Medición de Tiempos de Ejecución

Algoritmos y Estructuras de Datos I

27 de Mayo de 2019

- ¿Cómo medimos el tiempo que tarda en ejecutar un algoritmo?
- Vamos a utilizar:
  - **clock()**: tiempo aproximado de CPU que transcurrió desde que nuestro programa fue iniciado, expresado en ticks de reloj.
  - *CLOCKS_PER_SEC*: representa el número de ticks de reloj por segundo.

Ejemplo

Queremos saber cuanto tiempo tarda en ejecutar la siguiente función

```
1  int indicePrimeraAparicion(vector<int>& v, int elem){
2      int res = -1;
3      for(int i = 0; i < v.size(); i++){
4          if(v[i] == elem){
5              res = i;
6          }
7      }
8      return res;
9  }
```

¿Cómo podemos hacer?

Medición de tiempo con clock

```
1
2  vector<int> v = {1, 2, 3, 4, 5, 6}
3
4  double t0 = clock();
5  int indice = indicePrimeraAparicion(v, 1);
6  double t1 = clock();
7
8  double tiempo = (double(t1-t0)/CLOCKS_PER_SEC);
```

Guardar tiempos

¿Cómo guardamos en un archivo cuanto tiempo tarda nuestro programa para diferentes tamaños de vectores?

Formato

```
n    tiempo
1    1e-06
501  0.001791
1001 0.006865
1501 0.015039
2001 0.02694
2501 0.042895
3001 0.064085
3501 0.084179
4001 0.107197
4501 0.136611
5001 0.170028
5501 0.203299
6001 0.240701
6501 0.281691
7001 0.329036
7501 0.376581
```

Guardar tiempos continuación

```
1
2  int i = 1; int n = 10000; int paso = 1000;
3  ofstream fout;
4  fout.open("datos.csv");
5
6  fout << "n \t" << "tiempo" <<endl;
7
8  while(i < n){
9          vector<int> v = construir_vector(i, "asc");
10
11         double t0=clock();
12         int indice = indicePrimeraAparicion(v, 1);
13         double t1 = clock();
14
15         tiempo = (double(t1-t0)/CLOCKS_PER_SEC);
16
17         fout << i << "\t" << tiempo << endl;
18
19         i +=paso;
20 }
21 fout.close()
```

Graficador

¿Cómo gráficamos los tiempos en función del tamaño de la entrada?

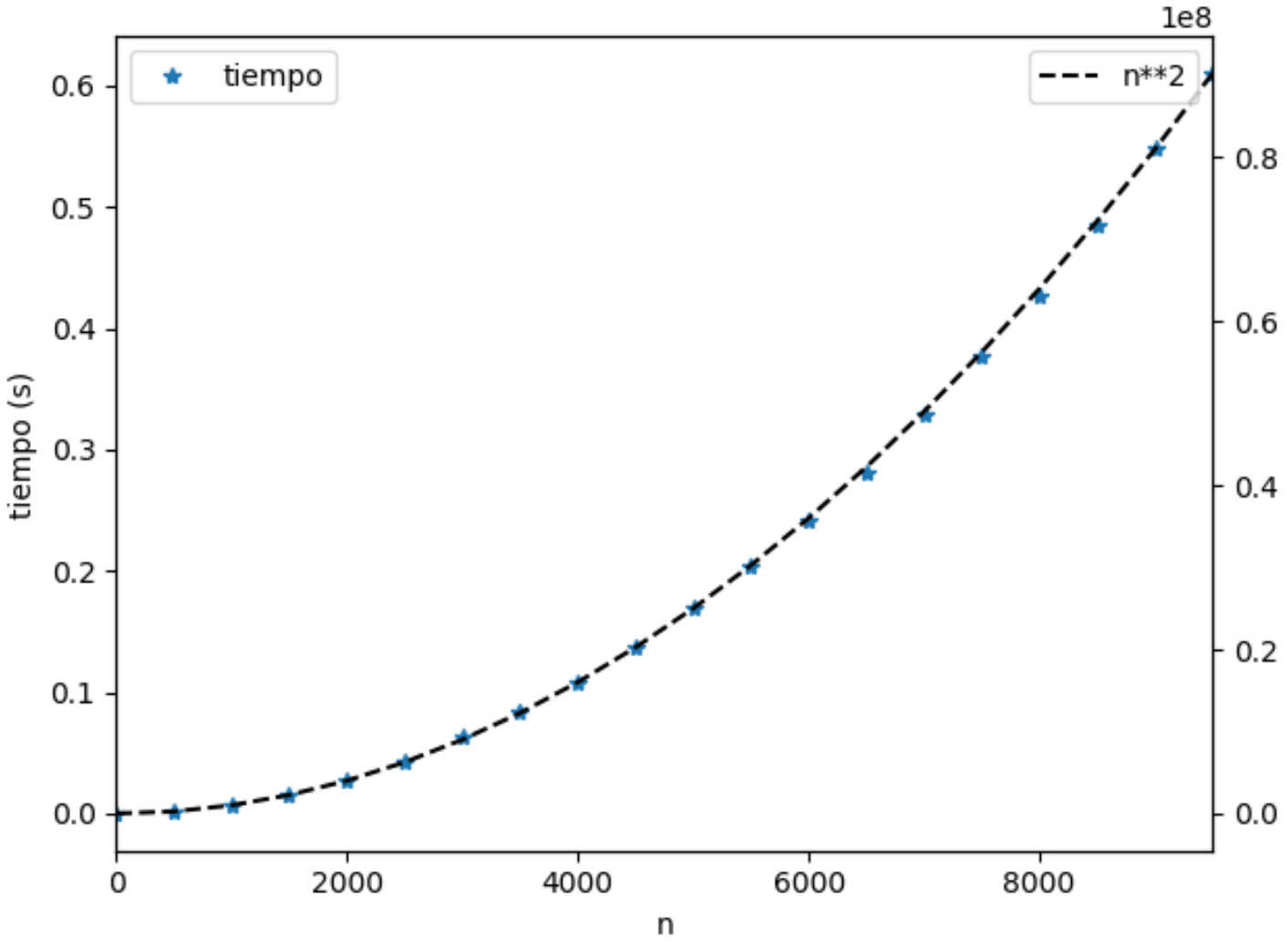
```
$python graficar.py --help
```

```
usage: graficar.py [-h] -i INPUT [-o SALIDA]
[-g {sqrt,logn,n,n2,n3,nlogn}]

graficador!

optional arguments:
  -h, --help            show this help message and exit
  -i INPUT, --input INPUT
  -o SALIDA, --salida SALIDA
  -g {sqrt,logn,n,n2,n3,nlogn}, --guia {sqrt,logn,n,n2,n3,nlogn}
```

Volvemos al ejemplo

```
$python graficar.py -i datos.csv -o lineal.png --guia n
```

Gráfico


1e8
tiempo
n**2
0.6
0.5
0.4
0.3
0.2
0.1
0.0
tiempo (s)
0.8
0.6
0.4
0.2
0.0
0
2000
4000
6000
8000
n