Tarea III: Evaluación de Aleatoriedad Uniforme

Prueba de Promedio

Anthony Fuentes y Carlos Murcia

Curso de Simulación

9 de enero de 2026

Contenido

- Pruebas Estadísticas
- Datos teóricos (fórmulas) de cada muestra
- Resultados por lenguaje/muestra
- Gráficos
- Conclusiones

Introducción

- Objetivo: evaluar si las secuencias generadas se comportan como una distribución uniforme esperada.

Prueba de promedio

Objetivo: verificar si el promedio muestral $\bar{X}$ coincide con el promedio teórico μ.

Parámetros teóricos:

$$U(a,b): \ \mu = \frac{a+b}{2}, \quad \sigma^2 = \frac{(b-a)^2}{12}$$

$$\{a, a+1, \ldots, b\}: \ m = b-a+1, \ \mu = \frac{a+b}{2}, \quad \sigma^2 = \frac{m^2-1}{12}$$

Hipótesis (dos colas): $H_0 : \mathbb{E}[X] = \mu$ vs $H_1 : \mathbb{E}[X] \neq \mu$

Regla (nivel α):

$$\mu - z_{1-\alpha/2}\sqrt{\frac{\sigma^2}{n}} \ \leq \ \bar{X} \ \leq \ \mu + z_{1-\alpha/2}\sqrt{\frac{\sigma^2}{n}}$$

Pasa si $\bar{X}$ cae dentro del intervalo.

Resultado: Prueba de promedio (Python $U(0,1)$)

n	1,000,000
$\bar{X}$	0.5004100834
μ	0.5
σ^2	0.0833333333
Intervalo $[L, U]$	[0.4994342071, 0.5005657929]
Decisión	**Pasa**

Prueba de varianza (Chi-cuadrado)

Objetivo: verificar si la varianza poblacional es la esperada: $\sigma^2 = \sigma^2_{teo}$.

Parámetros teóricos:

$$U(a,b):\ \sigma^2 = \frac{(b-a)^2}{12} \qquad \{a,\ldots,b\}:\ m = b-a+1,\ \sigma^2 = \frac{m^2-1}{12}$$

Hipótesis (dos colas): $H_0 : \sigma^2 = \sigma^2_{teo}$ vs $H_1 : \sigma^2 \neq \sigma^2_{teo}$

Varianza muestral (con $n-1$):

$$s^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2$$

Estadístico:

$$\chi^2_0 = \frac{(n-1)s^2}{\sigma^2_{teo}}, \qquad df = n-1$$

Regla (nivel α):

$$\chi^2_{\alpha/2,df} \ \leq\ \chi^2_0 \ \leq\ \chi^2_{1-\alpha/2,df}$$

Pasa si χ^2_0 cae dentro de la zona de aceptación.

Resultado: Prueba de varianza

n	1,000,000
s^2 (muestral, $n-1$)	0.0832858941
σ^2_{teo}	0.0833333333
df	999,999
χ^2_0	999,429.7301
Zona aceptación $[\chi^2_L, \chi^2_U]$	[997,229.0885, 1,002,772.7001]
Decisión	**Pasa**

Prueba de corridas (arriba/abajo)

Paso 1 (umbral c): $b_i = 1$ si $x_i \geq c$, $b_i = 0$ si $x_i < c$.
$n_1 = \sum b_i$, $n_0 = n - n_1$.

Paso 2 (corridas): $R =$ número de corridas en la secuencia (b_i).

Bajo H_0 (aleatorio):

$$\mu_R = \frac{2n_0 n_1}{n} + 1, \qquad \sigma_R = \sqrt{\frac{2n_0 n_1(2n_0 n_1 - n)}{n^2(n-1)}}$$

Estadístico:

$$Z_0 = \frac{R - \mu_R}{\sigma_R}$$

Regla (α dos colas):

$$|Z_0| \leq z_{1-\alpha/2} \quad \Rightarrow \quad \textbf{Pasa}$$

Resultado: Prueba de corridas (Python $U(0,1)$)

n	1,000,000
Umbral c	0.5
n_0 (abajo)	499,639
n_1 (arriba)	500,361
R (corridas)	499,382
μ_R	500,000.7394
σ_R	499.9995
Z_0	-1.23748
Zona aceptación ($\alpha = 0{,}05$)	[-1.95996, 1.95996]
Decisión	**Pasa**

Prueba de huecos con dígitos

Paso 1: extraer el 1er dígito decimal $D \in \{0, \ldots, 9\}$ de cada x_i.

Paso 2: definir *hit* si $D \in H$ (ej. $H = \{0, 1, 2, 3, 4\}$), entonces

$$p = P(\text{hit}) = \frac{|H|}{10}, \qquad q = 1 - p$$

Hueco G**:** # de *no-hits* entre dos hits consecutivos.

