Aplicación del algoritmo Divide y Vencerás

Yábir G. Benchakhtir

31 de marzo de 2018

Doble Grado en Ingeniería Informática y Matemáticas

Descripción del problema

Para poner en práctica este algoritmo se nos presenta inicialmente un vector de números comprendido en un intervalo $[-M, M]$ de modo que se escojen numeros de manera aleatoria de ese intervalo y se crea un vector de un tamaño N. Posteriormente se aplica el algoritmo sort que incorporla la libería *algorithm* de *C++*.

Bajo esta hipótesis tenemos que crear un algoritmo de manera que encontremos, si existe, un elemento del vector(v) tal que $v[i] = i$.

Primer algoritmo

```
int inpos(vector<int> &v){
  int min = 0;
  int max = v.size();
  int mid;

  while (min <= max){
    mid = (max+min)/2;

    if (v[mid] == mid)
      return mid;
    else if(v[mid] < mid)
      min = mid + 1;
    else if(v[mid] > mid)
      max = mid-1;
  }

  return -1;
}
```

Comparación

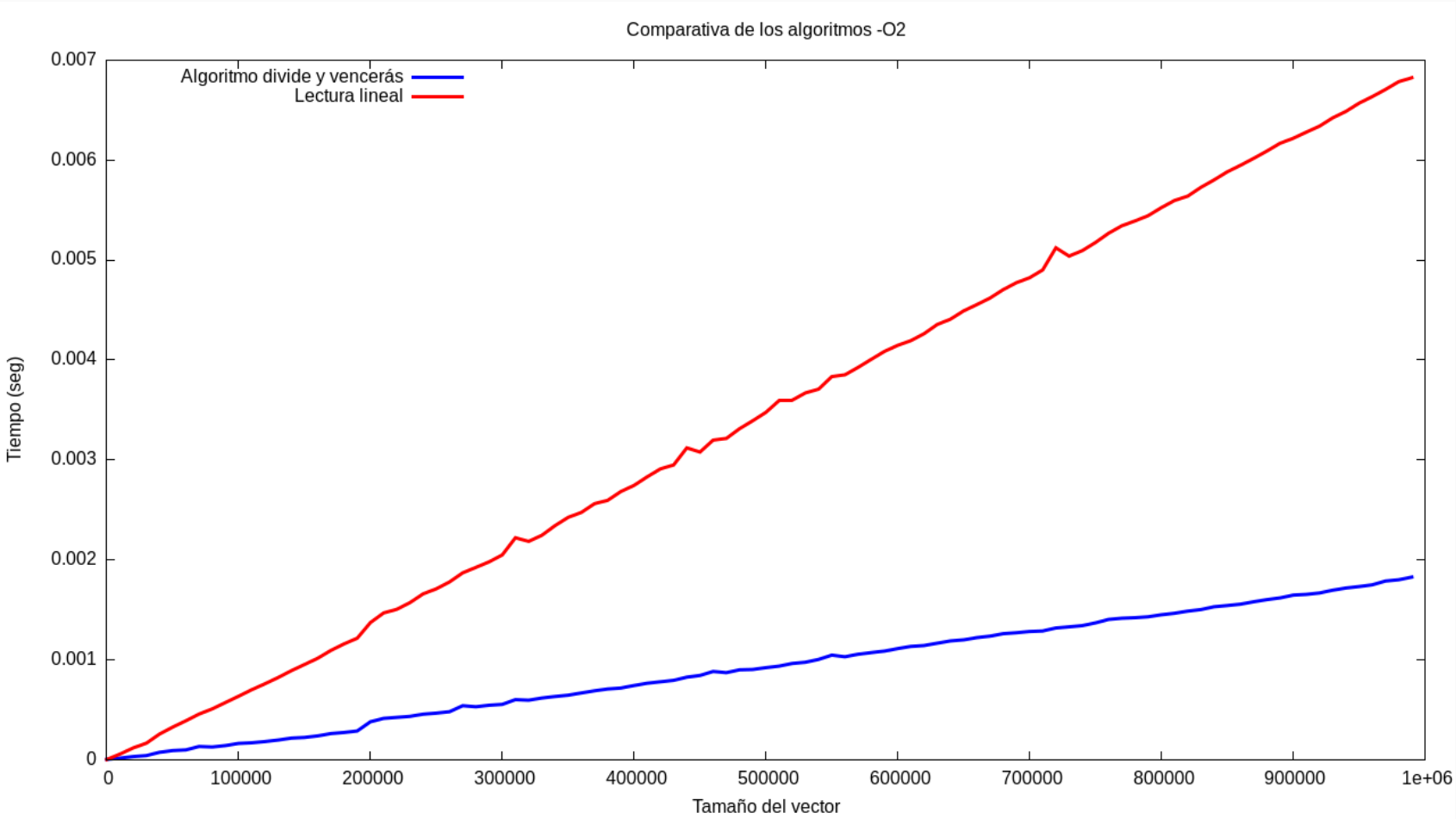
```
int inposOdd(vector<int> v){

  int i;

  for(i = 0; i < v.size(); i++)
    if(v[i] == i)
      return i;

  return -1;
}
```

