A HyFlex Module for the Max Cut Problem*

Steven Adriaensen, Gabriela Ochoa

March 11, 2015

1 Problem Formulation

Given a weighted graph G, with vertices V, edges $E \subset V \times V$ and weight function $w : E \to R$. Find a cut, i.e. a partition of V into two disjoint subsets, such that the sum of the weights of the edges crossing both partitions is maximized. The search-space consists of all possible cuts $V \to \{1, 2\}$ of G. Let p be a cut of G and $E_\times = \{(v_i, v_j) \in E | p(v_i) \neq p(v_j)\}$ the set of crossing edges, the cost of p is then given by $-\sum_{e \in E_\times} w(e)$. This domain provides 10 benchmark instances: Instances 2-7 were generated using Rudy, a graph generator by Giovanni Rinaldi[1]. Instances 0-1, 8-9 are torus graphs taken from the 7th DIMACS Implementation Challenge.[2] The properties and best known solution qualities (f_{prev}) of these instances are summarized in Table 1.

*This description is an extract from [1]

[1]The full set can be found at `http://web.stanford.edu/~yyye/yyye/Gset/`

[2]`http://dimacs.rutgers.edu/Challenges/Seventh/Instances/`

Table 1: Instances provided in the MAC domain

index	**name**	**type**	**weights**	$\lvert V\rvert$	$\lvert E\rvert$	f_{prev}
0	g3-8	torus	$\mathbb{Z}$	512	1536	41684814
1	g3-15	torus	$\mathbb{Z}$	3375	10125	283206561
2	g14	planar	1	800	4694	3064
3	g15	planar	1	800	4661	3050
4	g16	planar	1	800	4672	3052
5	g22	random	1	2000	19990	13359
6	g34	torus	1, -1	2000	4000	1384
7	g55	random	1	5000	12498	10299
8	pm3-8-50	torus	1, -1	512	1536	458
9	pm3-15-50	torus	1, -1	3375	10125	3014

2 Solution Initialisation

Vertices are greedily inserted in a random order in the partition that minimizes the cost of the cut on the sub-graph $G_{partial}$ of G, containing only the vertices and edges between vertices, that were already inserted.

3 Low Level Heuristics

3.1 Local search heuristics

0. SWAPFIRST: Changes the partition of a random vertex that improves the quality of the solution.

1. SWAPBEST: Changes the partition of the vertex improving the quality of the solution most.

2. SWAPNEIGHBOURS: Changes the partition of the 2 neighbouring vertices improving the quality of the solution most. This move is performed only once.

The SWAPFIRST and SWAPBEST heuristics are repeated for $\lceil 100\beta \rceil$ iterations or until no improving move exists.

3.2 Mutational heuristics

3. SWAPRANDOM: Changes the partition of a randomly selected vertex (repeated $\lceil 10\alpha \rceil$ times).

4. SWAPRANDOMNEIGHBOURS: Changes the partition of the 2 randomly selected neighbouring vertices (repeated $\lceil 5\alpha \rceil$ times).

3.3 Ruin-Recreate heuristics

5. RANDOMRR: Removes $\lceil 50\alpha \rceil$ random vertices and re-inserts them in a random partition.

6. GREEDYRR: Removes $\lceil 50\alpha \rceil$ random vertices and re-inserts them greedily, inserting the vertex in the partition, resulting in the best sub-cut.

7. RADIALRR: Removes $\lceil 5\alpha \rceil$ random vertices and all their neighbours, and re-inserts them as in 6.

3.4 Crossover heuristics

8. ONEPOINTXO: Performs the one point crossover on the partitioning of vertices (ordered by id).

9. MULTIPLEPARENTXO: Performs the multiple parent crossover described in [2].

References

[1] Steven Adriaensen, Gabriela Ochoa, and Ann Nowé. A benchmark set extension and comparative study for the hyflex framework. In *Evolutionary Computation (CEC), 2015 IEEE Congress on*. IEEE, 2015.

[2] Qinghua Wu and Jin-Kao Hao. A memetic approach for the max-cut problem. In *Parallel Problem Solving from Nature-PPSN XII*, pages 297–306. Springer, 2012.