi : sum{i in ZONE} d[i] <= 2;
```

localizzazione.dat

```
set CLIENTI:= C1 C2 C3 C4 C5;
set ZONE := dep1 dep2 dep3;

param : costo_costruzione capacita_max :=
dep1 10000  180
dep2 15000  230
dep3 13000  500;

param richieste :=
C1  91
C2  170
C3  135
C4  153
C5  110;

param costi_trasp : C1   C2  C3  C4  C5  :=
dep1  15 13 27 9 7
dep2  12 21 34 21 3
dep3  7  10  2 17 12;
```

6.4 VARIABILI BINARIE PER INDICARE IL SODDISFACIMENTO DI VINCOLI DISGIUNTIVI

Nell'usuale definizione di problemi di ottimizzazione si assume che tutti i vincoli debbano essere soddisfatti simultaneamente da una soluzione ammissibile. Tuttavia in molte applicazioni può accadere che solo un sottoinsieme dei vincoli debba essere soddisfatto e che tale sottoinsieme sia specificato dal valore che assume un'opportuna variabile di decisione. In questo caso si dice che i vincoli sono *disgiuntivi.*
Come esempio di questo uso delle variabili binarie, analizziamo una importante classe di problemi.

6.4.1 Problemi di "scheduling" (sequenziamento)

Si tratta di problemi di produzione in cui si deve decidere l'ordine di processamento di una sequenza di lavori su una macchina in grado di eseguire un lavoro alla volta (capacità unitaria). Si deve quindi esprimere la condizione disgiuntiva

"il lavoro i-esimo precede il lavoro j-esimo"
oppure
"il lavoro j-esimo precede il lavoro i-esimo".

Questo tipo di problema si presenta spesso in ambito industriale e nei sistemi di elaborazione.
Formalmente si ha la seguente situazione: siano dati n lavori *indipendenti* (il tempo di esecuzione di ciascun lavoro non dipende da quando viene eseguito rispetto agli altri lavori) e *indivisibili* (ciascun lavoro deve essere completato prima di poter eseguire il successivo).
Supponiamo inoltre che ciascun lavoro sia presente nel sistema fin dall'inizio, cioè che la macchina possa iniziare la lavorazione di un qualunque lavoro in qualsiasi istante.
Sia noto $p_i, \quad i = 1, \ldots, n$ il tempo di processamento di ciascun lavoro sulla macchina.
Il problema consiste nel determinare la sequenza di lavorazione dei lavori sulla macchina, cioè gli istanti $t_i, \ i = 1, \ldots, n$ in cui la macchina inizia la lavorazione del lavoro i-esimo, in modo da ottimizzare un opportuno criterio.
Avendo introdotto le variabili t_i indicanti gli istanti di tempo in cui la macchina inizia a processare l'i-esimo lavoro, formulare un problema di scheduling significa determinare i vincoli sulle variabili t_i in modo che esse rappresentino sequenze effettivamente realizzabili sulla macchina.
Formulazione.

– *Variabili.* Introduciamo formalmente le seguenti variabili: per indicare se il lavoro i precede il lavoro j o viceversa, per ogni $1 \leq i < j \leq n$, si introducono le

variabili $0-1$ così definite

$$y_{ij} = \begin{cases} 1 & \text{se il lavoro } i \text{ precede il lavoro } j \\ 0 & \text{se il lavoro } j \text{ precede il lavoro } i. \end{cases}$$

Si introducono, inoltre, le variabili temporali $t_i,\ \ i = 1, \ldots, n$ indicanti gli istanti di tempo di inizio dei lavori.

– *Vincoli.* Come già osservato, la macchina ha capacità unitaria e deve completare un lavoro prima di iniziarne un altro. Quindi uno solo dei due vincoli "il lavoro i-esimo precede il lavoro j-esimo", oppure "il lavoro j-esimo precede il lavoro i-esimo" deve essere soddisfatto.
Se il lavoro i è iniziato sulla macchina prima del lavoro j, si deve avere

$$t_j \geq t_i + p_i.$$

Se invece il lavoro j inizia prima del lavoro i, allora si deve avere

$$t_i \geq t_j + p_j.$$

Si devono, quindi, esplicitare le seguenti condizioni logiche:

$$y_{ij} = 1 \quad \Rightarrow \quad t_i - t_j \leq -p_i \tag{6.4.1}$$

$$y_{ij} = 0 \quad \Rightarrow \quad t_j - t_i \leq -p_j. \tag{6.4.2}$$

Se M è un limite superiore sia per $t_i - t_j + p_i$ sia per $t_j - t_i + p_j$, allora usando la (6.3.1), le condizioni (6.4.1) e (6.4.2) possono essere rispettivamente modellate dai vincoli

$$t_i - t_j + p_i \leq M(1 - y_{ij}) \qquad 1 \leq i < j \leq n \tag{6.4.3}$$

$$t_j - t_i + p_j \leq M y_{ij} \qquad 1 \leq i < j \leq n. \tag{6.4.4}$$

Infatti se $y_{ij} = 1$ la (6.4.3) esprime la condizione che la lavorazione del lavoro j può iniziare solo dopo il completamento del lavoro i mentre la (6.4.4) è sempre soddisfatta (per la scelta di M) e quindi non introduce ulteriori restrizioni. Se $y_{ij} = 0$, allora la (6.4.4) esprime la condizione che la lavorazione del lavoro i può iniziare solo dopo il completamento del lavoro j, mentre la (6.4.3) è sempre soddisfatta e quindi non introduce alcuna ulteriore restrizione. La (6.4.3) e la (6.4.4) sono di solito chiamati *vincoli di sequenziamento.*
Si devono inoltre esplicitare i vincoli di non negatività sulle variabili t_i, cioè

$$t_i \geq 0 \quad i = 1, \ldots, n.$$

Si può riassumere quanto fino ad ora esposto nel seguente risultato:

Teorema 6.4.1 *Se un vettore* $(t, y)^T$ *con* $t \in \mathbb{R}^n$ *ed* $y \in \{0,1\}^{n \times n}$ *soddisfa il sistema*

$$\begin{cases} t_i - t_j + p_i \leq M(1 - y_{ij}) \\ t_j - t_i + p_j \leq M y_{ij} \end{cases} \qquad 1 \leq i < j \leq n$$

allora ciascuna componente del vettore t *rappresenta un istante ammissibile di inizio processamento per il corrispondente lavoro. Viceversa, per ogni vettore ammissibile* t *esiste sicuramente un vettore* y *(che rappresenta l'ordine di processamento dei lavori sulla macchina) tale che il vettore* (t, y) *è ammissibile per il precedente sistema di vincoli.*

Naturalmente possono essere facilmente inseriti nel modello vincoli di precedenza o altre restrizioni temporali aggiungendo vincoli lineari sulle variabili t ed y.

– *Funzione obiettivo.* Nei problemi di scheduling la funzione obiettivo è di solito costruita in modo da ottimizzare un opportuno criterio. Analizziamo, ora, due dei criteri più diffusi:

a) *Tempo medio di permanenza nel sistema.*

   Ogni istante t_i può essere, infatti, anche interpretato come tempo di attesa nel sistema del lavoro i prima di essere processato. Quindi, il tempo medio di permanenza nel sistema può essere scritto

$$\frac{\sum_{i=1}^{n}(t_i + p_i)}{n}.$$

b) *Tempo complessivo di utilizzazione della macchina.*

   Questo criterio è significativo nel caso dell'uso di più macchine, perché nel caso di una sola macchina questo tempo complessivo è noto; infatti esso è dato da $\sum_{i=1}^{n} p_i$. Tuttavia anche in questo caso esso è esprimibile come nel caso generale cioè nella forma

$$\max_{1 \leq i < j \leq n} (t_i + p_i).$$

   Si osservi che questa fuzione obiettivo da minimizzare è di tipo "max" e quindi non è lineare.

Analizziamo, ora, un semplice esempio di problema di scheduling.

Esempio 6.4.1 *Sia data una macchina a capacità unitaria che deve effettuare tre lavori aventi tempo di processamento* $p_1 = 2,\ p_2 = 3,\ p_3 = 4,$. *Formulare il problema di scheduling che consenta di determinare la sequenza che minimizza il tempo medio di permanenza nel sistema, tenendo conto che, se il primo lavoro precede il secondo, l'inizio del terzo lavoro deve aspettare un tempo* $\Delta_3 = 2$ *dopo*

il termine del secondo lavoro, mentre, se il terzo lavoro precede il primo, l'inizio del secondo deve attendere un tempo $\Delta_2 = 3$ *dopo il termine del primo lavoro.*

Formulazione.
Formuliamo questo problema come appena esposto nel caso generale.

– *Variabili.* Introduciamo tre variabili continue t_1, t_2, t_3, indicanti gli istanti di inizio dei lavori sulla macchina e tre variabili $0-1$ per esprimere i vincoli di sequenziamento così definite:

$$y_{ij} = \begin{cases} 1 & \text{se il lavoro } i \text{ precede il lavoro } j \\ 0 & \text{se il lavoro } j \text{ precede il lavoro } i \end{cases} \quad 1 \leq i < j \leq 3.$$

– *Vincoli di sequenziamento.* Introducendo una costante positiva M che sia una limitazione superiore per $t_i - t_j + p_i$ e per $t_j - t_i + p_j$, i vincoli di sequenziamento possono essere scritti

$$\begin{aligned} t_1 - t_2 + 2 &\leq M(1 - y_{12}) \\ t_2 - t_1 + 3 &\leq M y_{12} \\ t_1 - t_3 + 2 &\leq M(1 - y_{13}) \\ t_3 - t_1 + 4 &\leq M y_{13} \\ t_2 - t_3 + 3 &\leq M(1 - y_{23}) \\ t_3 - t_2 + 4 &\leq M y_{23} \end{aligned}$$

– *Altri vincoli.* Gli altri vincoli di attese reciproche possono essere rappresentati utilizzando le variabili binarie precedentemente introdotte e la costante positiva M.

$$\begin{aligned} t_2 + 3 + 2 - t_3 &\leq M(1 - y_{12}) \\ t_1 + 2 + 3 - t_2 &\leq M y_{13} \end{aligned}$$

Inoltre, si devono esplicitare i vincoli di non negatività

$$t_1 \geq 0 \quad t_2 \geq 0 \quad t_3 \geq 0.$$

– *Funzione obiettivo.* La funzione obiettivo da minimizzare è data dal tempo medio di permanenza nel sistema e quindi può essere scritta

$$\frac{1}{3}(t_1 + 2 + t_2 + 3 + t_3 + 4).$$

La formulazione finale sarà quindi

$$\begin{cases} \min \dfrac{1}{3}(t_1 + t_2 + t_3 + 9) \\ t_1 - t_2 + 2 \leq M(1 - y_{12}) \\ t_2 - t_1 + 3 \leq M y_{12} \\ t_1 - t_3 + 2 \leq M(1 - y_{13}) \\ t_3 - t_1 + 4 \leq M y_{13} \\ t_2 - t_3 + 3 \leq M(1 - y_{23}) \\ t_3 - t_2 + 4 \leq M y_{23} \\ t_2 + 3 + 2 - t_3 \leq M(1 - y_{12}) \\ t_1 + 2 + 3 - t_2 \leq M y_{13} \\ t_1 \geq 0 \quad t_2 \geq 0 \quad t_3 \geq 0 \\ y_{12} \in \{0,1\}, \quad y_{13} \in \{0,1\}, \quad y_{23} \in \{0,1\}. \end{cases}$$

□

7

Teoria e Metodi della Programmazione Lineare Intera

7.1 INTRODUZIONE

Come visto precedentemente, molti problemi particolarmente importanti dal punto di vista applicativo sono riconducibili alla soluzione di un Problema di Programmazione Intera. In generale un Problema di Programmazione Intera può essere descritto nella seguente maniera:

$$\begin{cases} \min c^T x \\ x \in P \\ x \in \mathbf{Z}^n, \end{cases}$$

dove $P \subseteq \mathbb{R}^n$ è un generico poliedro e $\mathbf{Z}^n$ indica l'insieme dei vettori n-dimensionali a componenti intere. Quindi l'insieme ammissibile del precedente problema è costituito dai soli vettori a componenti intere contenuti nel dato poliedro P.
Intuitivamente sembrerebbe che un possibile modo per affrontare i problemi di Programmazione Lineare Intera sia quello di trascurare il vincolo di interezza sulle componenti del vettore x, di risolvere, quindi, il seguente problema di Programmazione Lineare

$$\begin{cases} \min c^T x \\ x \in P \\ x \in \mathbb{R}^n, \end{cases}$$

ottenendo come soluzione il punto $\hat{x}$ e, infine, di scegliere un vettore a componenti intere "vicino" al punto $\hat{x}$ (cioè un vettore che si ottiene da $\hat{x}$ sostituendo le sue componenti non intere con degli interi "vicini").
