BENDERS DECOMPOSITION WITH GAMS

ERWIN KALVELAGEN

Abstract. This document describes an implementation of Benders Decomposition using GAMS.

1. Introduction

Benders' Decomposition[2] is a popular technique in solving certain classes of difficult problems such as stochastic programming problems[7, 13] and mixed-integer nonlinear programming problems[6, 5]. In this document we describe how a Benders' Decomposition algorithm for a MIP problem can be implemented in a GAMS environment. For stochastic programming examples of Benders' Decomposition implemented in GAMS see [9, 11]. For a simple generalized Benders' model for an MINLP model see [10].

2. Benders' Decomposition for MIP Problems

Using the notation in [12] , we can state the MIP problem as:

$$\boxed{\begin{array}{lll} \text{MIP} \qquad\qquad & \underset{x,y}{\text{minimize}} & c^T x + f^T y \\ & & Ax + By \geq b \\ & & y \in Y \\ & & x \geq 0 \end{array}}$$

If y is fixed to a feasible integer configuration, the resulting model to solve is:

$$\begin{aligned} \min_x \;& c^T x \\ & Ax \geq b - B\overline{y} \\ & x \geq 0 \end{aligned} \tag{1}$$

The complete minimization problem can therefore be written as:

$$\min_{y \in Y} \left[f^T y + \min_{x \geq 0} \{ c^T x | Ax \geq b - By \} \right] \tag{2}$$

The dual of the inner LP problem is:

$$\begin{aligned} \max_u \;& (b - B\overline{y})^T u \\ & A^T u \leq c \\ & u \geq 0 \end{aligned} \tag{3}$$

Date: 20 december 2002, revised march,april 2005.

In the Benders' decomposition framework two different problems are solved. A restricted master problem which has the form:

$$\begin{aligned} \min_y \ & z \\ & z \geq f^T y + (b - By)^T \overline{u}_k, \ k = 1, \ldots, K \\ & (b - By)^T \overline{u}_\ell \leq 0, \ell = 1, \ldots, L \\ & y \in Y \end{aligned} \tag{4}$$

and subproblems of the form:

$$\begin{aligned} \max_u \ & f^T \overline{y} + (b - B\overline{y})^T u \\ & A^T u \leq c \\ & u \geq 0 \end{aligned} \tag{5}$$

The Benders' Decomposition algorithm can be stated as:

```
{initialization}
$y :=$ initial feasible integer solution
$LB := -\infty$
$UB := \infty$
while $UB - LB > \epsilon$ do
  {solve subproblem}
  $\max_u \{f^T \overline{y} + (b - B\overline{y})^T u | A^T u \leq c, u \geq 0\}$
  if Unbounded then
    Get unbounded ray $\overline{u}$
    Add cut $(b - By)^T \overline{u} \leq 0$ to master problem
  else
    Get extreme point $\overline{u}$
    Add cut $z \geq f^T y + (b - By)^T \overline{u}$ to master problem
    $UB := \min\{UB, f^T \overline{y} + (b - B\overline{y})^T \overline{u}\}$
  end if
  {solve master problem}
  $\min_y \{z | \text{cuts}, y \in Y\}$
  $LB := \overline{z}$
end while
```

The subproblem is a dual LP problem, and the master problem is a pure IP problem (no continuous variables are involved). Benders' Decomposition for MIP is of special interest when the Benders' subproblem and the relaxed master problem are easy to solve, while the original problem is not.

3. The Fixed Charge Transportation Problem

The problem we consider is the Fixed Charge Transportation Problem (FCTP)[1, 14]. The standard transportation problem can be described as:

TP
$$\begin{aligned}
\underset{x}{\text{minimize}}\ & \sum_{i,j} c_{i,j}x_{i,j}\\
& \sum_j x_{i,j} = s_i\\
& \sum_i x_{i,j} = d_j\\
& x_{i,j} \geq 0
\end{aligned}$$

The fixed charge transportation problem adds a fixed cost $f_{i,j}$ to a link $i \rightarrow j$. This can be modeled using extra binary variables $y_{i,j}$ indicating whether a link is open or closed:

