CHEMICAL ENGINEERING TRANSACTIONS, Volume 11, 2007
Edited by Sauro Pierucci
Copyright © 2007, AIDIC Servizi S.r.l., ISBN 978-88-95608-00-6

Comparison of Some High-Performance MINLP Solvers

Toni Lastusilta[1], Michael R. Bussieck[2] and Tapio Westerlund[1,*]

[1,*]Process Design Laboratory, Åbo Akademi University
Biskopsgatan 8, FIN-20500 ÅBO, Finland

[2] GAMS Software GmbH
Eupener Str. 135-137, D-50933 Cologne, Germany

In this paper some high-performance mixed integer non-linear programming (MINLP) solvers in GAMS (General Algebraic Modeling System) are evaluated. AlphaECP, Baron, Dicopt and SBB are the MINLP (mixed integer non-linear programming) solvers in focus. A large test set with 250 problems is used and special attention is given to a new GAMS solver, GAMS/AlphaECP, which has recently been included in GAMS. The results show that each solver improves the overall performance of the GAMS system. Furthermore the results reveal that the combined performance of some pairs of solvers is significantly better than other pairs.

1. Introduction

The selection of a flexible modeling system as well as the selection of suitable solvers by which a generated model can be solved is often a critical issue in obtaining valuable results in modeling applications. For mathematical programming application the GAMS (General Algebraic Modeling System) (Brooke, 1992) is a frequently used high-level modeling system with several features tailored for solving complex models. The language allows the user to build large maintainable models and the system is especially designed for modeling of linear, nonlinear and mixed integer optimization problems. Other frequently used modeling languages for similar types of applications are, for example, AMPL (Fourer, 2002) and LINDO (Schrage, 1997).
GAMS includes several optional solvers with different performance and characteristics. In the present paper three high-performance MINLP solvers in the GAMS software, SBB, Dicopt and Baron are tested on a large set of different mathematical programming problems together with a new GAMS MINLP-solver implementation. The latter MINLP-solver, GAMS/AlphaECP, in this paper called AlphaECP, is a new solver in GAMS. The purpose of the test runs is to compare the performance of the new solver with the already available and frequently used GAMS MINLP-solvers.

2. Setup

MINLP World (http://www.gamsworld.org/minlp/) offers a MINLP problem library, MINLPLib, with currently 250 models from small scale literature models to large scale real world models. The models are from areas like Agricultural Economics, Chemical-, Civil-, and Electrical Engineering, Finance, Management and Operations Research

(Bussieck et al., 2003). The MINLPLib library has earlier been used to compare the performance of, for example, Dicopt and SBB (Bussieck and Drud, 2001), but at that time it contained only 139 problems. The MINLP problem library, now containing 250 problems, is used in the present paper to test the performance of the aforementioned solvers. The test problems were executed with GAMS version 22.3 on a 3,2 GHz Pentium 4 PC with 1 GB memory. The default settings for each solver were used and the maximum allowed CPU time for each problem was 1000 seconds (~17 minutes). An additional test for AlphaECP was performed by increasing the time limit to 21600 seconds (6 hours) to determine the specific performance of this solver. For AlphaECP, Dicopt and Baron the default MILP (mixed integer linear programming) solver for sub problems is GAMS/Cplex 10 from Ilog. For Baron, Dicopt and SBB the default NLP (non linear programming) solvers for sub problems were used. The default NLP solver for Dicopt as well as SBB is Conopt (Drud, 1992), but for Baron the default NLP solver is Minos (Murtagh et al., 2002). Optimality, indicated in the result section, is determined with a relative tolerance of 0.01% from the global optimal value. For 10 problems in MINLPLib the global optimal value is not known and therefore the solution obtained, with the solvers, is reported only as feasible for these problems.