Tale strategia, in alcuni casi permette di ottenere una buona approssimazione di una soluzione $x^\star$ del problema di Programmazione Lineare Intera di partenza. Tuttavia, in generale, può essere non utilizzabile o non efficiente.

Infatti, in molti casi, i punti ottenuti dal precedente arrondamento delle componenti non intere del vettore $\hat{x}$ possono *non appartenere all'insieme ammissibile* del problema Programmazione Lineare Intera e quindi possono non aver nessun significato dal punto di vista applicativo.
Per superare la precedente difficoltà, si può scegliere, come approssimazione della soluzione $x^\star$ del problema di Programmazione Lineare Intera, un punto intero *ammissibile* "vicino" al punto $\hat{x}$. Tuttavia, può capitare che questi punti interi ammissibili siano delle *pessime approssimazioni* delle soluzione suluzioni del problema di partenza.
Da quanto detto, emerge chiaramente la necessità di cercare di sviluppare dei metodi per affrontare i problemi di Programmazione Lineare Intera che considerino direttamente la presenza del vincolo che le componenti del vettore x siano intere.

7.2 RELAZIONI TRA PROGRAMMAZIONE LINEARE INTERA E PROGRAMMAZIONE LINEARE

Consideriamo un problema di Programmazione Lineare Intera

$$\text{(PLI)} \quad \begin{cases} \min c^T x \\ Ax \geq b \\ x \geq 0, \quad \text{intero} \end{cases}$$

e il problema di Programmazione Lineare ottenuto dal problema (PLI) eliminando il vincolo di interezza sulle variabili, cioè il problema

$$\text{(PL)} \quad \begin{cases} \min c^T x \\ Ax \geq b \\ x \geq 0. \end{cases}$$

Poiché risulta che

a) l'insieme ammissibile di (PLI) è incluso nell'insieme ammissibile di (PL), cioè

$$\{x \in \mathbf{Z}^n \;\mid\; Ax \geq b, \;\; x \geq 0\} \subseteq \{x \in \mathbb{R}^n \;\mid\; Ax \geq b, \;\; x \geq 0\}$$

b) le funzioni obiettivo dei due problemi coincidono

allora il problema (PL) è un *rilassamento* del problema (PLI) e prende il nome di *rilassamento continuo* o *lineare*. Naturalmente se $x^\star$ è soluzione ottima del problema (PLI) e $y^\star$ è soluzione ottima del rilassamento (PL), allora risulta $c^T y^\star \leq c^T x^\star$.

Proposizione 7.2.1 *Se una soluzione ottima del problema rilassato (PL) è intera, allora essa è soluzione ottima del problema intero (PLI).*

Proposizione 7.2.2 *Sia $\hat{x}$ una soluzione ottima (non necessariamente a componenti intere) del problema rilassato (PL) e sia $\tilde{x}$ un punto ammissibile per il problema intero (PLI). Se $c^T\hat{x} = c^T\tilde{x}$ allora il punto $\tilde{x}$ è una soluzione ottima del problema intero (PLI).*

7.3 FORMULAZIONI LINEARI DI PROBLEMI DI PROGRAMMAZIONE LINEARE INTERA

Il questo paragrafo verrà messo in evidenza che ci sono più poliedri in grado di identificare (cioè di separare dagli altri) i punti interi che costituiscono l'insieme ammissibile di un problema di Programmazione Lineare Intera. In particolare verrà introdotto il concetto di "formulazione lineare" di un problema di Programmazione Lineare Intera e verrà fornita una caratterizzazione delle diverse formulazioni lineari di uno stesso problema di Programmazione Lineare Intera. Iniziamo considerando il seguente esempio di problema di Programmazione Lineare Intera

$$\begin{cases} \min -x_1 - x_2 \\ 2x_1 - 5x_2 \geq -5 \\ -2x_1 + 2x_2 \geq -1 \\ x_1 \geq 0, x_2 \geq 0, \quad \text{intere} . \end{cases} \tag{7.3.1}$$

L'insieme ammissibile di questo problema è costituito dai punti $(0,0), (0,1)$ e $(1,1)$.

Se si denota con P il poliedro del rilassamento lineare del problema (7.3.1) cioè

$$P = \{x \in \mathbb{R}^2 \ | \ 2x_1 - 5x_2 \geq -5, \ \ -2x_1 + 2x_2 \geq -1, \ \ x_1 \geq 0, \ \ x_2 \geq 0\}, \tag{7.3.2}$$

si può riscrivere il problema nella forma

$$\begin{cases} \min -x_1 - x_2 \\ x \in P \cap \mathbf{Z}^2. \end{cases}$$

Lo stesso insieme ammissibile del problema intero (7.3.1) (formato quindi dai tre punti $(0,0), (0,1)$ e $(1,1)$) può essere ottenuto, ad esempio, sostituendo il secondo

vincolo del problema intero (7.3.1) con il vincolo $-x_1 + x_2 \geq 0$ ottenendo così il seguente problema di Programmazione Lineare Intera

$$\begin{cases} \min -x_1 - x_2 \\ 2x_1 - 5x_2 \geq -5 \\ -x_1 + x_2 \geq 0 \\ x_1 \geq 0, x_2 \geq 0, \quad \text{intere} . \end{cases} \tag{7.3.3}$$

Con questa sostituzione la regione ammissibile del rilassamento lineare cambia e diventa

$$P' = \{x \in \mathbb{R}^2 \ \mid \ 2x_1 - 5x_2 \geq -5, \quad -x_1 + x_2 \geq 0, \quad x_1 \geq 0, \quad x_2 \geq 0\} \tag{7.3.4}$$

ma l'insieme ammissibile di questo nuovo problema di Programmazione Lineare Intera (7.3.3) continua ad essere costituito dai soli tre punti $(0,0), (0,1)$ e $(1,1)$; quindi il problema (7.3.3), che può essere riscritto nella forma

$$\begin{cases} \min -x_1 - x_2 \\ x \in P' \cap \mathbf{Z}^2, \end{cases}$$

è equivalente al problema (7.3.1).
Si possono ulteriormente variare i vincoli e ottenere gli stessi punti ammissibili per il problema intero; infatti possiamo, ad esempio, modificare il primo vincolo del problema (7.3.3) sostituendolo con il vincolo $x_2 \leq 1$, ottenendo così il seguente problema di Programmazione Lineare Intera

$$\begin{cases} \min -x_1 - x_2 \\ x_2 \leq 1 \\ -x_1 + x_2 \geq 0 \\ x_1 \geq 0, x_2 \geq 0, \quad \text{intere} . \end{cases} \tag{7.3.5}$$

Con questa sostituzione la regione ammissibile del rilassamento lineare cambia e diventa

$$P'' = \{x \in \mathbb{R}^2 \ \mid \ x_2 \leq 1, \quad -x_1 + x_2 \geq 0, \quad x_1 \geq 0, \quad x_2 \geq 0\} \tag{7.3.6}$$

ma l'insieme ammissibile di questo nuovo problema di Programmazione Lineare Intera (7.3.5) continua ad essere costituito dai soli tre punti $(0,0), (0,1)$ e $(1,1)$; quindi il problema (7.3.5), che può essere riscritto nella forma

$$\begin{cases} \min(-x_1 - x_2) \\ x \in P'' \cap \mathbf{Z}^2 \end{cases}$$

è equivalente al problema (7.3.3).

I tre problemi (7.3.1), (7.3.3), (7.3.5) ora considerati sono equivalenti dal punto di vista della Programmazione Lineare Intera avendo essi la stessa funzione obiettivo e lo stesso insieme ammissibile. Sono però, diverse le rappresentazioni fornite. Questo concetto si può formalizzare nella seguente definizione.

Definizione 7.3.1 FORMULAZIONE LINEARE
Un poliedro P è una formulazione lineare *per un problema di Programmazione Lineare Intera*

$$\begin{cases} \min c^T x \\ Ax \geq b \\ x \geq 0, \quad \textit{intero} \end{cases}$$

se detto

$$S = \{x \in \mathbb{R}^n \ \mid \ Ax \geq b, \ \ x \geq 0, \quad x \quad \textit{intero}\}$$

l'insieme ammissibile del problema di Programmazione Lineare Intera, risulta

$$S = P \cap \mathbf{Z}^n.$$

In accordo a questa definizione, i tre problemi poliedri P, P' e P'' rispettivamente definiti in (7.3.2), (7.3.4), (7.3.6) costituiscono formulazioni lineari di uno stesso problema di Programmazione Lineare Intera.
Quindi per una stesso problema di Programmazione Lineare Intera possono esistere piú formulazioni lineari; infatti, in generale, definendo due polideri

$$\bar{P} = \{x \in \mathbb{R}^n \ \mid \ \bar{A}x \geq \bar{b}, \ \ x \geq 0\}, \qquad \widetilde{P} = \{x \in \mathbb{R}^n \ \mid \ \widetilde{A}x \geq \widetilde{b}, \ \ x \geq 0\},$$

se risulta $P \cap \mathbf{Z}^n = P \cap \mathbf{Z}^n$, allora $\bar{P}$ e $\widetilde{P}$ rappresentano formulazioni per uno stesso problema di Programmazione Lineare Intera. Naturalmente i rilassamenti lineari associati sono diversi; infatti se si considerano i problema rilassati

$$\begin{cases} \min c^T x \\ x \in \bar{P} \end{cases} \qquad \begin{cases} \min c^T x \\ x \in \widetilde{P} \end{cases}$$

si avranno diversi valori ottimi; siano $\bar{z}$ e $\widetilde{z}$ rispettivamente i valori ottimi di questi due problemi rilassati; se $z^\star$ è la soluzione ottima del corrispondente problema intero, ovviamente si ha

$$\bar{z} \leq z^\star \qquad \text{e} \qquad \widetilde{z} \leq z^\star.$$

Quindi in entrambi i casi il valore ottimo dei due rilassamenti fornisce una limitazione inferiore (*"lower bound"*) del valore ottimo della soluzione ottima del problema di Programmazione Lineare Intera. Se inoltre vale $\bar{P} \subseteq \widetilde{P}$ risulta

$$\widetilde{z} \leq \bar{z} \leq z^\star,$$

e si ha quindi che il primo rilassamento fornisce un "lower bound" piú stringente del valore ottimo $z^\star$ e in questo senso è da preferirsi all'altro; questo permette di definire un ordinamento delle formulazioni. In particolare, il concetto di formulazione migliore si può formalizzare nella seguente definizione.

Definizione 7.3.2 *Date due formulazioni P_1 e P_2 di un problema di Programmazione Lineare Intera, si dice che P_1 è* migliore *di P_2 se $P_1 \subseteq P_2$.*

La definizione è giustificata dal fatto che, se risulta $P_1 \subseteq P_2$, allora la soluzione del rilassamento corrispondente a P_1 approssima meglio il valore dell'ottimo intero. Sulla base di questa definizione è possibile affermare che dato un problema di Programmazione Lineare Intera, può esistere una formulazione "ottima", cioè una formulazione migliore, nel senso specificato dalla definizione, di qualsiasi altra formulazione lineare del problema di Programmazione Lineare Intera; tale formulazione ottima è costituita dal poliedro contenuto in tutti i poliedri che contengono la regione ammissibile del problema intero. Infatti, formalmente si ha il seguente teorema.

Teorema 7.3.1 *Sia dato un problema di Programmazione Lineare Intera*

$$\begin{cases} \min c^T x \\ x \in P \cap \mathbf{Z}^n \end{cases}$$

con $P = \{x \in \mathbb{R}^n \mid Ax \geq b, \quad x \geq 0\}$. Se A e b sono a componenti razionali, allora esiste sempre una formulazione ottima costituita da un poliedro $P^\star$ che è il piú piccolo insieme convesso che contiene $P \cap \mathbf{Z}^n$.

Se fosse noto il poliedro $P^\star$, il problema di Programmazione Lineare Intera si potrebbe riformulare come

$$\begin{cases} \min c^T x \\ x \in P^\star. \end{cases} \tag{7.3.7}$$

Negli esempi visti nelle pagine precedenti la formulazione P'' data dalla (7.3.6) è la formulazione ottima del problema di Programmazione Lineare Intera (7.3.1). È molto importante notare che il poliedro P'' definito nella (7.3.6) ha tutti vertici a componenti intere. Questo è vero in generale; infatti vale il seguente teorema.

Teorema 7.3.2 *La formulazione ottima $P^\star$ di un problema di Programmazione Lineare Intera ha tutti i vertici a componenti intere. Viceversa, se un polidero P ha tutti i vertici interi allora esso costituisce la formulazione ottima di ogni problema di Programmazione Lineare Intera con insieme ammissibile $P \cap \mathbf{Z}^n$.*

Sarebbe molto importante conoscere $P^\star$ perché i vertici di $P^\star$ risultano essere tutti a componenti intere e quindi sarebbe sufficiente risolvere (ad esempio con il metodo del simplesso) il problema rilassato (7.3.7) ed ottenere la soluzione del problema di Programmazione Lineare Intera originario.
