genetic Minimal Cut Sets - gMCS

Authors: Iñigo Apaolaza, University of Navarra, TECNUN School of Engineering, Spain

Luis V. Valcarcel, University of Navarra, TECNUN School of Engineering, Spain

Francisco J. Planes, University of Navarra, TECNUN School of Engineering, Spain

Reviewer(s):

INTRODUCTION

Minimal Cut Sets (MCSs) are minimal sets of reaction knockouts which deprive a network from accomplishing a given metabolic task[1]. On the other hand, genetic Minimal Cut Sets (gMCSs) consist of minimal sets of genes whose simultaneous inhibition would render the functioning of a given metabolic task impossible[2]. Therefore, the concepts of MCSs and gMCSs are equivalent but at different levels: at the reaction level for the former and at the gene level for the latter.

MCSs are not directly converted into feasible knockout strategies at a gene level due to the fact that the Gene-Protein-Reaction (GPR) rules, typically provided in genome-scale metabolic reconstructions, are not one-to-one. Certainly, we can find impractical reaction knockouts because they are catalyzed by enzymes involving several genes, which increase the cost of the intervention. On the other hand, the resulting gene knockout interventions from MCSs may provoke undesired metabolic effects, as we may have reactions (included in a particular MCS) catalyzed by an enzyme that additionally participate in other relevant reactions (not included in the MCS under consideration). This fact is further discussed in Apaolaza et al., 2017 (a)[2], (b)[3].

Moreover, state-of-the-art methods which calculate combined gene intervention strategies are based on combinatorial approaches and, as a consequence, are not viable for seeking for high order therapeutic strategies in large metabolic networks. This tutorial aims also to demonstrate that the calculation gMCSs involving a large number of genes is viable with the function provided.

EQUIPMENT SETUP

Initialize The Cobra Toolbox and select the solver (~25 sec)

If necessary, initialise the Cobra Toolbox:

```
initCobraToolbox(false) % false, as we don't want to update
```

```
      _____   _____   _____   _____     _____     |
     /  ___| /  _  \ |  _  \ |  _  \   / ___ \    |   COnstraint-Based Reconstruction and Analysis
     | |     | | | | | |_| | | |_| |  | |___| |   |   The COBRA Toolbox - 2018
     | |     | | | | |  _  { |  _  /  |  ___  |   |
     | |___  | |_| | | |_| | | | \ \  | |   | |   |   Documentation:
     \_____| \_____/ |_____/ |_|  \_\ |_|   |_|   |   http://opencobra.github.io/cobratoolbox
```

```
                                                 |

> Checking if git is installed ...  Done.
> Checking if the repository is tracked using git ...  Done.
> Checking if curl is installed ...  Done.
> Checking if remote can be reached ...  Done.
> Initializing and updating submodules (this may take a while)... Done.
> Adding all the files of The COBRA Toolbox ...  Done.
> Define CB map output... set to svg.
> TranslateSBML is installed and working properly.
> Configuring solver environment variables ...
  - [---*] ILOG_CPLEX_PATH: C:\Program Files\ibm\ILOG\CPLEX_Studio128\cplex\matlab\x64_win64
  - [----] GUROBI_PATH: --> set this path manually after installing the solver ( see instructions )
  - [----] TOMLAB_PATH: --> set this path manually after installing the solver ( see instructions )
  - [----] MOSEK_PATH: --> set this path manually after installing the solver ( see instructions )
  Done.
> Checking available solvers and solver interfaces ... Done.
> Setting default solvers ... Done.
> Saving the MATLAB path ... Done.
  - The MATLAB path was saved in the default location.

> Summary of available solvers and solver interfaces

                   Support           LP      MILP        QP      MIQP       NLP
   ----------------------------------------------------------------------
   gurobi           active                0         0         0         0         -
   ibm_cplex        active                1         1         1         -         -
   tomlab_cplex     active                0         0         0         0         -
   glpk             active                1         1         -         -         -
   mosek            active                0         -         0         -         -
   matlab           active                1         -         -         -         1
   cplex_direct     active                0         0         0         0         -
   dqqMinos         active                0         -         -         -         -
   pdco             active                1         -         1         -         -
   quadMinos        active                0         -         -         -         -
   qpng             passive               -         -         1         -         -
   tomlab_snopt     passive               -         -         -         -         0
   gurobi_mex       legacy                0         0         0         0         -
   lindo_old        legacy                0         -         -         -         -
   lindo_legacy     legacy                0         -         -         -         -
   lp_solve         legacy                1         -         -         -         -
   opti             legacy                0         0         0         0         0
   ----------------------------------------------------------------------
   Total            -                     5         2         3         0         1

+ Legend: - = not applicable, 0 = solver not compatible or not installed, 1 = solver installed.


> You can solve LP problems using: 'ibm_cplex' - 'glpk' - 'matlab' - 'pdco' - 'lp_solve'
> You can solve MILP problems using: 'ibm_cplex' - 'glpk'
> You can solve QP problems using: 'ibm_cplex' - 'pdco' - 'qpng'
> You can solve MIQP problems using:
> You can solve NLP problems using: 'matlab'

> Checking for available updates ...
--> You cannot update your fork using updateCobraToolbox(). [06867a @ gmcs_v3].
    Please use the MATLAB.devTools (https://github.com/opencobra/MATLAB.devTools).
```

Note that the approaches to search for MCS and gMCS problems are based on Mixed Integer Linear Programming (MILP). The solver selected will be Cplex.

```
changeCobraSolver('ibm_cplex', 'all');
```

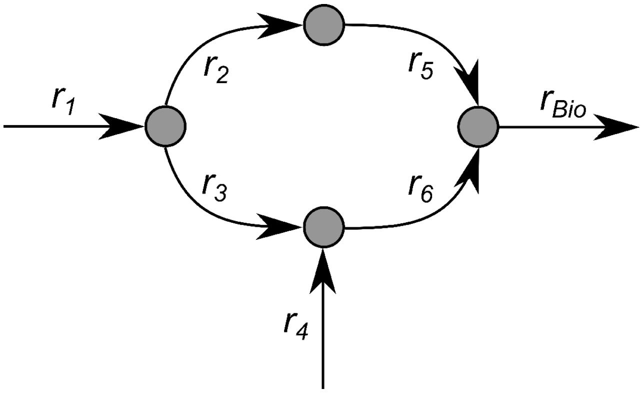
```
> IBM ILOG CPLEX interface added to MATLAB path.
> The solver compatibility is not tested with MATLAB R2017a.
> Solver for LP problems has been set to ibm_cplex.

> IBM ILOG CPLEX interface added to MATLAB path.
> The solver compatibility is not tested with MATLAB R2017a.
> Solver for MILP problems has been set to ibm_cplex.

> IBM ILOG CPLEX interface added to MATLAB path.
> The solver compatibility is not tested with MATLAB R2017a.
> Solver for QP problems has been set to ibm_cplex.
> Solver ibm_cplex not supported for problems of type MIQP. No solver set for this problemtype
> Solver ibm_cplex not supported for problems of type NLP. Currently used: matlab
```

PROCEDURE

This tutorial will be divided in two different parts. First, a toy example will be used to illustrate the difference between MCSs and gMCSs and, second, gMCSs will be calculated for Recon2.v04 metabolic model[4].

1. Toy example (~5 sec)

The toy example under study is presented below:



First, we are going to load the MAT-file which contains the metabolic network in the toy example.

```
load('gMCStoyExample.mat');
```

As different metabolic models in COBRA format, it contains the following fields: *S*, *ub*, *lb*, *rxns*, *mets*, *genes*, *rules*, *grRules*, *rxnGeneMat* and *c*.

```
rxns = model.rxns
```

```
rxns = 7×1 cell array
    'r1'
    'r2'
    'r3'
    'r4'
    'r5'
    'r6'
    'rBio'
```

```
grRules = model.grRules
```

```
grRules = 7×1 cell array
```

```
    '(g1)'
    '(g2)'
    '(g2) or (g3)'
    '(g4)'
    '(g5) and (g6)'
    '(g5)'
    ''
```

We can observe that all reactions and genes have a one-to-one relationship except for *r3* and *r5*, which have rules containing 2 different genes and an "or" and an "and" relationship, respectively. On the other hand, *g2* and *g5* take part in two different reactions each. The existence of non-trivial relationships between genes and reactions are the cause of not obtaining the same solutions with MCSs and gMCSs.