Contexto

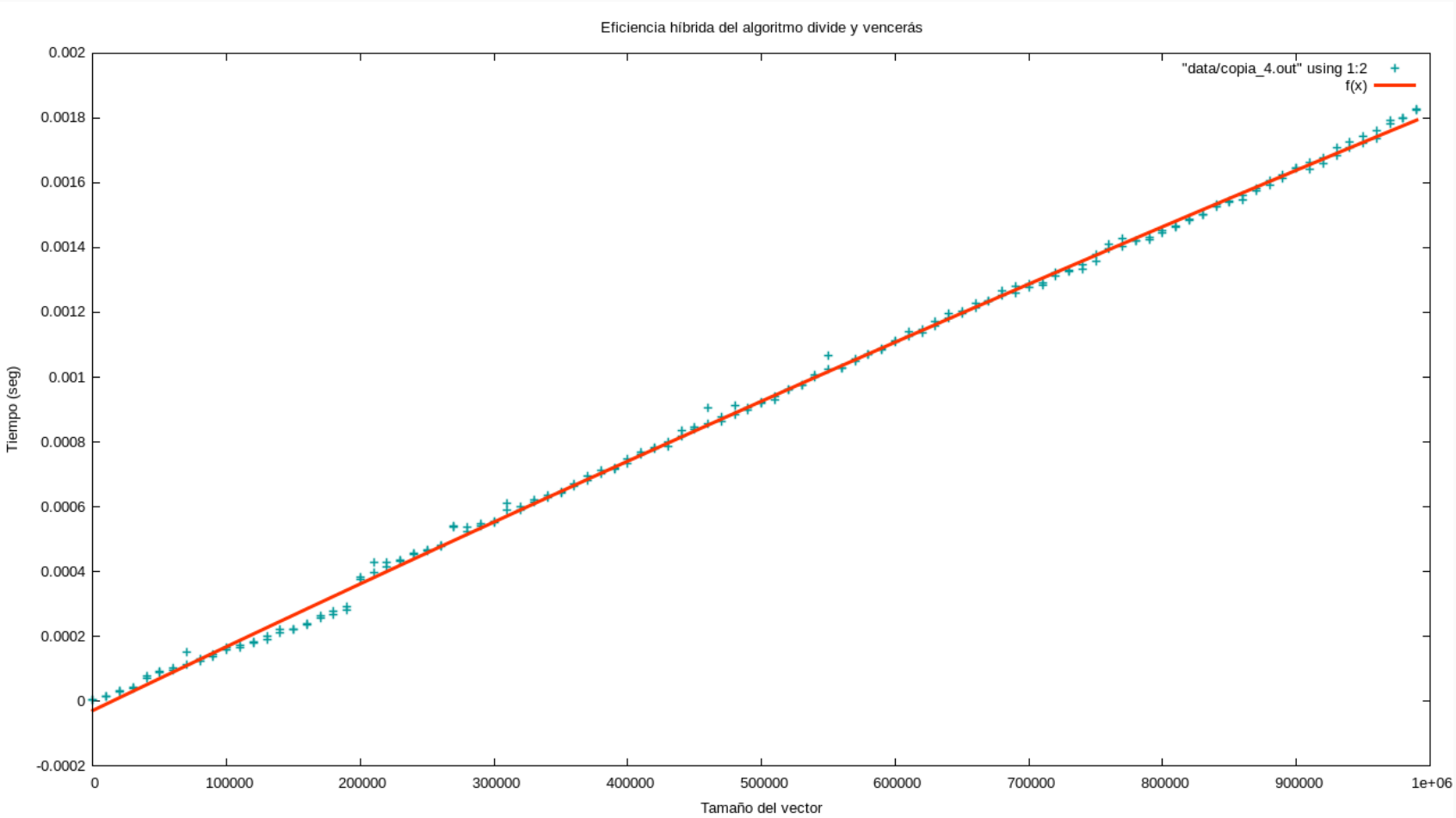
El programa descrito se ha compilado usando *g++* en su versión *g++ (Ubuntu 5.4.0-6ubuntu1 16.04.9) 5.4.0 20160609* en una máquina con las siguientes especificaciones:

