Gráficas aleatorias, redes sociales y el internet

Parte 3

Mariana Olvera-Cravioto

UNC Chapel Hill

molvera@email.unc.edu

7 de octubre de 2021

El modelo Albert-Barabási

- Todos los modelos de gráficas aleatorias que hemos visto son **estáticos**.
- Los modelos estáticos no explican como crecen las gráficas.
- Los modelos que **evolucionan** proponen un mecanismo para escoger cómo se conecta un vértice que acaba de llegar.
- Numeramos los vértices en el orden en el que se agregan a la gráfica.
- Uno de los modelos más famosos de gráficas que evolucionan es el modelo **Albert-Barabási** o de **conexión preferencial**.
- Este modelo asume que un vértice que va llegando escoge el vértice al que va a conectarse de acuerdo a una probabilidad proporcional a su grado.
- En otras palabras, los recién llegados "prefieren" conectarse a los vértices con grados grandes.

El modelo Albert-Barabási... cont.

- Este modelo comienza con un solo vértice que tiene un bucle.
- En cada paso, un nuevo vértice llega y se agrega a la gráfica ya sea conectándose con una arista a un vértice existente, o creando un bucle.
- Sea $D_i(k)$ el grado del vértice i después de que k vértices han sido agregados.
- Cuando llega el vértice $k+1$ se conecta al vértice i con probabilidad:

$$p_i(k) = \begin{cases} \frac{D_i(k)}{2k+1}, & i = 1, \ldots, k, \\ \frac{1}{2k+1}, & i = k+1. \end{cases}$$

- Este modelo produce gráficas **libres de escala** con distribución del los grados:

$$P_k(n) = \frac{1}{n} \sum_{i=1}^{n} 1(D_i(n) = k) \approx 4k^{-3}$$

cuando n es grande.

Modelos de conexión preferencial

- El modelo se puede generalizar de tal manera que cada vértice llega con $m \geq 1$ aristas, y la arista j del vértice $k+1$ escoge conectarse al vértice i con probabilidad:

$$p_i(k) = \frac{D_i(k, j-1) + \delta}{\sum_{v=1}^{k} (D_v(k, j-1) + \delta)}, \qquad i = 1, \ldots, k, k+1,$$

  donde $\delta > -m$ y $D_i(t, j)$ es el grado del vértice i después de que t vértices han llegado y j aristas del vérice $t+1$ han sido conectadas.

- Este modelo genera gráficas **libres de escala** cuyos grados tienen distribución

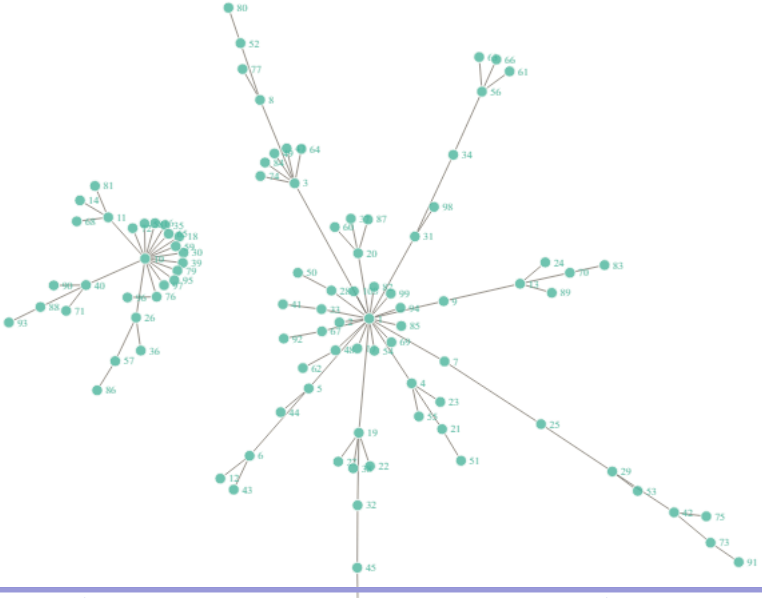
$$P_k(n) = \frac{1}{n} \sum_{i=1}^{n} 1(D_i(n, m) = k) \approx C_{m,\delta} k^{-\tau}$$

  cuando n es grande, donde $\tau = 3 + \delta/m$.

Modelos de conexión preferencial... cont.

- En gráficas de conexión preferencial, los grados de los vértices más **viejos** son muy diferentes que los de los más **jóvenes**.
- La etiqueta de un vértice, i.e., su orden de llegada, nos da mucha información acerca de sus propiedades.
- Los vértices más viejos tienden a tener grados más grandes.
- El **grado más grande** crece como $O(n^{-1/(2+\beta/m)})$ cuando $n \to \infty$.