3. Solvers

AlphaECP is a C programming language implementation of the ECP (Extended Cutting Plane) method, presented in (Westerlund and Pörn, 2002). In AlphaECP only a sequence of MILP problems are solved. The intermediate MILP problems are solved to feasibility or optimality. The nonlinear constraints are evaluated at every MILP solution and for a set of nonlinear constraints that are not satisfied at the solution point, linearizations are added to the MILP problem. Linearizations are added and MILP problems are solved until all constraints are satisfied. Note that no NLP problems need to be solved and that the number of function and gradient evaluations is therefore often several magnitudes less than in solvers based on solving NLP sub problems (Emet, 2004). On the other hand the time usage when solving the MILP sub problems may be quite large (Emet, 2004) if the sub problems are solved to optimality and in some cases also when solving them only to feasibility. The method guarantees global optimum for pseudo-convex problems. In the test runs the default parameter settings are used. The default parameter settings guarantee global optimum for problems with convex objective function and pseudo-convex constraints.
The performance of AlphaECP is compared with the GAMS solvers, Baron, Dicopt and SBB. The solver manuals for these solvers can be found from http://www.gams.com/solvers/. Baron (Branch And Reduce Optimization Navigator) combines constraint propagation, interval analysis, and duality in its reduce arsenal with enhanced branch and bound concepts in its search for globally optimal solutions (Ryoo and Sahinidis, 1995) and (Nikolaos and Sahinidis, 2000). Dicopt (DIscrete and Continuous OPTimizer) is based on the outer approximation method and presented in (Viswanathan and Grossmann, 1990). SBB combines a standard Branch and Bound (B&B) method (Dakin, 1965) with some of the NLP solvers in GAMS, for example, CONOPT, MINOS or SNOPT.

4. Results and analysis

All solvers show a very good individual performance and in addition it is remarkable that each solver contribute to an overall performance within the used time limit of 1000 seconds. The relative performance between Dicopt and SBB was found to be similar to what was observed in an earlier comparison (Bussieck and Drud, 2001), where SBB performed slightly better. It was found that each solver was able to solve some problem to a better solution than all the other solvers. The combined effort of two solvers revealed that some solvers are better in some problems than other solvers. For 18 of the 250 problems no solution was found, with any of the four compared solvers, within the time limit of 1000 seconds. In Figure 1 we can see the number of feasible solutions and how many of the feasible solutions were optimal, when the problems are solved solely by the indicated solver. We can also see the number of problems for which the solvers interupted at 1000 seconds and, thus, might still improve the result. In Figure 2 the performance of any combination of two solvers is illustrated. From the figure it can be found that any combination of two solvers gives a better performance than the best single solver in some respect (compare Figure 2 with Figure 1). Also the combination D&S which gives slightly fewer feasible solutions (175) than the best single solver (177), solves 123 problems to optimality while the best single solver solves only 115 problems to optimality. Furthermore, it can be found that the overall performance can still be improved by using three and four solvers. Figure 3 reveals that each solver gives a unique contribution to the overall performance. We can see that AlphaECP found a solution to 13 problems when no other solver found any solution, and for 11 problems AlphaECP was the only solver that found an optimal solution for these problems. In

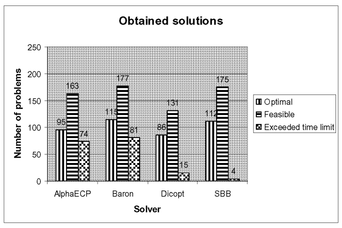

Figure 1. The solvers obtained the following amount of solutions.

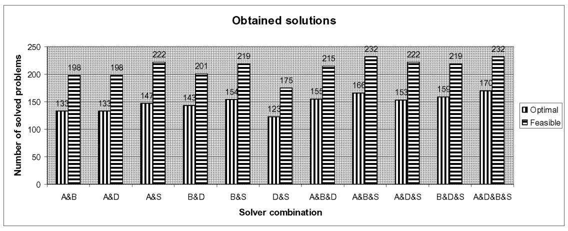

Figure 2. The combined performance of solvers (A=AlphaECP, B=Baron, D=Dicopt, S=SBB).

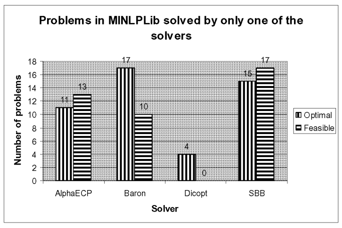

Figure 3. All solvers give a unique contribution.

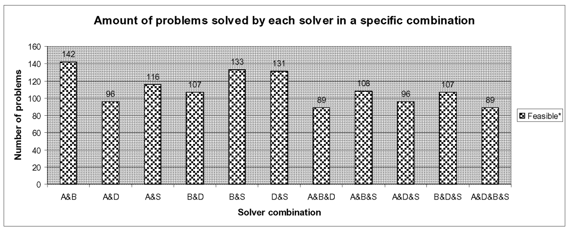

Figure 4. Amount of problems solved to feasibility, by each solver in a combination, for all possible solver combinations.