- CPU: Intel Pentium G3258 (2) @ 3.200GHz
- Memoria RAM: 7876MiB
- Kernel: 4.13.0-36-generic
- OS: Linux Mint 18.3 Sylvia x86_64

Resultados


Comparativa de los algoritmos -O2
Algoritmo divide y vencerás
Lectura lineal
Tiempo (seg)
0.007
0.006
0.005
0.004
0.003
0.002
0.001
0
0
100000
200000
300000
400000
500000
600000
700000
800000
900000
1e+06
Tamaño del vector

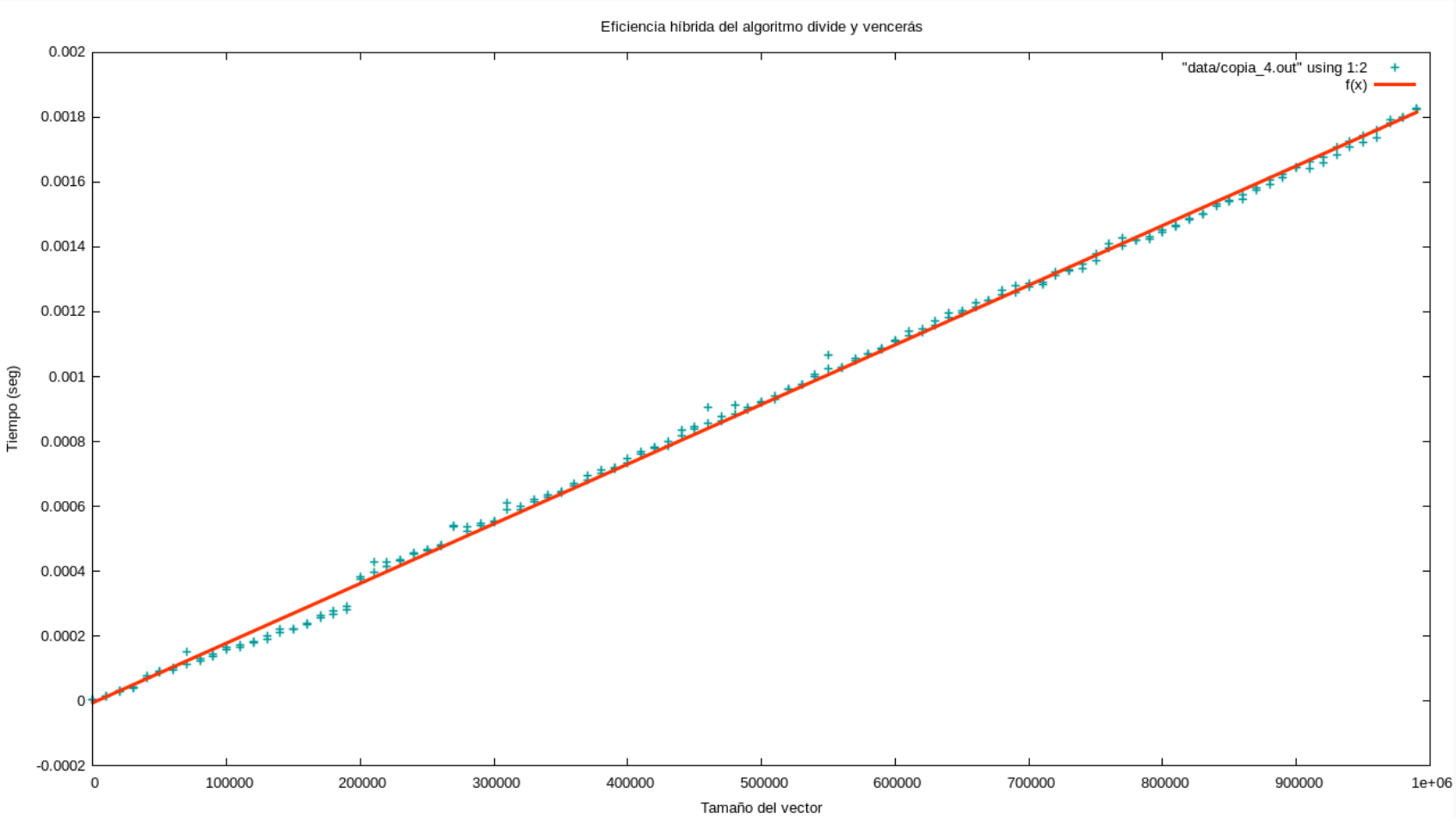
Eficiencia híbrida $log(n)$



$$f(x) = b\log(x + c) + a$$

```
Final set of parameters            Asymptotic Standard Error
=======================            ==========================

a                 = -0.189463      +/- 0.02642      (13.95%)
b                 = 0.0121242      +/- 0.001582     (13.05%)
c                 = 6.10404e+06    +/- 8.595e+05    (14.08%)
```

Eficiencia híbrida $mx + n$



$$f(x) = bx + a$$

```
Final set of parameters            Asymptotic Standard Error
=======================            ==========================

a                = -5.18976e-06    +/- 2.963e-06    (57.1%)
b                = 1.83948e-09     +/- 5.171e-12    (0.2811%)
```

Mejora de nuestro algoritmo

```
int conRepetidos(vector<int> &v, int top, int bot){

  if (bot > top)
    return -1;

  int mid = (top+bot)/2;
  int midv = v[mid];

  if (midv == mid)
    return mid;

  int lefti = min(mid-1, midv);
  int left  = conRepetidos(v, lefti, bot);

  if (left != -1)
    return left;

  int righti = max(mid+1, midv);
  int right = conRepetidos(v, bot, top);

  return right;
```

Una primera implementación

```
int inpos(int T[], int init, int len){
int mid = len/2;

if(len != 0){
  if (T[init + mid] == init+mid )
    return init+mid;
  else{
    int below = inpos(T, init, mid);
    int up = inpos(T, init + mid+1, len - mid-1);

    if(below != -1)
return below;
    if (up != -1)
return up;
  }
}

return -1;
}
```

