Analyzing the Instances of the MaxSAT Evaluation *

Josep Argelich[1], Chu Min Li[2], Felip Manyà[3], and Jordi Planes[1]

[1] Universitat de Lleida, Lleida, Spain
[2] MIS, Université de Picardie Jules Verne, Amiens, France
[3] Artificial Intelligence Research Institute (IIIA-CSIC), Bellaterra, Spain

The MaxSAT Evaluation [1] is an affiliated event of the SAT Conference that is held every year since 2006, and is devoted to empirically evaluate exact MaxSAT algorithms solving any of the following problems: MaxSAT, Weighted MaxSAT (WMaxSAT), Partial MaxSAT (PMaxSAT), and Weighted Partial MaxSAT (WPMaxSAT).

The objective of this paper is to analyze the instances of the 2010 MaxSAT Evaluation in order to gain new insights into their computational hardness, answer some questions that have been asked to us as organizers, and evaluate how appropriate are the current settings of parameters such as timeout and available RAM memory. To this end, we conducted a number of experiments, which were performed on a cluster with 160 2 GHz AMD Opteron 248 Processors with 1 GB of RAM memory.

In the experiments, we considered the 2,675 instances of the 2010 MaxSAT Evaluation: 544 MaxSAT instances, 349 WMaxSAT instances, 1,122 PMaxSAT instances, and 660 WPMaxSAT instances. Instances were assigned to one of the following three categories: random, crafted and industrial. We used the 17 solvers that participated in MaxSAT-2010. They can be classified into three main types: branch and bound (B&B) solvers, satisfiability-based (sat-based) and unsatisfiability-based (unsat-based) solvers. In the first type, we find 10 solvers: akmaxsat, akmaxsat_ls, IncMaxSatz, IncWMaxSatz, Maxsat_Power, LS_Power, WMaxsat_Power, LSW_Power, WMaxSatz-2009, and WMaxSatz+. In the second type, we find 2 solvers: SAT4J-Maxsat,and QMaxSAT. In the third type, we find 5 solvers: WPM1, PM2, WPM2, wbo 1.4a, and wbo 1.4b.

In what follows, we summarize the experiments, point out the lessons we have learned, and suggest to introduce some modifications in forthcoming evaluations:

Experiment 1: Historical evolution. We compared how fast are the best solvers of the last evaluation compared with the best solvers that participated in previous evaluations but have not been submitted to MaxSAT-2010 on the PMaxSAT instances of the random, crafted and industrial categories. The results provide evidence that some older solvers are yet highly competitive in some categories, and suggest that we should consider the best previous solver for each problem and category until it is beaten by new solvers. On the other hand, taking into account the number of unsolved instances not yet solved by any participating solver, we should report the number of instances that have been solved for the first time in the results of the evaluation.

* Research supported by Generalitat de Catalunya (2009-SGR-1434), *Ministerio de Ciencia e Innovación* (CONSOLIDER CSD2007-0022, INGENIO 2010, Acción Integrada HA2008-0017, TIN2009-14704-C03-01, and TIN2010-20967-C04-01/03), and the *Secretaría General de Universidades del Ministerio de Educación: Programa Nacional de Movilidad de Recursos Humanos*.

Experiment 2: Analysis of the timeout. We evaluated the impact of setting a timeout of 7,200 seconds instead of the timeout of previous evaluations (1,800 seconds). The idea is to find out if it is necessary to change the current timeout because it introduces a bias in favor of some solvers. The results indicate that the current timeout is adequate for the evaluation. The introduction of a higher timeout could complicate the development of the evaluation without introducing significant differences in the results.
Experiment 3: Analysis of RAM memory. The amount of available RAM memory may produce quite different performance profiles. The cluster used in the evaluation has 2 processors per node, and they share 1 GB of RAM memory. So, we decided to evaluate the impact of setting 1GB of RAM memory instead of 512MB. The results indicate that the fact of doubling the available RAM memory does not lead to remarkable differences in performance. However, it is interesting to double the memory from time to time in order to detect anomalous situations: Maxsat_Power and LSW_Power showed a much better performance profile with 1GB, due to the way these solvers manage dynamic memory but not to the solving techniques they implement. On the other hand, it would be interesting to perform the evaluation with a cluster allowing 4GB or more of RAM memory to every solver, but this is beyond the reach of the organizers for the time being.
Experiment 4: Size of instance sets. The submitted instance sets have different size, and we rank solvers by the total number of solved instances. This may bias the results in that there may be sets of instances with a large number of instances and sets with just a few instances. Therefore, for ranking solvers by their ability to solve instances from different sets, we normalized the results taking into account the number of instances in each set. We observed that in some cases the resulting rankings are different, and propose, for future evaluations, to set a maximum of 100 instances per set, and present the results using both the ranking based on total number of solved instances and the ranking based on percentage of solved instances.
Experiment 5: Parameters of instances. We have analyzed several parameters of the instance sets: the median number of variables and clauses, the mean size of the first core found, the mean value of the solutions, and the core size multiplied by the solution. The results indicate that the problem size, the first size of the unsatisfiable core, and the number of unsatisfiability cores can give very useful indications when selecting a MaxSAT solver: when the instance has fewer than, e.g. 5,000 clauses, use a B&B solver; otherwise, search for a unsatisfiable core of the instance, if the core contains more than, e.g. 10 clauses, again use a B&B solver, if the hard clauses of the instances are of very simple form (e.g. binary clauses with negative literals), always use a B&B solver. In all other cases, use a sat-based or unsat-based solver. Regarding sat-based solvers, which are good on large size instances, it seems to be decisive the quality of the first upper bound.

References

1. J. Argelich, C. M. Li, F. Manyà, and J. Planes. The first and second Max-SAT evaluations. *Journal on Satisfiability, Boolean Modeling and Computation*, 4:251–278, 2008.