We aim to calculate MCSs which will render the biomass production impossible via the function named *calculateMCS()*. Therefore, the desired target metabolic task (represented in the field *c* of the structure) will be the biomass reaction.

```
model.rxns(logical(model.c))
```

```
ans = cell

    'rBio'
```

We will calculate 10 MCSs and we will set the length of the largest MCS to be calculated to 7 (the number of reactions), in order to calculate all the existing MCSs.

```
n_MCS = 10;
max_len_MCS = 7;
```

Now, the different optional arguments of the function must be set, if needed. In this case, we will calculate MCSs involving reactions 1 to 6. The biomass reaction (*rBio*) is omitted since it is the target metabolic task.

```
optional_inputs.rxn_set = {'r1'; 'r2'; 'r3'; 'r4'; 'r5'; 'r6'};
```

The time limit for the calculation of each MCS will be set to 30 seconds.

```
optional_inputs.timelimit = 30;
```

Although not shown here, other optional arguments may be set, as detailed in the documentation of the function.

We will now proceed with the calculation of the MCSs.

```
[MCSs, MCS_time] = calculateMCS(model, n_MCS, max_len_MCS, optional_inputs);
```

```
CPXPARAM_Preprocessing_Fill                      50
CPXPARAM_Preprocessing_Aggregator                50
CPXPARAM_Preprocessing_Dependency                1
CPXPARAM_TimeLimit                               30
CPXPARAM_Preprocessing_Dual                      1
CPXPARAM_Preprocessing_NumPass                   50
CPXPARAM_Output_CloneLog                         -1
CPXPARAM_Preprocessing_CoeffReduce               2
CPXPARAM_Preprocessing_BoundStrength             1
CPXPARAM_MIP_Strategy_HeuristicFreq              1000
```

```
CPXPARAM_Preprocessing_Relax                     1
CPXPARAM_Emphasis_MIP                            4
CPXPARAM_Preprocessing_Symmetry                  1
CPXPARAM_MIP_Strategy_RINSHeur                   50
CPXPARAM_MIP_Pool_RelGap                         0.10000000000000001
CPXPARAM_MIP_Limits_Populate                     40

Populate: phase I
Tried aggregator 2 times.
MIP Presolve eliminated 2 rows and 4 columns.
Aggregator did 3 substitutions.
Reduced MIP has 10 rows, 12 columns, and 26 nonzeros.
Reduced MIP has 6 binaries, 0 generals, 0 SOSs, and 6 indicators.
Presolve time = 0.00 sec. (0.02 ticks)

Root node processing (before b&c):
  Real time             =    0.00 sec. (0.06 ticks)
Parallel b&c, 8 threads:
  Real time             =    0.00 sec. (0.00 ticks)
  Sync time (average)   =    0.00 sec.
  Wait time (average)   =    0.00 sec.
                          ------------
Total (root+branch&cut) =    0.00 sec. (0.06 ticks)
CPXPARAM_Preprocessing_Fill                      50
CPXPARAM_Preprocessing_Aggregator                50
CPXPARAM_Preprocessing_Dependency                1
CPXPARAM_TimeLimit                               30
CPXPARAM_Preprocessing_Dual                      1
CPXPARAM_Preprocessing_NumPass                   50
CPXPARAM_Output_CloneLog                         -1
CPXPARAM_Preprocessing_CoeffReduce               2
CPXPARAM_Preprocessing_BoundStrength             1
CPXPARAM_MIP_Strategy_HeuristicFreq              1000
CPXPARAM_Preprocessing_Relax                     1
CPXPARAM_Emphasis_MIP                            4
CPXPARAM_Preprocessing_Symmetry                  1
CPXPARAM_MIP_Strategy_RINSHeur                   50
CPXPARAM_MIP_Pool_RelGap                         0.10000000000000001
CPXPARAM_MIP_Limits_Populate                     40

Populate: phase I
Tried aggregator 2 times.
MIP Presolve eliminated 2 rows and 4 columns.
Aggregator did 3 substitutions.
Reduced MIP has 10 rows, 12 columns, and 26 nonzeros.
Reduced MIP has 6 binaries, 0 generals, 0 SOSs, and 6 indicators.
Presolve time = 0.00 sec. (0.02 ticks)
Found incumbent of value 2.000000 after 0.00 sec. (0.03 ticks)
Probing time = 0.00 sec. (0.00 ticks)
Tried aggregator 1 time.
Reduced MIP has 10 rows, 12 columns, and 26 nonzeros.
Reduced MIP has 6 binaries, 0 generals, 0 SOSs, and 6 indicators.
Presolve time = 0.00 sec. (0.01 ticks)
Probing time = 0.00 sec. (0.00 ticks)
MIP emphasis: hidden feasible solutions.
MIP search method: dynamic search.
Parallel mode: deterministic, using up to 8 threads.
Root relaxation solution time = 0.00 sec. (0.01 ticks)

        Nodes                                         Cuts/
   Node  Left     Objective  IInf  Best Integer    Best Bound    ItCnt     Gap

*     0+    0                            2.0000        0.0000           100.00%
      0     0        cutoff              2.0000                     0    0.00%
```

```
Root node processing (before b&c):
  Real time             =    0.00 sec. (0.07 ticks)
Parallel b&c, 8 threads:
  Real time             =    0.00 sec. (0.00 ticks)
  Sync time (average)   =    0.00 sec.
  Wait time (average)   =    0.00 sec.
                          ------------
Total (root+branch&cut) =    0.00 sec. (0.07 ticks)

Populate: phase II
MIP emphasis: hidden feasible solutions.
MIP search method: dynamic search.
Parallel mode: deterministic, using up to 8 threads.

Root node processing (before b&c):
  Real time             =    0.00 sec. (0.01 ticks)
Parallel b&c, 8 threads:
  Real time             =    0.00 sec. (0.00 ticks)
  Sync time (average)   =    0.00 sec.
  Wait time (average)   =    0.00 sec.
                          ------------
Total (root+branch&cut) =    0.00 sec. (0.01 ticks)
CPXPARAM_Preprocessing_Fill                      50
CPXPARAM_Preprocessing_Aggregator                50
CPXPARAM_Preprocessing_Dependency                1
CPXPARAM_TimeLimit                               30
CPXPARAM_Preprocessing_Dual                      1
CPXPARAM_Preprocessing_NumPass                   50
CPXPARAM_Output_CloneLog                         -1
CPXPARAM_Preprocessing_CoeffReduce               2
CPXPARAM_Preprocessing_BoundStrength             1
CPXPARAM_MIP_Strategy_HeuristicFreq              1000
CPXPARAM_Preprocessing_Relax                     1
CPXPARAM_Emphasis_MIP                            4
CPXPARAM_Preprocessing_Symmetry                  1
CPXPARAM_MIP_Strategy_RINSHeur                   50
CPXPARAM_MIP_Pool_RelGap                         0.10000000000000001
CPXPARAM_MIP_Limits_Populate                     40

Populate: phase I
Reduced MIP has 17 rows, 20 columns, and 43 nonzeros.
Reduced MIP has 7 binaries, 0 generals, 0 SOSs, and 7 indicators.
Presolve time = 0.00 sec. (0.01 ticks)
Warning:  No solution found from 1 MIP starts.
Retaining values of one MIP start for possible repair.
Tried aggregator 2 times.
MIP Presolve eliminated 2 rows and 5 columns.
Aggregator did 3 substitutions.
Reduced MIP has 11 rows, 12 columns, and 28 nonzeros.
Reduced MIP has 6 binaries, 0 generals, 0 SOSs, and 6 indicators.
Presolve time = 0.00 sec. (0.02 ticks)
Probing time = 0.00 sec. (0.00 ticks)
Tried aggregator 1 time.
Reduced MIP has 11 rows, 12 columns, and 28 nonzeros.
Reduced MIP has 6 binaries, 0 generals, 0 SOSs, and 6 indicators.
Presolve time = 0.00 sec. (0.01 ticks)
Probing time = 0.00 sec. (0.00 ticks)
Clique table members: 1.
MIP emphasis: hidden feasible solutions.
MIP search method: dynamic search.
