Project Checkpoint

Shengyun Peng Yuan Ma Qiang Wang

In this checkpoint, we've implemented two algorithms to solve Minimum Vertex Cover problem: Maximum Degree Greedy[1] for approximation and FastVC[2] for local search. And the pseudocode and respective results table are below:

Algorithm 1 Maximum Degree Greedy

Data: graph $G = (V, E)$
Result: a vertex cover of G
$C \leftarrow \emptyset$
while $E \neq \emptyset$ **do**
    select a vertex u of maximum degree
    $V \leftarrow V - \{u\}$
    $C = C \cup \{u\}$
end while
Return C

Algorithm 2 FastVC

Data: graph $G = (V, E)$, the cutoff time
Result: a vertex cover of G
$C := ConstructVC()$
$gain(v) := 0$ for each vertex $v \notin C$
while elapsed time $<$ cutoff **do**
    if C covers all edges **then**
        $C^* := C$
        remove a vertex with minimum *loss* from C
        continue
    end if
    $u := ChooseRmVertex(C)$
    $C := C \setminus \{u\}$
    $e :=$ a random uncovered edge
    $v :=$ the endpoint of e with greater *gain*, breaking ties in favor of the older one
    $C := C \cup \{v\}$
end while
Return C^*

For the **remaining** algorithms, we are going to implement Branch and Bound, and Stochastic Local Search(possibly) for Local Search.

Branch and Bound: Given a set of selected vertices which don't form a Vertex Covers set, the edges can be partitioned into two parts: edges covered by current set and ones not covered. The subproblem is to find the minimum vertex cover of those uncovered edges. To solve this, speed up the algorithm and avoid duplicated searching, for each of the nodes in our

Table 1: Result

	Approximation: Maximum Degree Greedy [1]			Fast Local search[2]		
Dataset	Time(s)	VC Value	RelErr	Time(s)	VC Value	RelErr
jazz.graph	0.03	175	0.11	0.008448804	160	0.013
karate.graph	0.002	25	0.79	0.001936625	14	0.000
football.graph	0.0016	103	0.100	0.005724291	97	0.032
as-22july06.graph	5.50	3500	0.06	0.412248884	3325	0.007
hep-th.graph	2.44	4214	0.07	0.143352889	3942	0.004
star.graph	8.24	8535	0.24	0.187682665	7040	0.0200
star2.graph	6.10	5174	0.14	0.257217559	4862	0.070
netscience.graph	0.17	920	0.02	0.104795025	899	0.000
email.graph	0.15	691	0.16	0.056203149	613	0.032
delaunay_n10.graph	0.15	889	0.26	0.082553542	747	0.063
power.graph	0.89	2910	0.32	0.095896463	2271	0.031

search tree, there will be two branches to search. One includes one more vertex remaining unused and according to best first search strategy, the vertex will be chosen as the node with the most degree considering only uncovered edges. Another, on the contrary, will not include the previous vertex, but all of its neighbor vertices, which are not covered yet. The lower bound of each branch is measured as the number of vertices used. If current branch has exceeded the global lower bound so far, the search on this branch will be terminated since it is not possible for the solution of this branch will give us a better solution than previous known one.

Stochastic Local Search[3] : This algorithm is for k-vertex cover. It takes $G = (V.E)$ and a parameter k, and searches for a vertex cover of size k of G. First, build a candidate solution by iteratively adding vertices that a maximum number of incident edges which are not covered by C until the cardinality of C is k. Second, exchange two vertices to build a neighbouring candidate solution: a vertex u in C is taken out of C according to heuristics, and a vertex v not in C is put into C randomly. And the termination criterion is when either a vertex is found, or a maximum number of steps has been reached.

References

[1] Delbot, Franois, and Christian Laforest. "Analytical and experimental comparison of six algorithms for the vertex cover problem." Journal of Experimental Algorithmics (JEA) 15 (2010): 1-4.

[2] Cai, Shaowei. "Balance between complexity and quality: local search for minimum vertex cover in massive graphs." Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI. 2015.

[3] Richter, Silvia, Malte Helmert, and Charles Gretton. "A stochastic local search approach to vertex cover." KI 2007: Advances in Artificial Intelligence. Springer Berlin Heidelberg, 2007. 412-426.