Práctica 8

Pablo Gutiérrez Aguirre
pgutierrez2018@udec.cl

16 de mayo 2022

Tabla de contenidos

Certamen 1: Pregunta 1

Una familia de agricultores posee una parcela familiar de 125 hectáreas de tierra y tiene 4.000.000 de pesos en fondos disponibles para inversión para el próximo año. Sus miembros pueden producir un total de 3.500 horas en mano de obra en invierno y 4.000 horas durante el verano. En caso que no se necesite una parte de estas horas, los jóvenes de la familia pueden trabajar en un campo vecino por 2.500 pesos la hora durante el invierno y 3.000 pesos hora en el verano. El ingreso familiar efectivo puede obtenerse a partir de tres cultivos (porotos, maíz y avena) y dos tipos de crianza de animales (vacas lecheras y gallinas ponedoras). No se necesita invertir para los cultivos, sin embargo cada animal puede ser adquirido al comienzo de cada año a un precio de 500.000 pesos por vaca y 8.000 pesos por gallina. Cada vaca requiere de 1,5 hectáreas de tierra, 100 horas de trabajo en invierno y 50 horas en verano, y cada vaca produce una utilidad anual de 450.000 pesos. Para las gallinas se requiere de 0,6 horas en invierno, 0,3 horas en verano, y cada una de ellas produce anualmente 10.000 pesos en venta de huevos. El gallinero puede acomodar un máximo de 3.000 gallinas y el tamaño del establo limita el rebaño a un máximo de 32 vacas. Las horas requeridas por hectáreas y los ingresos estimados por hectárea sembrada, para cada uno de los posibles cultivos vienen dados en la Tabla 1. Formule un modelo de programación que permita a la familia de agricultores, planificar las operaciones para el próximo año en forma conveniente.

Certamen 1: Pregunta 1

Variables de decisión

- x_1 : Cantidad de hectáreas a sembrar de frijoles
- x_2 : Cantidad de hectáreas a sembrar de maíz
- x_3 : Cantidad de hectáreas a sembrar de avena
- y_1 : Cantidad de vacas a criar
- y_2 : Cantidad de gallinas a criar
- h_1 : Cantidad de horas trabajadas en ranchos vecinos en invierno
- h_2 : Cantidad de horas trabajadas en ranchos vecinos en verano

Certamen 1: Pregunta 1

Función objetivo

$$\begin{aligned} Max \quad z = & 150.000x_1 + 250.000x_2 + 100.000x_3 + \\ & 450.000y_1 + 10.000y_2 + \\ & 3.500h_1 + 4.000h_2 \end{aligned} \tag{1}$$

Sujeto a

$$\begin{aligned} 20x_1 + 35x_2 + 10x_3 + 100y_1 + 0.6y_2 + h_1 &\leq 3.000 && \text{Mano de obra invierno} \\ 50x_1 + 75x_2 + 40x_3 + 50y_1 + 0.3y_2 + h_2 &\leq 4.000 && \text{Mano de obra verano} \\ x_1 + x_2 + x_3 + 1.5y_1 &\leq 125 && \text{Restricción de tierra} \\ 500.000y_1 + 8.000y_2 &\leq 4.000.000 && \text{Restricción de capital} \\ y_1 &\leq 32 && \text{Capacidad establo} \\ y_2 &\leq 3000 && \text{Capacidad gallinero} \\ x_1, x_2, x_3, y_1, y_2 &\geq 0 && \text{Restricciones de dominio} \\ y_1, y_2 &\in \mathbb{N} && \text{Restricciones de dominio} \end{aligned}$$

Certamen 1: Pregunta 2

A comienzo de enero, 50 técnicos especializados trabajan para una empresa líder en electrónica en el gran Concepción, llamada CSL. Cada técnico especializado puede trabajar hasta 160 horas al mes. Para satisfacer futuras demandas, es necesario capacitar a nuevos técnicos. La capacitación de un nuevo técnico dura un mes. Un técnico experimentado tiene que supervisar al aprendiz durante 50 horas del mes de entrenamiento. A cada técnico experimentado se le pagan mensualmente 2.000 dólares (aunque no trabaje las 160 horas). Durante el mes de entrenamiento, se pagan al aprendiz 1.000 dólares al mes. Al final de cada mes, 5% de los técnicos experimentados de CSL, cambian de trabajo para irse a empresas de la competencia. La empresa desea optimizar los costos de mano de obra, cumpliendo con los requerimientos de servicio durante los próximos 5 meses. Donde son 6.000, 7.000, 8.000, 9.500 y 11.000 horas, para los meses 1, 2, 3, 4 y 5, respectivamente. Formule un modelo de programación que permita a la empresa optimizar los costos de mano de obra.

Certamen 1: Pregunta 2

Variables de decisión

- x_i : Número de técnicos aprendices en el mes i.
- y_i : Número de técnicos experimentados al inicio del mes i.