Parallel mode: deterministic, using up to 8 threads.
Root relaxation solution time = 0.00 sec. (0.01 ticks)
```

```
        Nodes                                         Cuts/
   Node  Left     Objective  IInf  Best Integer    Best Bound    ItCnt     Gap

      0     0        2.0000     1                      2.0000        0
*     0+    0                            2.0000        2.0000             0.00%

Root node processing (before b&c):
  Real time             =    0.02 sec. (0.16 ticks)
Parallel b&c, 8 threads:
  Real time             =    0.00 sec. (0.00 ticks)
  Sync time (average)   =    0.00 sec.
  Wait time (average)   =    0.00 sec.
                          ------------
Total (root+branch&cut) =    0.02 sec. (0.16 ticks)

Populate: phase II
MIP emphasis: hidden feasible solutions.
MIP search method: dynamic search.
Parallel mode: deterministic, using up to 8 threads.

Root node processing (before b&c):
  Real time             =    0.02 sec. (0.09 ticks)
Parallel b&c, 8 threads:
  Real time             =    0.33 sec. (0.11 ticks)
  Sync time (average)   =    0.24 sec.
  Wait time (average)   =    0.00 sec.
                          ------------
Total (root+branch&cut) =    0.34 sec. (0.20 ticks)
CPXPARAM_Preprocessing_Fill                      50
CPXPARAM_Preprocessing_Aggregator                50
CPXPARAM_Preprocessing_Dependency                1
CPXPARAM_TimeLimit                               30
CPXPARAM_Preprocessing_Dual                      1
CPXPARAM_Preprocessing_NumPass                   50
CPXPARAM_Output_CloneLog                         -1
CPXPARAM_Preprocessing_CoeffReduce               2
CPXPARAM_Preprocessing_BoundStrength             1
CPXPARAM_MIP_Strategy_HeuristicFreq              1000
CPXPARAM_Preprocessing_Relax                     1
CPXPARAM_Emphasis_MIP                            4
CPXPARAM_Preprocessing_Symmetry                  1
CPXPARAM_MIP_Strategy_RINSHeur                   50
CPXPARAM_MIP_Pool_RelGap                         0.10000000000000001
CPXPARAM_MIP_Limits_Populate                     40

Populate: phase I
Reduced MIP has 20 rows, 23 columns, and 52 nonzeros.
Reduced MIP has 7 binaries, 0 generals, 0 SOSs, and 7 indicators.
Presolve time = 0.00 sec. (0.01 ticks)
Warning:  No solution found from 3 MIP starts.
Retaining values of one MIP start for possible repair.
Tried aggregator 2 times.
MIP Presolve eliminated 2 rows and 8 columns.
Aggregator did 3 substitutions.
Reduced MIP has 14 rows, 12 columns, and 34 nonzeros.
Reduced MIP has 6 binaries, 0 generals, 0 SOSs, and 6 indicators.
Presolve time = 0.00 sec. (0.03 ticks)

Root node processing (before b&c):
  Real time             =    0.00 sec. (0.11 ticks)
Parallel b&c, 8 threads:
  Real time             =    0.00 sec. (0.00 ticks)
  Sync time (average)   =    0.00 sec.
  Wait time (average)   =    0.00 sec.
```

```
                          ------------
Total (root+branch&cut) =    0.00 sec. (0.11 ticks)
CPXPARAM_Preprocessing_Fill                      50
CPXPARAM_Preprocessing_Aggregator                50
CPXPARAM_Preprocessing_Dependency                1
CPXPARAM_TimeLimit                               30
CPXPARAM_Preprocessing_Dual                      1
CPXPARAM_Preprocessing_NumPass                   50
CPXPARAM_Output_CloneLog                         -1
CPXPARAM_Preprocessing_CoeffReduce               2
CPXPARAM_Preprocessing_BoundStrength             1
CPXPARAM_MIP_Strategy_HeuristicFreq              1000
CPXPARAM_Preprocessing_Relax                     1
CPXPARAM_Emphasis_MIP                            4
CPXPARAM_Preprocessing_Symmetry                  1
CPXPARAM_MIP_Strategy_RINSHeur                   50
CPXPARAM_MIP_Pool_RelGap                         0.10000000000000001
CPXPARAM_MIP_Limits_Populate                     40

Populate: phase I
Reduced MIP has 20 rows, 23 columns, and 52 nonzeros.
Reduced MIP has 7 binaries, 0 generals, 0 SOSs, and 7 indicators.
Presolve time = 0.00 sec. (0.01 ticks)
Warning:  No solution found from 3 MIP starts.
Retaining values of one MIP start for possible repair.
Tried aggregator 2 times.
MIP Presolve eliminated 2 rows and 8 columns.
Aggregator did 3 substitutions.
Reduced MIP has 14 rows, 12 columns, and 34 nonzeros.
Reduced MIP has 6 binaries, 0 generals, 0 SOSs, and 6 indicators.
Presolve time = 0.02 sec. (0.03 ticks)
Probing fixed 4 vars, tightened 3 bounds.
Probing time = 0.00 sec. (0.00 ticks)
Tried aggregator 2 times.
MIP Presolve eliminated 10 rows and 8 columns.
Aggregator did 1 substitutions.
Reduced MIP has 3 rows, 3 columns, and 6 nonzeros.
Reduced MIP has 1 binaries, 0 generals, 0 SOSs, and 2 indicators.
Presolve time = 0.02 sec. (0.01 ticks)
Probing time = 0.00 sec. (0.00 ticks)
Tried aggregator 1 time.
Reduced MIP has 3 rows, 3 columns, and 6 nonzeros.
Reduced MIP has 1 binaries, 0 generals, 0 SOSs, and 2 indicators.
Presolve time = 0.00 sec. (0.00 ticks)
Probing time = 0.00 sec. (0.00 ticks)
Clique table members: 1.
MIP emphasis: hidden feasible solutions.
MIP search method: dynamic search.
Parallel mode: deterministic, using up to 8 threads.
Root relaxation solution time = 0.00 sec. (0.00 ticks)

        Nodes                                         Cuts/
   Node  Left     Objective  IInf  Best Integer    Best Bound    ItCnt     Gap

*     0     0      integral     0        3.0000        3.0000        0    0.00%

Root node processing (before b&c):
  Real time             =    0.05 sec. (2.06 ticks)
Parallel b&c, 8 threads:
  Real time             =    0.00 sec. (0.00 ticks)
  Sync time (average)   =    0.00 sec.
  Wait time (average)   =    0.00 sec.
                          ------------
Total (root+branch&cut) =    0.05 sec. (2.06 ticks)
```

```
Populate: phase II
MIP emphasis: hidden feasible solutions.
MIP search method: dynamic search.
Parallel mode: deterministic, using up to 8 threads.

        Nodes                                         Cuts/
   Node  Left     Objective  IInf  Best Integer    Best Bound    ItCnt     Gap

      0     2        3.0000     1        3.0000        3.0000        0    0.00%
Elapsed time = 0.05 sec. (2.08 ticks, tree = 0.01 MB, solutions = 1)

Root node processing (before b&c):
  Real time             =    0.00 sec. (0.02 ticks)
Parallel b&c, 8 threads:
  Real time             =    0.00 sec. (0.01 ticks)
  Sync time (average)   =    0.00 sec.
  Wait time (average)   =    0.00 sec.
                          ------------
Total (root+branch&cut) =    0.00 sec. (0.03 ticks)
CPXPARAM_Preprocessing_Fill                      50
CPXPARAM_Preprocessing_Aggregator                50
CPXPARAM_Preprocessing_Dependency                1
CPXPARAM_TimeLimit                               30
CPXPARAM_Preprocessing_Dual                      1
CPXPARAM_Preprocessing_NumPass                   50
CPXPARAM_Output_CloneLog                         -1
CPXPARAM_Preprocessing_CoeffReduce               2
CPXPARAM_Preprocessing_BoundStrength             1
CPXPARAM_MIP_Strategy_HeuristicFreq              1000
CPXPARAM_Preprocessing_Relax                     1
CPXPARAM_Emphasis_MIP                            4
CPXPARAM_Preprocessing_Symmetry                  1