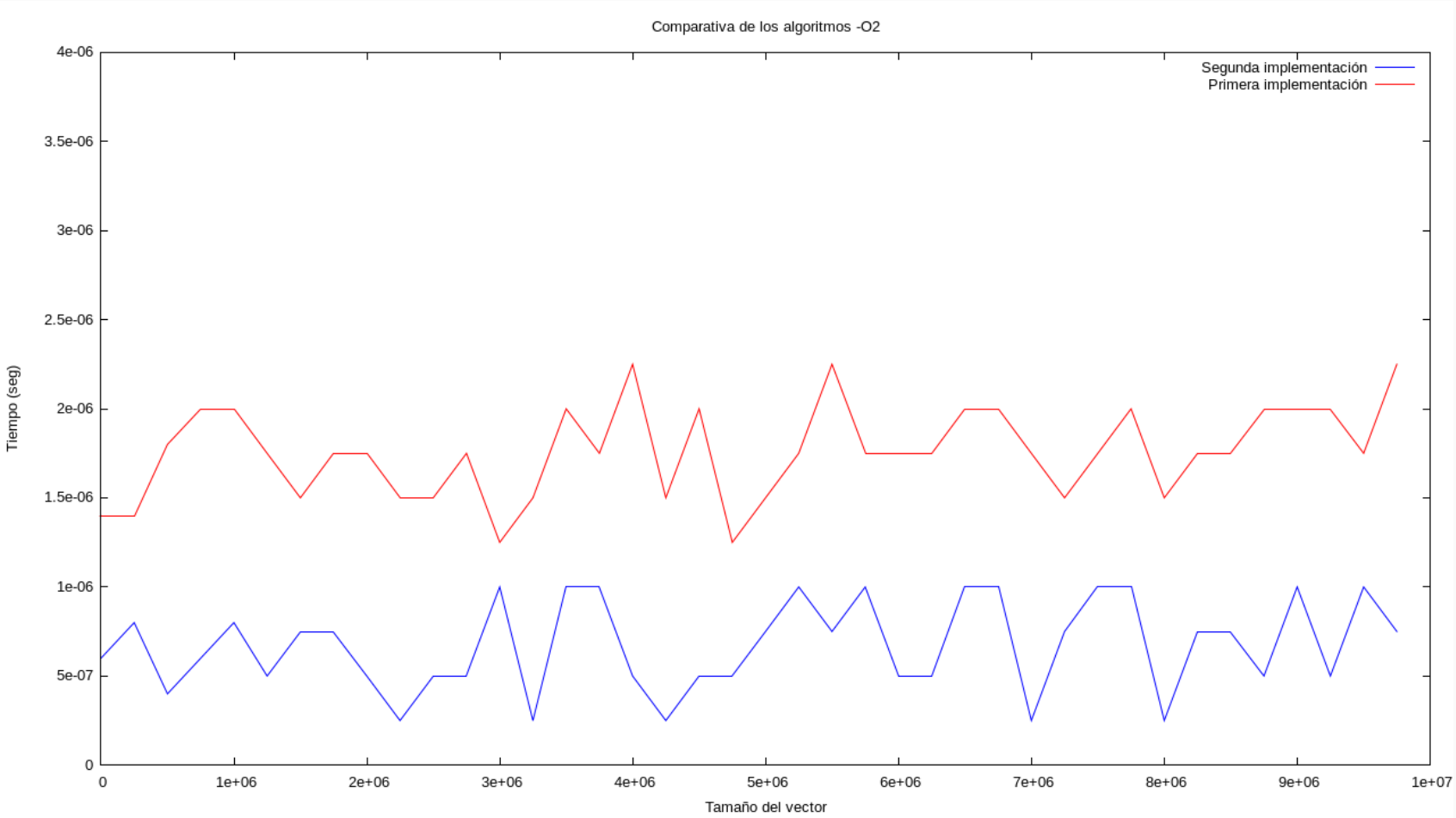


Figura 1: Comparación de ambos algoritmos DyV