# IN3701 - Modelamiento y Optimización

## Auxiliar Nº 3 - Algoritmo de Ford-Fulkerson

Profesores: Victor Bucarey, Pablo Rey.
Auxiliares: Cristiam Gil, Dana Pizarro, Diego Bernstein, Diego Fuentealba, Eduardo Lara, Macarena Osorio, Mario Morales, Tomás Lagos.

**P1.**
Las placas madre de los computadores son manufacturados (en unidades de mil) por tres compañías $c_1$, $c_2$ y $c_3$. Luego, son distribuidos a dos fábricas de computadores, $m_1$ y $m_2$, a través de la red de transporte señalada a continuación.

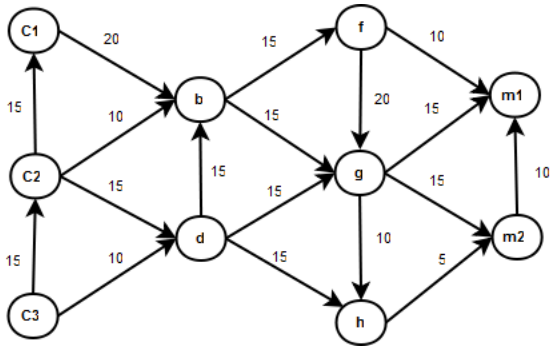


Fig. 1: Red para el problema 1.

La compañía $c_1$ puede producir hasta 15 unidades, la compañía $c_2$ hasta 20 unidades y la compañía $c_3$ hasta 25 unidades. Si cada fábrica necesita 25 unidades, ¿Cuántas unidades debe producir cada compañía de manera tal que en conjunto satisfagan la demanda de cada fábrica o al menos suministrarles la mayor cantidad posible que permita la red?

**P2.**
Aplique el algoritmo de Ford-Fulkerson al siguiente Grafo:

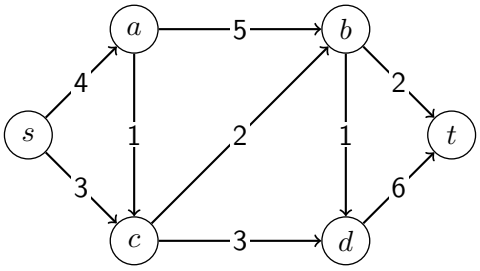


Fig. 2: Red para el problema 2.

**P3.**
Un problema fundamental en el diseño urbano es la localización de servicios básicos como colegios, hospitales y áreas recreacionales. En este problema formularemos un modelo simplificado para decidir la localización de estaciones de bomberos en una ciudad.

La ciudad se puede dividir en $I$ distritos, en que cada uno contiene $p_i$ habitantes. Análisis preliminares (estudios de terrenos, factores políticos, etc.) han establecido que las estaciones de bomberos solo pueden ser

ubicadas en $J$ sitios predeterminados dentro de la ciudad. Sea $d_{ij} > 0$ la distancia desde el centro del distrito $i$ hasta el sitio $j$. Se deben seleccionar los sitios en los cuales construir una estación (en un sitio cabe a lo más una) y además se debe asignar una estación a cada distrito. Es decir, cada distrito de la ciudad debe tener una (y solo una) estación de bomberos asociada. Una estación puede tener más de un distrito asociado. Construir una estación en el sitio $j$ tiene un costo fijo asociado igual a $c_j$. Además, existe un costo variable que es linealmente proporcional (constante de proporcionalidad es $f_j$) a la cantidad total de gente que debe servir la estación. O sea, si se construye una estación en el sitio $j$, entonces el costo asociado es $c_j + f_j s_j$, en que $s_j$ es la población total que debe servir la estación ubicada en $j$ (es la suma de las poblaciones de todos los distritos asociados a esa estación). El presupuesto total destinado para construir las estaciones de bomberos es igual a $B$ y no debe ser sobrepasado.

Formule un modelo de programación lineal que minimice la distancia máxima entre un distrito y su respectiva estación.

**Resumen:**

1. Un **Grafo simple** $G(V, A)$ se define por vertices y arcos (caso dirigido, o arístas en caso no-dirigido), tal que todo arco conecta dos vertices distintos, y no existen dos arcos que conectan los mismos dos vertices (en la misma dirección).

2. Un **camino** desde v hasta u es una forma de conectar a v con u utilizando los arcos del grafo en su dirección factible de manera de que ningún nodo es visitado dos veces.

3. Un grafo se dice que **está conectado** si es que para cada par de nodos existe un camino que los conectan.

4. El problema de **encontrar el flujo máximo** entre un par de nodos en un grafo se puede resolver mediante la siguiente formulación:

$$\begin{array}{ll} \max_{\{l_a \leq x_a \leq u_a : a \in E \setminus \{ts\}\}} & x_{ts} \\ div_i(x) = 0 & i \in V \end{array}$$

   Donde $div_i(x) = \sum_{\{a:a=(i,j)\in E\}} x_a - \sum_{\{a:a=(j,i)\in E\}} x_a$ es el operador de divergencia en el nodo $i$ sobre los arcos del grafo que se conectan con tal nodo.

5. El **algoritmo de Ford-Fulkerson** resuelve el problema anterior:

**Algorithm 1** PROCEDURE W:

**Require:** $m(i) = null \ \forall i$;
**Require:** $m(s) = (r, +\infty)$; % $r$ es el arco de retorno
**Ensure:** $L = [s]$;
**while** $t \notin L$ AND $\exists i \in L$ **do**
  Eliminar $i$ de $L$;
  **for** $a = (i, j) \in A$ **do**
    **if** $m(j) == null$ AND $x_a < c_a$ **then**
      $m(j) = (a, \min\{c_a - x_a, \delta_i\})$;
      $L = L \cup [j]$;
  **for** $a = (j, i) \in A$ **do**
    **if** $m(j) == null$ AND $x_a > 0$ **then**
      $m(j) = (a, \min\{x_a, \delta_i\})$;
      $L = L \cup [j]$;

**Algorithm 2** FORD-FULKERSON:

**Require:** Optimo=$False$;
**while** !Optimo **do**
  Aplicar Procedure W;
  **if** $t$ no fue marcado **then**
    Optimo = $True$;
    Print x;
  **else**
    Aumentar el flujo en una cantidad $\delta_t$;