CPXPARAM_MIP_Strategy_RINSHeur                   50
CPXPARAM_MIP_Pool_RelGap                         0.10000000000000001
CPXPARAM_MIP_Limits_Populate                     40

Populate: phase I
Reduced MIP has 22 rows, 25 columns, and 60 nonzeros.
Reduced MIP has 7 binaries, 0 generals, 0 SOSs, and 7 indicators.
Presolve time = 0.00 sec. (0.01 ticks)
Warning:  No solution found from 2 MIP starts.
Retaining values of one MIP start for possible repair.
Tried aggregator 2 times.
MIP Presolve eliminated 2 rows and 10 columns.
Aggregator did 3 substitutions.
Reduced MIP has 16 rows, 12 columns, and 40 nonzeros.
Reduced MIP has 6 binaries, 0 generals, 0 SOSs, and 6 indicators.
Presolve time = 0.00 sec. (0.04 ticks)

Root node processing (before b&c):
  Real time             =    0.02 sec. (0.12 ticks)
Parallel b&c, 8 threads:
  Real time             =    0.00 sec. (0.00 ticks)
  Sync time (average)   =    0.00 sec.
  Wait time (average)   =    0.00 sec.
                          ------------
Total (root+branch&cut) =    0.02 sec. (0.12 ticks)
CPXPARAM_Preprocessing_Fill                      50
CPXPARAM_Preprocessing_Aggregator                50
CPXPARAM_Preprocessing_Dependency                1
CPXPARAM_TimeLimit                               30
CPXPARAM_Preprocessing_Dual                      1
CPXPARAM_Preprocessing_NumPass                   50
```

```
CPXPARAM_Output_CloneLog                         -1
CPXPARAM_Preprocessing_CoeffReduce               2
CPXPARAM_Preprocessing_BoundStrength             1
CPXPARAM_MIP_Strategy_HeuristicFreq              1000
CPXPARAM_Preprocessing_Relax                     1
CPXPARAM_Emphasis_MIP                            4
CPXPARAM_Preprocessing_Symmetry                  1
CPXPARAM_MIP_Strategy_RINSHeur                   50
CPXPARAM_MIP_Pool_RelGap                         0.10000000000000001
CPXPARAM_MIP_Limits_Populate                     40

Populate: phase I
Reduced MIP has 22 rows, 25 columns, and 60 nonzeros.
Reduced MIP has 7 binaries, 0 generals, 0 SOSs, and 7 indicators.
Presolve time = 0.00 sec. (0.01 ticks)
Warning:  No solution found from 2 MIP starts.
Retaining values of one MIP start for possible repair.
Row 'c6' infeasible, all entries at implied bounds.
Presolve time = 0.00 sec. (0.02 ticks)

Root node processing (before b&c):
  Real time             =    0.00 sec. (0.04 ticks)
Parallel b&c, 8 threads:
  Real time             =    0.00 sec. (0.00 ticks)
  Sync time (average)   =    0.00 sec.
  Wait time (average)   =    0.00 sec.
                          ------------
Total (root+branch&cut) =    0.00 sec. (0.04 ticks)
CPXPARAM_Preprocessing_Fill                      50
CPXPARAM_Preprocessing_Aggregator                50
CPXPARAM_Preprocessing_Dependency                1
CPXPARAM_TimeLimit                               30
CPXPARAM_Preprocessing_Dual                      1
CPXPARAM_Preprocessing_NumPass                   50
CPXPARAM_Output_CloneLog                         -1
CPXPARAM_Preprocessing_CoeffReduce               2
CPXPARAM_Preprocessing_BoundStrength             1
CPXPARAM_MIP_Strategy_HeuristicFreq              1000
CPXPARAM_Preprocessing_Relax                     1
CPXPARAM_Emphasis_MIP                            4
CPXPARAM_Preprocessing_Symmetry                  1
CPXPARAM_MIP_Strategy_RINSHeur                   50
CPXPARAM_MIP_Pool_RelGap                         0.10000000000000001
CPXPARAM_MIP_Limits_Populate                     40

Populate: phase I
Reduced MIP has 22 rows, 25 columns, and 60 nonzeros.
Reduced MIP has 7 binaries, 0 generals, 0 SOSs, and 7 indicators.
Presolve time = 0.00 sec. (0.01 ticks)
Warning:  No solution found from 2 MIP starts.
Retaining values of one MIP start for possible repair.
Row 'c9' infeasible, all entries at implied bounds.
Presolve time = 0.00 sec. (0.02 ticks)

Root node processing (before b&c):
  Real time             =    0.00 sec. (0.04 ticks)
Parallel b&c, 8 threads:
  Real time             =    0.00 sec. (0.00 ticks)
  Sync time (average)   =    0.00 sec.
  Wait time (average)   =    0.00 sec.
                          ------------
Total (root+branch&cut) =    0.00 sec. (0.04 ticks)
CPXPARAM_Preprocessing_Fill                      50
CPXPARAM_Preprocessing_Aggregator                50
```

```
CPXPARAM_Preprocessing_Dependency                1
CPXPARAM_TimeLimit                               30
CPXPARAM_Preprocessing_Dual                      1
CPXPARAM_Preprocessing_NumPass                   50
CPXPARAM_Output_CloneLog                         -1
CPXPARAM_Preprocessing_CoeffReduce               2
CPXPARAM_Preprocessing_BoundStrength             1
CPXPARAM_MIP_Strategy_HeuristicFreq              1000
CPXPARAM_Preprocessing_Relax                     1
CPXPARAM_Emphasis_MIP                            4
CPXPARAM_Preprocessing_Symmetry                  1
CPXPARAM_MIP_Strategy_RINSHeur                   50
CPXPARAM_MIP_Pool_RelGap                         0.10000000000000001
CPXPARAM_MIP_Limits_Populate                     40

Populate: phase I
Reduced MIP has 22 rows, 25 columns, and 60 nonzeros.
Reduced MIP has 7 binaries, 0 generals, 0 SOSs, and 7 indicators.
Presolve time = 0.00 sec. (0.01 ticks)
Warning:  No solution found from 2 MIP starts.
Retaining values of one MIP start for possible repair.
Presolve time = 0.00 sec. (0.00 ticks)

Root node processing (before b&c):
  Real time             =    0.02 sec. (0.02 ticks)
Parallel b&c, 8 threads:
  Real time             =    0.00 sec. (0.00 ticks)
  Sync time (average)   =    0.00 sec.
  Wait time (average)   =    0.00 sec.
                          ------------
Total (root+branch&cut) =    0.02 sec. (0.02 ticks)
CPXPARAM_Preprocessing_Fill                      50
CPXPARAM_Preprocessing_Aggregator                50
CPXPARAM_Preprocessing_Dependency                1
CPXPARAM_TimeLimit                               30
CPXPARAM_Preprocessing_Dual                      1
CPXPARAM_Preprocessing_NumPass                   50
CPXPARAM_Output_CloneLog                         -1
CPXPARAM_Preprocessing_CoeffReduce               2
CPXPARAM_Preprocessing_BoundStrength             1
CPXPARAM_MIP_Strategy_HeuristicFreq              1000
CPXPARAM_Preprocessing_Relax                     1
CPXPARAM_Emphasis_MIP                            4
CPXPARAM_Preprocessing_Symmetry                  1
CPXPARAM_MIP_Strategy_RINSHeur                   50
CPXPARAM_MIP_Pool_RelGap                         0.10000000000000001
CPXPARAM_MIP_Limits_Populate                     40

Populate: phase I
Reduced MIP has 22 rows, 25 columns, and 60 nonzeros.
Reduced MIP has 7 binaries, 0 generals, 0 SOSs, and 7 indicators.
Presolve time = 0.00 sec. (0.01 ticks)
Warning:  No solution found from 2 MIP starts.
Retaining values of one MIP start for possible repair.
Presolve time = 0.00 sec. (0.00 ticks)

Root node processing (before b&c):
  Real time             =    0.00 sec. (0.02 ticks)
Parallel b&c, 8 threads:
  Real time             =    0.00 sec. (0.00 ticks)
  Sync time (average)   =    0.00 sec.
  Wait time (average)   =    0.00 sec.
                          -----------
Total (root+branch&cut) =    0.00 sec. (0.02 ticks)
```