Una gráfica Albert-Barabási



Simulación estocástica

- Suppose we want to run experiments using one of the random graph models we have seen.
- Por ejemplo, podemos cambiar el número de vértices y los parámetros del modelo para explorar sus propiedades.
- Es posible que queramos evaluar otro fenómeno sobre la gráfica.
- **Pregunta:** ¿Cómo lo hacemos?

Simulación estocástica

- Suppose we want to run experiments using one of the random graph models we have seen.
- Por ejemplo, podemos cambiar el número de vértices y los parámetros del modelo para explorar sus propiedades.
- Es posible que queramos evaluar otro fenómeno sobre la gráfica.
- **Pregunta:** ¿Cómo lo hacemos?
- **Respuesta:** Podemos usar una computadora para generar números aleatorias que pueden ser usados para crear una gráfica.
- La técnica de generar números o processos aleatorios usando la computadora se conoce como **simulación estocástica**.

Números aleatorios uniformes

- Cualquier lenguaje de programación (e.g. Python, C, R, Matlab), e incluso Excel, puede generar números aleatorios.
- Usaremos U para denotar a una variable aleatoria uniforme en $[0, 1]$.
- Para cualquier $0 \leq a < b \leq 1$,

$$P(a < U < b) = b - a$$

- Informalmente, U toma todos los valores en $[0, 1]$ con la misma probabilidad.
- Las computadoras pueden generar secuencias largas de variables aleatorias independientes y uniformes en $[0, 1]$.
- Los números aleatorios **uniformes** pueden usarse para generar cualquier otro tipo de números aleatorios.

Volados virtuales

- Supongamos que queremos simular un volado que cae cara con probabilidad p.
- Vamos a generar un número aleatorio X tal que

$$X = \begin{cases} 1, & \text{con probabilidad } p, \\ 0, & \text{con probabilidad } 1 - p. \end{cases}$$

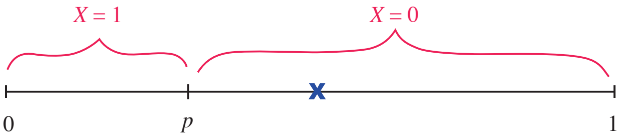
- Asociamos $X = 1$ con el evento que el volado cayó **cara**, y $X = 0$ con el evento que cayó **cruz**.
- **Pregunta:** ¿Cómo podemos usar un número aleatorio uniforme U para generar X?

Volados virtuales

- Supongamos que queremos simular un volado que cae cara con probabilidad p.
- Vamos a generar un número aleatorio X tal que

$$X = \begin{cases} 1, & \text{con probabilidad } p, \\ 0, & \text{con probabilidad } 1-p. \end{cases}$$

- Asociamos $X = 1$ con el evento que el volado cayó **cara**, y $X = 0$ con el evento que cayó **cruz**.
- **Pregunta:** ¿Cómo podemos usar un número aleatorio uniforme U para generar X?



Volados virtuales... cont.

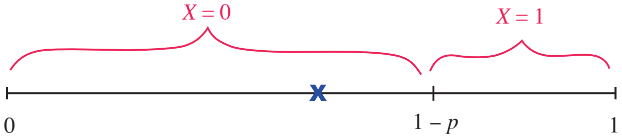
- Esto es equivalente a definir:

$$X = 1(U \leq p),$$

donde $1(A) = 1$ si A ocurre y $1(A) = 0$ si A^c ocurre.

- **Observación:** También hubiéramos podido definir

$$X = 1(U > 1 - p)$$



Cómo generar números aleatorios discretos

- Supongamos que queremos generar una variable aleatoria X que toma los valores $\{x_1, x_2, \ldots, x_k\}$.
- Sea $\{p_i : 1 \leq i \leq k\}$ la función de probabilidad de X, i.e.,

$$p_i = P(X = x_i), \qquad 1 \leq i \leq k$$

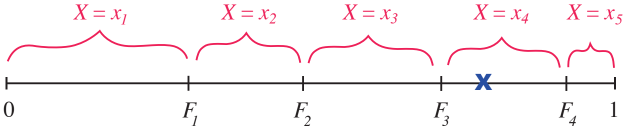
- Ahora calculemos la función de distribución de X definiendo $F_0 = 0$ y:

$$F_i = p_1 + \cdots + p_i, \qquad 1 \leq i \leq k$$

- **Pregunta:** ¿Cómo podemos usar un número aleatorio uniforme U para generar X?

