Assignment 2 - Integer Linear Programming

Andrea Piancone

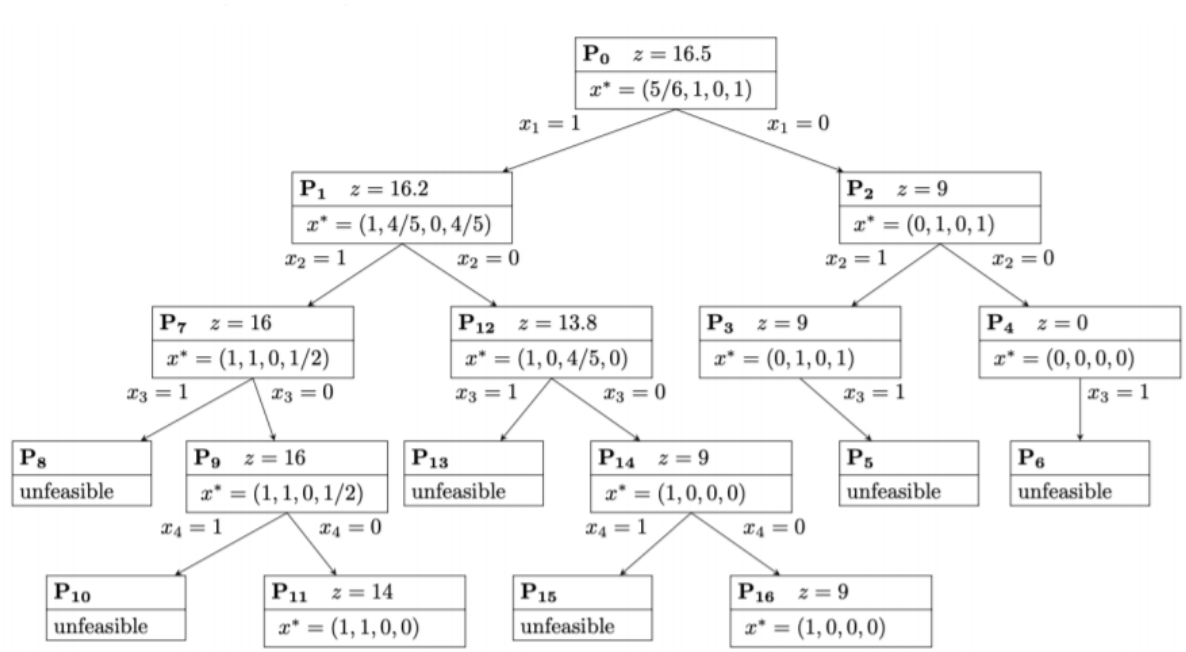
Contents

1 Problem 1

Consider the following ILP:

$$
\begin{aligned}
&\max \quad 9x_1 + 5x_2 + 6x_3 + 4x_4 \\
&\text{s.t.} \\
&6x_1 + 3x_2 + 5x_3 + 2x_4 \leq 10, \\
&x_3 + x_4 \leq 1, \\
&-x_1 + x_3 \leq 0, \\
&-x_2 + x_4 \leq 0 \\
&x_1, x_2, x_3, x_4 \in \{0, 1\}
\end{aligned}
$$

The following tree represents the solutions of all possible relaxations of the problem in which no sub-problem has been excluded (fathoming).



Suppose that the Branch and Bound (BB) algorithm applies to this problem. Also, let's suppose that the algorithm visits the sub-problems in the following order P0 , P1 , . . . , P16. Clearly, the algorithm does not visit all nodes.

1.1 Esercizio 1

Determine the nodes that will be visited by the BB algorithm and for each of them get the upper and lower limit deduced by the algorithm in the execution

L'algoritmo di Branch & Bound, per giungere alla soluzione ottimale segue il seguente percorso:

