Facultad de Ciencias Exactas, Ingeniería y Agrimensura

Tópicos Avanzados en Optimización Combinatoria y Teoría de Grafos

Programación Lineal

Alumnos: Meli Sebastián, Rodríguez Jeremías

19 de mayo de 2017

1. Ejercicio 1C: Dieta de McDonalds

1.1. Apartado 1

Utilizamos el software LPSolve para hallar el costo buscado. Utilizamos las siguientes variables:

x_1 : Cantidad de Cuartos de Libra con queso a comprar
x_2 : Cantidad de McNificas con queso a comprar
...
x_9 : Cantidad de jugos de naranja a comprar

Y formalizamos el problema como la minimización de:

$$\sum_{i=1}^{9} x_i * Precio_i$$

Sujeto a 8 desigualdades indicando los nutrientes necesarios que aporta cada compra y las cotas correspondientes. El problema en formato lp es el siguiente:

```
/* Objective function */
min: 1.84 x1 + 2.19 x2 + 1.84 x3 + 1.44 x4 + 2.29 x5 + 0.77 x6 + 1.29 x7 + 0.60 x8 + 0.72 x9;

/* Variable bounds */

510 x1 + 370 x2 + 500 x3 + 370 x4 + 400 x5 + 220 x6 + 345 x7 + 110 x8 + 80 x9 >= 2000;
34 x1  + 35 x2  + 42 x3 + 38 x4 + 42 x5 + 26 x6 + 27 x7 + 12 x8 + 20 x9 >= 350;
34 x1  + 35 x2  + 42 x3 + 38 x4 + 42 x5 + 26 x6 + 27 x7 + 12 x8 + 20 x9 <= 375;
28 x1  + 24 x2  + 25 x3 + 14 x4 + 31 x5 + 3 x6   + 15 x7 + 9 x8 + 1 x9 >= 55;
15 x1  + 15 x2  + 6 x3 + 2 x4 + 8 x5 + 0 x6     + 4 x7 + 10 x8 + 2 x9 >= 100;
6 x1   + 10 x2  + 2 x3 + 0 x4 + 15 x5 + 15 x6  + 0 x7 + 4 x8 + 120 x9 >= 100;
30 x1  + 20 x2  + 25 x3 + 15 x4 + 15 x5 + 0 x6 + 20 x7 + 30 x8 + 2 x9 >= 100;
20 x1  + 20 x2  + 20 x3 + 10 x4 + 8 x5 + 2 x6  +  15 x7 + 0 x8 + 2 x9 >= 100;
x1>=0;
x2>=0;
x3>=0;
x4>=0;
x5>=0;
x6>=0;
x7>=0;
x8>=0;
x9>=0;
```

El resultado obtenido es:

Variables	result
	14,855737704918
x1	4,38524590163934
x2	0
x3	0
x4	0
x5	0
x6	6,14754098360656
x7	0
x8	3,42213114754098
x9	0

Es decir, el precio mínimo es US$ 14.85.

1.2. Apartado 2

La cantidad de calorías es (truncando algunos decimales):

$$4.385 * 510 + 6.147 * 220 + 3.422 * 110 = 3965.11$$

1.3. Apartado 3

Agregamos la restricción indicada:

```
510 x1 + 370 x2 + 500 x3 + 370 x4 + 400 x5 + 220 x6 + 345 x7 + 110 x8 + 80 x9 >= 2500;
```

Y obtenemos el siguiente resultado:

Variables	result
	16,6709741550696
x1	0,231941683233911
x2	3,85464987850677
x3	0
x4	0
x5	0
x6	0
x7	0
x8	2,04329578087031
x9	9,13408438259332

Sale un poco más caro intentar comer menos calorías en McDonalds.