Despite having tried to calculate 10 MCSs, only 6 exist for this Toy Example. The results are shown in the following piece of code:

```
MCSs{1}
```

```
ans = 2×1 cell array
    'r5'
    'r6'
```

```
MCSs{2}
```

```
ans = 2×1 cell array
    'r1'
    'r6'
```

```
MCSs{3}
```

```
ans = 2×1 cell array
    'r1'
    'r4'
```

```
MCSs{4}
```

```
ans = 2×1 cell array
    'r2'
    'r6'
```

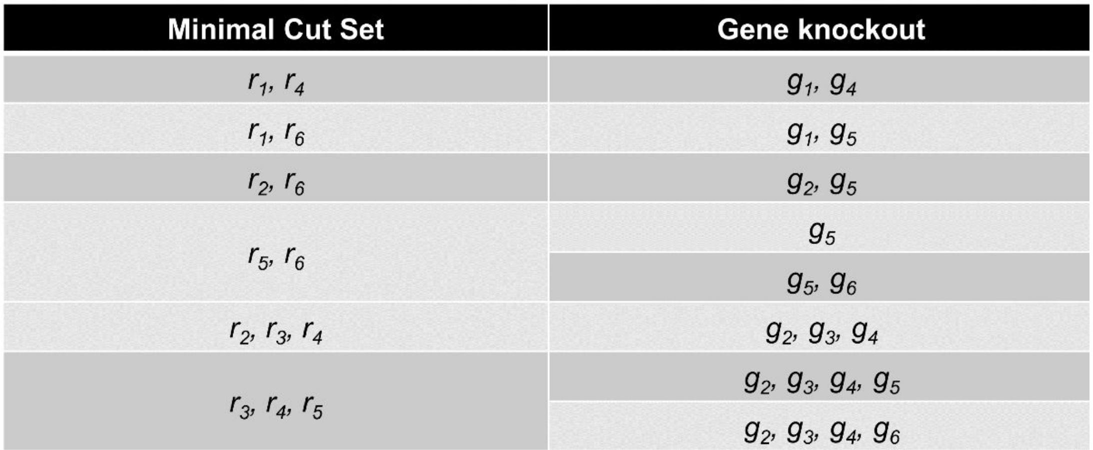
```
MCSs{5}
```

```
ans = 3×1 cell array
    'r2'
    'r3'
    'r4'
```

```
MCSs{6}
```

```
ans = 3×1 cell array
    'r3'
    'r4'
    'r5'
```

We now translate these minimal reaction knockout strategies to the gene level following the *grRules*. The following table shows the genetic interventions which must be fulfiilled to accomplish the respective reaction knockouts:



Minimal Cut Set	Gene knockout
r_1, r_4	g_1, g_4
r_1, r_6	g_1, g_5
r_2, r_6	g_2, g_5
r_5, r_6	g_5
	g_5, g_6
r_2, r_3, r_4	g_2, g_3, g_4
r_3, r_4, r_5	g_2, g_3, g_4, g_5
	g_2, g_3, g_4, g_6

In order to check if the gene knockouts that have to be carried out to perform the inhibition of the found MCSs are minimal, we will calculate gMCSs for the same toy example using the function *calculateGeneMCS()*.

First, we need to define the name of the model to create the gene knockout constraints matrix, *G*. The binary *G* matrix defines for each row the set of blocked reactions arising from the knockout of an irreducible subset of genes. The subset of genes associated with each row in G is interrelated and their simultaneous knockout is required to delete at least one of the reactions in the metabolic network. This matrix may be needed for other calculations. We will name here 'toy_example_gMCS' and again will calculate 10 gMCSs. The function will look for the G matrix in ['G_' model_name '.mat'] (here “G_toy_example_gMCS.mat). If this structure is not available, the function buildGmatrix() is called and the G matrix for the model under consideration generated and saved. In this case, the length of the largest gMCS calculated will be set to 6 (the number of genes).

```
model_name = 'toy_example_gMCS';
n_gMCS = 10;
max_len_gMCS = 6;
```

Next, we set the optional inputs. The maximum time for the calculation of each gMCS will be 30 seconds.

```
optional_inputs.timelimit = 30;
```

Note that, again, some more optional inputs may be set, as detailed in the documentation of the function. However, they are not needed for the calculation of the problem presented in this tutorial.

