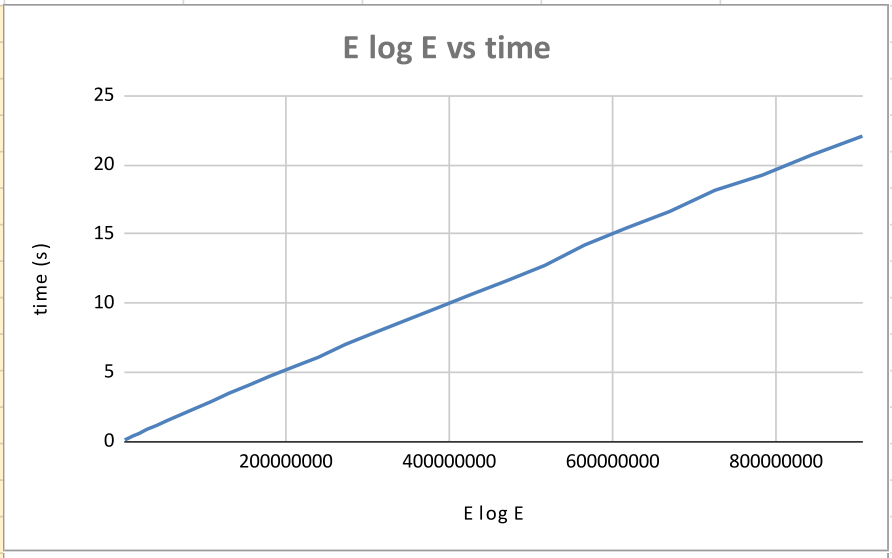
Utilizando o gerador de grafos densos (dgg) conseguimos saber exactamente quantas arestas E existem em função do número de vértices V. E = V(V-1)/2

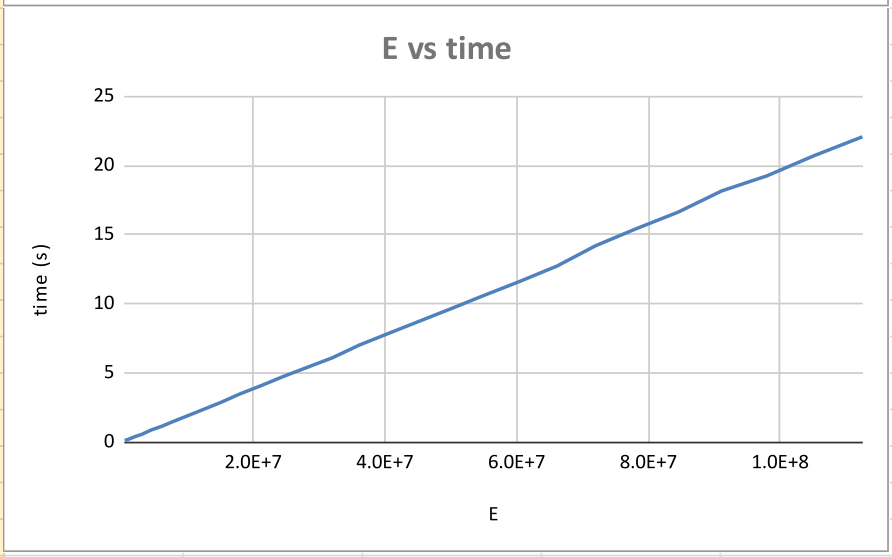
```
for i in $(seq 1000 500 10000); do echo $i; ./dgg $i > teste_$i.txt; done
```

Correndo o programa em todos os ficheiros de teste gerados

```
for file in teste_*.txt; do echo "----------------------"; echo $file; time ./p2 < $file; done
```

| V | E = V(V-1) / 2 | E log E | tempo (s) |
|---|---|---|---|
| 1000 | 499500 | 2846418.479 | 0.115 |
| 1500 | 1124250 | 6802682.611 | 0.214 |
| 2000 | 1999000 | 12595324.78 | 0.388 |
| 2500 | 3123750 | 20287745 | 0.57 |
| 3000 | 4498500 | 29928825.16 | 0.875 |
| 3500 | 6123250 | 41558387.48 | 1.145 |
| 4000 | 7998000 | 55210045.24 | 1.519 |
| 4500 | 10122750 | 70912885.51 | 1.919 |
| 5000 | 12497500 | 88692547.26 | 2.374 |
| 5500 | 15122250 | 108571954.3 | 2.877 |
| 6000 | 17997000 | 130571836.5 | 3.487 |
| 6500 | 21121750 | 154711113.7 | 4.067 |
| 7000 | 24496500 | 181007187.1 | 4.73 |
| 7500 | 28121250 | 209476164.8 | 5.39 |
| 8000 | 31996000 | 240133041.6 | 6.091 |
| 8500 | 36120750 | 272991842.5 | 7.017 |
| 9000 | 40495500 | 308065740.7 | 7.85 |
| 9500 | 45120250 | 345367155.6 | 8.72 |
| 10000 | 49995000 | 384907834 | 9.637 |
| 10500 | 55119750 | 426698919.7 | 10.62 |
| 11000 | 60494500 | 470751012 | 11.622 |
| 11500 | 66119250 | 517074216.7 | 12.726 |
| 12000 | 71994000 | 565678190.1 | 14.198 |
| 12500 | 78118750 | 616572177 | 15.42 |
| 13000 | 84493500 | 669765044.5 | 16.621 |
| 13500 | 91118250 | 725265312 | 18.156 |
| 14000 | 97993000 | 783081176.9 | 19.247 |
| 14500 | 105117750 | 843220538.3 | 20.699 |
| 15000 | 112492500 | 905691018 | 22.071 |




Ambos os plots "parecem" lineares. Talvez com mais pontos conseguissemos ver melhor as tendências, com o E log E vs time a tender ligeiramente para baixo, e o E vs time a tender ligeiramente para cima.

O melhor caso do Kruskal é o sort apenas percorrer os arcos, Ω(E). No 1º gráfico, não o distinguimos do melhor caso.

O pior caso do Kruskal é o sort ter os arcos da pior forma possível O(E log E). No 2º gráfico, não o distinguimos do pior caso.

Dado que não controlamos os pesos dos arcos e a ordem pela qual estes arcos são processados pelo sort, a complexidade do nosso algoritmo estará algures entre estas duas rectas.