FCTP
$$\begin{aligned}
\underset{x,y}{\text{minimize}}\ & \sum_{i,j}(f_{i,j}y_{i,j} + c_{i,j}x_{i,j})\\
& \sum_j x_{i,j} = s_i\\
& \sum_i x_{i,j} = d_j\\
& x_{i,j} \leq M_{i,j}y_{i,j}\\
& x_{i,j} \geq 0, y_{i,j} \in \{0,1\}
\end{aligned}$$

where $M_{i,j}$ are large enough numbers. When solving this as a straight MIP problem, it is important to assign reasonable values to $M_{i,j}$. As $M_{i,j}$ can be considered as an upper bound on $x_{i,j}$, we can find good values:

$$M_{i,j} = \min\{s_i, d_j\} \tag{6}$$

When we rewrite the problem as

$$\begin{aligned}
\min_{x,y}\ & \sum_{i,j} c_{i,j}x_{i,j} + \sum_{i,j} f_{i,j}y_{i,j}\\
& -\sum_j x_{i,j} \geq -s_i\\
& \sum_i x_{i,j} \geq d_j\\
& -x_{i,j} + M_{i,j}y_{i,j} \geq 0\\
& x_{i,j} \geq 0\\
& y_{i,j} \in \{0,1\}
\end{aligned} \tag{7}$$

we see that the Benders' subproblem can be stated as:

$$\begin{aligned}
\max_{u,v,w}\ & \sum_i(-s_i)u_i + \sum_j d_jv_j + \sum_{i,j}(-M_{i,j}\overline{y}_{i,j})w_{i,j}\\
& -u_i + v_j - w_{i,j} \leq c_{i,j}\\
& u_i \geq 0, v_j \geq 0, w_{i,j} \geq 0
\end{aligned} \tag{8}$$

The Benders' Relaxed Master Problem can be written as:

$$
\begin{aligned}
&\min_{y} z \\
&\qquad z \geq \sum_{i,j} f_{i,j} y_{i,j} + \sum_i (-s_i)\overline{u}_i^{(k)} + \sum_j d_j \overline{v}_j^{(k)} + \sum_{i,j} (-M_{i,j}\overline{w}_{i,j}^{(k)}) y_{i,j} \\
&\qquad \sum_i (-s_i)\overline{u}_i^{(\ell)} + \sum_j d_j \overline{v}_j^{(\ell)} + \sum_{i,j} (-M_{i,j}\overline{w}_{i,j}^{(\ell)}) y_{i,j} \leq 0 \\
&\qquad y_{i,j} \in \{0,1\}
\end{aligned}
\tag{9}
$$

Using this result the GAMS model can now be formulated as:

Model benders.gms. [1]

```
$ontext

   An example of Benders Decomposition on fixed charge transportation
   problem bk4x3.

   Optimal objective in reference : 350.

   Erwin Kalvelagen, December 2002

   See:
   http://www.in.tu-clausthal.de/~gottlieb/benchmarks/fctp/
   http://www.gams.com/~erwin/benders/benders.pdf

$offtext


set i 'sources' /i1*i4/;
set j 'demands' /j1*j3/;

parameter supply(i) /
   i1 10
   i2 30
   i3 40
   i4 20
/;


parameter demand(j) /
   j1 20
   j2 50
   j3 30
/;


table c(i,j) 'variable cost'
       j1    j2    j3
i1    2.0   3.0   4.0
i2    3.0   2.0   1.0
i3    1.0   4.0   3.0
i4    4.0   5.0   2.0
;



table f(i,j) 'fixed cost'
        j1     j2     j3
i1    10.0   30.0   20.0
i2    10.0   30.0   20.0
i3    10.0   30.0   20.0
i4    10.0   30.0   20.0
;