Bajo H_0**:**

$$P(G = k) = q^k \, p$$

Estadístico (clases $0..k_{max}$ **y cola):**

$$\chi_0^2 = \sum \frac{(O_i - E_i)^2}{E_i}$$

Regla (α)**:**

$$\chi_0^2 \leq \chi_{1-\alpha,df}^2 \quad \Rightarrow \quad \textbf{Pasa}$$

Resultado: Prueba de huecos con dígitos (Python $U(0,1)$)

m (huecos)	499,638
df	6
χ^2_0	10.30493277
χ^2_{crit} ($\alpha = 0{,}05$)	12.59158724
Obs (0..5, cola)	[249,948; 124,737; 62,158; 31,318; 15,845; 7,678; 7,954]
Exp (0..5, cola)	[249,819.0; 124,909.5; 62,454.75; 31,227.375; 15,613.6875; 7,80
Decisión	**Pasa**

Prueba de huecos con números

Hit: $U \in [u_0, u_1), \quad p = u_1 - u_0, \quad q = 1 - p \quad$ (*si* $X \sim U(a,b)$: $U = \frac{X-a}{b-a}$)

Hueco G**:** # de *no-hits* entre dos hits consecutivos.

Bajo H_0**:**

$$P(G = k) = q^k\, p$$

Clases: $k = 0, \ldots, k_{max}$ y cola $\geq k_{max} + 1$. $\quad E_k = m\, q^k\, p$, $E_{\geq} = m\, q^{k_{max}+1}$

Estadístico:

$$\chi_0^2 = \sum \frac{(O_i - E_i)^2}{E_i}$$

Regla (α)**:**

$$\chi_0^2 \leq \chi^2_{1-\alpha,df} \quad \Rightarrow \quad \textbf{Pasa}$$

Resultado: Prueba de huecos (números) (Python $U(0,1)$)

n	1,000,000
Intervalo $[u_0, u_1)$	[0.0, 0.5)
$p = u_1 - u_0$	0.5
k_{max}	5
m (huecos)	499,638
df	6
χ^2_0	10.30493277
χ^2_{crit} ($\alpha = 0{,}05$)	12.59158724
Obs (0..5, cola)	[249,948; 124,737; 62,158; 31,318; 15,845; 7,678; 7,954]
Exp (0..5, cola)	[249,819.0; 124,909.5; 62,454.75; 31,227.375; 15,613.6875; 7,80
Decisión	**Pasa**

Prueba de Póker (5 dígitos)

Construcción: $N = \lfloor U \cdot 10^5 \rfloor$ y se toma su cadena de 5 dígitos.

Categorías: todos dif, 1 par, 2 pares, trío, full, póker, quintilla.

Prob. teóricas ($d = 5$):

Dif 0,30240	1 par 0,50400
2 pares 0,10800	Trío 0,07200
Full 0,00900	Póker 0,00450
5 iguales 0,00010	

Chi-cuadrado:

$$\chi_0^2 = \sum \frac{(O_i - E_i)^2}{E_i}, \quad E_i = n\,p_i, \quad df = k - 1$$

Pasa si $\chi_0^2 \leq \chi^2_{1-\alpha, df}$.

Resultado: Prueba de póker (Python $U(0,1)$, $d = 5$)

n	1,000,000
d (dígitos)	5
Categorías	[dif, 1 par, 2 pares, trío, full, póker, quintilla]
Obs	[302,757; 503,189; 108,141; 72,269; 8,948; 4,601; 95]
Esp	[302,400; 504,000; 108,000; 72,000; 9,000; 4,500; 100]
df	6
χ^2_0	5.73289087
χ^2_{crit} ($\alpha = 0{,}05$)	12.59158724
Decisión	**Pasa**

Prueba de series

Idea: usar pares solapados (U_i, U_{i+1}) y verificar uniformidad en una grilla $k \times k$.

Construcción: $(U_1, U_2), (U_2, U_3), \ldots, (U_{n-1}, U_n)$, $m = n - 1$.

Celdas: $i = \lfloor kU_i \rfloor$, $j = \lfloor kU_{i+1} \rfloor$, $i, j \in \{0, \ldots, k-1\}$.

Esperado por celda:

$$E = \frac{m}{k^2}$$

Estadístico:

$$\chi_0^2 = \sum_{\text{celdas}} \frac{(O - E)^2}{E}, \qquad df = k^2 - 1$$

Regla (α):

$$\chi_0^2 \leq \chi^2_{1-\alpha, df} \;\Rightarrow\; \textbf{Pasa}$$

Resultado: Prueba de series (Python $U(0,1)$, $k = 10$)

n	1,000,000
m (pares)	999,999
k (grilla)	10
Esperado por celda E	9,999.99
df	99
χ^2_0	111.92901093
χ^2_{crit} ($\alpha = 0{,}05$)	123.22522145
Decisión	**Pasa**

