Using a stabilized Benders algorithm for cell suppression
Daniel Baena, Jordi Castro, Antonio Frangioni (Universitat Politecnica de Catalunya, Catalonia)
jordi.castro@upc.edu

Abstract and Paper

The authors recently developed a stabilized Benders decomposition, which was applied to the cell suppression problem [1]. In this work we provide some numerical results with this algorithm in the solution of two sets of cells suppression instances: (i) synthetic hierarchical tables; (ii) real world instances. The resulting optimization problems have up to 24,000 binary variables, 181 million continuous variables, and 367 million constraints. The stabilized Benders approach is compared with the Benders implementation in the Tau-Argus package. In some instances Stabilized Benders provided a good solution in one minute whereas the other approach found no feasible solution in one hour.
Using a stabilized Benders algorithm for cell suppression

Daniel Baena∗, Jordi Castro**∗, Antonio Frangioni***

∗ Dept. of Statistics and Operations Research, Universitat Politècnica de Catalunya, danibaena@gmail.com

** Dept. of Statistics and Operations Research, Universitat Politècnica de Catalunya, jordi.castro@upc.edu

*** Dipartimento di Informatica, Universit di Pisa, frangio@di.unipi.it

Abstract. (This is a brief abstract of the recently accepted paper [1]).

The cell suppression problem (CSP) is one of the most widely applied post-tabular protection methods for tabular data. Given a set of sensitive cells to be protected, its aim is to find set of additional cells whose removal guarantees that estimates of values of sensitive cells fall out of a predefined protection interval. CSP is formulated as a very large mixed integer linear optimization problem. Due to its large scale is a very challenging problem for current general optimization solvers, and specialized approaches (namely, cutting planes or Benders decomposition) are needed for its solution (for instance, Benders decomposition is the algorithm used in the state-of-the-art optimal CSP implementation in Tau-Argus). However, the convergence to the optimal solution is often too slow due to well known instability issues of Benders decomposition. This work discusses a recently developed stabilized Benders method, in an attempt to avoid some of its instabilities. Some results are reported in the solution of generated and real-world CSP instances, showing the effectiveness of this approach. In some instances, stabilized Benders provided a very good solution in less than one minute, while the other available approaches (i.e., general or specialized codes, such as CPLEX or the optimal option in Tau-Argus) found no feasible solution in one hour.

1 Extended abstract

This work was recently accepted for publication in [1]. Due to copyright reasons we will only provide a short description, omitting most details (which can be found in the accepted paper).
The Cell Suppression Problem (CSP) has been used for years as one of the main post-tabular methods for statistical disclosure control in tabular data. CSP is formulated as a large mixed integer linear problem (MILP). Indeed even for tables of moderate size the resulting MILP can not be tackled by state-of-the-art solvers, and specialized methods that exploit the problem structure are needed. The usual specialized method for CSP is Benders decomposition. Benders decomposition is meant for problems with two types of variables: the “difficult” (usually binary variables) and the “easy” ones (continuous variables in most applications). In CSP the difficult binary variables model the suppression of table cells, whereas the easy continuous variables are used to impose the protection of sensitive cells.

Benders decomposition iteratively alternates between the solution of a problem in the difficult binary variables (named the “master” problem) and the easy continuous variables (named the “subproblem”). The binary variables computed by the master are passed to the subproblem. The solution of the subproblem provides information (namely, Lagrange multipliers) to refine the new master at the next iteration (namely, new constraints or cuts are added to the master). This procedure iterates until convergence is obtained. Unfortunately, convergence—although it is always guaranteed—can be slow because of well-known instability issues. Indeed the master solutions tend to oscillate between different feasible regions, sometimes jumping from a good point to a worse one at the next iteration. To avoid this bad behaviour, the authors proposed in [1] a stabilization technique for Benders decomposition, which was tuned and implemented for CSP. Briefly, the idea of the stabilization is, once a good point for the binary variables has been found, to look within a ball of a certain radius centered at the good point (this ball is named the “trust region”). Since variables are binary, we use a Hamming distance to formulate the trust region. When no better point can be found within the ball, the trust region can be discarded, and a reverse trust-region can be used. The resulting stabilized Benders approach always converges to an optimal solution (if the problems is feasible). Full details about this procedure can be found in [1].

This approach was tested on a set of 48 synthetic two-dimensional tables with one hierarchical dimension (1H2D), and 15 real-world instances from the literature. In 92% of the synthetic 1H2D tables, stabilized Benders outperformed Benders in terms of both CPU time and optimality gap of the feasible solution found. With the stabilized strategy, the average optimality gap was 0.87% whereas for CSP Benders was 2.51%. Moreover, the stabilized Benders was 1.8
faster than Benders. For real-world tables the results were not so conclusive: both algorithms reached the one hour time limit, but stabilized Benders reported a solution with a lower (thus better) optimality gap only in half of the instances. However it computed a feasible solution within the one hour time limit for all the instances, whereas Benders was not able to do it in two cases. For instance, Table 1 show the CPU time an optimality gap for two synthetic and two real world tables with both stabilized Benders and the optimal algorithm of Tau-Argus (which is based on Benders decomposition). The full set of results is available in [1].

References


<table>
<thead>
<tr>
<th>Instance</th>
<th>Stabilized Benders</th>
<th>Tau-Argus</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>gap</td>
<td>CPU</td>
</tr>
<tr>
<td>synth-1</td>
<td>0.00%</td>
<td>60.37</td>
</tr>
<tr>
<td>synth-2</td>
<td>0.00%</td>
<td>73.42</td>
</tr>
<tr>
<td>real-1</td>
<td>54.08%</td>
<td>—</td>
</tr>
<tr>
<td>real-2</td>
<td>28.92%</td>
<td>—</td>
</tr>
</tbody>
</table>

— One hour time limit reached
† One hour time limit reached without a feasible solution

Table 1: Comparison between stabilized Benders and the Tau-Argus optimal algorithm in two 1H2D synthetic and two real instances