Questo fatto è di notevole importanza in quanto se si utilizza il metodo del simplesso per risolvere il problema rilassato, la soluzione ottenuta, che è un vertice del poliedro, sicuramente è a componenti intere.
Purtroppo però, tale formulazione ottima in generale non è nota oppure non è utilizzabile per il numero eccessivo dei vincoli (c'è una crescita esponenziale dei vincoli con la dimensione del problema).

7.4 IL METODO DEL "BRANCH AND BOUND"

Il "Branch and Bound" (BB) è una metodologia di ricerca della soluzione ottima che effettua un'esplorazione *parziale* dell'insieme delle soluzioni ammissibili. In particolare la funzione obiettivo viene calcolata per un sottoinsieme di cardinalità abbastanza piccola delle soluzioni ammissibili con la proprietà di contenere almeno una soluzione ottima.

Per descrivere in dettaglio questa tecnica facciamo riferimento al generico problema di Programmazione Lineare Intera

$$\begin{cases} \min c^T x \\ Ax \geq b \\ x \geq 0, \quad \text{intero} \end{cases} \tag{7.4.1}$$

che può essere riscritto nella forma

$$\begin{cases} \min c^T x \\ x \in S \end{cases}$$

dove $S = \{x \in \mathbb{R}^n \mid Ax \geq b, \quad x \geq 0, \quad \text{intero}\}$ è l'insieme ammissibile. Indicheremo con $x^\star$ l'ottimo del problema (7.4.1) e con $z^* = c^T x^\star$ il suo valore ottimo corrispondente.

La strategia che è alla base della tecnica del "branch and bound" è la decomposizione del problema originario (7.4.1) in sottoproblemi. Questo viene realizzato effettuando una partizione dell'insieme ammissibile S in una famiglia $\{S_1, \ldots, S_q\}$ di sottoinsiemi di S con $q \geq 2$, cioè tali che

$$S_i \cap S_j = \emptyset \qquad \text{per ogni coppia} \qquad 1 \leq i < j \leq r$$

e

$$\bigcup_{i=1}^{q} S_i = S.$$

A seguito di questa partizione si possono considerare q sottoproblemi (che indichiamo con $Prob^{(i)}$) del tipo

$$\begin{cases} \min c^T x \\ x \in S_i \end{cases} \qquad i = 1, \ldots, q. \qquad (Prob^{(i)})$$

Ora, se $x^{(i)}$ è l'ottimo dell'i-esimo sottoproblema $Prob^{(i)}$ e $z^{(i)} = c^T x^{(i)}$ il valore ottimo corrispondente, si ha che la soluzione ottima del problema originario è data dalla $x^{(i)}$ corrispondente al minimo tra i valori $z^{(i)} = c^T x^{(i)}$, $i = 1, \ldots, q$. Identificando il problema originario (7.4.1) con il problema $Prob^{(0)}$ e il suo insieme ammissibile S con S_0 si può dire che i nuovi problemi generati $Prob^{(1)}, \ldots, Prob^{(q)}$ sono *"figli"* del problema *"padre"* $Prob^{(0)}$.
Se un sottoproblema $Prob^{(i)}$ dovesse risultare, a sua volta, di difficile soluzione si partiziona ulteriormente l'insieme S_i producendo nuovi sottoproblemi (figli) ed iterando la procedura fino a che il problema originario non risulti decomposto in problemi elementari di facile soluzione. Si osservi che questa generazione progressiva di sottoproblemi (figli) produce un *albero genealogico* detto *albero di enumerazione.*

In generale però, risolvere un sottoproblema può essere difficile tanto quanto risolvere il problema originario ed è per questo motivo che invece della soluzione esatta del problema $Prob^{(i)}$ si preferisce calcolare una limitazione inferiore *"lower bound"* L_i di $z^{(i)}$ e cioè un valore $L_i \leq z^{(i)}$. Tale valore viene poi confrontato, ad un certo passo della procedura, con il miglior valore della funzione obiettivo trovato fino a quel momento che chiameremo *valore ottimo corrente* e indicheremo con $\widetilde{z}$. Se il lower bound trovato L_i risulta non inferiore a quello del valore ottimo corrente, ovvero se

$$\widetilde{z} \leq L_i \leq z^{(i)}$$

allora nell'insieme S_i non esiste un punto in cui la funzione obiettivo abbia un valore migliore (cioè minore) di $\widetilde{z}$. Questo permette di sospendere l'esame del sottoproblema $Prob^{(i)}$ senza risolverlo e di non considerarlo ulteriormente nella soluzione del problema originario.

Da quanto detto emerge che la tecnica del "branch and bound" è caratterizzata da due fasi principali:

i) Fase di "bounding"
Calcolo dei "lower bound" dei sottoproblemi allo scopo di acquisire l'informazione necessaria per capire se scartare o meno un sottoproblema.

ii) Fase di "branching"
Generazione di sottoproblemi e quindi dell'albero di enumerazione.

È importante osservare che entrambi le fasi devono essere realizzabili attraverso procedure efficienti. Inoltre il numero dei sottoproblemi generati deve essere estremamente limitato affinché la strategia nel suo complesso risulti efficiente.

Riportiamo, ora, uno schema algoritmico che implementa il metodo del "branch and bound". Indicheremo con $\mathcal{L}$ l'insieme dei sottoproblemi candidati (generati nelle varie fasi di branching) che devono ancora essere analizzati; tali sottoproblemi vengono detti *aperti*. Con $\widetilde{z}$ indicheremo il *valore ottimo corrente* e con $\widetilde{x}$ la *soluzione ottima corrente* che vengono dapprima inizializzati e poi aggiornati nel corso dell'algoritmo non appena vengono individuate soluzioni ammissibili per il problema originario "migliori" cioè con valori inferiori della funzione obiettivo. Per ciascun problema $Prob^{(i)} \in \mathcal{L}$ indicheremo con L_i il valore di un "lower bound" e con $\bar{x}^{(i)}$ un vettore in corrispondenza del quale è raggiunto questo "lower bound", cioè un vettore $\bar{x}^{(i)}$ tale che risulti $L_i = c^T \bar{x}^{(i)}$.

Metodo del "Branch and Bound"

Inizializzazione

- Si inizializzano $\widetilde{z}$ e $\widetilde{x}$ determinando una soluzione ammissibile (facile da calcolare) del problema originario. Naturalmente risulta $\widetilde{z} \geq z^\star$. Se tale soluzione $\widetilde{x}$ non è facile da individuare si lascia per ora non definito $\widetilde{x}$ e si pone $\widetilde{z} = +\infty$.
- Si applica una *strategia di "bounding"* per determinare un "lower bound" L_0 del valore ottimo del problema $Prob^{(0)}$ e un vettore $\bar{x}^{(0)}$ in corrispondenza del quale si ha $L_0 = c^T \bar{x}^{(0)}$.
  - Se $\bar{x}^{(0)} \in S_0$ oppure $L_0 = \widetilde{z}$, l'algoritmo termina; nel primo caso si ha che la soluzione ottima del problema originario è $\bar{x}^{(0)}$, nel secondo tale soluzione ottima è l'ottimo corrente $\widetilde{x}$.
  - Altrimenti si applica una *strategia di "branching"* per generare nuovi sottoproblemi che vengono inseriti nella lista dei sottoproblemi aperti $\mathcal{L}$.

Iterazione generica

Si esamina la lista $\mathcal{L}$ dei sottoproblemi aperti. Se $\mathcal{L} = \emptyset$ si pone $x^\star = \widetilde{x}$ e $z^\star = \widetilde{z}$ e l'algoritmo termina. Se invece $\mathcal{L} \neq \emptyset$ si estrae un sottoproblema aperto $Prob^{(j)}$ dalla lista $\mathcal{L}$ e si procede come segue:

- si applica una *strategia di "bounding"* per calcolare un "lower bound" L_j del valore ottimo del sottoproblema $Prob^{(j)}$ e un vettore $\bar{x}^{(j)}$ in corrispondenza del quale si ha $L_j = c^T\bar{x}^{(j)}$, con la convenzione di porre $L_j = +\infty$ se il problema risultasse inammissibile.

  - Se $L_j \geq \widetilde{z}$ allora il sottoproblema $Prob^{(j)}$ viene chiuso in quanto nessuna soluzione "migliore" dell'ottimo corrente può essere contenuta nell'insieme ammissibile del sottoproblema $Prob^{(j)}$.
  - Se invece $L_j < \widetilde{z}$ il sottoproblema $Prob^{(j)}$ potrebbe permettere di migliorare l'ottimo corrente e quindi
    - se $\bar{x}^{(j)} \in S_j$, allora $\bar{x}^{(j)}$ è soluzione ottima del sottoproblema $Prob^{(j)}$ e poiché si sta considerando il caso $L_j < \widetilde{z}$, si ha che $\bar{x}^{(j)}$ è una soluzione ammissibile del problema originario (in quanto $S_i \subseteq S_0$) in cui il valore $L_j = c^T\bar{x}^{(j)}$ è minore (e pertanto migliore) del valore ottimo corrente. Si aggiorna quindi l'ottimo corrente ponendo $\widetilde{x} = \bar{x}^{(j)}$ e $\widetilde{z} = L_j = c^T\bar{x}^{(j)}$ e si chiude il problema $Prob^{(j)}$.

    - se invece $\bar{x}^{(j)} \notin S_i$ allora si applica una *strategia di "branching"* al problema $Prob^{(j)}$ in modo da generare nuovi sottoproblemi che vengono inseriti nella lista dei problemi candidati $\mathcal{L}$.

Evidentemente la risoluzione del problema originario sarà tanto più efficiente quanto migliori saranno i valori dei "lower bound" ed a loro volta tali valori approssimeranno tanto meglio il valore ottimo del sottoproblema quanto più efficace sarà stata la decomposizione del problema originario. Di conseguenza l'efficienza del metodo del "branch and bound" dipende essenzialmente dalla qualità delle strategie che ne caratterizzano la struttura che sono:

a) *la strategia di bounding*, ovvero la strategia per determinare i "lower bound" cioè per calcolare un valore che approssimi per difetto il valore ottimo dei sottoproblemi.

b) *la strategia di branching*, ovvero la strategia per determinare la partizione dell'insieme delle soluzioni ammissibili di un sottoproblema.

c) *la strategia per la scelta del sottoproblema da esaminare*, ovvero come decidere, ad ogni iterazione, quale sottoproblema selezionare dalla lista $\mathcal{L}$ dei problemi aperti.

Ovviamente a seconda della strategia di bounding, di branching e di scelta del sottoproblema da estrarre da $\mathcal{L}$ adottate, lo schema generale appena descritto si concretizzerà in un algoritmo differente.

Qui di seguito descriveremo alcune delle strategie più utilizzate nella pratica.

Strategie di "bounding".
Esistono varie strategie per il calcolo dei "lower bound". Per descriverne due delle piú utilizzate consideriamo

$$P^i = \{x \in \mathbb{R}^n \ \mid \ A^i x \geq b^i, \ \ x \geq 0\}$$

una possibile formulazione del sottoproblema $Prob^{(i)}$ che quindi può essere scritto nella forma

$$\begin{cases} \min c^T x \\ x \in P^i \cap \mathbf{Z}^n. \end{cases} \qquad (Prob^{(i)})$$

1. *Rilassamento lineare.*
   Un possibile modo di calcolare i "lower bound" è quello di considerare il rilassamento lineare del problema $Prob^{(i)}$ cioè il problema ottenuto eliminando il vincolo di interezza, ovvero il problema

   $$\begin{cases} \min c^T x \\ x \in P^i \end{cases} \qquad (7.4.2)$$

   e di porre

   $$L_i = c^T \bar{x}^{(i)}$$

   dove $\bar{x}^{(i)}$ è soluzione ottima del rilassamento lineare, cioè del problema (7.4.2). Infatti, come abbiamo già visto nel paragrafo 7.2, il valore ottimo del problema rilassato è sempre minore o uguale al valore ottimo del problema intero $Prob^{(i)}$ ed inoltre se la soluzione ottima del problema rilassato è intera, allora essa è anche soluzione ottima del problema $Prob^{(i)}$ (si veda la Proposizione 7.2.1). Si osservi che il problema rilassato è risolubile in maniera molto efficiente (ad esempio con il metodo del simplesso).