1. Nel nodo P_0, il problema rilassato porta ad un valore della funzione obiettivo pari 16.5, la miglior valore della funzione attualmemnte noto dall'algoritmo (che d'ora in avanti si indicherà con Z_{best}) in corrispondenza di una soluzione intera è $-\infty$. Poichè le variabili decisionali non sono ancora intere, l'algoritmo procede.
2. L'algoritmo passa ad ispezionare il nodo P_1. In tale nodo, il problema rilassato porta ad una valore della funzione obiettivo pari 16.2. Anche in questo nodo Z_{best} è $-\infty$. In questo nodo, le variabili decisionali in corrispondenza della soluzione ottimale non sono intere.
3. L'algoritmo ispeziona il nodo P_2. In P_2, la soluzione ottimale rispetta i vincoli di interezza, ed il valore della funzione obiettivo è pari 9. In questo nodo quindi, estremo superiore ed inferiore coincidono e sono pari a 9. Di conseguenza, Z_{best}, si aggiorna e diventa pari a 9. L'algoritmo non ispeziona più i nodi derivanti da P_2, in quanto nessun altro sottoproblema che si può estrarre da P_2, può consentire di raggiungre un valore della funzione obiettivo superiore a 9, che attualmente è la miglior soluzione intera del problema.
4. L'algoritmo, tuttavia non è ancora giunto a convergenza in quanto in P_1 la soluzione non è ancora intera. Pertanto, l'algoritmo ispeziona i nodi che derivano da P_1. L'algoritmo ora ispeziona P_7. In P_7, la soluzione non è conforme ai vincoli di interezza del problema, e la funzione obiettivo ora è pari 16 e rappresenta l'estremo superiore del nodo. La miglior soluzione intera raggiungibile (che rappresenta l'estremo inferiore in tale nodo) è ancora pari a 9. Poichè le variabili decisionali non sono intere, l'algoritmo procede.
5. L'algoritmo ispeziona il nodo P_8, tuttavia è caratterizzato da una regione non ammissibile, quindi non è presente nè un estremo superiore nè un estremo inferiore.
6. L'algoritmo ora ispeziona P_9, in P_9, la funzione obiettivo assume un valore pari a 16 (estremo superiore), mentre la miglior soluzione intera attualmente nota dall'algorimto (estremo inferiore) è pari a 9. Poichè

le variabili decisionali non sono intere, l'algoritmo deve procedere.

7. L'algoritmo ispeziona il nodo P_{10}, tuttavia è caratterizzato da una regione non ammissibile, quindi non è presente nè un estremo superiore nè un estremo inferiore.
8. L'algoritmo ora esamina il nodo P_{11}. In P_{11}, le variabili decisionali sono intere e la funzione obiettivo assume un valore pari a 14. Quindi la miglior soluzione intera si aggiorna diventando pari a 14. In questo nodo estremo superiore ed inferiore coincidono con il valore di 14.
9. L'algoritmo ispeziona il nodo P_{12}. In P_{12}, l'algoritmo si interrpome in quanto la soluzione del problema rilassato porta ad un valore ottimale della funzione obiettivo pari 1 13.8, quando però la soluzione intera attualmente nota del problema è pari a 14. Quindi, l'algoritmo non può trovare un altro nodo che garantisce una soluzione conforme ai vincoli di interezza, migliore rispetto a quella trovata in P_{11}.

Variabile	Valore
x_1	1
x_2	1
x_3	0
x_4	0

Con un valore della funzione obiettivo pari a 14.

In sintesi, l'algoritmo compie il seguente percorso:

Nodo	Lower Bound	Upper Bound
P_0	$-\infty$	16.5
P_1	$-\infty$	16.2
P_2	9	9
P_7	9	16
P_8	na	na
P_9	9	16
P_{10}	na	na
P_{11}	14	14
P_{12}	13.8	14

1.2 Esercizio 2

Solve the problem with an ILP solver and check the value of the objective function matches the one found at point 1

```
library(lpSolveAPI)
library(dplyr)
library(pander)
library(tidyr)
```

Si definisce il problema come un problema di massimo e si definisce la funzione obiettivo.

```
model <- make.lp(0, 4)
lp.control(model, sense = "max")
set.objfn(model, obj = c(9, 5, 6, 4))
```

Si introducono i vincoli all'interno del modello

```
add.constraint(model,
               xt = c(6, 3, 5, 2),
               type = "<=",
```

```
                rhs = 10,
                indices = c(1, 2, 3, 4))
```

```
add.constraint(model,
                xt = c(1,1 ),
                type = "<=",
                rhs = 1,
                indices = c(3, 4))
```

```
add.constraint(model,
                xt = c(-1, 1),
                type = "<=",
                rhs = 0,
                indices = c(1,3))
```

```
add.constraint(model,
                xt = c(-1, 1),
                type = "<=",
                rhs = 0,
                indices = c(2, 4))
```