*
```

[1]http://www.amsterdamoptimization.com/models/benders/benders.gms

```
* check supply-demand balance
*
scalar totdemand, totsupply;
totdemand = sum(j, demand(j));
totsupply = sum(i, supply(i));
abort$(abs(totdemand-totsupply)>0.001) "Supply does not equal demand.";

*
* for big-M formulation we need tightest possible upperbounds on x
*
parameter xup(i,j) 'tight upperbounds for x(i,j)';
xup(i,j) = min(supply(i),demand(j));


*-------------------------------------------------------------------
* standard MIP problem formulation
*-------------------------------------------------------------------

variables
   cost   'objective variable'
   x(i,j) 'shipments'
   y(i,j) 'on-off indicator for link'
;
positive variable x;
binary variable y;

equations
   obj      'objective'
   cap(i)   'capacity constraint'
   dem(j)   'demand equation'
   xy(i,j)  'y=0 => x=0'
;

obj..        cost =e= sum((i,j), f(i,j)*y(i,j) + c(i,j)*x(i,j));
cap(i)..     sum(j, x(i,j)) =l= supply(i);
dem(j)..     sum(i, x(i,j)) =g= demand(j);
xy(i,j)..    x(i,j) =l= xup(i,j)*y(i,j);

display "--------------------- standard MIP formulation----------------------";
option optcr=0;
option limrow=0;
option limcol=0;
model fscp /obj,cap,dem,xy/;
solve fscp minimizing cost using mip;

*--------------------------------------------------------------------
* Benders Decomposition Initialization
*--------------------------------------------------------------------


display "--------------------- BENDERS ALGORITHM ----------------------------";

scalar UB 'upperbound' /INF/;
scalar LB 'lowerbound' /-INF/;

y.l(i,j) = 1;


*--------------------------------------------------------------------
* Benders Subproblem
*--------------------------------------------------------------------

variable z 'objective variable';

positive variables
   u(i) 'duals for capacity constraint'
   v(j) 'duals for demand constraint'
   w(i,j) 'duals for xy constraint'
;

equations
   subobj          'objective'
   subconstr(i,j)  'dual constraint'
;

* to detect unbounded subproblem
scalar unbounded /1.0e6/;
z.up = unbounded;

subobj..  z =e= sum(i, -supply(i)*u(i)) + sum(j, demand(j)*v(j))
                + sum((i,j), -xup(i,j)*y.l(i,j)*w(i,j))
                 ;

subconstr(i,j)..  -u(i) + v(j) - w(i,j) =l= c(i,j);

model subproblem /subobj, subconstr/;
* reduce output to listing file:
subproblem.solprint=2;
* speed up by keeping GAMS in memory:
subproblem.solvelink=2;

*---------------------------------------------------------------------
* Benders Modified Subproblem to find unbounded ray
*---------------------------------------------------------------------

variable dummy 'dummy objective variable';

equations
   modifiedsubobj          'objective'
   modifiedsubconstr(i,j)  'dual constraint'
   edummy;
;

modifiedsubobj..
    sum(i, -supply(i)*u(i)) + sum(j, demand(j)*v(j))
           + sum((i,j), -xup(i,j)*y.l(i,j)*w(i,j)) =e= 1;

modifiedsubconstr(i,j)..
    -u(i) + v(j) - w(i,j) =l= 0;

edummy.. dummy =e= 0;

model modifiedsubproblem /modifiedsubobj, modifiedsubconstr, edummy/;
* reduce output to listing file:
modifiedsubproblem.solprint=2;
* speed up by keeping GAMS in memory:
modifiedsubproblem.solvelink=2;


*---------------------------------------------------------------------
* Benders Relaxed Master Problem
*---------------------------------------------------------------------

set iter /iter1*iter50/;

set cutset(iter) 'dynamic set';
cutset(iter)=no;
set unbcutset(iter) 'dynamic set';
unbcutset(iter)=no;


variable z0 'relaxed master objective variable';
equations
   cut(iter)           'Benders cut for optimal subproblem'
   unboundedcut(iter)  'Benders cut for unbounded subproblem'
;

parameters
   cutconst(iter)     'constant term in cuts'
   cutcoeff(iter,i,j)
;

cut(cutset).. z0 =g= sum((i,j), f(i,j)*y(i,j))
                      + cutconst(cutset)
```

```
                       + sum((i,j), cutcoeff(cutset,i,j)*y(i,j));
unboundedcut(unbcutset)..
                cutconst(unbcutset)
                + sum((i,j), cutcoeff(unbcutset,i,j)*y(i,j)) =l= 0;



model master /cut,unboundedcut/;
* reduce output to listing file:
master.solprint=2;
* speed up by keeping GAMS in memory:
master.solvelink=2;
* solve to optimality
master.optcr=0;


*----------------------------------------------------------------------
* Benders Algorithm
*----------------------------------------------------------------------

scalar converged /0/;
scalar iteration;
scalar bound;
parameter ybest(i,j);
parameter log(iter,*) 'logging info';

loop(iter$(not converged),

*
* solve Benders subproblem
*
   solve subproblem maximizing z using lp;

*
* check results.
*

   abort$(subproblem.modelstat>=2) "Subproblem not solved to optimality";

*
* was subproblem unbounded?
*

   if (z.l+1 < unbounded,

*
* no, so update upperbound
*
      bound = sum((i,j), f(i,j)*y.l(i,j)) + z.l;
      if (bound < UB,
          UB = bound;
          ybest(i,j) = y.l(i,j);
          display ybest;
      );

*
* and add Benders' cut to Relaxed Master
*
      cutset(iter) = yes;

   else

*
* solve modified subproblem
*

     solve modifiedsubproblem maximizing dummy using lp;

*
* check results.
*
```

```
      abort$(modifiedsubproblem.modelstat>=2)
             "Modified subproblem not solved to optimality";


*
* and add Benders' cut to Relaxed Master
*
      unbcutset(iter) = yes;
   );


*
* cut data
*
   cutconst(iter) = sum(i, -supply(i)*u.l(i)) + sum(j, demand(j)*v.l(j));
   cutcoeff(iter,i,j) = -xup(i,j)*w.l(i,j);

*
* solve Relaxed Master Problem
*

   option optcr=0;
   solve master minimizing z0 using mip;

*
* check results.
*

   abort$(master.modelstat=4) "Relaxed Master is infeasible";
   abort$(master.modelstat>=2) "Masterproblem not solved to optimality";

*
* update lowerbound
*

   LB = z0.l;

   log(iter,'LB') = LB;
   log(iter,'UB') = UB;

   iteration = ord(iter);
   display iteration,LB,UB;

   converged$( (UB-LB) < 0.1 ) = 1;
   display$converged "Converged";

);

display log;

abort$(not converged) "No convergence";

*
* recover solution
*
y.fx(i,j) = ybest(i,j);
fscp.solvelink=2;
fscp.solprint=2;
solve fscp minimizing cost using rmip;
abort$(fscp.modelstat<>1) "final lp not solved to optimality";

display "Benders solution",y.l,x.l,cost.l;
```