2. *Rilassamento della formulazione.*
   Per ottenere un "lower bound" è possibile considerare oltre al rilassamento lineare un qualsiasi altro rilassamento del problema intero $Prob^{(i)}$. Infatti, il valore ottimo di qualunque problema del tipo

   $$\begin{cases} \min c^T x \\ x \in P' \end{cases}$$

   con P' poliedro tale che $S_i \subseteq P'$, fornisce un "lower bound" per il problema $Prob^{(i)}$. È però molto importante notare che il poliedro P' non è necessariamente una formulazione del problema $Prob^{(i)}$ e quindi può contenere

punti a coordinate intere che non appartengono all'insieme ammissibile S_i del problema $Prob^{(i)}$. Quindi non è piú vero che una soluzione ottima intera del problema rilassato è una soluzione ottima del problema intero

Strategie di "branching".
Vediamo ora una semplice strategia per separare un generico problema $Prob^{(i)}$. Limiteremo la trattazione ad una strategia di tipo binario ovvero ad una strategia che genera sempre due sottoproblemi; nonostante la sua semplicità questa strategia si rivela in pratica molto efficiente.
Supponiamo di aver risolto il rilassamento lineare di $Prob^{(i)}$ e sia, come già detto, $\bar{x}^{(i)}$ la sua soluzione ottima e $L_i = c^T \bar{x}^{(i)}$ il corrispondente valore ottimo.
Se $\bar{x}^{(i)}$ ha tutte componenti intere allora $\bar{x}^{(i)} \in S_i$ e quindi è una soluzione ottima del problema $Prob^{(i)}$ e quindi il problema non va separato, ma chiuso.
Se L_i è maggiore o uguale al valore ottimo corrente $\widetilde{z}$, il problema non può dare origine ad un punto in cui il valore della funzione obiettivo sia migliore di quello corrente e non è necessario separarlo per trovare la sua soluzione ottima intera e quindi va chiuso.
Supponiamo quindi che nessuno di questi due casi si sia verificato, cioè $\bar{x}^{(i)}$ abbia almeno una componente frazionaria e $L_i < \widetilde{z}$, e vediamo come separare questo sottoproblema $Prob^{(i)}$.
Sia $\bar{x}_k^{(i)}$ una componente non intera del vettore $\bar{x}^{(i)}$. Separiamo il problema $Prob^{(i)}$ nei seguenti due problemi:

$$Prob^{(i,1)} \begin{cases} \min c^T x \\ x \in S_i \\ x_k \leq \lfloor \bar{x}_k^{(i)} \rfloor \end{cases} \qquad \text{e} \qquad Prob^{(i,2)} \begin{cases} \min c^T x \\ x \in S_i \\ x_k \geq \lceil \bar{x}_k^{(i)} \rceil \end{cases}$$

dove $\lfloor \bar{x}_k^{(i)} \rfloor$ indica la sua parte intera inferiore (ossia il più grande intero minore di $\bar{x}_k^{(i)}$) e $\lceil \bar{x}_k^{(i)} \rceil$ la sua parte intera superiore (ossia il più piccolo intero maggiore di $\bar{x}_k^{(i)}$). $Prob^{(i,1)}$ è ottenuto da $Prob^{(i)}$ semplicemente aggiungendo a $Prob^{(i)}$ il vincolo $x_k \leq \lfloor \bar{x}_k^{(i)} \rfloor$ e $Prob^{(i,2)}$ aggiungendo a $Prob^{(i)}$ il vincolo $x_k \geq \lceil \bar{x}_k^{(i)} \rceil$. È facile verificare che l'unione delle regioni ammissibili di questi due problemi coincide con la regione ammissibile S_i e che la loro intersezione è vuota; abbiamo così realizzato una partizione di S_i.

Strategie per la scelta del sottoproblema da esaminare.
Esistono diverse strategie di scelta, tra queste le più usate sono le seguenti:

1. Scelta del sottoproblema con il minimo "lower bound". Tale scelta ha lo scopo di esaminare per primi quei sottoproblemi in cui è più probabile trovare una soluzione ottima. Infatti, se la strategia di bounding produce buone approssimazioni dei valori ottimi dei sottoproblemi, a valori bassi del

"lower bound" corrisponderanno bassi valori delle soluzioni ottime dei sottoproblemi. Pertanto, esaminando il sottoproblema aperto cui corrisponde il minimo valore del "lower bound" si avrà una maggiore probabilità di individuare la soluzione ottima del problema originario.

2. Scelta con criterio di priorità LIFO (Last In First Out). In questo caso i sottoproblemi da esaminare sono gestiti dalla procedura secondo lo schema a *pila* (*stack*). In particolare, il sottoproblema scelto in $\mathcal{L}$ è quello che da meno tempo si trova in $\mathcal{L}$.

3. Scelta con criterio di priorità FIFO (First In First Out). In questo caso i sottoproblemi da esaminare sono gestiti dalla procedura secondo lo schema a *coda*. In particolare, il sottoproblema scelto in $\mathcal{L}$ è quello che da più tempo si trova in $\mathcal{L}$.

Integrando lo schema algoritmico già esaminato con queste strategie si ottiene un algoritmo effettivamente realizzabile.

Le Figure 7.4.1 e 7.4.2 riassumono con dei diagrammi di flusso i passi fondamentali di un algoritmo *Branch and Bound* per la soluzione del generico problema di PLI

$$\begin{cases} \min c^T x \\ x \in S = P \cap \mathbf{Z}^n. \end{cases}$$

che utilizza il *rilassamento lineare* per calcolare i lower bound.

Osservazione 7.4.1 Se il problema originario è un problema di Programmazione Lineare 0–1, ovvero le variabili del problema possono assumere solo i valori 0 o 1, si può supporre che la formulazione del problema sia contenuta nell'insieme $\{x \in \mathbb{R}^n \mid 0 \leq x_h \leq 1, \quad h = 1, \ldots, n\}$. Quindi ogni componente $\bar{x}_k^{(i)}$ dell'ottimo $\bar{x}^{(i)}$ del rilassato lineare del problema $Prob^{(i)}$ è compresa tra 0 e 1; quindi poiché ovviamente risulta $\lfloor \bar{x}_k^{(i)} \rfloor = 0$ e $\lceil \bar{x}_k^{(i)} \rceil = 1$, i sottoproblemi che eventualmente vengono generati nella fase di "branching" si ottengono ponendo $x_k = 0$ in uno e $x_k = 1$ nell'altro.

Osservazione 7.4.2 Se il problema di Programmazione Lineare Intera in esame è un problema di massimizzazione invece che di minimizzazione (come assunto fino ad ora), la tecnica del "branch and bound" si applica in maniera analoga sostituendo i "lower bound" L_i con "upper bound" U_i, ovvero con delle limitazioni superiori del valore ottimo del problema intero. Ovviamente l'"upper bound" associato ad un sottoproblema con regione ammissibile vuota sarà posto per convenzione uguale a $-\infty$. Inoltre la chiusura dei sottoproblemi dominati dall'ottimo corrente avverrà nel caso in cui $U_i \leq \widetilde{z}$.

Vediamo ora un esempio di applicazione del metodo del "Branch and Bound" appena descritto.

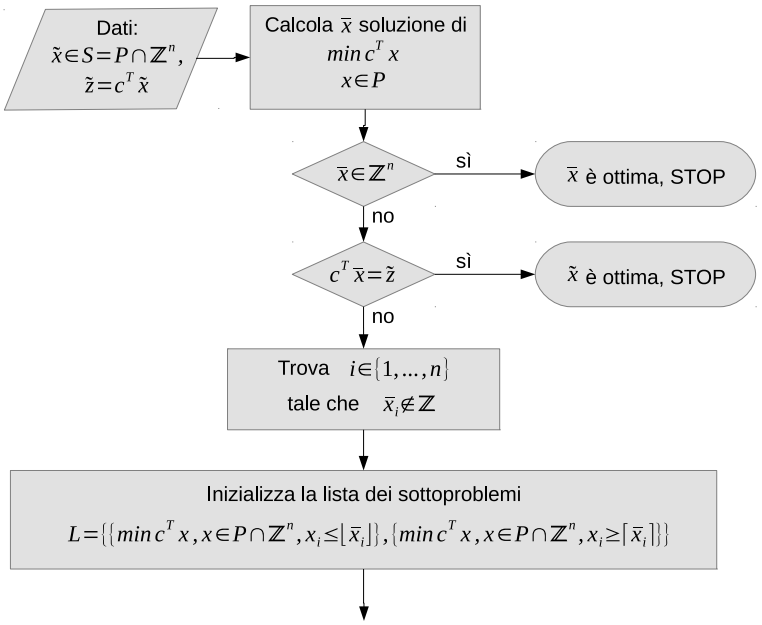


Fig. 7.4.1 Inizializzazione del Branch and Bound

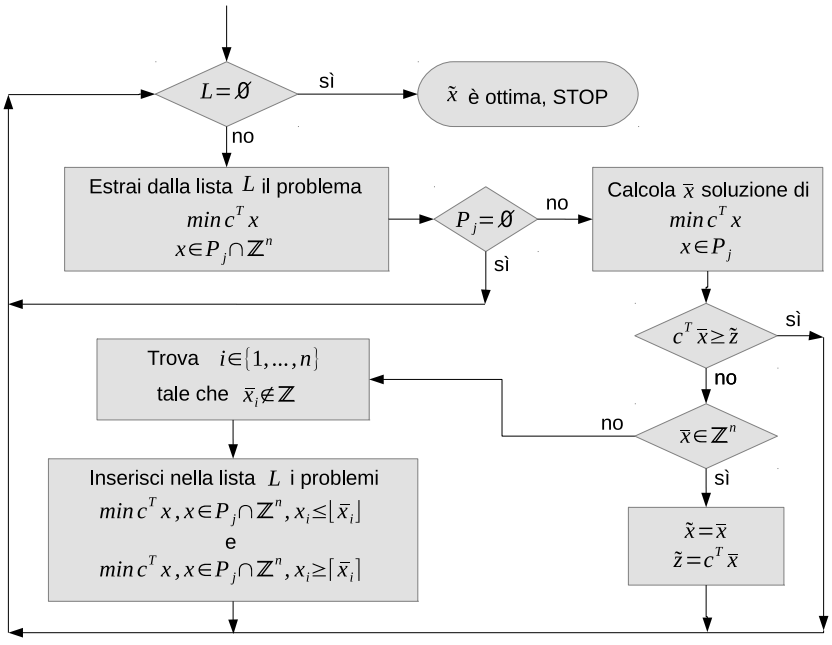


Fig. 7.4.2 Generica iterazione del Branch and Bound

Esempio 7.4.3 *Sia dato il seguente problema lineare intero:*

$$\begin{cases} \min -2x_1 - 8x_2 \\ -2x_1 + 6x_2 \geq -2 \\ x_1 - 4x_2 \geq -5 \\ -2x_1 - 2x_2 \geq -13 \\ x_1 \geq 0, x_2 \geq 0 \\ x_1, x_2 \in Z. \end{cases}$$

Dato un punto ammissibile $(3,2)^T$*, risolvere il problema con la tecnica del Branch and Bound.*

Chiameremo $Prob^{(0)}$ il problema dato. Poiché il problema è in due variabili, possiamo risolvere tutti i rilassamenti lineari per via grafica.

Inizializzazione
Innanzitutto inizializziamo l'ottimo corrente utilizzando la soluzione ammissibile fornita, ovvero $\widetilde{x} = (3,2)^T$ e $\widetilde{z} = -22$.

Calcoliamo L_0 risolvendo graficamente il rilassamento lineare di $Prob^{(0)}$ e otteniamo

$$\bar{x}^{(0)} = (21/5, \ \ 23/10)^T, \qquad L_0 = -2 \cdot 21/5 - 8 \cdot 23/10 = -26.8.$$

Poichè $\bar{x}^{(0)}$ non è a componenti intere, non possiamo dichiararlo soluzione del problema. Inoltre poichè $L_0 < \widetilde{z} = -22$, neanche $\widetilde{x}$ può essere dichiarato ottimo del problema. Il vettore $\bar{x}^{(0)}$ non è intero e quindi separiamo rispetto a una sua componente frazionaria, per esempio rispetto a quella con indice piú basso, cioè x_1; vengono allora generati i due sottoproblemi

$$Prob^{(1)} \quad \begin{cases} \min -2x_1 - 8x_2 \\ -2x_1 + 6x_2 \geq -2 \\ x_1 - 4x_2 \geq -5 \\ -2x_1 - 2x_2 \geq -13 \\ x_1 \geq \lceil 21/5 \rceil = 5 \\ x_1 \geq 0, x_2 \geq 0 \\ x_1, x_2 \in Z. \end{cases} \qquad \text{e} \qquad Prob^{(2)} \begin{cases} \min -2x_1 - 8x_2 \\ -2x_1 + 6x_2 \geq -2 \\ x_1 - 4x_2 \geq -5 \\ -2x_1 - 2x_2 \geq -13 \\ x_1 \leq \lfloor 21/5 \rfloor = 4 \\ x_1 \geq 0, x_2 \geq 0 \\ x_1, x_2 \in Z. \end{cases}$$

e viene inizializzata la lista $\mathcal{L}$:

$$\mathcal{L} = \left\{ Prob^{(1)}, Prob^{(2)} \right\}.$$

Prima iterazione
La lista $\mathcal{L}$ non è vuota, e quindi estraiamo $Prob^{(1)}$ dalla lista:

$$\mathcal{L} = \left\{ Prob^{(2)} \right\}.$$

Siccome $Prob^{(1)}$ risulta ammissibile, risolviamo geometricamente il suo rilassamento lineare. Si ottiene

$$\bar{x}^{(1)} = (5,\ 3/2)^T, \qquad L_1 = -22,$$

Poiché risulta $L_1 = \widetilde{z} = -22$ il problema $Prob^{(1)}$ si può chiudere.