We will now proceed with the calculation of the gMCSs.

```
[gMCSs, gMCS_time] = calculateGeneMCS(model_name, model, n_gMCS, max_len_gMCS, optional
```

```
G MATRIX - Summary
    'n_rxns_0_genes'     [1]
    'n_rxns_1_gene'      [4]
    'n_rxns_only_or'     [1]
    'n_rxns_only_and'    [1]
    'n_rxns_or_and'      [0]
    'n_rxns_total'       [7]
G MATRIX - STEP 1
G MATRIX - STEP 2
G MATRIX - STEP 3
G MATRIX - STEP 4

G MATRIX - Delete Repeats
```

```
G MATRIX - Check Relations
The G Matrix has been successfully calculated
CPXPARAM_Preprocessing_Fill                      50
CPXPARAM_Preprocessing_Aggregator                50
CPXPARAM_Preprocessing_Dependency                1
CPXPARAM_TimeLimit                               30
CPXPARAM_Preprocessing_Dual                      1
CPXPARAM_Preprocessing_NumPass                   50
CPXPARAM_Output_CloneLog                         -1
CPXPARAM_Preprocessing_CoeffReduce               2
CPXPARAM_Preprocessing_BoundStrength             1
CPXPARAM_MIP_Strategy_HeuristicFreq              1000
CPXPARAM_Preprocessing_Relax                     1
CPXPARAM_Emphasis_MIP                            4
CPXPARAM_Preprocessing_Symmetry                  1
CPXPARAM_MIP_Strategy_RINSHeur                   50
CPXPARAM_MIP_Pool_RelGap                         0.10000000000000001
CPXPARAM_MIP_Limits_Populate                     40

Populate: phase I
Tried aggregator 2 times.
MIP Presolve eliminated 4 rows and 6 columns.
MIP Presolve modified 2 coefficients.
Aggregator did 3 substitutions.
Reduced MIP has 8 rows, 10 columns, and 19 nonzeros.
Reduced MIP has 5 binaries, 0 generals, 0 SOSs, and 5 indicators.
Presolve time = 0.00 sec. (0.03 ticks)
Probing fixed 5 vars, tightened 5 bounds.
Probing time = 0.00 sec. (0.00 ticks)
Tried aggregator 1 time.
MIP Presolve eliminated 8 rows and 10 columns.
All rows and columns eliminated.
Presolve time = 0.00 sec. (0.01 ticks)

Root node processing (before b&c):
  Real time             =    0.02 sec. (0.06 ticks)
Parallel b&c, 8 threads:
  Real time             =    0.00 sec. (0.00 ticks)
  Sync time (average)   =    0.00 sec.
  Wait time (average)   =    0.00 sec.
                          ------------
Total (root+branch&cut) =    0.02 sec. (0.06 ticks)

Populate: phase II
Tried aggregator 2 times.
MIP Presolve eliminated 4 rows and 6 columns.
MIP Presolve modified 2 coefficients.
Aggregator did 3 substitutions.
Reduced MIP has 8 rows, 10 columns, and 19 nonzeros.
Reduced MIP has 5 binaries, 0 generals, 0 SOSs, and 5 indicators.
Presolve time = 0.00 sec. (0.03 ticks)
Probing fixed 5 vars, tightened 5 bounds.
Probing time = 0.00 sec. (0.00 ticks)
Clique table members: 1.
MIP emphasis: hidden feasible solutions.
MIP search method: dynamic search.
Parallel mode: deterministic, using up to 8 threads.
Root relaxation solution time = 0.00 sec. (0.00 ticks)

        Nodes                                         Cuts/
   Node  Left     Objective  IInf  Best Integer    Best Bound    ItCnt     Gap

*     0+    0                            1.0000        0.0000           100.00%
      0     0        1.0000     0        1.0000        1.0000        0    0.00%
      0     0        1.0000     0        1.0000        1.0000        0    0.00%
Elapsed time = 0.02 sec. (0.15 ticks, tree = 0.01 MB, solutions = 1)

Root node processing (before b&c):
  Real time             =    0.00 sec. (0.08 ticks)
Parallel b&c, 8 threads:
  Real time             =    0.00 sec. (0.01 ticks)
  Sync time (average)   =    0.00 sec.
  Wait time (average)   =    0.00 sec.
                          ------------
Total (root+branch&cut) =    0.00 sec. (0.09 ticks)
CPXPARAM_Preprocessing_Fill                      50
CPXPARAM_Preprocessing_Aggregator                50
CPXPARAM_Preprocessing_Dependency                1
CPXPARAM_TimeLimit                               30
CPXPARAM_Preprocessing_Dual                      1
CPXPARAM_Preprocessing_NumPass                   50
CPXPARAM_Output_CloneLog                         -1
CPXPARAM_Preprocessing_CoeffReduce               2
CPXPARAM_Preprocessing_BoundStrength             1
CPXPARAM_MIP_Strategy_HeuristicFreq              1000
CPXPARAM_Preprocessing_Relax                     1
CPXPARAM_Emphasis_MIP                            4
CPXPARAM_Preprocessing_Symmetry                  1
CPXPARAM_MIP_Strategy_RINSHeur                   50
CPXPARAM_MIP_Pool_RelGap                         0.10000000000000001
CPXPARAM_MIP_Limits_Populate                     40

Populate: phase I
Reduced MIP has 18 rows, 19 columns, and 45 nonzeros.
Reduced MIP has 7 binaries, 0 generals, 0 SOSs, and 7 indicators.
Presolve time = 0.00 sec. (0.01 ticks)
Warning:  No solution found from 1 MIP starts.
Retaining values of one MIP start for possible repair.
Row 'c3' infeasible, all entries at implied bounds.
Presolve time = 0.00 sec. (0.01 ticks)

Root node processing (before b&c):
  Real time             =    0.00 sec. (0.03 ticks)
Parallel b&c, 8 threads:
  Real time             =    0.00 sec. (0.00 ticks)
  Sync time (average)   =    0.00 sec.
  Wait time (average)   =    0.00 sec.
                          ------------
Total (root+branch&cut) =    0.00 sec. (0.03 ticks)
CPXPARAM_Preprocessing_Fill                      50
CPXPARAM_Preprocessing_Aggregator                50
CPXPARAM_Preprocessing_Dependency                1
CPXPARAM_TimeLimit                               30
CPXPARAM_Preprocessing_Dual                      1
CPXPARAM_Preprocessing_NumPass                   50
CPXPARAM_Output_CloneLog                         -1
CPXPARAM_Preprocessing_CoeffReduce               2
CPXPARAM_Preprocessing_BoundStrength             1
CPXPARAM_MIP_Strategy_HeuristicFreq              1000
CPXPARAM_Preprocessing_Relax                     1
CPXPARAM_Emphasis_MIP                            4
CPXPARAM_Preprocessing_Symmetry                  1
CPXPARAM_MIP_Strategy_RINSHeur                   50
CPXPARAM_MIP_Pool_RelGap                         0.10000000000000001
CPXPARAM_MIP_Limits_Populate                     40

Populate: phase I
Reduced MIP has 18 rows, 19 columns, and 45 nonzeros.
Reduced MIP has 7 binaries, 0 generals, 0 SOSs, and 7 indicators.
Presolve time = 0.00 sec. (0.01 ticks)
Warning:  No solution found from 1 MIP starts.
Retaining values of one MIP start for possible repair.
Tried aggregator 2 times.
MIP Presolve eliminated 7 rows and 10 columns.
MIP Presolve modified 2 coefficients.
Aggregator did 3 substitutions.
Reduced MIP has 5 rows, 6 columns, and 10 nonzeros.
Reduced MIP has 3 binaries, 0 generals, 0 SOSs, and 3 indicators.
Presolve time = 0.00 sec. (0.03 ticks)
Probing fixed 3 vars, tightened 2 bounds.
Probing time = 0.00 sec. (0.00 ticks)
Tried aggregator 1 time.
MIP Presolve eliminated 5 rows and 6 columns.
All rows and columns eliminated.
Presolve time = 0.00 sec. (0.00 ticks)

Root node processing (before b&c):
  Real time             =    0.00 sec. (0.05 ticks)
Parallel b&c, 8 threads:
  Real time             =    0.00 sec. (0.00 ticks)
  Sync time (average)   =    0.00 sec.
  Wait time (average)   =    0.00 sec.
                          ------------
Total (root+branch&cut) =    0.00 sec. (0.05 ticks)

Populate: phase II
Tried aggregator 2 times.
MIP Presolve eliminated 7 rows and 10 columns.
MIP Presolve modified 2 coefficients.
Aggregator did 3 substitutions.
Reduced MIP has 5 rows, 6 columns, and 10 nonzeros.
Reduced MIP has 3 binaries, 0 generals, 0 SOSs, and 3 indicators.
Presolve time = 0.00 sec. (0.03 ticks)
Probing fixed 3 vars, tightened 2 bounds.
Probing time = 0.00 sec. (0.00 ticks)
Clique table members: 1.
MIP emphasis: hidden feasible solutions.
MIP search method: dynamic search.
Parallel mode: deterministic, using up to 8 threads.
Root relaxation solution time = 0.00 sec. (0.00 ticks)

        Nodes                                         Cuts/
   Node  Left     Objective  IInf  Best Integer    Best Bound    ItCnt     Gap

*     0+    0                            2.0000        1.0000            50.00%
      0     0        2.0000     0        2.0000        2.0000        0    0.00%
      0     0        2.0000     0        2.0000        2.0000        0    0.00%
Elapsed time = 0.02 sec. (0.12 ticks, tree = 0.01 MB, solutions = 1)

Root node processing (before b&c):
  Real time             =    0.00 sec. (0.06 ticks)
Parallel b&c, 8 threads:
  Real time             =    0.27 sec. (0.01 ticks)
  Sync time (average)   =    0.21 sec.
  Wait time (average)   =    0.00 sec.
                          ------------
Total (root+branch&cut) =    0.27 sec. (0.07 ticks)
CPXPARAM_Preprocessing_Fill                      50
CPXPARAM_Preprocessing_Aggregator                50
CPXPARAM_Preprocessing_Dependency                1
CPXPARAM_TimeLimit                               30
CPXPARAM_Preprocessing_Dual                      1
```

```
CPXPARAM_Preprocessing_NumPass                   50
CPXPARAM_Output_CloneLog                         -1
CPXPARAM_Preprocessing_CoeffReduce               2
CPXPARAM_Preprocessing_BoundStrength             1
CPXPARAM_MIP_Strategy_HeuristicFreq              1000
CPXPARAM_Preprocessing_Relax                     1
CPXPARAM_Emphasis_MIP                            4