The Benders' algorithm will converge to the optimal solution in 17 cycles. The values of the bounds are as follows:

```
----    309 PARAMETER log  logging info

                LB          UB

iter1      250.000     460.000
iter2      260.000     460.000
```

```
iter3      310.000     460.000
iter4      310.000     460.000
iter5      330.000     460.000
iter6      330.000     460.000
iter7      330.000     460.000
iter8      340.000     410.000
iter9      340.000     410.000
iter10     340.000     410.000
iter11     340.000     410.000
iter12     340.000     410.000
iter13     340.000     410.000
iter14     350.000     410.000
iter15     350.000     400.000
iter16     350.000     360.000
iter17     350.000     350.000
```

4. Refinements

In the example above we can add to the master problem additional restrictions on y such that only attractive proposals are generated[14]. The first condition is:

$$\sum_i s_i y_{i,j} \ge d_j \tag{10}$$

i.e. enough links should be open so that demand can be met. If the assumption holds that $\sum_i s_i = \sum_j d_j$, then we can add:

$$\sum_j d_j y_{i,j} \ge s_i \tag{11}$$

or enough links should be open such that all supply can be absorbed. In GAMS these equations look like:

```
equations
   ycon1(i)            'extra conditions on y'
   ycon2(j)            'extra conditions on y'
;
ycon1(i)..  sum(j,demand(j)*y(i,j)) =g= supply(i);
ycon2(j)..  sum(i,supply(i)*y(i,j)) =g= demand(j);

model master /cut,unboundedcut,ycon1,ycon2/;
```

These additional constraints in the master problem cause a significant faster convergence:

```
----    333 PARAMETER log  logging info

              LB          UB

iter1     330.000     460.000
iter2     330.000     460.000
iter3     340.000     410.000
iter4     340.000     350.000
iter5     350.000     350.000
```

The complete model is listed below:

Model benders2.gms. [2]

```
$ontext

   An example of Benders Decomposition on fixed charge transportation
   problem bk4x3. This formulation has a few refinements to speed
   up convergence.