Seconda iterazione
La lista $\mathcal{L}$ non è vuota, e quindi estraiamo l'unico problema in esso contenuto, cioè $Prob^{(2)}$:

$$\mathcal{L} = \emptyset.$$

Siccome $Prob^{(2)}$ risulta ammissibile, risolviamo geometricamente il suo rilassamento lineare. Si ottiene

$$\bar{x}^{(2)} = (4,\ 9/4)^T, \qquad L_2 = -26.$$

Poiché risulta $L_2 = -26 < \widetilde{z} = -22$ ed inoltre $\bar{x}^{(2)}$ non è a componenti intere, il problema $Prob^{(2)}$ non si può chiudere e quindi si effettua un'operazione di separazione rispetto all'unica componente non intera di $\bar{x}^{(2)}$, ovvero la seconda componente; vengono generati i due sottoproblemi

$$Prob^{(3)} \quad \begin{cases} \min -2x_1 - 8x_2 \\ -2x_1 + 6x_2 \geq -2 \\ x_1 - 4x_2 \geq -5 \\ -2x_1 - 2x_2 \geq -13 \\ x_1 \leq 4 \\ x_2 \geq \lceil 9/4 \rceil = 3 \\ x_1 \geq 0, x_2 \geq 0 \\ x_1, x_2 \in Z. \end{cases} \quad \text{e} \quad Prob^{(4)} \quad \begin{cases} \min -2x_1 - 8x_2 \\ -2x_1 + 6x_2 \geq -2 \\ x_1 - 4x_2 \geq -5 \\ -2x_1 - 2x_2 \geq -13 \\ x_1 \leq 4 \\ x_2 \leq \lfloor 9/4 \rfloor = 2 \\ x_1 \geq 0, x_2 \geq 0 \\ x_1, x_2 \in Z. \end{cases}$$

e vengono inseriti nella lista $\mathcal{L}$:

$$\mathcal{L} = \left\{ Prob^{(3)}, Prob^{(4)} \right\}.$$

Terza iterazione
La lista $\mathcal{L}$ non è vuota, e quindi estraiamo $Prob^{(3)}$ dalla lista:

$$\mathcal{L} = \left\{ Prob^{(4)} \right\}.$$

Il $Prob^{(3)}$ risulta inammissibile e quindi si può chiudere.

Quarta iterazione
La lista $\mathcal{L}$ non è vuota, e quindi estraiamo l'unico problema in esso contenuto, cioè $Prob^{(4)}$:

$$\mathcal{L} = \emptyset.$$

Siccome $Prob^{(4)}$ risulta ammissibile, risolviamo geometricamente il suo rilassamento lineare. Si ottiene

$$\bar{x}^{(4)} = (4,2)^T, \qquad L_4 = -24.$$

Poiché si ha $L_4 = -24 < \widetilde{z} = -22$ e $\bar{x}^{(4)}$ è a componenti intere, si può chiudere il problema $Prob^{(4)}$ e si aggiorna la soluzione ottima corrente e il valore ottimo corrente ponendo

$$\widetilde{x} = (4,2)^T \qquad \widetilde{z} = -24.$$

Quinta iterazione
Siccome la lista $\mathcal{L}$ è vuota l'algoritmo termina e la soluzione ottima è data da

$$x^\star = \widetilde{x} = (4,2)^T \qquad \text{con valore ottimo} \qquad z^\star = \widetilde{z} = -24.$$

□

7.4.1 Il problema del knapsack binario.

Dato un problema di knapsack

$$\begin{aligned} &\max c^T x \\ &a^T x \leq b \\ &x \in \{0,1\}^n \end{aligned}$$

il suo rilassamento continuo è

$$\begin{aligned} &\max c^T x \\ &a^T x \leq b \\ &0 \leq x \leq 1. \end{aligned}$$

Possiamo supporre che i coefficienti a_i e c_i, $i = 1, \ldots, n$ siano positivi (vedere osservazione di seguito).
Per determinare la soluzione del problema rilassato si può procedere come segue

(i) Si riordinano le variabili in modo che i rapporti peso ingombro siano ordinati in modo non crescente, cioè

$$\frac{c_1}{a_1} \geq \frac{c_2}{a_2} \geq \ldots \geq \frac{c_n}{a_n}.$$

(ii) Si determina l'indice critico k per cui

$$\sum_{i=1}^{k} a_i > b \qquad \sum_{i=1}^{k-1} a_i \leq b.$$

(iii) La soluzione ottima è quindi

$$\begin{aligned} x_i^* &= 1, \quad \text{per} \quad i = 1, \ldots, k-1 \\ x_k^* &= \frac{b - \sum_{i=1}^{k-1} a_i}{a_k}, \\ x_i^* &= 0, \quad \text{per} \quad i = k+1, \ldots, n \end{aligned}$$

Osservazione Dato un problema di knapsack continuo limitato, possiamo sempre suppore che il coefficienti a_i e c_i per $i = 1, \ldots, n$ siano positivi. Infatti negli altri casi è possibile ricondursi a questa situazione come descritto nel seguito.

1) La variabile x_i non compare nella funzione obiettivo, ovvero $c_i = 0$. In questo caso si può fissare la variabile x_i in base al segno di a_i:

   - se $a_i > 0$ allora si fissa $x_i = 0$ e il problema può essere riformulato escludendo la variabile i-esima (senza modificare il termine noto b),
   - se $a_i < 0$ allora si fissa $x_i = 1$ e il problema può essere riformulato escludendo la variabile i-esima (ovviamente sommando al termine noto b il valore $-a_i$).

2) La variabile x_i non compare nel vincolo, ovvero $a_i = 0$. In questo caso si può fissare la variabile x_i in base al segno di c_i:

   - se $c_i > 0$ allora si fissa $x_i = 1$ e il problema può essere riformulato escludendo la variabile i-esima (senza modificare il termine noto b),
   - se $c_i < 0$ allora si fissa $x_i = 0$ e il problema può essere riformulato escludendo la variabile i-esima (senza modificare il termine noto b).

3) L'i-esimo coefficiente della funzione obiettivo è negativo, ed il corrispondente coefficiente nel vincolo è positivo: $c_i < 0,\ a_i > 0$. In questo caso si può fissare la variabile $x_i = 0$. Il problema può quindi essere riformulato escludendo la variabile i-esima senza modificare il termine noto b.

4) L'i-esimo coefficiente della funzione obiettivo è positivo, ed il corrispondente coefficiente nel vincolo è negativo: $c_i > 0,\ a_i < 0$. In questo caso si può fissare la variabile $x_i = 1$. Il problema può quindi essere riformulato escludendo la variabile i-esima e sommando al termine noto b il valore $-a_i$.

5) Entrambi i coefficienti $c_i < 0,\ a_i < 0$. In questo caso si può sostituire la variabile x_i con un'altra variabile x_i' ponendo $c_i' = -c_i,\ a_i' = -a_i$ (che risultano quindi entrambi positivi) e si somma al termine noto b il valore $-a_i$. Una volta risolto questo problema trasformato è possibile ottenere il volore ottimo della variabile originaria x_i ponendo $x_i^* = 1 - x_i'^{,*}$.

Esempio 7.4.4 *Risolvere con il metodo del Branch and Bound il seguente problema di knapsack binario:*

$$\begin{array}{ll}\max & 1.2x_1 - x_2 + x_3 + x_4 + 1.5x_5 + 0.3x_6 + 0.3x_7 \\ & 2x_1 + 2x_2 + 2x_3 + x_4 + x_5 + x_6 + 2x_7 \leq 3 \\ & x_i \in \{0,1\}, \quad i = 1, \ldots, 7.\end{array}$$

Innanzitutto si fissa $x_2 = 0$. Si deve risolvere il problema nelle variabili rimanenti. Si riordinano le variabili in modo decrescente rispetto al rapporto peso–ingombro c_k/a_k (rinominandole con y); si ottiene il problema

$$Prob^{(0)} \begin{cases} \max 1.5y_1 + y_2 + 1.2y_3 + y_4 + 0.3y_5 + 0.3y_6 \\ y_1 + y_2 + 2y_3 + 2y_4 + y_5 + 2y_6 \leq 3 \\ y_i \in \{0,1\} \quad i = 1, \ldots, 6. \end{cases}$$

Inizializzazione
Una soluzione del rilassamento lineare è data da

$$\bar{y}^{(0)} = \left(1,\ 1, \frac{3-2}{2},\ 0,\ 0,\ 0\right)^T = \left(1,\ 1, \frac{1}{2},\ 0,\ 0,\ 0\right)^T$$

in corrispondenza della quale abbiamo l'upper bound $U_0 = 3.1$. Siccome $\bar{y}^{(0)}$ non è intera, essa non può essere la soluzione del problema di knapsack binario. Per inizializzare l'ottimo corrente è sufficiente ottenere una qualunque soluzione ammissibile intera. Ne possiamo individuare una approssimando all'intero inferiore la componente frazionaria di $\bar{y}^{(0)}$, ovvero $\tilde{y} = (1,\ 1,\ 0,\ 0,\ 0,\ 0)^T$. Il valore dell'ottimo corrente è quindi $\tilde{z} = c^T\tilde{y} = 2.5$. Siccome risulta $2.5 = \tilde{z} < U_0 = 3.1$, $\tilde{y}$ non è ottimo.
Si separa rispetto alla variabile y_3 e si ottengono i due sottoproblemi:

$$Prob^{(1)} \begin{cases} \max 1.5y_1 + y_2 + y_4 + 0.3y_5 + 0.3y_6 \\ y_1 + y_2 + 2y_4 + y_5 + 2y_6 \leq 3 \\ y_3 = 0 \\ y_i \in \{0,1\} \quad i = 1, \ldots, 6. \end{cases}$$

$$Prob^{(2)} \begin{cases} \max 1.5y_1 + y_2 + 1.2 + y_4 + 0.3y_5 + 0.3y_6 \\ y_1 + y_2 + 2y_4 + y_5 + 2y_6 \leq 1 \\ y_3 = 1 \\ y_i \in \{0,1\} \quad i = 1, \ldots, 6. \end{cases}$$

Si inizializza la lista $\mathcal{L}$:

$$\mathcal{L} = \left\{Prob^{(1)}, Prob^{(2)}\right\}.$$

Prima iterazione
Si estrae $Prob^{(1)}$ dalla lista:

$$\mathcal{L} = \left\{Prob^{(2)}\right\}.$$

Si ottiene:

$$\bar{y}^{(1)} = \left(1,\ 1,\ 0, \frac{1}{2},\ 0,\ 0\right)^T$$

in corrispondenza della quale abbiamo l'upper bound $U_1 = 3$. Poiché $U_1 = 3 > 2.5 = \tilde{z}$ e $\bar{y}^{(1)}$ non è intera si separa rispetto alla variabile y_4 e si ottengono i due sottoproblemi:

$$Prob^{(3)} \begin{cases} \max 1.5y_1 + y_2 + 0.3y_5 + 0.3y_6 \\ y_1 + y_2 + y_5 + 2y_6 \leq 3 \\ y_3 = 0 \\ y_4 = 0 \\ y_i \in \{0,1\} \quad i = 1,\ldots,6. \end{cases}$$

$$Prob^{(4)} \begin{cases} \max 1.5y_1 + y_2 + 1 + 0.3y_5 + 0.3y_6 \\ y_1 + y_2 + y_5 + 2y_6 \leq 1 \\ y_3 = 0 \\ y_4 = 1 \\ y_i \in \{0,1\} \quad i = 1,\ldots,6. \end{cases}$$

e si inseriscono nella lista:

$$\mathcal{L} = \left\{ Prob^{(2)}, Prob^{(3)}, Prob^{(4)} \right\}.$$

Seconda iterazione
Si estrae $Prob^{(2)}$ dalla lista:

$$\mathcal{L} = \left\{ Prob^{(3)}, Prob^{(4)} \right\}.$$

Si ottiene:

$$\bar{y}^{(2)} = (1,\ 0,\ 1, 0,\ 0,\ 0)^T$$

in corrispondenza della quale abbiamo l'upper bound $U_2 = 2.7$. Poiché $U_2 = 2.7 > 2.5 = \tilde{z}$ e $\bar{y}^{(2)}$ è intera, si aggiorna l'ottimo corrente $\tilde{y} = \bar{y}^{(2)}$ e $\tilde{z} = 2.7$ e si chiude il $Prob^{(2)}$.