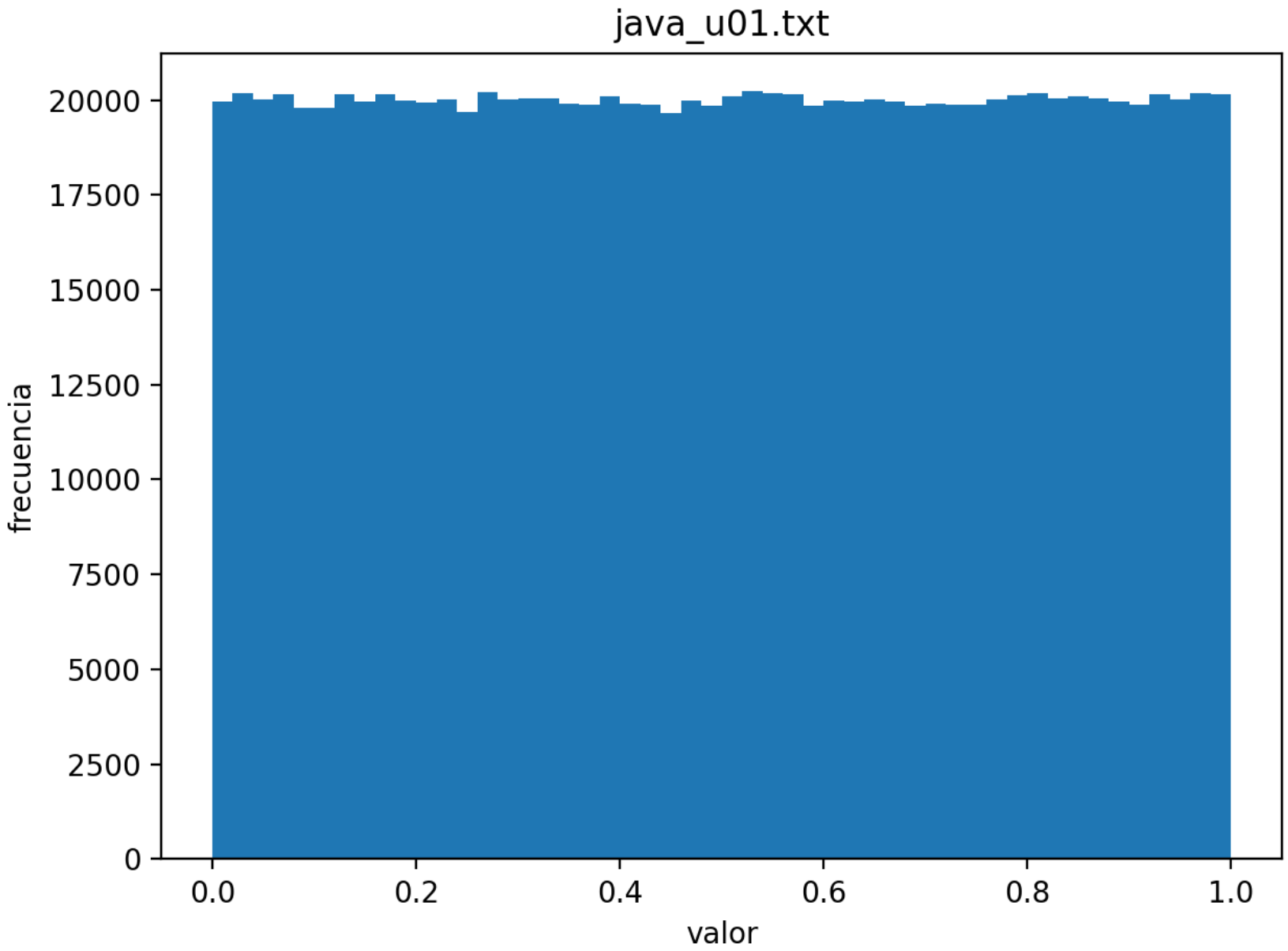
Generación de muestras por lenguaje

- Para cada lenguaje se generó una secuencia de $n = 1{,}000{,}000$ valores.
- Cada secuencia sigue el rango/distribución indicada en el enunciado (ej. $[0, 1)$ o conjuntos discretos).

Lenguaje: Java ($[0, 1)$)

- **Generador:** `SplittableRandom`
- **Función:** `nextDouble()` $\rightarrow$ valores en $[0, 1)$
- **Tamaño de muestra:** $n = 1{,}000{,}000$
- **Semilla:** `123456789` (resultados reproducibles)
- **Salida:** `java_u01.txt` (1 número por línea)

Histograma: Java (`java_u01.txt`)