Certamen 1: Pregunta 2

Función objetivo:

$$min \quad z = 1.000(x_1 + x_2 + x_3 + x_4 + x_5) + 2.000(y_1 + y_2 + y_3 + y_4 + y_5)$$

Sujeto a

$$160y_1 - 50x_1 \geq 6.000 \quad \text{(Tiempo de técnico disponible para el mes 1)}$$
$$160y_2 - 50x_2 \geq 7.000 \quad \text{(Tiempo de técnico disponible para el mes 2)}$$
$$160y_3 - 50x_3 \geq 8.000 \quad \text{(Tiempo de técnico disponible para el mes 3)}$$
$$160y_4 - 50x_4 \geq 9.500 \quad \text{(Tiempo de técnico disponible para el mes 4)}$$
$$160y_5 - 50x_5 \geq 11.000 \quad \text{(Tiempo de técnico disponible para el mes 5)}$$
$$50 * 0.95y_1 + x_1 = y_2 \quad \text{(Técnicos especializados que salen en el mes 2)}$$
$$0.95y_2 + x_2 = y_3 \quad \text{(Técnicos especializados que salen en el mes 3)}$$
$$0.95y_3 + x_3 = y_4 \quad \text{(Técnicos especializados que salen en el mes 4)}$$
$$0.95y_4 + x_4 = y_5 \quad \text{(Técnicos especializados que salen en el mes 5)}$$
$$x_i, y_i \geq 0, x_i, y_i \in \mathbb{N}, \forall i = \{1, ..., 5\} \quad \text{Restricciones de dominio}$$

Certamen 1: Pregunta 2

Parámetros:

- T : Conjunto de periodos, $T = \{1, ..., n\}$
- d_t :Demanda horaria del periodo t
- a: Sueldo técnico aprendiz
- e: Sueldo técnico especializado
- h_e: Horas máximas que puede trabajar un técnico especializado
- h_a: Horas de capacitación de un técnico aprendiz
- c: Técnicos especializados iniciales
- r: Porcentaje de técnicos que renuncian

Certamen 1: Pregunta 2

Función objetivo:

$$min \quad z = a \sum_{t \in T} x_t + e \sum_{t \in T} y_t$$

Sujeto a

$$h_e \cdot y_t - h_a \cdot x_t \geq d_t \quad \forall t \in T \tag{1}$$

$$c \cdot (1 - r) y_1 + x_1 = y_2 \tag{2}$$

$$(1 - r) y_t + x_t = y_{t+1} \qquad \forall t \in \{2, ..., n\} \tag{3}$$

$$x_t, y_t \geq 0, x_t, y_t \in \mathbb{N}, \forall t \in T \tag{4}$$

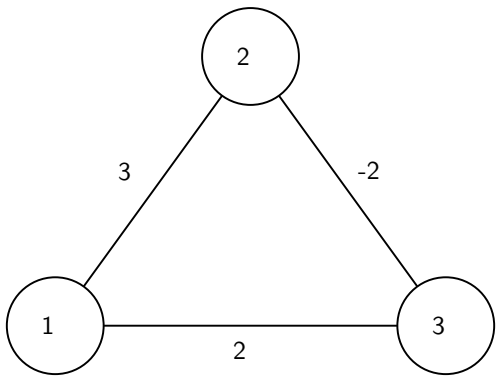
Certamen 1: Pregunta 3

Demostrar si los algoritmos de Prim, Dijkstra y Kruskal funcionan correctamente con aristas con costos negativos. Justifique su respuesta con un argumento o un contraejemplo.

- **Algoritmos de Prim y Kruskal:** Ambos problemas resuelven el problema de árbol de cobertura mínima. Por tanto, ambos algoritmos trabajan desde la determinación de guardar una arista y no acumular aristas. Por ende, se puede aplicar incluso si existen aristas con pesos negativos en el grafo. En conclusión, la existencia de los pesos de las aristas negativas no afecta a la exactitud de los dos algoritmos.

Certamen 1: Pregunta 3

Algoritmo de Dijkstra: No puede encontrar la ruta más corta con aristas negativas. Por ejemplo, en un grafo de 3 vértices, donde hay un camino de 1 al 3 de costo 2, de 1 al 2 de costo 3 y otro de 2 al 3 de costo -2, Dijkstra no encuentra la ruta más corta de 1 a 3; encuentra el camino de costo 2 (camino 1 – 2) y no el de longitud 1 (camino 1 – 3 – 2). Esto es debido a que en la última iteración no se alcanza a actualizar el costo de llegar al vértice 2 antes de finalizar el algoritmo.



Certamen 1: Pregunta 4

Diseñe un algoritmo en Python de tiempo lineal para determinar si un grafo de n vértices contiene algún ciclo. En caso de usar algún algoritmo visto en clases, solo indicar que parte se debe cambiar.

Certamen 1: Pregunta 4

Usar el algoritmo de DFS (búsqueda primero en profundidad), cuando un vértice en la función recursiva ya está marcado como gris, significa que ya se visitó previamente, por tanto, es un ciclo. Además, agregar que π del vértice u es diferente al adyacente al vértice v en un grafo no dirigido, ya que en la lista de adyancencia de un grafo no dirigido, todos tienen doble dirección.

Certamen 1: Pregunta 4

```
1  def DFS_visitar(self,u):
2      self.tiempo += 1
3      self.d[u] = self.tiempo
4      self.color[u] = "gris"
5      print(u, end = " ")
6      for v in self.adyacentes[u]:
7          if self.color[v] == "gris" and self.pi[u] != v:  #NUEVA LINEA
8              print("Hay un ciclo")                        #NUEVA LINEA
9          if self.color[v] == "blanco:":
10             self.pi[v] = u
11             self.DFS_visitar(v)
12     self.color[u] = "negro"
13     self.tiempo += 1
14     self.f[u] = self.tiempo
15
16 def DFS(self):
17     for u in self.vertices:
18         self.color[u] = "blanco"
19         self.pi[u] = -1
20         self.tiempo = 0
21     for u in self.vertices:
22         if self.color[u] == "blanco":
23             self.DFS_visitar(u)
```

Gracias por su atención

Dudas o consultas:

pgutierrez2018@udec.cl