Cómo generar números aleatorios discretos... cont.

- Sea U un número aleatorio uniforme en $[0, 1]$.



- Esto es equivalente a definir:

$$X = \sum_{i=1}^{k} x_i 1(F_{i-1} \leq U < F_i)$$

Cómo simular una distribución continua

- Supongamos que queremos generar una variable aleatoria X tal que:
  - X tiene función de distribución $F(x) = P(X \leq x)$
  - $F(x)$ es continua y estrictamente creceiente cuando $0 < F(x) < 1$ (puede ser no decreciente para $F(x) = 0$ o $F(x) = 1$)
  - $F(x)$ tiene inversa $F^{-1}(x)$.

- **Algoritmo:**
  1. Generar $U \sim$ Uniforme[0,1]
  2. Regresaar $X = F^{-1}(U)$

- Esto es conocido como el **método de la transformación inversa**.

Ejemplo: números aleatorios Pareto

- Para generar gráficas aleatorias libres de escala usando los modelos que hemos visto requiere simular números aleatorios Pareto.
- La función de distribución Pareto está dada por:

$$F(x) = 1 - (x/b)^{-\alpha}, \qquad x \geq b$$

- El parámetro $\alpha > 0$ controla la **forma** de la distribución (qué tan pesada es su cola), y el parámetro $b > 0$ controla su **escala** (y su soporte).
- Podemos calcular su inversa de la manera siguiente:

Ejemplo: números aleatorios Pareto

- Para generar gráficas aleatorias libres de escala usando los modelos que hemos visto requiere simular números aleatorios Pareto.
- La función de distribución Pareto está dada por:

$$F(x) = 1 - (x/b)^{-\alpha}, \qquad x \geq b$$

- El parámetro $\alpha > 0$ controla la **forma** de la distribución (qué tan pesada es su cola), y el parámetro $b > 0$ controla su **escala** (y su soporte).
- Podemos calcular su inversa de la manera siguiente:

$$u = 1 - (x/b)^{-\alpha} \iff (x/b)^{-\alpha} = 1 - u \iff x = b(1-u)^{-1/\alpha}$$

- Por lo tanto, $F^{-1}(u) = b(1-u)^{-1/\alpha}$, y $X = F^{-1}(U)$ tiene una distribución Pareto(α, b).

Cómo generar gráficas aleatorias

- Lo único que necesitamos para generar los modelos de gráficas aleatorias que hemos visto es saber generar volados, números aleatorios discretos, y variables aleatorias continuas usando el método de la transformación inversa.
- Los modelos que hemos visto son:
  - Gráfica Erdős-Rényi
  - Modelo Chung-Lu
  - Modelo estocástico por bloques
  - Gráfica aleatoria de intersección
  - Modelo Albert-Barabási
- Para cada uno de ellos vamos a dar un algoritmo en **pseudocódigo** para generarlo.
- En todos los casos, la meta es obtener la matriz de adyacencia A para la gráfica generada.

Gráfica Erdős-Rényi

- Este algoritmo genera una gráfica Erdős-Rényi no dirigida con n vértices y probabilidad para las aristas p.
- Definir $p \in (0,1)$ y n; inicializar la matriz $A \in \mathbb{R}^n \times \mathbb{R}^n$.
- Para $i = 1 : n$
  - Para $j = 1 : n$
    - Si $i = j$ asignar
$$a_{i,j} = 0$$
    - Si $i \neq j$ generar un número aleatorio uniforme en $[0,1]$, $U_{i,j}$, y asignar
$$a_{i,j} = a_{j,i} = 1(U_{i,j} \leq p)$$
- Regresar la matriz de adyacencia A.

Modelo Chung-Lu model

- Este algoritmo genera un modelo Chung-Lu con n vértices y pesos con distribución $F(x) = 1 - (x/b)^{-\alpha}$.
- Definir n e inicializar la matriz $A \in \mathbb{R}^n \times \mathbb{R}^n$ y el vector de pesos $W \in \mathbb{R}^n$.
- Para $i = 1 : n$
  - Generar un número aleatorio uniforme en $[0, 1]$, U_i, y asignar

$$W_i = b(1 - U_i)^{-1/\alpha}$$

- Calcular $L_n = \sum_{i=1}^{n} W_i$.
- Para $i = 1 : n$
  - Para $j = 1 : n$
    - Si $i = j$ asignar

$$a_{i,j} = 0$$

    - Si $i \neq j$ generar un número aleatorio uniforme en $[0, 1]$, $U_{i,j}$, y asignar

$$p_{i,j} = \frac{W_i W_j}{L_n} \wedge 1 \qquad \text{y} \qquad a_{i,j} = a_{j,i} = 1\,(U_{i,j} \leq p_{i,j})$$

- Regresar la matriz de adyacencia A.