```

[2]http://www.amsterdamoptimization.com/models/benders/benders2.gms

```
   Optimal objective in reference : 350.

   Erwin Kalvelagen, December 2002

   See:
   http://www.in.tu-clausthal.de/~gottlieb/benchmarks/fctp/
   http://www.gams.com/~erwin/benders/benders.pdf


$offtext


set i 'sources' /i1*i4/;
set j 'demands' /j1*j3/;

parameter supply(i) /
   i1 10
   i2 30
   i3 40
   i4 20
/;


parameter demand(j) /
   j1 20
   j2 50
   j3 30
/;


table c(i,j) 'variable cost'
       j1    j2    j3
i1    2.0   3.0   4.0
i2    3.0   2.0   1.0
i3    1.0   4.0   3.0
i4    4.0   5.0   2.0
;



table f(i,j) 'fixed cost'
        j1     j2     j3
i1    10.0   30.0   20.0
i2    10.0   30.0   20.0
i3    10.0   30.0   20.0
i4    10.0   30.0   20.0
;

*
* check supply-demand balance
*
scalar totdemand, totsupply;
totdemand = sum(j, demand(j));
totsupply = sum(i, supply(i));
abort$(abs(totdemand-totsupply)>0.001) "Supply does not equal demand.";

*
* for big-M formulation we need tightest possible upperbounds on x
*
parameter xup(i,j) 'tight upperbounds for x(i,j)';
xup(i,j) = min(supply(i),demand(j));


*---------------------------------------------------------------------
* standard MIP problem formulation
*---------------------------------------------------------------------

variables
   cost   'objective variable'
   x(i,j) 'shipments'
   y(i,j) 'on-off indicator for link'
;
```

```
positive variable x;
binary variable y;

equations
   obj      'objective'
   cap(i)   'capacity constraint'
   dem(j)   'demand equation'
   xy(i,j)  'y=0 => x=0'
;

obj..        cost =e= sum((i,j), f(i,j)*y(i,j) + c(i,j)*x(i,j));
cap(i)..     sum(j, x(i,j)) =l= supply(i);
dem(j)..     sum(i, x(i,j)) =g= demand(j);
xy(i,j)..    x(i,j) =l= xup(i,j)*y(i,j);

model fscp /obj,cap,dem,xy/;

option optcr=0;
option limrow=0;
option limcol=0;


*----------------------------------------------------------------------
* Benders Subproblem
*----------------------------------------------------------------------

variable z 'objective variable';

positive variables
   u(i) 'duals for capacity constraint'
   v(j) 'duals for demand constraint'
   w(i,j) 'duals for xy constraint'
;

equations
   subobj          'objective'
   subconstr(i,j)  'dual constraint'
;

* to detect unbounded subproblem
scalar unbounded /1.0e6/;
z.up = unbounded;

subobj..  z =e= sum(i, -supply(i)*u(i)) + sum(j, demand(j)*v(j))
                + sum((i,j), -xup(i,j)*y.l(i,j)*w(i,j))
                 ;

subconstr(i,j)..  -u(i) + v(j) - w(i,j) =l= c(i,j);

model subproblem /subobj, subconstr/;
* reduce output to listing file:
subproblem.solprint=2;
* speed up by keeping GAMS in memory:
subproblem.solvelink=2;

*----------------------------------------------------------------------
* Benders Modified Subproblem to find unbounded ray
*----------------------------------------------------------------------

variable dummy 'dummy objective variable';

equations
   modifiedsubobj          'objective'
   modifiedsubconstr(i,j)  'dual constraint'
   edummy;
;

modifiedsubobj..
    sum(i, -supply(i)*u(i)) + sum(j, demand(j)*v(j))
           + sum((i,j), -xup(i,j)*y.l(i,j)*w(i,j)) =e= 1;

modifiedsubconstr(i,j)..
```

```
    -u(i) + v(j) - w(i,j) =l= 0;

edummy.. dummy =e= 0;

model modifiedsubproblem /modifiedsubobj, modifiedsubconstr, edummy/;
* reduce output to listing file:
modifiedsubproblem.solprint=2;
* speed up by keeping GAMS in memory:
modifiedsubproblem.solvelink=2;


*---------------------------------------------------------------------
* Benders Relaxed Master Problem
*---------------------------------------------------------------------