Si impone che le variabili decisionali debbano essere binarie:

```
set.type(model, c(1,2,3,4), "binary")
```

```
model
```

```
## Model name:
##            C1   C2   C3   C4
## Maximize    9    5    6    4
## R1          6    3    5    2  <=  10
## R2          0    0    1    1  <=   1
## R3         -1    0    1    0  <=   0
## R4          0   -1    0    1  <=   0
## Kind      Std  Std  Std  Std
## Type      Int  Int  Int  Int
## Upper       1    1    1    1
## Lower       0    0    0    0
```

Il problema è stato costruito correttamente, quindi si procede a risolvere il problema di programmazione intera.

```
solve(model)
```

```
## [1] 0
```

Si estrae il valore della funzione obiettivo, in corrispondenza della soluzione ottimale.

```
get.objective(model)
```

```
## [1] 14
```

La funzione obiettivo ha un valore ottimale pari a 14

Si estrae ora il valore delle variabili decisonali in corrispondenza della soluzione ottimale.

```
get.variables(model)
```

```
## [1] 1 1 0 0
```

In corrispondenza della soluzione ottimale si ha:

Variabile	Valore
x_1	1
x_2	1
x_3	0
x_4	0

Il valore della funzione obiettivo in corrispondenza della soluzione ottimale, coincide con quanto ricavato al punto precedente, così come coincidono i valori delle variabili decisionali.

2 Problem 2

SunNet is a residential Internet Service Provider (ISP) in the central Florida area. Presently, the company operates one centralized facility that all of its clients call into for Internet access.

To improve service, the company is planning to open three satellite offices in the cities of Pine Hills, Eustis, and Sanford. The company has identified five different regions to be serviced by these three offices. The following table summarizes the number of customers in each region, the service capacity at each office, and the monthly average cost per customer for providing the service to each region from each office. Table entries of "n.a." indicate infeasible region-to-service center combinations.

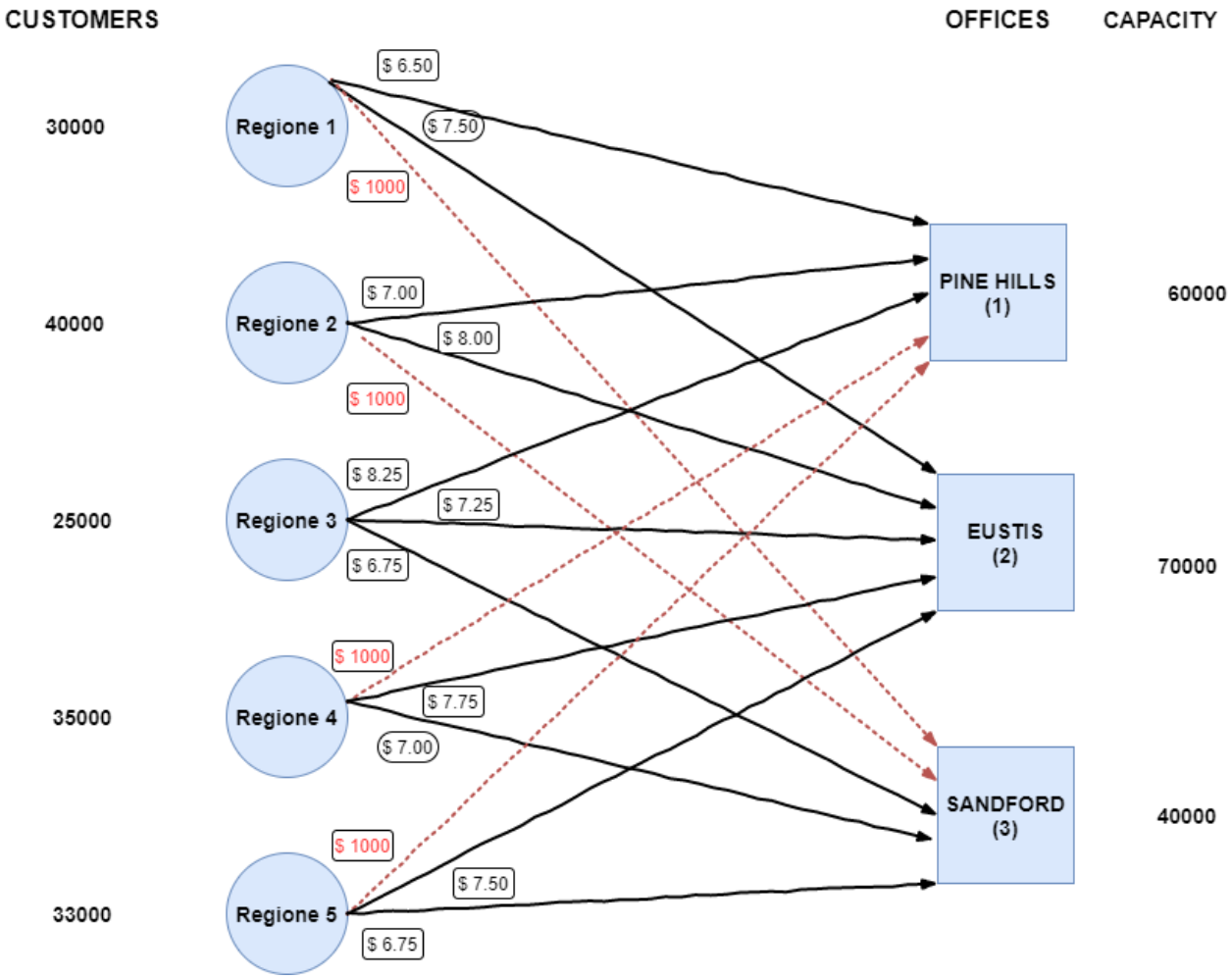
SunNet would like to determine how many customers from each region to assign to each service center to minimize the total cost.