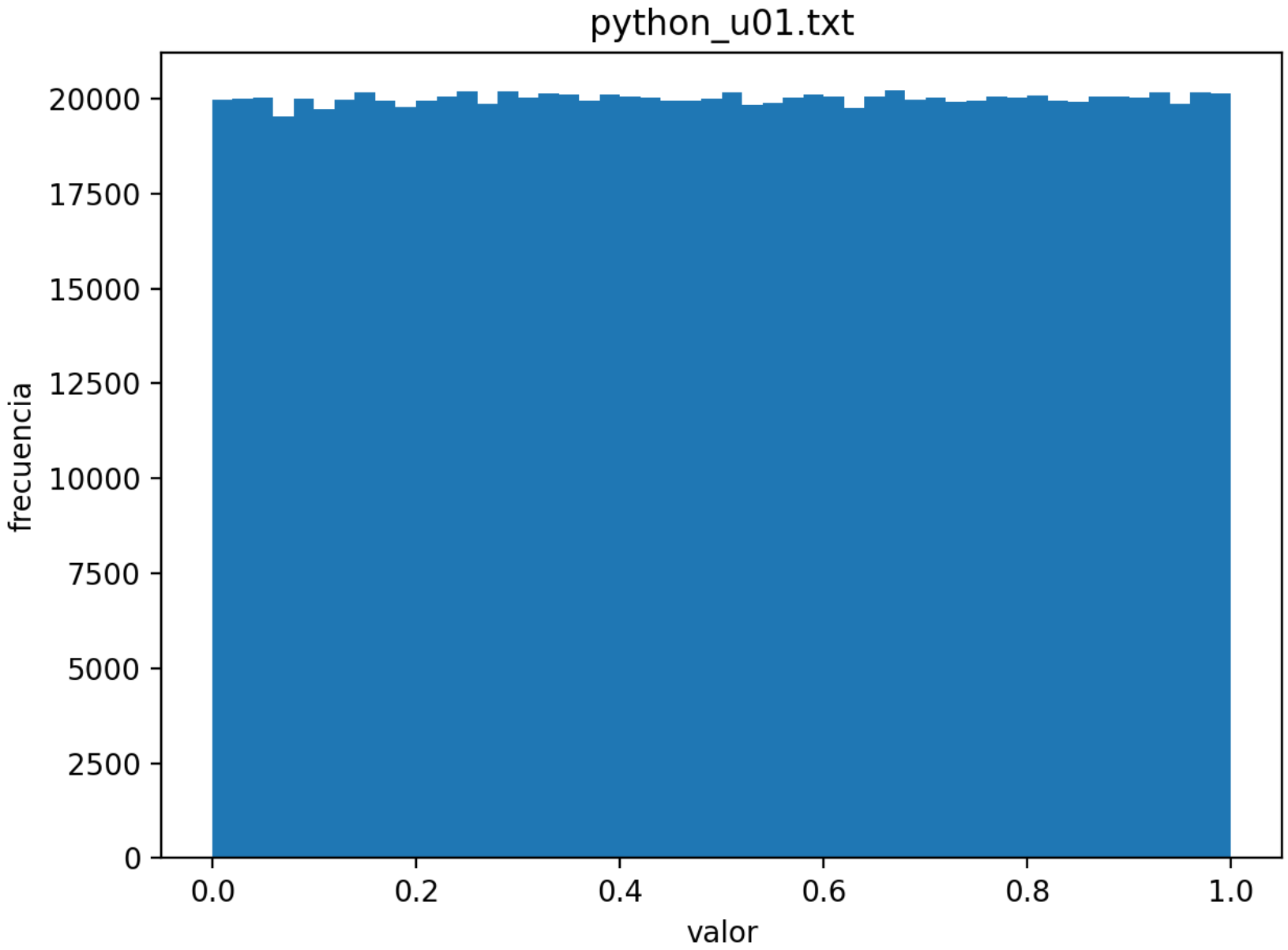
Lenguaje: Erlang ($[0, 1)$)

- **Script:** `escript` (ejecución directa)
- **Generador:** `rand` con `seed(exsplus, {...})`
- **Método:** entero uniforme $I \in \{0, \ldots, 2^{53} - 1\}$ y $X = I/2^{53} \in [0, 1)$
- **Tamaño de muestra:** $n = 1{,}000{,}000$
- **Semilla:** `123456789` (reproducible)
- **Salida:** `erlang_u01.txt` (1 número por línea)

