Estadística Computacional
2^{do} Cuatrimestre 2025
Trabajo Práctico

Departamento de Computación
Facultad de Ciencias Exactas y Naturales
Universidad de Buenos Aires

Análisis de distribuciones de tiempos de cómputo

Objetivo

El objetivo de este trabajo es analizar estadísticamente el tiempo de corrida de un algoritmo matemático y determinar cómo cambia en función de la carga de tarea pedida.

Introducción

La forma más sencilla de medir el tiempo de ejecución de un algoritmo es utilizando funciones provistas por el sistema operativo o el lenguaje de programación que permiten obtener la hora actual del sistema antes y después de la ejecución del algoritmo. La diferencia entre ambos tiempos nos da una estimación del tiempo que tardó en ejecutarse el algoritmo.

Algorithm 1 Medición del tiempo de ejecución simple de un algoritmo

```
tic ← obtener_hora_actual()
ejecutar_algoritmo()
toc ← obtener_hora_actual()
tiempo_de_ejecucion ← toc − tic
```

A la hora de medir tiempos de cómputo muy breves que son comparables a la precisión del reloj del sistema, es necesario repetir la ejecución de un algoritmo varias veces consecutivas, medir el tiempo total, y luego dividirlo por el número de veces que corrió el algoritmo. Nos vamos a referir a este procedimiento como *medición en lote*.

Algorithm 2 Medición del tiempo de ejecución promedio de un algoritmo en m repeticiones

```
tic ← obtener_hora_actual()
for i ← 1 to N do
    ejecutar_algoritmo()
end for
toc ← obtener_hora_actual()
tiempo_de_ejecucion ← (toc−tic)/m
```

Vamos a utilizar una función sencilla cuya complejidad algorítmica es conocida. En este caso, se eligió una función que calcula el n-ésimo número de Fibonacci utilizando una implementación recursiva ingenua. Esta función tiene una complejidad exponencial (tiempo de ejecución $O(2^n)$), lo que la hace adecuada para observar variaciones en los tiempos de ejecución.

Algorithm 3 Cálculo del n-ésimo número de Fibonacci

```
1: function FIBONACCI(n)
2:     if n ≤ 1 then
3:         return n
4:     else
5:         return Fibonacci(n − 1) + Fibonacci(n − 2)
6:     end if
7: end function
```

Vamos a medir el tiempo de ejecución de la función Fibonacci para distintos valores de n, utilizando tanto la medición simple como la medición en lote de tamaño m. Nos interesa analizar cómo es la dependencia de algún estadístico (por ejemplo, la media o la mediana) de los tiempos de ejecución a medida que aumentamos el valor de n y m.

Metodología

Entregamos un código base en Python que implementa las mediciones de tiempo descritas anteriormente. El código incluye funciones para medir el tiempo de ejecución simple y en lote, así como la implementación recursiva de la función Fibonacci. El código tiene una lista de valores de n y m para los cuales se realizarán las mediciones. Dependiendo de la computadora puede tardar hasta media hora. Se sugiere ejecutar el código en un entorno controlado para evitar interferencias que puedan afectar las mediciones de tiempo.

Luego de ejecutar el código, se obtendrá un archivo json con los tiempos de ejecución medidos para cada combinación de n y m. Cada entrada se ve así:

```
{"n":2,"m":1,"tiempo_ms":0.0007199124}
```

Test estadísticos

Nos interesa hacer tests sobre las distribuciones de tiempos obtenidas para distinguir por ejemplo si la medición en lote reduce la variabilidad de los tiempos medidos o si existen diferencias significativas entre las distribuciones de tiempos obtenidas con ambos métodos de medición. Las hipótesis nulas a testear son:

- Las distribuciones de tiempos siguen una distribución normal.
- Dado un n. Las medias de las distribuciones de tiempos obtenidas para $m = 1$ y $m > 1$ son iguales.
- La variabilidad de las distribuciones de tiempos obtenidas para $m = 1$ y $m > 1$ son iguales.