1.4. Apartado 4

Reformulamos el problema como sigue:

```
/* Objective function */
min: 1.84 x1 + 2.19 x2 + 1.84 x3 + 1.44 x4 + 2.29 x5 + 0.77 x6 + 1.29 x7 + 0.60 x8 + 0.72 x9;

/* Variable bounds */

510 x1 + 370 x2 + 500 x3 + 370 x4 + 400 x5 + 220 x6 + 345 x7 + 110 x8 + 80 x9 <= 2000;
34 x1  + 35 x2  + 42 x3 + 38 x4 + 42 x5 + 26 x6 + 27 x7 + 12 x8 + 20 x9 >= 350;
34 x1  + 35 x2  + 42 x3 + 38 x4 + 42 x5 + 26 x6 + 27 x7 + 12 x8 + 20 x9 <= 375;
28 x1  + 24 x2  + 25 x3 + 14 x4 + 31 x5 + 3 x6   + 15 x7 + 9 x8 + 1 x9 >= 55;
15 x1  + 15 x2  + 6 x3 + 2 x4 + 8 x5 + 0 x6     + 4 x7 + 10 x8 + 2 x9 >= 100;
6 x1   + 10 x2  + 2 x3 + 0 x4 + 15 x5 + 15 x6  + 0 x7 + 4 x8 + 120 x9 >= 100;
30 x1  + 20 x2  + 25 x3 + 15 x4 + 15 x5 + 0 x6 + 20 x7 + 30 x8 + 2 x9 >= 100;
20 x1  + 20 x2  + 20 x3 + 10 x4 + 8 x5 + 2 x6  +  15 x7 + 0 x8 + 2 x9 >= 100;

x1>=0;
```

```
x2>=0;
x3>=0;
x4>=0;
x5>=0;
x6>=0;
x7>=0;
x8>=0;
x9>=0;
```

El software no logra encontrar una solución, dado que el problema es insatisfactible. La comida de McDonalds tiene demasiadas calorías.

1.5. Apartado 5

Para hallar la menor cantidad de calorías con que podemos satisfascer nuestras necesidades nutricionales, reformulamos el problema como sigue:

```
/* Objective function */
min: 510 x1 + 370 x2 + 500 x3 + 370 x4 + 400 x5 + 220 x6 + 345 x7 + 110 x8 + 80 x9 ;
/* Variable bounds */

34 x1  + 35 x2  + 42 x3 + 38 x4 + 42 x5 + 26 x6 + 27 x7 + 12 x8 + 20 x9 >= 350;
34 x1  + 35 x2  + 42 x3 + 38 x4 + 42 x5 + 26 x6 + 27 x7 + 12 x8 + 20 x9 <= 375;
28 x1  + 24 x2  + 25 x3 + 14 x4 + 31 x5 + 3 x6   + 15 x7 + 9 x8 + 1 x9 >= 55;
15 x1  + 15 x2  + 6 x3 + 2 x4 + 8 x5 + 0 x6     + 4 x7 + 10 x8 + 2 x9 >= 100;
6 x1   + 10 x2  + 2 x3 + 0 x4 + 15 x5 + 15 x6  + 0 x7 + 4 x8 + 120 x9 >= 100;
30 x1  + 20 x2  + 25 x3 + 15 x4 + 15 x5 + 0 x6 + 20 x7 + 30 x8 + 2 x9 >= 100;
20 x1  + 20 x2  + 20 x3 + 10 x4 + 8 x5 + 2 x6  +  15 x7 + 0 x8 + 2 x9 >= 100;

x1>=0;
x2>=0;
x3>=0;
x4>=0;
x5>=0;
x6>=0;
x7>=0;
x8>=0;
x9>=0;
```

Y obetenemos el siguiente resultado:

Variables	result
	2466,98113207547
x1	0
x2	4,08805031446541
x3	0
x4	0
x5	0
x6	0
x7	0
x8	2,0440251572327
x9	9,11949685534591

Reafirmando nuestro resultado del apartado anterior, donde queríamos satisfascer las necesidades nutricionales con solo 2000 calorías.