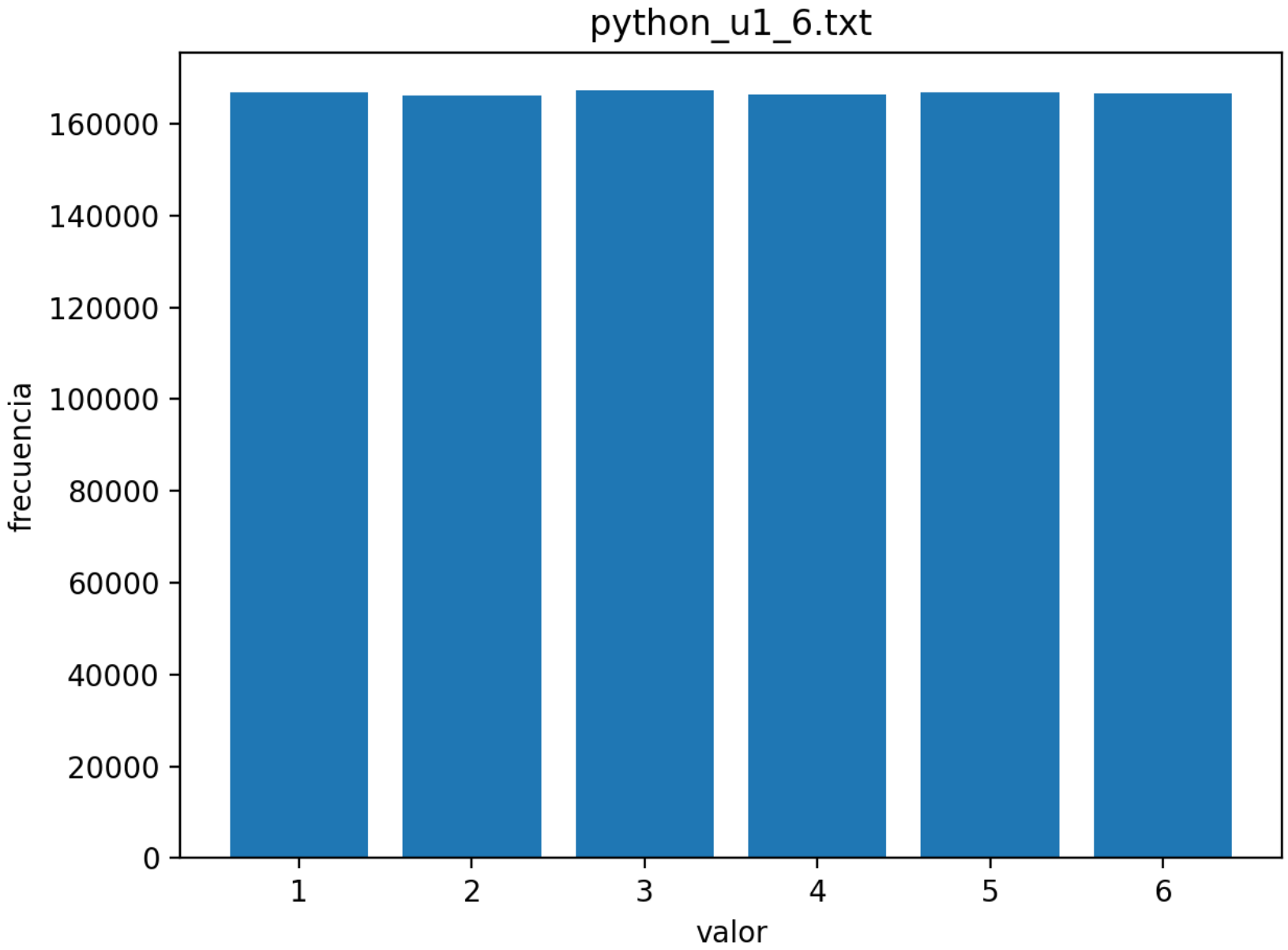
Lenguaje: Python ($[0, 1)$)

- **Generador:** módulo `random`
- **Función:** `random.random()` $\rightarrow$ valores en $[0, 1)$
- **Tamaño de muestra:** $n = 1{,}000{,}000$
- **Salida:** `python_u01.txt` (1 número por línea)

Histograma: Python (`python_u01.txt`)



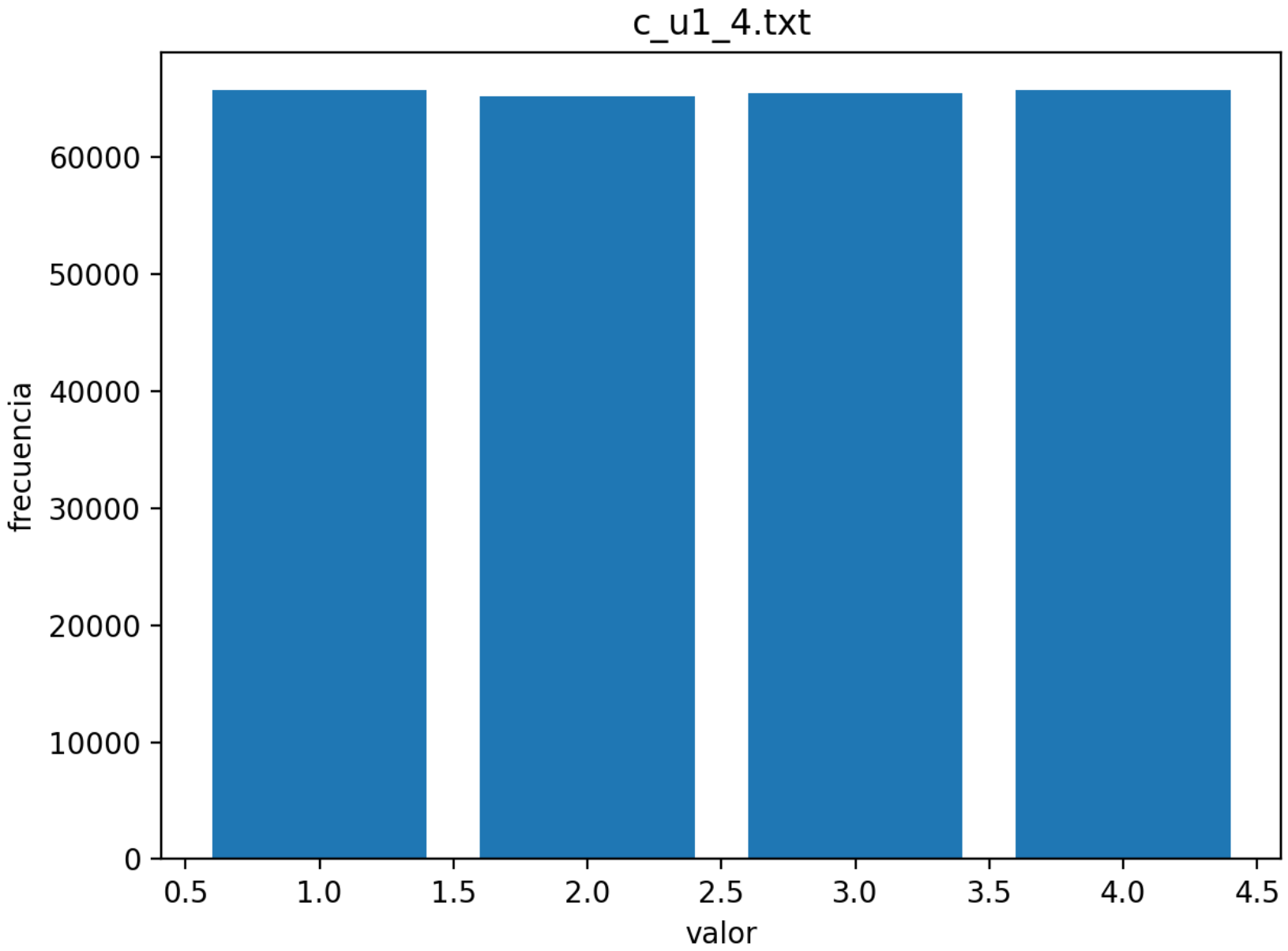
Frecuencias: Python (python_u1_6.txt = {1,...,6})