Modelo estocástico por bloques

- Este algoritmo genera un modelo SBM con n vértices, K comunidades de tamaños $\{\pi_{1,n}, \ldots, \pi_{K,n}\}$, y kernel κ.
- Definir n e inicializar la matriz $A \in \mathbb{R}^n \times \mathbb{R}^n$.
- Asignar a cada vértice una etiqueta con su comunidad, digamos en un vector $(C_1, \ldots, C_n)$, e.g., asignándoles a los primeros $\pi_{1,n}$ vértices etiqueta 1, los siguientes $\pi_{2,n}$ etiqueta 2, etc.
- Para $i = 1 : n$
  - Para $j = 1 : n$
    - Si $i = j$ asignar
    $$a_{i,j} = 0$$
    - Si $i \neq j$ generar un número aleatorio uniforme en $[0, 1]$, $U_{i,j}$, y asignar
    $$p_{i,j} = \frac{\kappa(C_i, C_j)}{n} \qquad \text{and} \qquad a_{i,j} = a_{j,i} = 1\,(U_{i,j} \leq p_{i,j})$$
- Regresar la matriz de adyacencia A.

Gráfica aleatoria de intersección

- Este algoritmo genera una gráfica aleatoria de intersección con coeficiente de agrupapiento ajustable y distribución de los pesos $F(x) = 1 - (x/b)^{-\alpha}$.
- Definir $\beta, \gamma > 0$ y n; inicializar la matriz $A \in \mathbb{R}^n \times \mathbb{R}^n$ y el vector de pesos $W \in \mathbb{R}^n$.
- **Para generar la gráfica bipartita:**
- Definir $m = \lfloor \beta n \rfloor$ e inicializar la matriz $B \in \mathbb{R}^n \times \mathbb{R}^m$.
- Para $i = 1 : n$
  - Generar un número aleatorio uniforme en $[0, 1]$, U_i' , y asignar

$$W_i = b(1 - U_i')^{-1/\alpha}$$

  - Asignar $p_i = \frac{\gamma W_i}{n} \wedge 1$.
  - Para $j = 1 : m$
    - Generar un número aleatorio uniforme en $[0, 1]$, $U_{i,j}$, y asignar

$$b_{i,j} = 1(U_{i,j} \leq p_i)$$

- Regresar la matriz de adyacencia B de la gráfica bipartita.

Gráfica aleatoria de intersección... cont.

- **Para generar la gráfica de intersección:**
- Para $i = 1 : n$
  - Para $j = 1 : m$
    - Definir $N_i = B_{i\bullet}$, donde $B_{i\bullet}$ es la fila i de la matriz B (vecinos de i en la gráfica bipartita).
    - Definir $N_j = B_{j\bullet}$ (vecinos de j en la gráfica bipartita).
    - Asignar $J = N_i - N_j$.
    - Si $\min |J| = 0$ y $i \neq j$, entonces asignar $a_{i,j} = 1$; de lo contrario $a_{i,j} = 0$.
- Regresar la matriz de adyacencia A para la gráfica de intersección.
- **Observación:** En el código, para cualquier vector $X = (x_1, \ldots, x_m)$ escribimos $|X| = (|x_1|, \ldots, |x_m|)$.

Modelo Albert-Barabási

- Este algoritmo genera un modelo Albert-Barabási con n vértices.
- Inicializar el vector de grados $D \in \mathbb{R}^n$ y la matriz de adyacencia $A \in \mathbb{R}^n \times \mathbb{R}^n$.
- Asignar $D(1) = 2$ y $a_{1,1} = 1$.
- Para $k = 2 : n$
  - Asignar $D(k) = 1$ y construir el vector F_k de la siguiente manera:
  - Asignar $F_k(1) = D(1)$, y para $j = 2 : k$
    - Asignar $F_k(j) = F_k(j-1) + D(j)$
  - Normalizar F_k actualizando $F_k \to F_k / \sum_{j=1}^{k} F_k(j)$
  - Generar un número aleatorio uniforme en $[0, 1]$, U, y calcular

$$J = \sum_{j=1}^{k} j 1(F_k(j-1) < U \leq F_k(j))$$

  - Actualizar:

$$a_{k,J} = a_{J,k} = 1, \qquad D(J) = D(J) + 1$$

- Regresar la matriz de adyacencia A.

Gracias por su atención.