Region	Pine Hills	Eustis	Sanford	Customers
1	\$6.50	\$7.50	n.a.	30,000
2	\$7.00	\$8.00	n.a.	40,000
3	\$8.25	\$7.25	\$6.75	25,000
4	n.a.	\$7.75	\$7.00	35,000
5	n.a.	\$7.50	\$6.75	33,000
Capacity	60,000	70,000	40,000	

1) Draw a network flow model to represent this problem.
2) Implement your model and solve it.
3) What is the optimal solution?

2.1 Grafo del problema

Lo scopo del problema è assegnare i clienti delle 5 regioni, ai 3 nuovi uffici che la società vuole aprire, minimizzando il costo totale. Si tratta dunque di un problema di assegnamento rappresentato dal seguente grafo:



Ogni arco del grafo, rappresenta una variabile decisionale, i pesi degli archi rappresentano i coefficienti della funzione obiettivo (ossia il costo medio mensile per cliente) ed i clienti associati ad ogni regione e la capacità associata ad ogni ufficio, rappresentano i vincoli del problema. Inotre, è importante specificare che gli archi rappresentati da una linea rossa tratteggiata sono degli archi aggiunti artificialmente all'interno del problema per collegare i nodi *Region 1* e *Region 2* con il nodo *Eustis* e i nodi *Region 4* e *Region 5* con il nodo *Pine Hills*. Poichè tali archi sono artificiali e non reali (questi nodi non sono effettivamente collegati tra loro), si è assegnato un peso, ossia un costo mensile medio per cliente, arbitrariamente elevato pari 1000 dollari, affinchè tali archi diventino proibitivi e che quindi non siano scelti dalla soluzione ottimale.

Di seguito si formalizza il problema.