set iter /iter1*iter50/;

set cutset(iter) 'dynamic set';
cutset(iter)=no;
set unbcutset(iter) 'dynamic set';
unbcutset(iter)=no;


variable z0 'relaxed master objective variable';
equations
   cut(iter)           'Benders cut for optimal subproblem'
   unboundedcut(iter)  'Benders cut for unbounded subproblem'
   ycon1(j)            'extra condition on y'
   ycon2(i)            'extra condition on y'
;

parameters
   cutconst(iter)     'constant term in cuts'
   cutcoeff(iter,i,j)
;

cut(cutset).. z0 =g= sum((i,j), f(i,j)*y(i,j))
                      + cutconst(cutset)
                      + sum((i,j), cutcoeff(cutset,i,j)*y(i,j));
unboundedcut(unbcutset)..
                cutconst(unbcutset)
                + sum((i,j), cutcoeff(unbcutset,i,j)*y(i,j)) =l= 0;

ycon1(j)..        sum(i, y(i,j)*supply(i)) =g= demand(j);
ycon2(i)..        sum(j, y(i,j)*demand(j)) =g= supply(i);


model master /cut,unboundedcut,ycon1,ycon2/;
* reduce output to listing file:
master.solprint=2;
* speed up by keeping GAMS in memory:
master.solvelink=2;
* solve to optimality
master.optcr=0;


*---------------------------------------------------------------------
* Benders Decomposition Initialization
*---------------------------------------------------------------------


display "--------------------- BENDERS ALGORITHM ----------------------------";

scalar UB 'upperbound' /INF/;
scalar LB 'lowerbound' /-INF/;

model feasy /edummy,ycon1,ycon2/;
* reduce output to listing file:
feasy.solprint=2;
* speed up by keeping GAMS in memory:
feasy.solvelink=2;
solve feasy minimizing dummy using mip;
```

```
display "Initial values",y.l;


*---------------------------------------------------------------------
* Benders Algorithm
*---------------------------------------------------------------------

scalar converged /0/;
scalar iteration;
scalar bound;
parameter ybest(i,j);
parameter log(iter,*) 'logging info';


loop(iter$(not converged),

*
* solve Benders subproblem
*
   solve subproblem maximizing z using lp;

*
* check results.
*

   abort$(subproblem.modelstat>=2) "Subproblem not solved to optimality";

*
* was subproblem unbounded?
*

   if (z.l+1 < unbounded,

*
* no, so update upperbound
*

      bound = sum((i,j), f(i,j)*y.l(i,j)) + z.l;
      if (bound < UB,
          UB = bound;
          ybest(i,j) = y.l(i,j);
          display ybest;
      );


*
* and add Benders' cut to Relaxed Master
*
      cutset(iter) = yes;

   else

*
* solve modified subproblem
*

     solve modifiedsubproblem maximizing dummy using lp;

*
* check results.
*

     abort$(modifiedsubproblem.modelstat>=2)
            "Modified subproblem not solved to optimality";


*
* and add Benders' cut to Relaxed Master
*
      unbcutset(iter) = yes;
   );
*
* cut data
*
   cutconst(iter) = sum(i, -supply(i)*u.l(i)) + sum(j, demand(j)*v.l(j));
   cutcoeff(iter,i,j) = -xup(i,j)*w.l(i,j);

*
* solve Relaxed Master Problem
*

   option optcr=0;
   solve master minimizing z0 using mip;

*
* check results.
*

   abort$(master.modelstat=4) "Relaxed Master is infeasible";
   abort$(master.modelstat>=2) "Masterproblem not solved to optimality";

*
* update lowerbound
*

   LB = z0.l;

   log(iter,'LB') = LB;
   log(iter,'UB') = UB;

   iteration = ord(iter);
   display iteration,LB,UB;

   converged$( (UB-LB) < 0.1 ) = 1;
   display$converged "Converged";

);

display log;

abort$(not converged) "No convergence";

*
* recover solution
*
y.fx(i,j) = ybest(i,j);
fscp.solvelink=2;
fscp.solprint=2;
solve fscp minimizing cost using rmip;
abort$(fscp.modelstat<>1) "final lp not solved to optimality";

display "Benders solution",y.l,x.l,cost.l;
```