Terza iterazione
Si estrae $Prob^{(3)}$ dalla lista:

$$\mathcal{L} = \left\{ Prob^{(4)} \right\}.$$

Si ottiene:

$$\bar{y}^{(3)} = (1,\ 1,\ 0, 0,\ 1,\ 0)^T$$

in corrispondenza della quale abbiamo l'upper bound $U_3 = 2.8$. Poiché $U_3 = 2.8 > 2.7 = \tilde{z}$ e $\bar{y}^{(3)}$ è intera, si aggiorna l'ottimo corrente $\tilde{y} = \bar{y}^{(3)}$ e $\tilde{z} = 2.8$ e si chiude il $Prob^{(3)}$.

Quarta iterazione
Si estrae $Prob^{(4)}$ dalla lista:

$$\mathcal{L} = \emptyset.$$

Si ottiene:

$$\bar{y}^{(4)} = (1,\ 0,\ 0,1,\ 0,\ 0)^T$$

in corrispondenza della quale abbiamo l'upper bound $U_4 = 2.5$. Poiché $U_4 = 2.5 < \tilde{z}$ si chiude il $Prob^{(4)}$.

Terza iterazione
A questo punto la lista $\mathcal{L}$ è vuota per cui si ha

$$y^* = \tilde{y} = \bar{y}^{(3)} = (1,\ 1,\ 0,\ 0,\ 1,\ 0)^T$$

con valore ottimo pari a 2.8. Nelle variabili originarie la soluzione ottima è $x^* = (0,\ 0,\ 0,1,\ 1,\ 1,\ 0)^T$. □

Esempio 7.4.5 *Risolvere con il metodo del Branch and Bound il seguente problema di knapsack binario:*

$$\begin{array}{ll} \max & 0.8x_1 + 0.6x_2 + 3x_3 + x_4 + 2.2x_5 + 5x_6 + 0.5x_7 \\ & x_1 + 2x_2 + 3x_3 - 2x_4 + 2x_5 + 2x_6 + 2x_7 \leq 4 \\ & x_i \in \{0,1\}, \quad i = 1,\ldots,7. \end{array}$$

Innanzitutto si fissa $x_4 = 1$. Si deve risolvere il problema nelle variabili rimanenti. Si riordinano le variabili in modo decrescente rispetto al rapporto peso–ingombro c_k/a_k (rinominandole con y) e si aumenta di 2 il valore del termine noto (in seguito alla rimozione della variabile x_4); si ottiene il problema

$$Prob^{(0)} \begin{cases} \max 5y_1 + 2.2y_2 + 3y_3 + 0.8y_4 + 0.6y_5 + 0.5y_6 + 1 \\ 2y_1 + 2y_2 + 3y_3 + y_4 + 2y_5 + 2y_6 \leq 6 \\ y_i \in \{0,1\} \quad i = 1,\ldots,6. \end{cases}$$

Inizializzazione
Una soluzione del rilassamento lineare è data da

$$\bar{y}^{(0)} = \left(1,\ 1, \frac{6-4}{3},\ 0,\ 0,\ 0\right)^T = \left(1,\ 1, \frac{2}{3},\ 0,\ 0,\ 0\right)^T$$

in corrispondenza della quale abbiamo l'upper bound $U_0 = 10.2$. Siccome $\bar{y}^{(0)}$ non è intera, essa non può essere la soluzione del problema di knapsack binario. Per inizializzare l'ottimo corrente è sufficiente ottenere una soluzione ammissibile intera, approssimando all'intero inferiore la componente frazionaria, ovvero $\tilde{y} = (1,\ 1,\ 0,\ 0,\ 0,\ 0)^T$ ed il valore dell'ottimo corrente è $\tilde{z} = 8.2$. Siccome risulta $8.2 = \tilde{z} < U_0 = 10.2$, $\tilde{y}$ non è ottimo.
Si separa rispetto alla variabile y_3 e si ottengono i due sottoproblemi:

$$Prob^{(1)} \begin{cases} \max 5y_1 + 2.2y_2 + 0.8y_4 + 0.6y_5 + 0.5y_6 + 1 \\ 2y_1 + 2y_2 + y_4 + 2y_5 + 2y_6 \leq 6 \\ y_3 = 0 \\ y_i \in \{0,1\} \quad i = 1,\ldots,6. \end{cases}$$

$$Prob^{(2)} \begin{cases} \max 5y_1 + 2.2y_2 + 0.8y_4 + 0.6y_5 + 0.5y_6 + 4 \\ 2y_1 + 2y_2 + y_4 + 2y_5 + 2y_6 \leq 3 \\ y_3 = 1 \\ y_i \in \{0,1\} \quad i = 1,\ldots,6. \end{cases}$$

Si inizializza la lista $\mathcal{L}$:

$$\mathcal{L} = \left\{ Prob^{(1)}, Prob^{(2)} \right\}.$$

Prima iterazione

Si estrae $Prob^{(1)}$ dalla lista:

$$\mathcal{L} = \left\{ Prob^{(2)} \right\}.$$

Si ottiene:

$$\bar{y}^{(1)} = \left(1,\ 1,\ 0,1,\ \frac{1}{2},\ 0 \right)^T$$

in corrispondenza della quale abbiamo l'upper bound $U_1 = 9.3$. Poiché $U_1 = 9.3 > 8.2 = \tilde{z}$ e $\bar{y}^{(1)}$ non è intera, si separa rispetto alla variabile y_5, si ottengono i seguenti due sottoproblemi:

$$Prob^{(3)} \begin{cases} \max 5y_1 + 2.2y_2 + 0.8y_4 + 0.5y_6 + 1 \\ 2y_1 + 2y_2 + y_4 + 2y_6 \leq 6 \\ y_3 = 0 \\ y_5 = 0 \\ y_i \in \{0,1\} \quad i = 1,\ldots,6. \end{cases}$$

$$Prob^{(4)} \begin{cases} \max 5y_1 + 2.2y_2 + 0.8y_4 + 0.5y_6 + 1.6 \\ 2y_1 + 2y_2 + y_4 + 2y_6 \leq 4 \\ y_3 = 0 \\ y_5 = 1 \\ y_i \in \{0,1\} \quad i = 1,\ldots,6. \end{cases}$$

e si inseriscono nella lista:

$$\mathcal{L} = \left\{ Prob^{(2)}, Prob^{(3)}, Prob^{(4)} \right\}.$$

Seconda iterazione

Si estrae $Prob^{(2)}$ dalla lista:

$$\mathcal{L} = \left\{ Prob^{(3)}, Prob^{(4)} \right\}.$$

Si ottiene:

$$\bar{y}^{(2)} = \left(1,\ \frac{1}{2},\ 1,0,\ 0,\ 0\right)^T$$

in corrispondenza della quale abbiamo l'upper bound $U_2 = 10.1$. Poiché $U_2 = 10.1 > 8.2 = \tilde{z}$ e $\bar{y}^{(2)}$ non è intera, si separa rispetto alla variabile y_2 e si ottengono i seguenti due sottoproblemi:

$$Prob^{(5)} \begin{cases} \max 5y_1 + 0.8y_4 + 0.6y_5 + 0.5y_6 + 4 \\ 2y_1 + y_4 + 2y_5 + 2y_6 \leq 3 \\ y_3 = 1 \\ y_2 = 0 \\ y_i \in \{0,1\} \quad i = 1,\ldots,6. \end{cases}$$

$$Prob^{(6)} \begin{cases} \max 5y_1 + 0.8y_4 + 0.6y_5 + 0.5y_6 + 6.2 \\ 2y_1 + y_4 + 2y_5 + 2y_6 \leq 1 \\ y_3 = 1 \\ y_2 = 1 \\ y_i \in \{0,1\} \quad i = 1,\ldots,6. \end{cases}$$

e si inseriscono nella lista:

$$\mathcal{L} = \left\{Prob^{(3)}, Prob^{(4)}, Prob^{(5)}, Prob^{(6)}\right\}.$$

Terza iterazione
Si estrae $Prob^{(5)}$ dalla lista:

$$\mathcal{L} = \left\{Prob^{(3)}, Prob^{(4)}, Prob^{(6)}\right\}.$$

Si ottiene:

$$\bar{y}^{(5)} = (1,\ 0,\ 1,\ 1,\ 0,\ 0)^T$$

in corrispondenza della quale abbiamo l'upper bound $U_5 = 9.8$. Poiché $U_5 = 9.8 > 8.2 = \tilde{z}$ e $\bar{y}^{(5)}$ è intera, si aggiorna l'ottimo corrente $\tilde{y} = \bar{y}^{(5)}$ e $\tilde{z} = 9.8$ e si chiude il $Prob^{(5)}$.

Quarta iterazione
Si estrae $Prob^{(3)}$ dalla lista:

$$\mathcal{L} = \left\{Prob^{(4)}, Prob^{(6)}\right\}.$$

Si ottiene:

$$\bar{y}^{(3)} = \left(1,\ 1,\ 0,\ 1,\ 0,\ \frac{1}{2}\right)^T$$

in corrispondenza della quale abbiamo l'upper bound $U_3 = 9.25$. Poiché $U_3 = 9.25 < \tilde{z}$ si chiude il $Prob^{(3)}$.

Quinta iterazione
Si estrae $Prob^{(4)}$ dalla lista:

$$\mathcal{L} = \left\{ Prob^{(6)} \right\}.$$

Si ottiene:

$$\bar{y}^{(4)} = (1,\ 1,\ 0,0,\ 1,\ 0)^T$$

in corrispondenza della quale abbiamo l'upper bound $U_4 = 8.8$. Poiché $U_4 = 8.8 < \tilde{z}$ si chiude il $Prob^{(4)}$.

Sesta iterazione
Si estrae $Prob^{(6)}$ dalla lista:

$$\mathcal{L} = \emptyset.$$

Si ottiene:

$$\bar{y}^{(6)} = \left(\frac{1}{2},\ 1,\ 1,0,\ 0,\ 0 \right)^T$$

in corrispondenza della quale abbiamo l'upper bound $U_6 = 8.7$. Poiché $U_6 = 8.7 < \tilde{z}$ si chiude il $Prob^{(6)}$.

Settima iterazione
La lista $\mathcal{L}$ è vuota, quindi si ha

$$y^* = \tilde{y} = \bar{y}^{(5)} = (1,\ 0,\ 1,\ 1,\ 0,\ 0)^T$$

con valore ottimo pari a 9.8. Nelle variabili originarie la soluzione ottima è $x^* = (1,\ 0,\ 1,\ 1,\ 0,\ 1,\ 0)^T$. □

8

Ottimizzazione Non Lineare

8.1 FORME STANDARD E PRIME DEFINIZIONI

Un problema di Ottimizzazione consiste nel determinare il valore di un vettore di *variabili di decisione* $x \in \mathbb{R}^n$ che minimizza una *funzione obiettivo* $f : \mathbb{R}^n \to \mathbb{R}$, quando x è vincolato ad appartenere ad un *insieme ammissibile* $\mathcal{F} \subseteq \mathbb{R}^n$; cioè consiste nel problema:

$$\min_{x \in \mathcal{F}} f(x). \tag{8.1.1}$$

Osserviamo subito che un problema di massimo si può sempre ricondurre a un problema di minimo, cambiando di segno la funzione obiettivo. Infatti, i punti di massimo (ove esistano) del problema

$$\max_{x \in \mathcal{F}} \ f(x)$$

coincidono con i punti di minimo del problema

$$\min_{x \in \mathcal{F}} \ -f(x)$$

e risulta: $\max_{x \in \mathcal{F}} f(x) = -\min_{x \in \mathcal{F}}(-f(x))$. In base a tale osservazione ci si può riferire esclusivamente, senza perdita di generalità, a problemi di minimizzazione. Riportiamo la prima definizione utile.