1.6. Apartado 6

Agregamos las siguientes restricciones:

```
x1 <= 2;
x2 <= 2;
x3 <= 2;
x4 <= 2;
x5 <= 2;
x6 <= 2;
x7 <= 2;
x8 <= 2;
x9 <= 2;
```

Y repetimos los apartados anteriores con estas restricciones para completar la segunda fila de la siguiente tabla:

Dieta Variada	Costo mínimo	Calorías p/ costo mínimo	Calorías mínimas	Costo p/ calorías mínimas
No	14.85	3965.11	2466.98	16.74
Si	16.76	3793.95	3488.28	17.23

2. Ejercicio 3

Sea un problema de PL:

$$max\ cx$$
$$s/a\ Ax \leqslant b,\ x \geqslant 0$$

Sea $\{\hat{x_i}\ :\ \hat{x_i} = (x_{i_1}, x_{i_2}, \ldots, x_{i_n}),\ i = 1, \ldots, s\}$ un conjunto de s soluciones óptimas de PL con $c\hat{x_i} = k$. Sea $\lambda = (\lambda_1, \ldots, \lambda_s)\ /\ \sum_{i=1}^{s} \lambda_i = 1$

Sea $x^* = \sum_{i=1}^{s} \lambda_i \hat{x_i}$. Por definición de x^* observemos que su componente i-ésima es $x_i^* = \sum_{j=1}^{s} \lambda_j x_{j_i}$. Veamos que x^* es solución óptima de PL.

En primer lugar, veamos que es solución. Es decir, que satisface las restricciones del problema.

Sea una restricción cualquiera del problema:

$$a_1x_1 + a_2x_2 + \ldots + a_nx_n \leqslant b \;\Leftrightarrow\; \sum_{i=1}^{n} a_ix_i \leqslant b$$

Veamos que x^* la satisface.

$$\begin{aligned}
& \sum_{i=1}^{n} a_i x_i^* \\
= \quad & < Def\ de\ x^* > \\
& \sum_{i=1}^{n} a_i (\sum_{j=1}^{s} \lambda_j x_{j_i}) \\
= & \\
& \sum_{i=1}^{n} \sum_{j=1}^{s} a_i \lambda_j x_{j_i} \\
= & \\
& \sum_{j=1}^{s} \sum_{i=1}^{n} a_i \lambda_j x_{j_i} \\
= & \\
& \sum_{j=1}^{s} \lambda_j (\sum_{i=1}^{n} a_i x_{j_i}) \\
\leqslant \quad & < Cada\ \hat{x}_i\ cumple\ la\ restricción > \\
& \sum_{j=1}^{s} \lambda_j b \\
= & \\
& b \sum_{j=1}^{s} \lambda_j \\
= & \\
& b\ 1 \\
= & \\
& b
\end{aligned} \tag{1}$$

Luego x^* satisface todas las restricciones, y es solución de PL. Veamos ahora que es óptima:

$$
\begin{aligned}
& c\ x^* \\
= \quad & < Def\ de\ x^* > \\
& c\ \sum_{j=1}^{s} \lambda_j \hat{x}_j \\
= & \\
& \sum_{i=1}^{n} c_i\ \sum_{j=1}^{s} \lambda_j x_{j_i} \\
= & \\
& \sum_{i=1}^{n} \sum_{j=1}^{s} c_i \lambda_j x_{j_i} \\
= & \\
& \sum_{j=1}^{s} \sum_{i=1}^{n} c_i \lambda_j x_{j_i} \\
= & \\
& \sum_{j=1}^{s} \lambda_j \sum_{i=1}^{n} c_i x_{j_i} \\
= & \\
& \sum_{j=1}^{s} \lambda_j\ c\ \hat{x}_j \\
= & \\
& \sum_{j=1}^{s} \lambda_j k \\
= & \\
& k \sum_{j=1}^{s} \lambda_j \\
= & \\
& k\ 1 \\
= & \\
& k
\end{aligned}
$$

$\therefore\ x^*$ es solución óptima.

3. Ejercicio 6E

Sean $c = (c_1, \ldots, c_n)$ y $1 \leqslant k \leqslant n$, k entero. Queremos hallar:

$$
\begin{aligned}
& max\ cx \\
s/a & \\
& 0 \leqslant x \leqslant 1 \\
& \sum_{i=1}^{n} x_i \leqslant k
\end{aligned}
$$

Vamos a suponer que el vector c está ordenado de forma decreciente ($c_i \geqslant c_{i+1},\ 1 \leqslant i \leqslant n-1$). Suponer esto no nos hace perder generalidad, pues siempre podemos reordenar la sumatoria usando la conmutatividad de la suma.