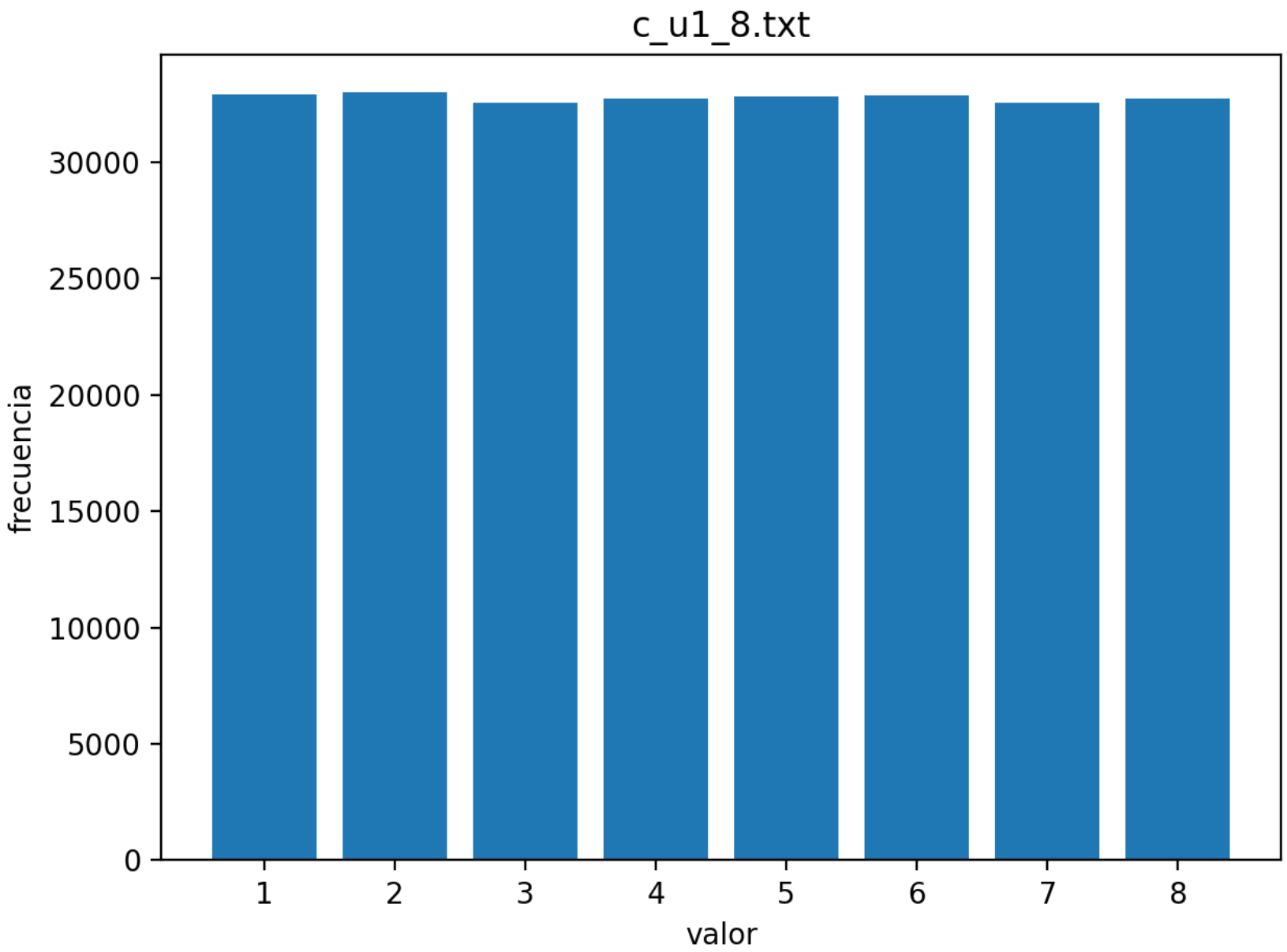
Lenguaje: C (uniforme discreta)