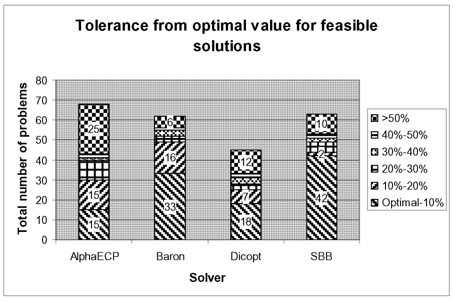

Figure 5. Quality of the feasible solutions.

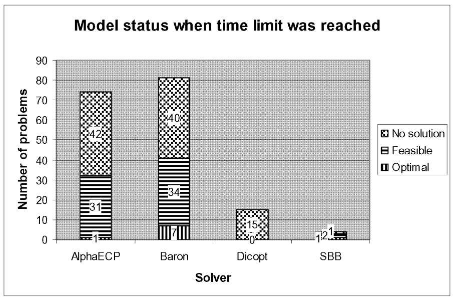

Figure 6. Model status for the problems that reached the time limit.

Figure 4 the number of problems that were solved individually to feasibility with all the solvers in the specific combination (for example both of the solvers when using only two solvers in the combination) is given. From Figure 5 we can see the quality of the feasible solutions, illustrated by the solution tolerance, defined as the difference of obtained and optimal solution relative to the optimal solution. In connection to Figure 4 and Figure 5 it was, in addition, found that the combination, AlphaECP **and** Baron, and the combination, Dicopt **and** SBB, found solutions to similar problems, when the other two solvers could not find a solution. AlphaECP **and** Baron found a solution to 34 problems when Dicopt **or** SBB did not obtain any solution. Dicopt **and** SBB found a solution to 17 problems, when the two other solvers did not obtain any solution. Moreover, AlphaECP **or** Baron found a solution to 57 problems when the two other solvers did not. For other combinations this strong coupling could not be found which implies that AlphaECP **and** Baron are able to solve different types of problems than Dicopt **and** SBB. In Figure 6 we can see that the model status for those problems which the solvers where still working on when the time limit was reached. We can see that especially AlphaECP and Baron might find additional feasible or optimal solutions to some problems if more solution time is given to these solvers. In theory Baron will always find the global optimum if given sufficient time which practically means infinite time for some problem. In total there where 115 problems for which some solver where still working when the time limit was reached. For 63 of these problems only a single solver was working when the time limit was reached. Two solvers where working on 45

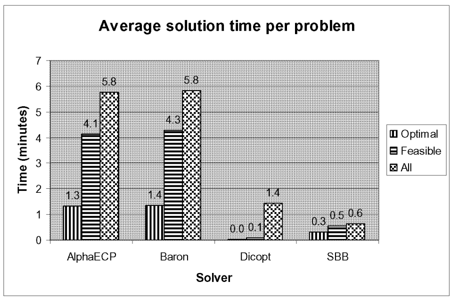

Figure 7. Average solution time per problem in MINLPLib.

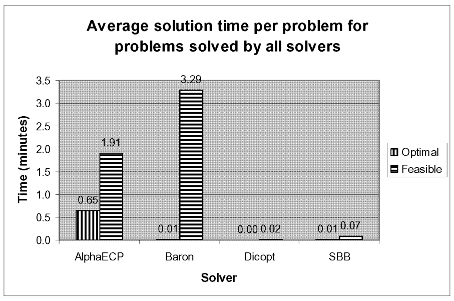

Figure 8. Average solution time per problem in MINLPLib that was solved by all compared solvers.

of these problems and three solvers were working on 7 of these problems. When for 63 problems only one solver would have needed more time it implies that different solvers had difficulties with different problems. In Figure 7 we can see the average solution time (calculated from the solution times for all the 250 problems) for each solver and also the average solution time for the problems, that were solved to feasibility and optimality with the solvers. Especially SBB can solve many of the problems very effectively. The efficiency of Dicopt and SBB is also true when comparing the time usage for the 89 problems, where all solvers found a solution. This is shown in Figure 8. On the other hand, several of the difficult problems that were solved to feasibility within 1000 CPU seconds with AlphaECP **or** Baron, terminated with no solution for Dicopt **and** SBB already after a few CPU seconds. This explains to a certain extent the higher average CPU usage, about 2 minutes, of AlphaECP and Baron.