CPXPARAM_Preprocessing_Symmetry                  1
CPXPARAM_MIP_Strategy_RINSHeur                   50
CPXPARAM_MIP_Pool_RelGap                         0.10000000000000001
CPXPARAM_MIP_Limits_Populate                     40

Populate: phase I
Reduced MIP has 19 rows, 20 columns, and 48 nonzeros.
Reduced MIP has 7 binaries, 0 generals, 0 SOSs, and 7 indicators.
Presolve time = 0.00 sec. (0.01 ticks)
Warning:  No solution found from 1 MIP starts.
Retaining values of one MIP start for possible repair.
Row 'c3' infeasible, all entries at implied bounds.
Presolve time = 0.02 sec. (0.02 ticks)

Root node processing (before b&c):
  Real time             =    0.02 sec. (0.04 ticks)
Parallel b&c, 8 threads:
  Real time             =    0.00 sec. (0.00 ticks)
  Sync time (average)   =    0.00 sec.
  Wait time (average)   =    0.00 sec.
                          ------------
Total (root+branch&cut) =    0.02 sec. (0.04 ticks)
CPXPARAM_Preprocessing_Fill                      50
CPXPARAM_Preprocessing_Aggregator                50
CPXPARAM_Preprocessing_Dependency                1
CPXPARAM_TimeLimit                               30
CPXPARAM_Preprocessing_Dual                      1
CPXPARAM_Preprocessing_NumPass                   50
CPXPARAM_Output_CloneLog                         -1
CPXPARAM_Preprocessing_CoeffReduce               2
CPXPARAM_Preprocessing_BoundStrength             1
CPXPARAM_MIP_Strategy_HeuristicFreq              1000
CPXPARAM_Preprocessing_Relax                     1
CPXPARAM_Emphasis_MIP                            4
CPXPARAM_Preprocessing_Symmetry                  1
CPXPARAM_MIP_Strategy_RINSHeur                   50
CPXPARAM_MIP_Pool_RelGap                         0.10000000000000001
CPXPARAM_MIP_Limits_Populate                     40

Populate: phase I
Reduced MIP has 19 rows, 20 columns, and 48 nonzeros.
Reduced MIP has 7 binaries, 0 generals, 0 SOSs, and 7 indicators.
Presolve time = 0.00 sec. (0.01 ticks)
Warning:  No solution found from 1 MIP starts.
Retaining values of one MIP start for possible repair.
Tried aggregator 1 time.
MIP Presolve eliminated 13 rows and 20 columns.
All rows and columns eliminated.
Presolve time = 0.00 sec. (0.03 ticks)

Root node processing (before b&c):
  Real time             =    0.00 sec. (0.04 ticks)
Parallel b&c, 8 threads:
  Real time             =    0.00 sec. (0.00 ticks)
  Sync time (average)   =    0.00 sec.
  Wait time (average)   =    0.00 sec.
                          ------------
```

```
Total (root+branch&cut) =    0.00 sec. (0.04 ticks)

Populate: phase II
Tried aggregator 1 time.
MIP Presolve eliminated 13 rows and 20 columns.
All rows and columns eliminated.
Presolve time = 0.00 sec. (0.03 ticks)

Root node processing (before b&c):
  Real time             =    0.00 sec. (0.03 ticks)
Parallel b&c, 8 threads:
  Real time             =    0.00 sec. (0.00 ticks)
  Sync time (average)   =    0.00 sec.
  Wait time (average)   =    0.00 sec.
                          ------------
Total (root+branch&cut) =    0.00 sec. (0.03 ticks)
CPXPARAM_Preprocessing_Fill                      50
CPXPARAM_Preprocessing_Aggregator                50
CPXPARAM_Preprocessing_Dependency                1
CPXPARAM_TimeLimit                               30
CPXPARAM_Preprocessing_Dual                      1
CPXPARAM_Preprocessing_NumPass                   50
CPXPARAM_Output_CloneLog                         -1
CPXPARAM_Preprocessing_CoeffReduce               2
CPXPARAM_Preprocessing_BoundStrength             1
CPXPARAM_MIP_Strategy_HeuristicFreq              1000
CPXPARAM_Preprocessing_Relax                     1
CPXPARAM_Emphasis_MIP                            4
CPXPARAM_Preprocessing_Symmetry                  1
CPXPARAM_MIP_Strategy_RINSHeur                   50
CPXPARAM_MIP_Pool_RelGap                         0.10000000000000001
CPXPARAM_MIP_Limits_Populate                     40

Populate: phase I
Reduced MIP has 20 rows, 21 columns, and 52 nonzeros.
Reduced MIP has 7 binaries, 0 generals, 0 SOSs, and 7 indicators.
Presolve time = 0.00 sec. (0.01 ticks)
Warning:  No solution found from 1 MIP starts.
Retaining values of one MIP start for possible repair.
Row 'c3' infeasible, all entries at implied bounds.
Presolve time = 0.00 sec. (0.02 ticks)

Root node processing (before b&c):
  Real time             =    0.00 sec. (0.04 ticks)
Parallel b&c, 8 threads:
  Real time             =    0.00 sec. (0.00 ticks)
  Sync time (average)   =    0.00 sec.
  Wait time (average)   =    0.00 sec.
                          ------------
Total (root+branch&cut) =    0.00 sec. (0.04 ticks)
CPXPARAM_Preprocessing_Fill                      50
CPXPARAM_Preprocessing_Aggregator                50
CPXPARAM_Preprocessing_Dependency                1
CPXPARAM_TimeLimit                               30
CPXPARAM_Preprocessing_Dual                      1
CPXPARAM_Preprocessing_NumPass                   50
CPXPARAM_Output_CloneLog                         -1
CPXPARAM_Preprocessing_CoeffReduce               2
CPXPARAM_Preprocessing_BoundStrength             1
CPXPARAM_MIP_Strategy_HeuristicFreq              1000
CPXPARAM_Preprocessing_Relax                     1
CPXPARAM_Emphasis_MIP                            4
CPXPARAM_Preprocessing_Symmetry                  1
CPXPARAM_MIP_Strategy_RINSHeur                   50
```

```
CPXPARAM_MIP_Pool_RelGap                         0.10000000000000001
CPXPARAM_MIP_Limits_Populate                     40

Populate: phase I
Reduced MIP has 20 rows, 21 columns, and 52 nonzeros.
Reduced MIP has 7 binaries, 0 generals, 0 SOSs, and 7 indicators.
Presolve time = 0.00 sec. (0.01 ticks)
Warning:  No solution found from 1 MIP starts.
Retaining values of one MIP start for possible repair.
Row 'c10' infeasible, all entries at implied bounds.
Presolve time = 0.00 sec. (0.02 ticks)

Root node processing (before b&c):
  Real time             =    0.00 sec. (0.03 ticks)
Parallel b&c, 8 threads:
  Real time             =    0.00 sec. (0.00 ticks)
  Sync time (average)   =    0.00 sec.
  Wait time (average)   =    0.00 sec.
                          ------------
Total (root+branch&cut) =    0.00 sec. (0.03 ticks)
CPXPARAM_Preprocessing_Fill                      50
CPXPARAM_Preprocessing_Aggregator                50
CPXPARAM_Preprocessing_Dependency                1
CPXPARAM_TimeLimit                               30
CPXPARAM_Preprocessing_Dual                      1
CPXPARAM_Preprocessing_NumPass                   50
CPXPARAM_Output_CloneLog                         -1
CPXPARAM_Preprocessing_CoeffReduce               2
CPXPARAM_Preprocessing_BoundStrength             1
CPXPARAM_MIP_Strategy_HeuristicFreq              1000
CPXPARAM_Preprocessing_Relax                     1
CPXPARAM_Emphasis_MIP                            4
CPXPARAM_Preprocessing_Symmetry                  1
CPXPARAM_MIP_Strategy_RINSHeur                   50
CPXPARAM_MIP_Pool_RelGap                         0.10000000000000001
CPXPARAM_MIP_Limits_Populate                     40

Populate: phase I
Reduced MIP has 20 rows, 21 columns, and 52 nonzeros.
Reduced MIP has 7 binaries, 0 generals, 0 SOSs, and 7 indicators.
Presolve time = 0.00 sec. (0.01 ticks)
Warning:  No solution found from 1 MIP starts.
Retaining values of one MIP start for possible repair.
Presolve time = 0.00 sec. (0.00 ticks)

Root node processing (before b&c):
  Real time             =    0.02 sec. (0.02 ticks)
Parallel b&c, 8 threads:
  Real time             =    0.00 sec. (0.00 ticks)
  Sync time (average)   =    0.00 sec.
  Wait time (average)   =    0.00 sec.
                          ------------
Total (root+branch&cut) =    0.02 sec. (0.02 ticks)
CPXPARAM_Preprocessing_Fill                      50
CPXPARAM_Preprocessing_Aggregator                50
CPXPARAM_Preprocessing_Dependency                1
CPXPARAM_TimeLimit                               30
CPXPARAM_Preprocessing_Dual                      1
CPXPARAM_Preprocessing_NumPass                   50
CPXPARAM_Output_CloneLog                         -1
CPXPARAM_Preprocessing_CoeffReduce               2
CPXPARAM_Preprocessing_BoundStrength             1
CPXPARAM_MIP_Strategy_HeuristicFreq              1000
CPXPARAM_Preprocessing_Relax                     1
```

```
CPXPARAM_Emphasis_MIP                              4
CPXPARAM_Preprocessing_Symmetry                    1
CPXPARAM_MIP_Strategy_RINSHeur                     50
CPXPARAM_MIP_Pool_RelGap                           0.10000000000000001
CPXPARAM_MIP_Limits_Populate                       40

Populate: phase I
Reduced MIP has 20 rows, 21 columns, and 52 nonzeros.
Reduced MIP has 7 binaries, 0 generals, 0 SOSs, and 7 indicators.
Presolve time = 0.00 sec. (0.01 ticks)
Warning:  No solution found from 1 MIP starts.
Retaining values of one MIP start for possible repair.
Presolve time = 0.00 sec. (0.00 ticks)

Root node processing (before b&c):
  Real time             =    0.00 sec. (0.02 ticks)
Parallel b&c, 8 threads:
  Real time             =    0.00 sec. (0.00 ticks)
  Sync time (average)   =    0.00 sec.
  Wait time (average)   =    0.00 sec.
                          ------------
Total (root+branch&cut) =    0.00 sec. (0.02 ticks)
```