2.2 Variabili decisionali

Lo scopo del problema è assegnare i clienti delle 5 regioni ad uno dei 3 nuovi uffici che la società desidera aprire, minimizzando il costo totale. Ogni combinazione regione-cliente, rappresenta una variabile decisionale $x_{i,j}$, dove $i = 1, ..., 5$ rappresenta l'indice della regione, $j = 1, ..., 3$, rappresenta l'indice dell'ufficio. Quindi le 15 variabili decisionali sono:

- $x_{1,1}$: numero di clienti della regione 1 assegnati all'ufficio di Pine Hills;
- $x_{1,2}$: numero di clienti della regione 1 assegnati all'ufficio di Eustis;
- $x_{1,3}$: numero di clienti della regione 1 assegnati all'ufficio di Sandford;
- $x_{2,1}$: numero di clienti della regione 2 assegnati all'ufficio di Pine Hills;
- $x_{2,2}$: numero di clienti della regione 2 assegnati all'ufficio di Eustis;

- $x_{2,3}$: numero di clienti della regione 2 assegnati all'ufficio di Sandford;
- $x_{3,1}$: numero di clienti della regione 3 assegnati all'ufficio di Pine Hills;
- $x_{3,2}$: numero di clienti della regione 3 assegnati all'ufficio di Eustis;
- $x_{3,3}$: numero di clienti della regione 3 assegnati all'ufficio di Sandford;
- $x_{4,1}$: numero di clienti della regione 4 assegnati all'ufficio di Pine Hills;
- $x_{4,2}$: numero di clienti della regione 4 assegnati all'ufficio di Eustis;
- $x_{4,3}$: numero di clienti della regione 4 assegnati all'ufficio di Sandford;
- $x_{5,1}$: numero di clienti della regione 5 asseganti all'ufficio di Pine Hills;
- $x_{5,2}$: numero di clienti della regione 5 asseganti all'ufficio di Eustis;
- $x_{5,3}$: numero di clienti della regione 5 assegnati all'ufficio di Sandford.

2.3 Funzione obiettivo

La funzione del problema, oggetto di minimizzazione è data dalla seguente combinazione lineare:

$$\begin{aligned} min\ z = 6.5x_{1,1} + 7.5x_{1,2} + 1000x_{1,3} \\ +7x_{2,1} + 8x_{2,2} + 1000x_{2,2} \\ +8.25x_{3,1} + 7.25x_{3,2} + 6.75x_{3,3} \\ +1000x_{4,1} + 7.75x_{4,2} + 7x_{4,3} \\ +1000x_{5,1} + 7.5x_{5,2} + 6.75x_{5,3} \end{aligned}$$

2.4 Vincoli

Si definiscono i vincoli del problema:

Vincoli di capacità dei tre uffici

$$\begin{cases} \sum_{i=1}^{5} x_{i,1} \leq 60000 \\ \sum_{i=1}^{5} x_{i,2} \leq 70000 \\ \sum_{i=1}^{5} x_{1,3} \leq 40000 \end{cases}$$

Supply constraints

$$\begin{cases} \sum_{j=1}^{3} x_{1,j} = 30000 \\ \sum_{j=1}^{3} x_{2,j} = 40000 \\ \sum_{j=1}^{3} x_{3,j} = 25000 \\ \sum_{j=1}^{3} x_{4,j} = 35000 \\ \sum_{j=1}^{3} x_{5,j} = 33000 \end{cases}$$

Vincolo di non negatività

$x_{i,j} \geq 0\ per\ i = 1, ...5\ j = 1, .., 3$

Vincolo di interezza

Poichè le variabili decisionali rappresentano i clienti delle cinque regioni tra i tre uffici che la società è intenzionata ad aprire, si deve imporre che le variabili decisioanli siano intere.

$x_{i,j} \in N\ \forall i, j$

2.5 Implementazione del modello

Si definisce che il problema è di minimo e si introduce la funzione obiettivo, oggetto di minimizzazione, che viene risolto tramite l'algoritmo di Branch & Bound essendo un problema di programmazione intera.

```
net.model <- make.lp(0, 15)
lp.control(net.model, sense = "min")
set.objfn(net.model, obj = c(6.5, 7.5, 1000,
                             7, 8, 1000,
                             8.25, 7.25, 6.75,
                             1000, 7.75, 7,
                             1000, 7.5, 6.75))
```

Vincoli del problema

Si introducono i vincoli di capacità

```
add.constraint(net.model,
               xt = c(1,1,1,1,1),
               type = "<=",
               rhs = 60000,
               indices = c(1, 4, 7, 10, 13))
```

```
add.constraint(net.model,
               xt = c(1,1,1,1,1),
               type ="<=",
               rhs = 70000,
               indices = c(2, 5, 8, 11, 14)
)
```

```
add.constraint(net.model,
               xt = c(1,1,1,1,1),
               type = "<=",
               rhs = 40000,
               indices = c(3,6,9,12,15))
```