Regresión lineal

Utilizaremos regresión lineal simple ordinaria para analizar la relación entre el tiempo de ejecución y los parámetros de entrada (tamaño m del lote y valor de n). Dado que la complejidad temporal de la función Fibonacci es exponencial, cuando se analizan los tiempos de ejecución en función de n se sugiere transformar la variable de respuesta utilizando el

logaritmo antes de realizar la regresión. Esto nos permitirá interpretar los coeficientes de la regresión en términos de la tasa de crecimiento del tiempo t de ejecución individual con respecto a n. El modelo que se propone es:

$$\ln t = \beta_0 + \beta_1 n + \epsilon$$

donde β_0 es la ordenada al origen, β_1 es el coeficiente que representa el efecto de n, y ϵ es el término de error.

Cuando se analiza el tiempo total de corrida T en función del tamaño m del lote, en cambio, se puede observar un crecimiento lineal que está mejor representado por

$$T = \beta_0 + \beta_1 m + \epsilon$$

donde β_0 representa el costo fijo de ejecución y β_1 el tiempo de cada corrida individual (es decir, cuánto aumenta T por cada unidad de m).

Tareas

Estadística descriptiva

1. Mida los tiempos de ejecución de la función Fibonacci para los distintos valores de n y tamaños de lote m.

2. Considere muestras de tamaño N (por ejemplo 500) y muestre que el tiempo de ejecución es una variable aleatoria. ¿Tiene alguna distribución que parezca conocida? Utilizar histogramas, diagramas de caja u otras visualizaciones adecuadas.

3. Compare visualmente las distribuciones de tiempos obtenidas con la medición simple y la medición en lote. ¿Cómo afecta el tamaño del lote a la variabilidad de los tiempos medidos?

4. Calcule estadísticos descriptivos (media, desviación estándar, mediana, intervalo intercuartil) para los tiempos de ejecución medidos.

Inferencia: tiempo de ejecución individual (I)

5. Realice tests de normalidad sobre las distribuciones de tiempos obtenidas. ¿Las distribuciones se aproximan a una distribución normal?

6. Realice tests estadísticos (t-test[1]) para comparar las distribuciones de tiempos entre la medición simple y la medición en lote. Analice para los distintos n. ¿Existen diferencias significativas entre las dos formas de medición?

Inferencia: tiempo de ejecución individual (bootstrapping; OPTATIVO)

5. (bis) Tome por ejemplo $n = 2$, $N = 500$ y guarde los datos de tiempos individuales de $m = 5$ y $m = 10$ (o algún otro caso que le parezca interesante). Utilice bootstrapping (remuestreo con reposición) para estimar la distribución de las correspondientes medias muestrales.

[1] Utilizaremos excepcionalmente este test aunque los datos no respeten sus supuestos

6. (bis) Construya intervalos de confianza para las dos medias muestrales y decida si se puede rechazar la hipótesis nula de que las medias son iguales.

Inferencia: tiempo de ejecución individual (II)

7. Realice una regresión lineal para el tiempo total de corrida T en función del tamaño de lote m (analice diversos valores fijos de n y N; fije por ejemplo $n = 4$, $N = 2000$). ¿Hay un costo fijo en el tiempo de ejecución del algoritmo? Relacione esto con las distribuciones obtenidas previamente. ¿Cuál sería el valor del tiempo de ejecución individual?

8. Compare el tiempo de ejecución individual obtenido por promedio (I) y el tiempo obtenido por regresión lineal (II). ¿Cuál de los dos métodos espera que sobreestime sistemáticamente el tiempo de ejecución individual?

Inferencia: complejidad temporal

9. Muestre que el tiempo individual de ejecución t depende de manera no lineal del nivel n en la secuencia de Fibonacci (decida valores fijos de m y N). Linealice los datos por medio del logaritmo como se describe más arriba.

10. Realice una regresión lineal para la variable $\log t$ en función de n. A partir de esto, ¿puede estimar el valor de la base a de la exponencial en la expresión $t(n) \sim O(a^n)$)?

Forma y Fecha de entrega

- Arme una notebook con la resolución del TP. No arroje código o gráficos sin explicación: intercale celdas de texto (con ecuaciones si fuera necesario), utilice Markdown para dar formato a secciones y subsecciones, etc, redacte y describa sus resultados.

- Suba la notebook en el link habilitado a tal efecto (es suficiente con que lo suba un solo integrante del grupo). Fecha límite de entrega: domingo 23 de noviembre 23:59 hrs.