# AEA – The Maximum Clique problem

Algorithm behaviour description

1) Create an initial clique using a greedy algorithm based on non-increasing degrees of the nodes and call it gBest

2) Randomly remove two vertics from the clique created in step one.

3) Add vertices to the incomplete clique returned by step two in order of non-increasing degrees.

4) If the complete clique formed in step 3 is better than gBest, gBest = (3).

5) Continue from step 2 till some termination criteria (Number of Iterations)

Strategy improvement for “Random solution”

* We improved the usage of data structures. We used in most places ArrayList as implementation for a List, while being more feasible using a LinkedList, so we made sure we adjusted these usages in order to win as much performance as possible. We also found that using an implementation of Set may help a lot in gaining performance and using a TreeMap for storing nodes sorted by their degree also was an addition in performance.
* We changed the representation of graphs. Before, we stored them in adjacent lists. Because we use a lot the operation to check if two vertices are connected, we changed to the approach of storing the graphs in adjacent matrix. This lead us to reducing the complexity of this operation from O(n) to O(1), which was a big impact from a performance point of view.

Results and improvements

* On data set [MANN\_a45](http://iridia.ulb.ac.be/~fmascia/maximum_clique/DIMACS-benchmark#detMANN_a45)
* Before improvements, 5 consecutive runs:
* Duration: 4895 milliseconds
* Duration: 4793 milliseconds
* Duration: 4882 milliseconds
* Duration: 4826 milliseconds
* Duration: 4813 milliseconds
* After improvements, 5 consecutive runs:

Duration: 3687 milliseconds

Duration: 3697 milliseconds

Duration: 3601 milliseconds

Duration: 3667 milliseconds

Duration: 3646 milliseconds