SORBONNE UNIVERSITÉ
CRÉATEURS DE FUTURS DEPUIS 1257

Dominating Set.

BM Bui-Xuan

Geometric graph : A geometric graph in a 2D plane is defined by a set of points in the plane called vertices, and a threshold on the distance between the points : there is an edge between two vertices if and only if the Euclidean distance between the two vertices is smaller than this threshold.

Dominating Set : Given a graph $G = (V, E)$, the minimum Dominating Set (MinDomSet) problem consists in computing a minimum sized subset of vertices $D \subseteq V$ such that every vertex $v \in V$ either belong to D or is a neighbour of a vertex in D.

1 Greedy algorithm

Propose a greedy heuristics for MinDomSet. Implement this method in the canvas file.

2 Naive local search

Propose and implement a naive local search heuristics in the canvas file. It is recommended to inspire from the methods seen during the last session, for example, with the naive RULE. Like with MinFSV in the previous session, the computing time of local searching could be a problem. We propose to improve this computation time following a geometrical approach, in the following section.

3 Optimisation specific to geometric graphs

Note that the naive local search RULE described in the previous TME has high complexity : given a fixed k the worst-case complexity is in $O(n^{2k-1})$, with n the number of vertices of the graph. There is a popular runtime optimization, which could be controversial in some context : the use of the *break* and *continue* instructions [1]. For instance, consider that we plan to replace three vertices A, B, C in a current candidate solution for MinDomSet by two exterior vertices. From definition of the naive RULE, we need to loop over every pair of exterior vertices in the search of one that, after replacing A, B, C by them, would still result in a dominating set. However, if the three vertices A, B, C are too far apart in the geometric model of the graph (*cf.* the threshold distance for an edge to exist), then there is no point looking for the pair of exterior vertices for the replacement : they do not exist. Accordingly, a judicious use of the *break/continue* statements can greatly speed up local searching when using the naive RULE.

Propose and implement a runtime optimization for the local search using the naive RULE when $k = 3$.

4 Other geometric optimisation (optional for the project)

Note that a geometric graph can be decomposed into convex piece : inspire from the results reported in [Nieberg, Hurink and Kern. *ACM Transactions on Algorithm*, 2008] and propose a runtime optimisation for MinDomSet. Implement and test the implementation with some examples.

1. In particular, the practice is contrary to usual IT's best-practices and contrary to most design patterns used in the industry, except for (maybe) when these *break/continue* instructions are located at the very beginning of a loop.