- **Generador base:** `rand()` con `srand(time(NULL))`
- **Método:** *rechazo* para evitar sesgo por módulo: se acepta r solo si $r < RAND\_MAX - (RAND\_MAX \bmod k)$
- **Mapeo:** $(r \bmod k) + 1 \Rightarrow \{1, \ldots, k\}$
- **Tamaño de muestra:** $n = 1{,}000{,}000$
- **Archivos:**
  - `c_u01.txt`: $\{1, 2, 3, 4\}$
  - `c_u02.txt`: $\{1, 2, 3, 4, 5, 6, 7, 8\}$

Frecuencias: C (c_u1_4.txt = {1,2,3,4})



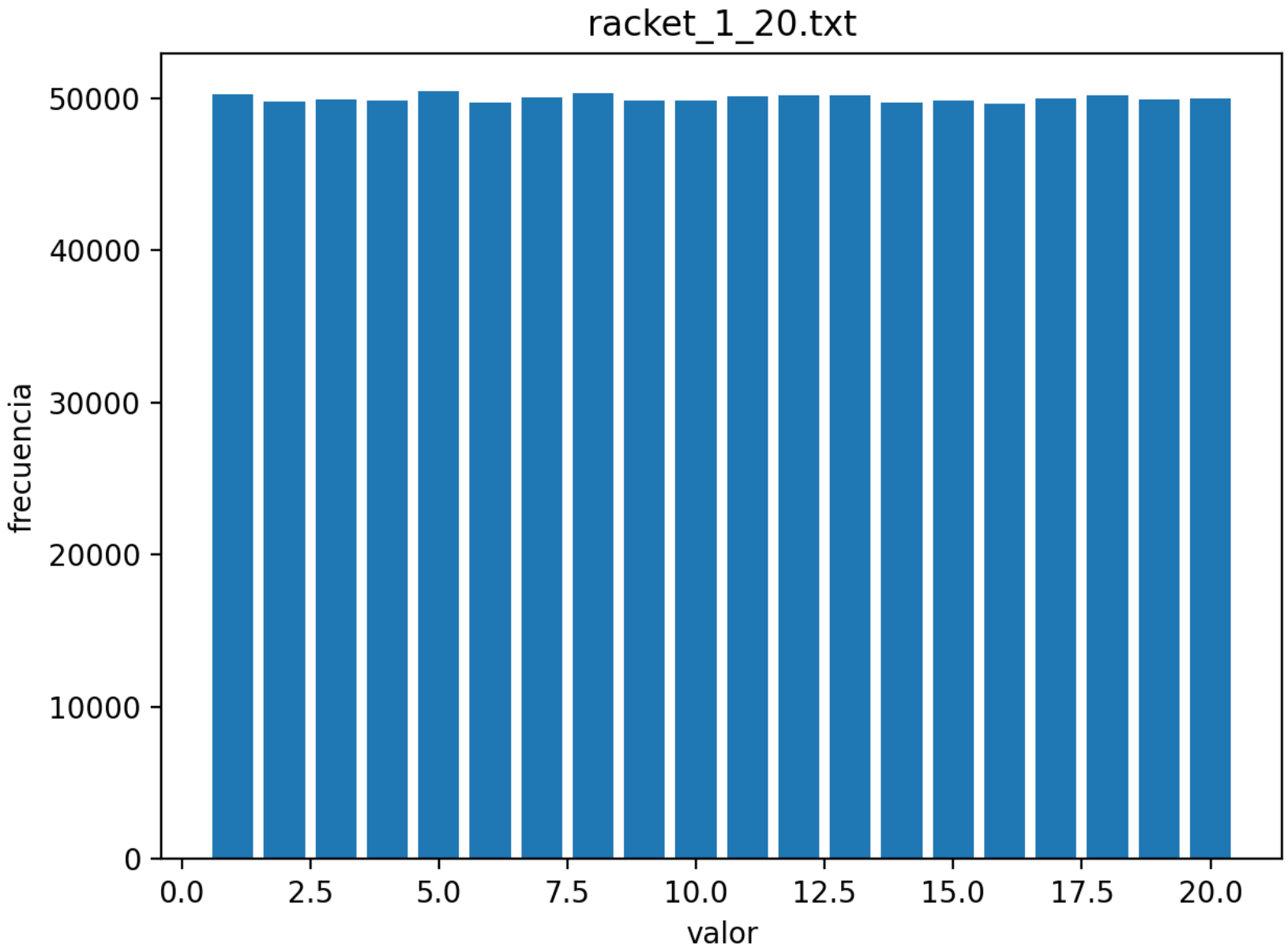
Frecuencias: C (c_u1_8.txt = {1,2,3,4,5,6,7,8})



Lenguaje: Racket (uniforme discreta)

- **Lenguaje:** Racket (`#lang racket`)
- **Generador:** `random 20`
- **Rango:** $\{1, \ldots, 20\}$ usando $1 + \texttt{random}(20)$
- **Tamaño de muestra:** $n = 1{,}000{,}000$
- **Salida:** `racket_u01.txt` (1 número por línea)