Definition 8.1.1 (Punto di minimo globale) *Un punto $x^* \in \mathcal{F}$ si dice* punto di minimo globale (o assoluto) *di f su $\mathcal{F}$ se risulta:*

$$f(x^*) \leq f(x), \quad \text{per ogni} \quad x \in \mathcal{F},$$

e, in tal caso, si dice che $f(x^*)$ *è* il minimo (o il valore minimo) globale *di* f *su* $\mathcal{F}$*, ossia*

$$f(x^*) = \min_{x \in \mathcal{F}} f(x).$$

Si dice che $x^* \in \mathcal{F}$ *è un* punto di minimo globale stretto *di* f *su* $\mathcal{F}$ *se risulta:*

$$f(x^*) < f(x), \quad \text{per ogni} \quad x \in \mathcal{F},\ x \neq x^*. \quad \square$$

È opportuno mettere in evidenza che, assegnati $\mathcal{F}$ e $f : \mathcal{F} \to R$ potrebbero anche non esistere soluzioni ottime. Una prima possibilità è che l'insieme ammissibile $\mathcal{F}$ sia vuoto; in tal caso non esistono *punti ammissibili* e di conseguenza non esistono soluzioni ottime.
Se $\mathcal{F}$ è non vuoto, possono verificarsi, nel caso generale, le situazioni seguenti:

- la funzione obiettivo è illimitata inferiormente su $\mathcal{F}$ ossia:

  $$\inf_{x \in \mathcal{F}} f(x) = -\infty$$

  e, in tal caso, non esiste un valore minimo di f su $\mathcal{F}$;

- la funzione obiettivo è limitata inferiormente su $\mathcal{F}$ ossia:

  $$\inf_{x \in \mathcal{F}} f(x) > -\infty,$$

  ma tuttavia *non esistono punti di minimo globale* di f su $\mathcal{F}$;

- esistono punti di minimo globale di f su $\mathcal{F}$; in tal caso la funzione obiettivo è necessariamente limitata inferiormente su $\mathcal{F}$ e si ha

  $$\inf_{x \in \mathcal{F}} f(x) = \min_{x \in \mathcal{F}} f(x).$$

Solo nell'ultimo caso, ovviamente, ci si può porre il problema della ricerca di una soluzione ottima.
"Risolvere" un problema di ottimizzazione può quindi significare, in pratica:

- stabilire se l'insieme ammissibile è non vuoto, oppure concludere che non esistono soluzioni ammissibili;

- stabilire se esistono soluzioni ottime, oppure dimostrare che il problema non ammette soluzioni ottime;

- determinare (eventualmente in modo approssimato) una soluzione ottima.

Due casi sono di particolare interesse: - l'insieme ammissibile $\mathcal{F}$ coincide con $\mathbb{R}^n$, cosicchè il Problema (8.1.1) diviene:

$$\min_{x \in \mathbb{R}^n} f(x); \tag{8.1.2}$$

in questo caso si dice che il Problema (8.1.1) e' *non vincolato*. Piu' in generale, il Problema (8.1.1) è non vincolato se $\mathcal{F}$ e' un insieme aperto in $\mathbb{R}^n$.
- l'insieme ammissibile è descritto da *vincoli di disuguaglianza* e/o *vincoli di uguaglianza* sulle variabili di decisione:

$$\mathcal{F} = \{x \in \mathbb{R}^n : g_i(x) \leq 0,\ i = 1, \ldots, p;\ h_j(x) = 0,\ j = 1, \ldots, m\};$$

in questo caso il Problema (8.1.1) diviene:

$$\begin{array}{ll} \min\limits_{x \in \mathbb{R}^n} & f(x) \\ & g(x) \leq 0 \\ & h(x) = 0, \end{array} \tag{8.1.3}$$

con $g : \mathbb{R}^n \to \mathbb{R}^p$ e $h : \mathbb{R}^n \to \mathbb{R}^m$. In questo caso diciamo che il Problema (8.1.1) è *vincolato*.
I problemi di Ottimizzazione vengono anche chiamati problemi di *Programmazione Matematica* quando si vuole mettere l'enfasi sui metodi risolutivi dei problemi stessi.
Notiamo che un problema di Programmazione Lineare è necessariamente vincolato, poiché altrimenti si tratterebbe sempre di un problema illimitato.
Si dice che il Problema (8.1.1) è un problema di *Programmazione Nonlineare* (PNL) se almeno una, tra le funzioni f, $g_i, i = 1, \ldots, p$, $h_j, j = 1, \ldots, m$, del Problema (8.1.2) o del Problema (8.1.3) risulta essere non lineare, rispetto ad almeno una delle componenti del vettore delle variabili di decisione x.
Per il problema vincolato (8.1.3) si assume usualmente che il numero m di vincoli di uguaglianza non sia maggiore del numero n di variabili di decisione, cioè si assume $m \leq n$. Altrimenti, dovendo le n variabili soddisfare m equazioni, l'insieme ammissibile potrebbe risultare vuoto, a meno che alcuni vincoli non siano tra di loro dipendenti, e quindi ridondanti. Una limitazione analoga non vale invece per i vincoli di disuguaglianza.
A volte, tra i vincoli di disuguaglianza, si mettono in esplicita evidenza i *vincoli semplici* sulle variabili, vincoli che esprimono limitazioni sul valore minimo m_i e massimo M_i che una variabile x_i può assumere. In questo caso, il Problema (8.1.3) diviene:

$$\begin{array}{ll} \min\limits_{x \in \mathbb{R}^n} & f(x) \\ & g(x) \leq 0 \\ & h(x) = 0 \\ m & \leq x \leq\ M. \end{array} \tag{8.1.4}$$

Se nel Problema (8.1.4) la variabile x_i non è limitata inferiormente (superiormente), si assume per convenzione che $m_i = -\infty$ ($M_i = +\infty$).
In generale, in problemi di PNL la ricerca di soluzioni globali può risultare difficile, e può avere interesse anche la ricerca di soluzioni di tipo "locale". Per poter

definire il concetto di "punto di minimo locale" occorre introdurre il concetto di *intorno sferico aperto* $\mathcal{S}$ di un punto. In particolare dato x^* , un intorno sferico aperto $\mathcal{S}$ di centro x^* e raggio $\rho > 0$ è definito come

$$\mathcal{S}(x^*, \rho) = \{x \in \mathbb{R}^n : \|x - x^*\| < \rho\}.$$

Possiamo allora introdurre la seguente definizione

Definition 8.1.2 (Punto di minimo locale) *Un punto $x^* \in \mathcal{F}$ si dice* punto di minimo locale (o relativo) *di f su $\mathcal{F}$ se* esiste *un intorno $\mathcal{S}(x^*, \rho)$ di x^* tale che:*

$$f(x^*) \leq f(x), \quad \text{per ogni} \quad x \in \mathcal{F} \cap \mathcal{S}(x^*, \rho),$$

e, in tal caso, si dice che $f(x^*)$ è un minimo locale *di f su $\mathcal{F}$. Si dice che $x^* \in \mathcal{F}$ è un* punto di minimo locale stretto *di f su $\mathcal{F}$ se* esiste *un intorno $\mathcal{S}(x^*, \rho)$ di x^* tale che:*

$$f(x^*) < f(x), \quad \text{per ogni} \quad x \in \mathcal{F} \cap \mathcal{S}(x^*, \rho),\ x \neq x^*. \quad \square$$

È immediato rendersi conto del fatto che *un punto di minimo globale è anche un punto di minimo locale.* Notiamo anche che nella definizione precedente l'intorno $\mathcal{S}(x^*, \rho)$ preso in considerazione *non è necessariamente tutto contenuto* in $\mathcal{F}$. Nel caso particolare in cui $\mathcal{F}$ abbia un interno non vuoto ed esista $\mathcal{S}(x^*, \rho) \subseteq \mathcal{F}$, tale che $f(x^*) \leq f(x)$ per ogni $x \in \mathcal{S}(x^*, \rho)$, diremo che x^* è un punto di minimo locale *non vincolato* di f su $\mathcal{F}$.
Nel seguito diremo che un problema di ottimizzazione è scritto in *forma standard* se è scritto nella forma (8.1.2) nel caso non vincolato, nella forma (8.1.3) nel caso vincolato.

8.2 MODELLI NON LINEARI

Esempio 8.2.1 (Discriminazione del prezzo)

Consideriamo un monopolista che operi su due mercati distinti (ad es. nazionale ed estero) ciascuno con una diversa funzione di domanda. Indichiamo con x_i l'offerta sul mercato $i = 1, 2$ e con $P_i = f_i(x_i)$ la funzione di domanda inversa sul mercato i. Il ricavo derivante dalla vendita di x_i unità di prodotto sul mercato i è $x_i f_i(x_i)$. Supponiamo inoltre che il costo unitario di produzione dipenda solo dal prodotto finale e non dal mercato e sia pari a c. Il problema consiste nel massimizzare il profitto del monopolista. La funzione ricavo è

$$x_1 f_1(x_1) + x_2 f(x_2),$$

mentre il costo è $c(x_1 + x_2)$. Il profitto totale sarà quindi

$$f(x) = x_1 f_1(x_1) + x_2 f(x_2) - c(x_1 + x_2).$$

Naturalmente possono essere presenti vincoli definiti dal processo di produzione del bene e dal mercato su cui il bene viene immesso. Questi vincoli sono specifici del processo di produzione e del mercato e non entriamo qui nel dettaglio. Li indicheremo semplicemente con $x \in S$. Posto $n = 2$ e $x = (x_1, \ldots, x_n)'$ si ha l'insieme ammissibile:

$$\mathcal{F} = \{x \in S : \ x \geq 0\}.$$

Più in generale nel caso di n mercati distinti, il problema di ottimizzazione corrispondente è

$$\begin{array}{ll} \max & \displaystyle\sum_{i=1}^{n} x_i f_i(x_i) - c\sum_{i=1}^{n} x_i \\ & x \in \mathcal{F}. \end{array}$$

Molto spesso le funzioni $f_i(x_i)$ hanno un andamento lineare del tipo

$$f_i(x_i) = a_i - m_i x_i \qquad \text{con } m_i > 0.$$

La funzione ricavo risulta essere quindi una funzione quadratica del tipo

$$\sum_{i=1}^{n} x_i(a_i - m_i x_i) = -\frac{1}{2} x'Qx + a'x$$

con Q matrice diagonale definita positiva con elementi diagonali $2m_i > 0$ e $a = (a_1 \ldots, a_n)'$. Il problema di discriminazione del prezzo diventa un problema di programmazione quadratica del tipo

$$\begin{array}{ll} max & -\frac{1}{2} x'Qx + (a + c)'x \\ & x \in \mathcal{F}. \end{array}$$

Example 8.2.2 (Problemi di approssimazione ai Minimi quadrati)

Supponiamo che siano noti n punti del piano (x_i, y_i) con $i = 1, \ldots, n$ che possono corrispondere ai valori una funzione continua $\phi : R \to R$ ottenuti per via sperimentale o con misurazioni.
Si vuole approssimare la funzione $y = \phi(x)$ per mezzo di un polinomio di primo grado (ovvero una retta) del tipo $y = mx + q$.
Si definiscono gli errori

$$e_i(x_i) = y_i - (mx_i + q), \quad i = 1, \ldots, n,$$

e si può considerare il problema di ottimizzazione non vincolata, (noto anche come problema di *curve fitting*)

$$\min \|e(x)\|^2 = \sum_{i=1}^{n} e_i(x_i)^2 = \sum_{i=1}^{n} (mx_i + q - y_i)^2$$

Si osservi che la funzione obiettivo è quadratica.
Più in generale, si può considerare il problema

$$\begin{array}{ll} min & \|e(x)\|^\alpha, \\ e \in \mathbb{R}^n & \end{array}$$

in cui $\|\cdot\|$ è una norma su $\mathbb{R}^n$ e $\alpha > 0$. I casi più comuni sono quelli in cui si richiede di minimizzare una norma ℓ_p con $p \geq 1$, o, equivalentemente, la $p-$ma potenza di una norma ℓ_p:

$$f(x) = \sum_{i=1}^{m} |e_i(x)|^p,$$

oppure la norma ℓ_∞:

$$f(x) = \max_{1 \leq i \leq m} |e_i(x)|.$$

Problemi differenti si ottengono ovviamente in corrispondenza ad altre scelte delle funzioni approssimanti che, nel caso più generale, possono dipendere in modo non lineare dai parametri incogniti.

Example 8.2.3 (Problemi di progettazione)

Un'industria chimica intende utilizzare della lamiera metallica residua, costruendo un serbatoio scoperto da adibire all'immagazzinamento di un prodotto liquido. La lamiera può essere tagliata e saldata a piacere, è disponibile per complessivi $150m^2$ e la si vuole utilizzare tutta. Il serbatoio deve essere contenuto in un capannone a pianta quadrata, con lato di $10m$, e con tetto spiovente dall'altezza di 4.5 all'altezza di $3m$. Per semplicità di progetto, si assume che il serbatoio abbia la forma di un prisma retto, con base quadrata.
Determiniamo le dimensioni del serbatoio, in modo da massimizzare il volume del liquido che vi può essere contenuto.