Sea $p = \max\ \{m \mid c_m > 0\}$ y $d = \min(p, k)$. Planteamos la siguiente solución (claramente factible) x^*:

$$x^*_m = \begin{cases} 1 & si \quad 1 \leqslant m \leqslant d \\ \\ 0 & si \quad d+1 \leqslant m \leqslant n \end{cases}$$

Cualquier otra solución factible y puede formarse a partir de x^* aplicando a cada componente desde la primera hasta la d-ésima exactamente una de las siguientes operaciones:

1. No hacer nada
2. Restarle una cantidad t menor a 1
3. Restarle una cantidad t menor a 1 y repartirla entre las componentes $d+1, d+2, ..., n$. Formalmente esto significa elegir un $\lambda \in \mathbb{R}^{n-d}$ tal que $\sum_{i=d+1}^{n} \lambda_i = t$ y sumarle λ_i a cada componente correspondiente.

De esta forma, aplicando exáctamente una operación de izquierda a derecha a cada una de las primeras d componentes de x^* podemos generar cualquier otra solución factible [1].

Veamos que, si tenemos una solución factible x y le aplicamos cualquiera de estas operaciones, el vector resultante y no mejorará el resultado de la función target. Supongamos entonces que aplicamos una operación en la componente x_p. Analicemos qué sucede con cáda una de las tres posibles:

1. Si no hacemos nada, el resultado será el mismo. ($cx = cy$)
2. Si aplicamos la segunda operación, el nuevo resultado será peor. Observemos que $c_p > 0$ ya que esta operación se está aplicando sobre una componente entre 1 y d. Al multiplicar $c_p > 0$ ahora por un valor menor, estará reportando menor beneficio a la función objetivo. ($cx > cy$)
3. Si le restamos t y lo distribuímos entre las otras, perderemos $t.c_p$ ganancia en la solución final y sumaremos $\sum_{i=d+1}^{n} \lambda_i c_i \leqslant \sum_{i=d+1}^{n} \lambda_i c_p = c_p \sum_{i=d+1}^{n} \lambda_i = c_p.t$. Es decir, no sumaremos más de lo que perdemos:

$$cy = cx - tc_p + \textstyle\sum_{i=d+1}^{n} \lambda_i c_i \leqslant cx - tc_p + tc_p = cx$$

Por lo tanto, aplicar una operación de estas tres no mejora el resultado. Dada una solución factible cualquiera y, que como ya discutimos puede generarse mediante d operaciones a partir de $x*$, no mejorará el resultado.

$\therefore\ x^*$ es óptima.

[1] Para ser rigurosos, deberíamos excluír y analizar antes algunas soluciones factibles en el caso m <k. En este caso, habrá soluciones factibles que no estamos considerando: aquellas que ponen x_i negativo a c_i negativo cuando m<k. Estas soluciones a pesar de ser factibles no las tenemos en cuenta porque al poner x_i positivo a un c_i negativo, estan restando valor a la función objetivo y claramente no pueden ser óptimas

4. Ejercicio 7

Nuestro problema es:

$$\begin{aligned} & max\ cx \\ s/a & \\ & 0 \leqslant x \leqslant 1 \\ & \sum_{i=1}^{n} x \leqslant k \end{aligned}$$

Y su dual es:

$$\begin{aligned} & min\ (\sum_{i=1}^{n} y_i) + y_{n+1}k \\ s/a & \\ & y_i + y_{n+1} \geqslant c_i,\ \forall\ 1 \leqslant i \leqslant n \\ & y \geqslant 0 \end{aligned}$$

Recordemos que x^* está definido igual que en el problema anterior; y seguimos bajo la hipótesis del problema anterior de que c está ordenado.