4.1 Further testing of the performance of AlphaECP

In a second test run the CPU time limit per problem was increased to 21600 seconds (6 hours) for AlphaECP. In Figure 9 we can see that the performance of AlphaECP is slightly enhanced. The number of feasible solutions was increased by 1 and the number of optimal solutions was increased by 16.

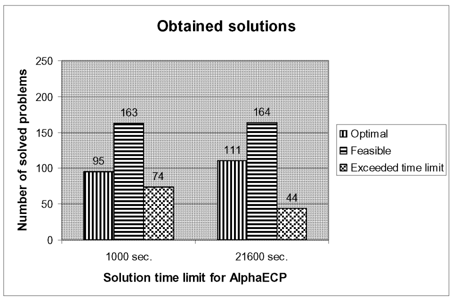

Figure 9. AlphaECP obtained the following amount of solutions with two different solution time limits.

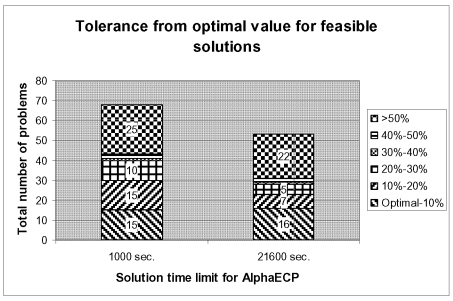

Figure 10. Comparison of feasible solution quality when using two separate solution time limits for AlphaECP.

The time limit was still reached for 44 of the problems. In Figure 10 we can see the quality of the feasible but nonoptimal solutions.

5. Conclusions

In these test runs we have seen that the new solver, AlphaECP, gives a considerable improvement to the GAMS system. From the test runs on the MINLPLib problems we found that the solvers SBB and Dicopt are in general faster to finish the optimization, but in many problems they did not find a solution when Baron or AlphaECP did. The most important observation made was that the different solvers were able to solve some problems, when no other solver obtained a solution within the specified time limit of 1000 CPU seconds, and thus giving a unique contribution to a software package where different solvers are available.

6. References

Brooke A., Kendrick D. and Meeraus A, 1992, GAMS: A User's Guide, Release 2.25. The Scientific Press, South San Francisco (CA). www.gams.com

Bussieck M.R. and Drud A.S., 2001, SBB: A New Solver for Mixed Integer Nonlinear Programming. http://www.gams.com/presentations/or01/sbb.pdf.

Bussieck M.R., Drud A.S., Meeraus A., 2003, MINLPLib - A Collection of Test Models for Mixed-Integer Nonlinear Programming. INFORMS J. Comput 15(1).

Dakin R.J., 1965, A Tree Search Algorithm for Mixed Integer Programming Problems. Computer Journal, 8, 250-255.

Drud A., 1992, CONOPT A Large-Scale GRG Code. Orsa Journal on Computing, 6, 207-216.

Emet S., 2004, A Comparative Study of Solving Some Nonconvex MINLP Problems. PhD thesis, Åbo Akademi University.

Fourer R., Kernighan B.W. and Gay D.M., 2002, AMPL: A Modeling Language for Mathematical Programming (2nd edition).

Murtagh B.A., Saunders M.A., Murray W., Gill P.E., Raman R. Kalvelagen E., 2002, GAMS/MINOS: A Solver for large-scale Nonlinear Optimization Problems.

Nikolaos V. and Sahinidis N.V., 2000, BARON Branch And Reduce Optimization Navigator. http://archimedes.scs.uiuc.edu/baron/manuse.pdf.

Ryoo H.S. and Sahinidis N.V., 1995, Global optimization of nonconvex NLPs and MINLPs with applications in process design. Computers. Chem. Engng vol. 19, 551-566.

Schrage L., 1997, Optimization Modeling with LINDO.

Viswanathan J. and Grossmann I.E., 1990, A combined penalty function and outer-approximation method for MINLP optimization. Computers. Chem. Engng vol. 14, 769-782.

Westerlund T. and Pörn R., 2002, Solving Pseudo-Convex Mixed Integer Optimization Problems by Cutting Plane Techniques. Optimization and Engineering, 3, 253-280.