Soluzione. Le variabili di decisione sono x_1 la misura del lato di base del serbatoio e x_2 la misura dell'altezza. Il volume del serbatoio è

$$V = A_b \cdot h = x_1^2 x_2.$$

Per quanto riguarda i vincoli abbiamo:
vincoli di disponibilità: deve essere usata esattamente una quantità di lamiera pari a 150 mq. Quindi, poichè il serbatoio è scoperto la quantità di lamiera necessaria è pari all'area di base A_b e alle 4 superfici laterali. Quindi $A_b + 4A_l = 150$ che corrisponde a

$$x_1^2 + 4x_1x_2 = 150.$$

Vincoli di spazio: il serbatoio deve essere collocato nel capannone, quindi

$$x_1 \leq 10;$$

per quanto riguarda x_2, poiché l'altezza del capannone è variabile da 4.5 a 3 metri e il lato è 10 m. abbiamo

$$x_2 \leq -0.15x_1 + 4.5.$$

Vincoli di non negatività: si tratta di lunghezze e quindi

$$x_1 \geq 0, \quad x_2 \geq 0.$$

Complessivamente possiamo scrivere:

$$\begin{array}{ll} \max & x_1^2 x_2 \\ & x_1^2 + 4x_1x_2 = 150 \\ & x_1 \leq 10 \\ & x_2 + 0.15x_1 \leq 4.5 \\ & x_1 \geq 0, \quad x_2 \geq 0 \end{array}$$

Example 8.2.4 (Problemi di gestione del rischio)

Un'industria dolciaria si rifornisce di zucchero acquistandolo in tre diversi paesi produttori, che indicheremo con A, B, C. I prezzi di acquisto dello zucchero nei tre paesi sono diversi, e, in ciascuno dei tre paesi, il prezzo subisce delle variazioni aleatorie, dovute alle variazioni dei cambi, alla maggiore o minore produzione stagionale, alle variazioni salariali ecc.. Per tener conto di queste variazioni aleatorie i prezzi per tonnellata sono caratterizzati mediante i valori medi p_A, p_B, p_C, e le varianze ${\sigma^2}_A, {\sigma^2}_B, {\sigma^2}_C$. Ovviamente se i prezzi non subissero variazioni aleatorie, converrebbe acquistare tutta la materia prima del paese che offre il prezzo minore. In presenza delle incertezze sui prezzi, l'industria deve fare riferimento, nella propria programmazione degli acquisti, ad un prezzo medio stimato pM, e acquista lo zucchero nei tre paesi secondo proporzioni tali da realizzare questo prezzo medio stimato. Indicando con x_1, x_2, x_3 le variabili che rappresentano, per ogni tonnellata di zucchero, le frazioni acquistate rispettivamente nei paesi A, B, C deve risultare:

$$\begin{array}{l} x_1 p_A + x_2 p_B + x_3 p_C = p_M \\ x_1 + x_2 + x_3 = 1; \end{array}$$

il primo vincolo infatti esprime il fatto che il prezzo medio stimato dello zucchero sia pari p_M per ogni tonnellata, e il secondo vincolo esprime il fatto che le variabili di decisioni sono frazioni di una quantità unitaria (la tonnellata).
Come obiettivo della programmazione degli acquisti, l'industria assume quello di minimizzare il rischio che il prezzo medio effettivo per tonnellata risultante dagli acquisti effettuati secondo le frazioni x_1, x_2, x_3, differisca da quello stimato p_M; ciò per rendere il più possibile certe, in termini di costi di produzione, le conseguenze della programmazione degli acquisti. Una ovvia misura di questo

rischio è data dalla varianza del prezzo medio ${\sigma^2}_M$, quantità data in questo caso dalla espressione

$${\sigma^2}_M = {x^2}_1{\sigma^2}_A + {x^2}_2{\sigma^2}_B + {x^2}_3{\sigma^2}_C.$$

Assumiamo che, nell'unità monetaria adottata risulti:

$$\begin{array}{ccc} p_A = 4, & p_B = 5.5 & p_C = 6 \\ \sigma_A^2 = 1, & \sigma_B^2 = 0.8, & \sigma_C^2 = 0.5 \end{array}$$

e che l'industria abbia programmato sulla base di un prezzo medio $P_M = 5$.
Soluzione. Si tratta di un problema di programmazione quadratica

$$\begin{array}{ll} \min & x_1^2 + 0.8x_2^2 + 0.5x_3^2 \\ & x_1 + x_2 + x_3 = 1 \\ & 4x_1 + 5.5x_2 + 6x_3 = 5 \end{array}$$

Example 8.2.5 (Problemi avanzati di localizzazione)

Una compagnia petrolifera si rifornisce di greggio in tre città portuali, che indicheremo con A, B, C. Il porto B è ubicato $300Km$ a est e $400Km$ a nord del porto A, e il porto C è ubicato $400Km$ ad est e $100Km$ a sud del porto B. La compagnia intende costruire una nuova raffineria per il greggio, e intende localizzare la nuova raffineria in modo da minimizzare la quantità totale di tubi occorrenti per collegare la raffineria ai porti. Determiniamo la formulazione di questo problema di localizzazione.
Supponiamo inoltre che non sia possibile situare la raffineria né a sud del porto A , né entro un raggio di 360 Km. dallo stesso.
Soluzione. Scegliamo un sistema di riferimento con il porto A nell'origine. Il porto B ha quindi coordinate (300,400), mentre il porto C (700,300). La raffineria si trova nella posizione incognita (x_1, x_2) e l'obiettivo è minimizzare la distanza, cioè

$$\min\ (x_1^2 + x_2^2)^{1/2} + ((x_1 - 300)^2 + (x_2 - 400)^2)^{1/2} + ((x_1 - 700)^2 + (x_2 - 300)^2)^{1/2}$$

Inoltre abbiamo i vincoli sulle coordinate della raffineria:

$$x_2 \geq 0$$

$$x_1^2 + x_2^2 \geq 360^2.$$

8.3 PROBLEMI DI OTTIMIZZAZIONE CONVESSI

Tra i problemi di Ottimizzazione sono di particolare interesse i cosiddetti problemi *convessi*.

Ricordiamo che un insieme $\mathcal{C} \subseteq \mathbb{R}^n$ è convesso se, presi comunque due punti $y, z \in \mathcal{C}$, risulta che anche $[y, z] \subseteq \mathcal{C}$, avendo denotato con $[y, z]$ il segmento che congiunge y e z, segmento dato dai punti x ottenuti come:

$$x = (1 - \beta)y + \beta z, \quad \beta \in [0, 1].$$

Si verifica facilmente che l'intersezione di un numero finito di insiemi convessi è un insieme convesso.

Definition 8.3.1 (Funzione (strettamente) convessa) *Una funzione $v(x)$ si dice* convessa *su un insieme convesso $\mathcal{C}$ se, presi comunque due punti $y, z \in \mathcal{C}$ risulta che:*

$$v((1 - \beta)y + \beta z) \leq (1 - \beta)v(y) + \beta v(z), \quad \beta \in [0, 1]. \tag{8.3.1}$$

La funzione $v(x)$ si dice strettamente convessa *se, per $y, z \in \mathcal{C}$, $y \neq z$, risulta*

$$v((1 - \beta)y + \beta z) < (1 - \beta)v(y) + \beta v(z), \quad \beta \in (0, 1).$$

□

Una funzione $v(x)$ si dice *(strettamente) concava* su un insieme convesso $\mathcal{C}$ se la funzione $-v(x)$ è (strettamente) convessa su $\mathcal{C}$.
A partire dalla definizione di funzione (strettamente) convessa, è possibile dedurre due importanti proprietà che riguardano le derivate, rispettivamente prime e seconde.

Proposition 8.3.2 *Una funzione $v(x)$ è convessa su un insieme convesso $\mathcal{C}$ se, e solo se, per ogni $y, z \in \mathcal{C}$ risulta:*

$$v(z) \geq v(y) + \nabla v(y)^T (z - y);$$

$v(x)$ è strettamente convessa su $\mathcal{C}$ se, e solo se, per ogni $y, z \in \mathcal{C}$, $y \neq z$, risulta:

$$v(z) > v(y) + \nabla v(y)^T (z - y).$$

Proposition 8.3.3 *Una funzione $v(x)$ è convessa su un insieme convesso $\mathcal{C}$ se, e solo se, per ogni $x \in \mathcal{C}$ risulta:*

$$\frac{1}{2} y^T \nabla^2 v(x) y \geq 0, \quad \text{per ogni } y \in \mathbb{R}^n;$$

inoltre, se risulta

$$\frac{1}{2} y^T \nabla^2 v(x) y > 0, \quad \text{per ogni } y \in \mathbb{R}^n,$$

$v(x)$ è strettamente convessa su $\mathcal{C}$.

Si noti che l'ultima condizione è solo sufficiente: ad esempio, la funzione $v(x) = x^4$ è strettamente convessa in $\mathcal{C} = \mathbb{R}^n$, ma non soddisfa la condizione, in quanto in

$x = 0$ la derivata seconda si annulla. Nel caso di funzioni quadratiche le condizioni del teorema diventano invece necessarie e sufficienti, ovvero si ha

Proposition 8.3.4 *Una funzione quadratica del tipo* $q(x) = \frac{1}{2}x^T Qx + c^T x$ *è convessa su un insieme convesso* $\mathcal{C}$ *se e solo se risulta* Q *semidefinita positiva. Inoltre,* $q(x)$ *è strettamente convessa su* $\mathcal{C}$ *se e solo se risulta* Q *definita positiva.*

Definition 8.3.5 (Problema (strettamente) convesso) *Si dice che il Problema (8.1.1) è un problema di ottimizzazione* convesso *se l'insieme ammissibile* $\mathcal{F}$ *è un insieme convesso e la funzione obiettivo* $f(x)$ *è una funzione convessa su* $\mathcal{F}$*. Se la funzione obiettivo è strettamente convessa su* $\mathcal{F}$*, il problema si dice* strettamente convesso*.*

I problemi di ottimizzazione convessi sono di particolare importanza per due motivi. Il primo è che la grande maggioranza dei problemi di ottimizzazione che si incontrano nella pratica sono convessi (vedi la Programmazione Lineare). Il secondo è che la convessità induce alcune proprietà che semplificano l'analisi e la soluzione di un problema convesso.

Proposition 8.3.6 *Un problema di ottimizzazione convesso o non ha soluzione, o ha solo soluzioni globali; non può avere soluzioni esclusivamente locali.*

Dimostrazione. (Facoltativa) La dimostrazione è per assurdo. Ammettiamo che x^* sia una soluzione locale, ma non globale, di un problema convesso $min_{x\in\mathcal{C}} f(x)$. Allora esisterà un altro punto $z \in \mathcal{C}$ tale che $f(z) < f(x^*)$. Consideriamo il segmento $[x^*, z]$: per la convessità di f, si ha:

$$f((1-\beta)x^* + \beta z) \leq (1-\beta)f(x^*) + \beta f(z) = f(x^*) + \beta(f(z) - f(x^*)), \text{per ogni } \beta \in [0,1].$$

Il termine $\beta(f(z) - f(x^*))$ risulta < 0, per ogni $\beta \in (0, 1]$, e si annulla solo per $\beta = 0$. Quindi, poichè in ogni punto $x \in (x^*, z]$ risulta $f(x) < f(x^*)$, non esiste nessun intorno di raggio $\rho > 0$ in cui x^* può soddisfare la definizione di minimo locale. □

Una seconda proprietà notevole è espressa dalla seguente proposizione:

Proposition 8.3.7 *In un problema di ottimizzazione strettamente convesso la soluzione globale, se esiste, è unica.*

Dimostrazione. Anche questa dimostrazione è per assurdo, e si lascia per esercizio. Sia $f(x)$ strettamente convessa. Si ammetta, per assurdo, che x^* e x^{**}, con $x^* \neq x^{**}$, siano due soluzioni globali di un problema convesso, e se ne traggano le conseguenze.◁

Riconoscere che un problema di ottimizzazione è convesso fornisce quindi importanti informazioni qualitative sulle sue soluzioni. Per riconoscere che un problema

è convesso dobbiamo verificare che $\mathcal{F}$ è un insieme convesso, e che $f(x)$ è convessa su $\mathcal{F}$, il che non è sempre facile. Ci aiuta la seguente proposizione, di facile utilizzo, e che è verificata molto spesso nella pratica. La proposizione fornisce una condizione sufficiente affinchè un problema sia convesso.

Proposition 8.3.8 *Si assuma che nel Problema (8.1.3) la funzione obiettivo $f(x)$ sia una funzione convessa in $\mathbb{R}^n$, che i vincoli di disuguaglianza siano dati da funzioni $g_i(x)$ convesse in $\mathbb{R}^n$, e che i vincoli di uguaglianza siano dati da funzioni affini del tipo $a_j^T x - b_j$. Allora il Problema (8.1.3) è convesso.*

Sommario