Definiré $y^* \in \mathbb{R}^{n+1}$ por casos:

Caso d=k Es decir, el vector c tiene al menos k valores positivos. Para este caso, habíamos definido $x^* = (1_1, 1_2, \ldots, 1_{k-1}, 1_k, 0, \ldots, 0)$ y definimos y^* como sigue

$$y_i^* = \begin{cases} c_i - c_d & si \quad 1 \leqslant i \leqslant d \\ 0 & si \quad d+1 \leqslant i \leqslant n \\ c_d & si \quad i = n+1 \end{cases}$$

Veamos que esta y^* es una solución factible del dual:

- Para $i \leqslant d,\ \ y_i + y_{n+1} = c_i - c_d + c_d = c_i \geqslant c_i$.
- Para $i > d,\ \ y_i + y_{n+1} = y_{n+1} = c_d \geqslant c_i$ pues $d \leqslant i$ y c está ordenado.

Veamos ahora cuanto vale la función objetivo en y^*:

$$by^* = (\sum_{i=1}^{k}(c_i - c_d)) + kc_d = \sum_{i=1}^{k}(c_i - c_d + c_d) = \sum_{i=1}^{k} c_i = cx^*$$

Caso d=m Es decir, el vector c tiene m<k valores positivos. En este caso habíamos definido $x^* = (1_1, 1_2, \ldots, 1_{m-1}, 1_m, 0, \ldots, 0)$ y definimos y^* como sigue:

$$y_i^* = \begin{cases} c_i & si \quad 1 \leqslant i \leqslant m \\ 0 & si \quad m+1 \leqslant i \leqslant n+1 \end{cases}$$

Veamos que esta y^* es una solución factible del dual:

- Para $i \leqslant m,\ \ y_i + y_{n+1} = c_i \geqslant c_i$.
- Para $i > m,\ \ y_i + y_{n+1} = 0 \geqslant c_i$ pues c_i es negativo.

Veamos ahora cuanto vale la función objetivo en y^*:

$$by^* = (\sum_{i=1}^{m} c_i) = cx^*$$

Por lo tengo, hemos obtenido un y^* (definido en función del problema) que es solu1ción factible del dual tal que $b.y^* = c.x^*$.

$\therefore$ x^* óptima.

5. Ejercicio 9

Formalizamos el problema de la planta frigorífica como sigue:

```
/* Objective function */
max: 8 j + 4 l + 4 s + 14 ja + 12 la + 13 sa + 11 je + 7 le + 9 se ;

/* Variable bounds */

j  + ja + je <= 480; // produccion de jamon        R1 - 8 - [40 , .. ]
l  + la + le <= 400; // produccion de lomos        R2 - 5 - [190, 420]
s  + sa + se <= 230; // produccion de salchichones R3 - 6 - [20 , 270]
ja + sa + la <= 420; // ahumado                    R4 - 7 - [400, 630]
je + se + le <= 250; // ahumado extra              R5 - 3 - [210, 690]
```

- ¿Cuál es el producto que más promete ganancias?

  El producto que mas promete ganancias es el jamón (8), comparado con el lomo (5) y el salchichón (6)

  La empresa estaría dispuesta a invertir en la produccion del jamon, siempre y cuando producir una unidad mas de jamón no tenga un costo mayor a 8, ya que en ese caso sería contraproducente

- ¿Hasta cuánto estaría dispuesta la empresa a invertir de manera que esta inversión resulte rentable? En este caso, como el rango no tiene limite ($[40, \ldots]$), la empresa podría invertir hasta que el costo de producir una unidad mas de jamon no tenga un costo mayor a 8 (que es la ganancia por cada unidad vendida)

- ¿Cuál sería la pérdida monetaria si la empresa debe reducir en 10 unidades la capacidad de ahumar sus productos en tiempo normal?

  Como la restricción del ahumado tiene estos valores: R4 - 7 - [400, 630]

  Vemos que mientras el valor (420 actualmente) se mantenga en el rango 400 - 630, el valor de cada unidad de ahumado es de 7. Esto quiere decir que restar 10 unidades (410, sigue en el rango), produciría un cambio en la ganancia de la empresa de

  $$(-10)*7 = -70$$

  Es decir, una pérdida de 70