5. Conclusion

We have shown how a standard Benders' Decomposition algorithm can be implemented in GAMS. Algorithmic development using a high level modeling language like GAMS is particular useful if complex subproblems need to be solved that can take advantage of the direct availability of the state-of-the-art LP, MIP or NLP capabilities of GAMS. Other examples of Benders algorithms written in GAMS include:

- a special form of a Generalized Benders Decomposition is used to solve a MINLP problem [8]. An example of this approach can be found in the model library model `nonsharp.gms`.

- Nested Benders Decomposition applied to a problem in hydro-thermal power generation [4].
- Generalized Benders Decomposition for large non-convex NLP's in water resource management [3]

GAMS is also used as a prototyping language. In this case a GAMS implementation of an algorithm is used to test the feasibility and usefulness of a certain computational approach. In a later stage the algorithm can be formalized and implemented in a more traditional language. Indeed, this is the way solvers like SBB and DICOPT have been developed.

6. Acknowledgements

I would like to thank Kourosh Marjani Rasmussen (Technical University of Denmark) for reporting a number of errors in an earlier version of this report.

References

1. M. L. Balinski, *Fixed Cost Transportation Problems*, Naval Research Logistics Quarterly **8** (1961), 41–54.
2. J. F. Benders, *Partitioning Procedures for Solving Mixed-Variables Programming Problems*, Numerische Mathematik **4** (1962), 238–252.
3. Ximing Cai, Daene C. McKinney, Leon S. Lasdon, and Jr. David W. Watkins, *Solving Large Nonconvex Water Resources Management Models Using Generalized Benders Decomposition*, Operations Research **49** (2001), no. 2, 235–245.
4. Santiago Cerisola and Andrès Ramos, *Benders Decomposition for Mixed-Integer Hydrothermal Problems by Lagrangean Relaxation*, 14th Power System Computation Conference, Sevilla, 2002.
5. C. A. Floudas, A. Aggarwal, and A. R. Ciric, *Global Optimum Search for Nonconvex NLP and MINLP Problems*, Computers and Chemical Engineering **13** (1989), no. 10, 1117–1132.
6. A. M. Geoffrion, *Generalized Benders Decomposition*, Journal of Optimization Theory and Applications **10** (1972), no. 4, 237–260.
7. Gerd Infanger, *Planning Under Uncertainty – Solving Large-Scale Stochastic Linear Programs*, Boyd & Fraser, 1994.
8. G. E. Paules IV and C. A. Floudas, *APROS: Algorithmic Development Methodology For Discrete-Continuous Optimization Problems*, Operations Research **37** (1989), no. 6, 902–915.
9. Erwin Kalvelagen, *Benders Decomposition for Stochastic Programming with GAMS*, `http://www.amsterdamoptimization.com/pdf/stochbenders.pdf`.
10. ———, *Some MINLP Solution Algorithms*, `http://www.amsterdamoptimization.com/pdf/minlp.pdf`.
11. ———, *Two Stage Stochastic Linear Programming with GAMS*, `http://www.amsterdamoptimization.com/pdf/twostage.pdf`.
12. Richard Kipp Martin, *Large Scale Linear and Integer Optimization; A Unified Approach*, Kluwer, 1999.
13. S. S. Nielsen and S.A. Zenios, *Scalable Parallel Benders Decomposition for Stochastic Linear Programming*, Parallel Computing **23** (1997), 1069–1088.
14. Kurt Spielberg, *On the Fixed Charge Transportation Problem*, Proceedings of the 1964 19th ACM national conference, ACM Press, 1964, pp. 11.101–11.1013.

Amsterdam Optimization Modeling Group LLC, Washington D.C./The Hague
E-mail address: `erwin@amterdamoptimization.com`