Si aggiungono i supply constraints

```
add.constraint(net.model,
               xt = c(1,1,1),
               type = "=",
               rhs = 30000,
               indices = c(1,2,3))
```

```
add.constraint(net.model,
               xt = c(1,1,1),
               type = "=",
               rhs = 40000,
               indices = c(4,5,6))
```

```
add.constraint(net.model,
               xt = c(1,1,1),
               type = "=",
               rhs = 25000,
               indices = c(7,8,9))
```

```
add.constraint(net.model,
               xt = c(1,1,1),
               type = "=",
               rhs = 35000,
               indices = c(10,11,12))
```

```
add.constraint(net.model,
               xt = c(1,1,1),
               type = "=",
               rhs = 33000,
               indices = c(13, 14, 15))
```

Si introduce la non negatività delle variabili decisionali

```
set.bounds(net.model, lower=c(rep(0, 15)))
```

Infine, si introduce la condizione di interezza delle variabili decisionali.

```
set.type(net.model, c(seq(1, 15)), type = "integer")
```

2.6 Risoluzione del problema

```
solve(net.model)
```

```
## [1] 0
```

Si estrae il valore ottimale della funzione obiettivo:

```
get.objective(net.model)
```

```
## [1] 1155000
```

Il costo in corrispondenza della soluzione ottimale è pari 1155000$. Si estrae il valore delle variabili decisionali, in corrispondenza della soluzione ottimale.

```
get.variables(net.model)
```

```
##  [1] 20000 10000     0 40000     0     0     0 25000     0     0     0 35000
## [13]     0 28000  5000
```

Il valore ottimale delle variabili decisionli, è riportao nella seguente tabella:

Variabile	Valore
$x_{1,1}$	20000
$x_{1,2}$	10000
$x_{2,1}$	40000
$x_{3,2}$	25000
$x_{4.3}$	35000
$x_{5,2}$	28000
$x_{5,3}$	5000

Tutte le altre variabili decisionali, hanno un valore pari a zero in corrispondenza della soluzione ottimale.

Al fine di minimizzare il costo è necessario seguire la seguente regola:

- assegnare 20000 clienti della regione 1 all'ufficio di Pine Hills;
- assegnare 10000 clienti della regione 1 all'ufficio di Eustis;
- assegnare 40000 clienit della regione 2 all'ufficio di Pine Hills;
- assegnare 25000 clienti della regione 3 all'ufficio di Eustis;
- assegnare 35000 clienti della regione 4 all'ufficio di Sandford;
- asseganre 28000 clienti della regione 5 all'ufficio di Eustis.
- assegnare 5000 clienti della regione 5 all'ufficio di Sandford.

Notare come, le variabili decisionali $x_{1,3}$, $x_{2,3}$, $x_{4,1}$ e $x_{5,1}$, relative agli archi aggiunti artificialmente con un

costo estremamente alto sono nulle, in coformità a quanto si voleva ottenere.

Una volta aver individuato la soluzione del problema, si stabilisce se tutti gli uffici sono pienamente utilizzati oppure se parte della loro capacità è inutilizzata.

```
get.constraints(net.model)
```

```
## [1] 60000 63000 40000 30000 40000 25000 35000 33000
```

Si osserva come l'ufficio della città di Eustis non è pienamente utilizzato, infatti tale ufficio potrebbe essere ancora in grado di gestire 7000 clienti.

Infine, si procede ad individuare quali sono le variabili di base in corrispondenza della soluzione ottimale.

```
get.basis(net.model)
```

```
## [1] -10  -2 -16  -9 -12 -22 -20 -23
```

Variabile	Valore
s_2	7000
$x_{1,1}$	20000
$x_{1,2}$	10000
$x_{2,1}$	40000
$x_{3,2}$	25000
$x_{4.3}$	35000
$x_{5,2}$	28000
$x_{5,3}$	5000

La soluzione di base ottimale non è degenere, la variabile di scarto s_2 è pari a 7000 e rappresenta i clieti che l'ufficio della città di Eustis potrebbe gestire ultriormente.