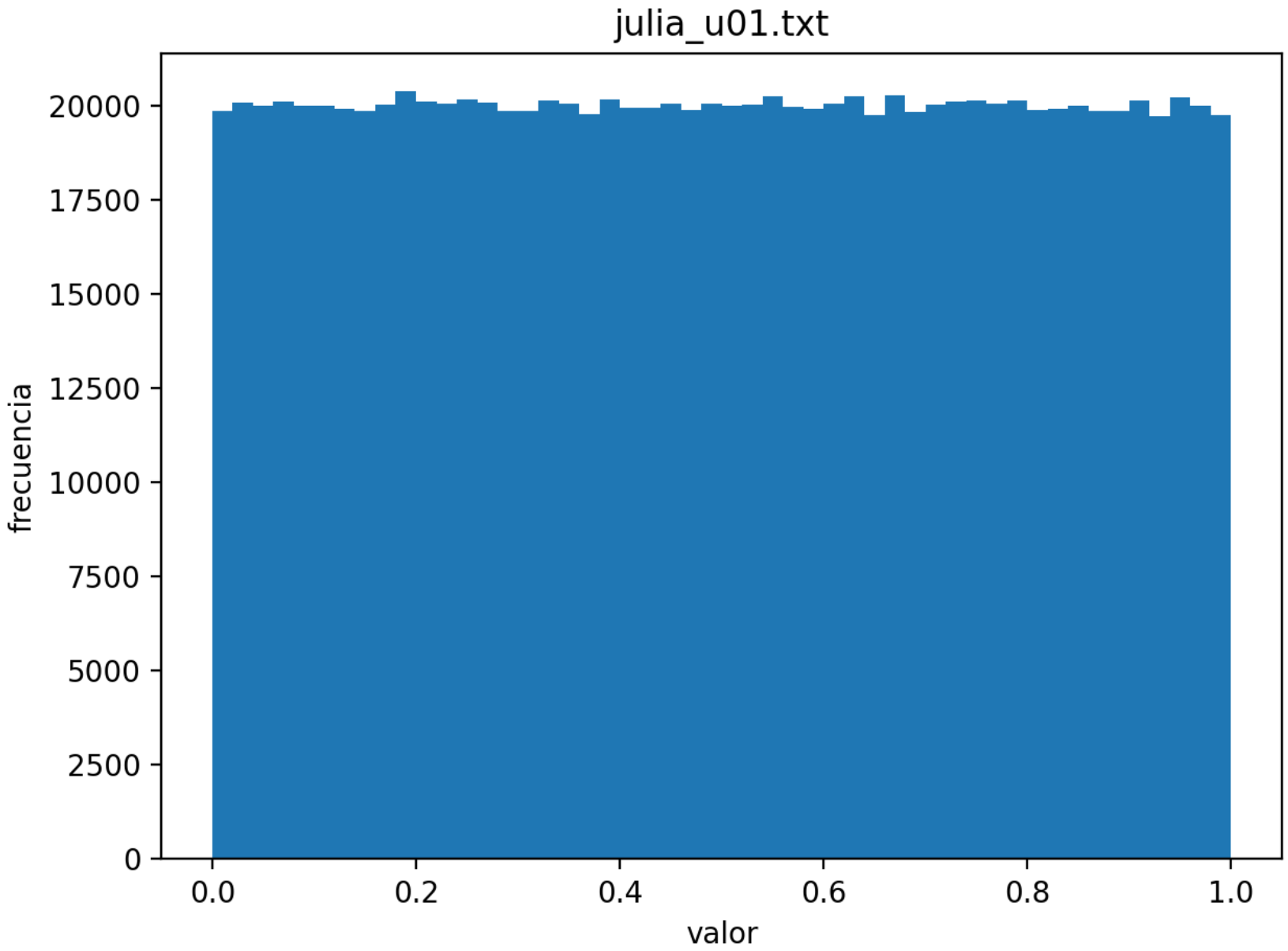
Frecuencias: Racket (`racket_1_20.txt`)



Otro Lenguaje: Julia ($[0, 1)$)

- **Generador:** función `rand()`
- **Rango:** valores reales en $[0, 1)$
- **Tamaño de muestra:** $n = 1{,}000{,}000$
- **Salida:** `julia_u01.txt`

Histograma: Julia (`julia_u01.txt`)



Conclusiones

- En las muestras continuas $U(0,1)$ (Java, Julia y Python) las pruebas de **promedio, varianza, huecos, póker y series** fueron satisfactorias, lo que respalda una buena uniformidad global.
- La prueba de **corridas** falló en varias secuencias continuas, lo que sugiere que el orden (patrón arriba/abajo) puede mostrar pequeñas desviaciones aun cuando la distribución sea uniforme.
- En discretas, **C** $\{\mathbf{1..4}\}$ y **Racket** $\{\mathbf{1..20}\}$ pasaron las pruebas aplicadas; **C** $\{\mathbf{1..8}\}$ falló en corridas, indicando posible dependencia en la secuencia.
- La muestra **Python** $\{\mathbf{1..6}\}$ no pasó las pruebas, lo cual indica que esa generación/archivo no se comportó como uniforme discreta bajo los criterios usados.
- En general, las pruebas muestran que **pasar una prueba no garantiza pasar todas**; se recomienda evaluar tanto **distribución** como **independencia** con varias pruebas.