In the same way as with MCSs, 3 gMCSs have been calculated in total.

```
gMCSs{1}
```

```
ans = cell

    'g5'
```

```
gMCSs{2}
```

```
ans = 2×1 cell array
    'g1'
    'g4'
```

```
gMCSs{3}
```

```
ans = 3×1 cell array
    'g2'
    'g3'
    'g4'
```

As shown in the previous table, calculating MCSs would result in 8 different genetic intervention strategies even when, as we have just demonstrated, only 3 minimal genetic interventions exist. Moving on to gMCSs seems, therefore, a more efficient strategy to calculate minimal gene knockout strategies.

2. gMCSs in large metabolic networks (20 min ~ 1 hour)

The algorithm presented in this tutorial is able to calculate intervention strategies in large metabolic networks. In addition, it is sufficiently flexible to calculate gMCSs involving a particular gene knockout. Interesting results regarding this issue can be found in Apaolaza et al., 2017(a)[2].

We are now going to calculate 6 gMCSs for the human metabolic model Recon2.v04[4] involving gene RRM1 (Entrez ID: 6240). To do so, the folllowing piece of code has to be executed.

First, we are going to load the metabolic model.

```
global CBTDIR
load([CBTDIR filesep 'test' filesep 'models' filesep 'mat' filesep 'Recon2.v04.mat']);
```

Next, in the same way as in the toy example, we will proceed to set the name of the model ('Recon2'), the number of gMCSs to be calculated (6), the length of the largest gMCS we want to calculate (10) and the optional input variables. Regarding the latter, the maximum time for the calculation of gMCSs will be set to 2 minutes (120 seconds), the KO will be '6240', namely the Entrez ID of RRM1. In this case, we will also have to set the "separate_transcript" field to '.', because, if we do not, the calculation of gMCSs will be done at the transcript level as a consequence of the usage of the aforementioned character in Recon2.v04 to differentiate the transcripts of the same gene.

```
model_name = 'Recon2';
n_gMCS = 6;
max_len_gMCS = 10;
optional_inputs.KO = '6240';
optional_inputs.separate_transcript = '.';
optional_inputs.timelimit = 2*60;
```

Finally, we will calculate the gMCSs. Turning the parallel pool on is recommended if the G matrix has not been calculated yet.

```
% parpool;
[gMCSs, gMCS_time] = calculateGeneMCS(model_name, modelR204, n_gMCS, max_len_gMCS, opti
```

```
G MATRIX - Summary
    'n_rxns_0_genes'     [2619]
    'n_rxns_1_gene'      [2917]
    'n_rxns_only_or'     [1426]
    'n_rxns_only_and'    [ 231]
    'n_rxns_or_and'      [ 247]
    'n_rxns_total'       [7440]
G MATRIX - STEP 1
G MATRIX - STEP 2
G MATRIX - STEP 3
Calculating Networks for GPR rules...
156 of 247 rxns
155 of 247 rxns
84 of 247 rxns
83 of 247 rxns
125 of 247 rxns
124 of 247 rxns
42 of 247 rxns
41 of 247 rxns
154 of 247 rxns
153 of 247 rxns
152 of 247 rxns
151 of 247 rxns
150 of 247 rxns
149 of 247 rxns
148 of 247 rxns
147 of 247 rxns
146 of 247 rxns
145 of 247 rxns
144 of 247 rxns
143 of 247 rxns
142 of 247 rxns
141 of 247 rxns
```

```
140 of 247 rxns
139 of 247 rxns
138 of 247 rxns
137 of 247 rxns
136 of 247 rxns
135 of 247 rxns
134 of 247 rxns
133 of 247 rxns
132 of 247 rxns
131 of 247 rxns
130 of 247 rxns
129 of 247 rxns
128 of 247 rxns
127 of 247 rxns
126 of 247 rxns
82 of 247 rxns
81 of 247 rxns
80 of 247 rxns
79 of 247 rxns
78 of 247 rxns
77 of 247 rxns
76 of 247 rxns
75 of 247 rxns
74 of 247 rxns
73 of 247 rxns
72 of 247 rxns
71 of 247 rxns
70 of 247 rxns
69 of 247 rxns
68 of 247 rxns
67 of 247 rxns
66 of 247 rxns
65 of 247 rxns
64 of 247 rxns
63 of 247 rxns
123 of 247 rxns
122 of 247 rxns
121 of 247 rxns
120 of 247 rxns
119 of 247 rxns
118 of 247 rxns
117 of 247 rxns
116 of 247 rxns
115 of 247 rxns
114 of 247 rxns
113 of 247 rxns
112 of 247 rxns
111 of 247 rxns
110 of 247 rxns
40 of 247 rxns
39 of 247 rxns
38 of 247 rxns
37 of 247 rxns
36 of 247 rxns
35 of 247 rxns
34 of 247 rxns
33 of 247 rxns
32 of 247 rxns
31 of 247 rxns
30 of 247 rxns
29 of 247 rxns
28 of 247 rxns
27 of 247 rxns
26 of 247 rxns
```

```
25 of 247 rxns
24 of 247 rxns
62 of 247 rxns
61 of 247 rxns
60 of 247 rxns
59 of 247 rxns
58 of 247 rxns
57 of 247 rxns
56 of 247 rxns
55 of 247 rxns
54 of 247 rxns
53 of 247 rxns
52 of 247 rxns
51 of 247 rxns
50 of 247 rxns
49 of 247 rxns
48 of 247 rxns
47 of 247 rxns
46 of 247 rxns
45 of 247 rxns
44 of 247 rxns
43 of 247 rxns
109 of 247 rxns
108 of 247 rxns
107 of 247 rxns
106 of 247 rxns
105 of 247 rxns
104 of 247 rxns
103 of 247 rxns
102 of 247 rxns
101 of 247 rxns
100 of 247 rxns
99 of 247 rxns
98 of 247 rxns
97 of 247 rxns
96 of 247 rxns
95 of 247 rxns
94 of 247 rxns
93 of 247 rxns
92 of 247 rxns
91 of 247 rxns
90 of 247 rxns
89 of 247 rxns
88 of 247 rxns
87 of 247 rxns
86 of 247 rxns
85 of 247 rxns
179 of 247 rxns
178 of 247 rxns
177 of 247 rxns
176 of 247 rxns
175 of 247 rxns
174 of 247 rxns
173 of 247 rxns
172 of 247 rxns
171 of 247 rxns
170 of 247 rxns
169 of 247 rxns
168 of 247 rxns
167 of 247 rxns
166 of 247 rxns
165 of 247 rxns
164 of 247 rxns
163 of 247 rxns
```

```
162 of 247 rxns
161 of 247 rxns
160 of 247 rxns
159 of 247 rxns
158 of 247 rxns
157 of 247 rxns
196 of 247 rxns
195 of 247 rxns
194 of 247 rxns
193 of 247 rxns
192 of 247 rxns
191 of 247 rxns
190 of 247 rxns
189 of 247 rxns
188 of 247 rxns
187 of 247 rxns
186 of 247 rxns
185 of 247 rxns
184 of 247 rxns
183 of 247 rxns
182 of 247 rxns
181 of 247 rxns
180 of 247 rxns
219 of 247 rxns
218 of 247 rxns
217 of 247 rxns
216 of 247 rxns
215 of 247 rxns
214 of 247 rxns
213 of 247 rxns
212 of 247 rxns
211 of 247 rxns
210 of 247 rxns
226 of 247 rxns
225 of 247 rxns
224 of 247 rxns
223 of 247 rxns
222 of 247 rxns
221 of 247 rxns
220 of 247 rxns
233 of 247 rxns
232 of 247 rxns
231 of 247 rxns
230 of 247 rxns
229 of 247 rxns
228 of 247 rxns
227 of 247 rxns
240 of 247 rxns
239 of 247 rxns
238 of 247 rxns
237 of 247 rxns
236 of 247 rxns
235 of 247 rxns
234 of 247 rxns
247 of 247 rxns
246 of 247 rxns
245 of 247 rxns
244 of 247 rxns
243 of 247 rxns
242 of 247 rxns
241 of 247 rxns
209 of 247 rxns
208 of 247 rxns
207 of 247 rxns
```

```
206 of 247 rxns
205 of 247 rxns
204 of 247 rxns
203 of 247 rxns
202 of 247 rxns
201 of 247 rxns
200 of 247 rxns
199 of 247 rxns
198 of 247 rxns
197 of 247 rxns
23 of 247 rxns
22 of 247 rxns
21 of 247 rxns
20 of 247 rxns
19 of 247 rxns
18 of 247 rxns
17 of 247 rxns
16 of 247 rxns
15 of 247 rxns
14 of 247 rxns
13 of 247 rxns
12 of 247 rxns
11 of 247 rxns
10 of 247 rxns
9 of 247 rxns
8 of 247 rxns
7 of 247 rxns
6 of 247 rxns
5 of 247 rxns
4 of 247 rxns
3 of 247 rxns
2 of 247 rxns
1 of 247 rxns
G MATRIX - STEP 4

0%      [                                        ]0%      [                                        ]1%
G MATRIX - Delete Repeats
G MATRIX - Check Relations
The G Matrix has been successfully calculated
CPXPARAM_Preprocessing_Fill                      50
CPXPARAM_Preprocessing_Aggregator                50
CPXPARAM_Preprocessing_Dependency                1
CPXPARAM_TimeLimit                               120
CPXPARAM_Preprocessing_Dual                      1
CPXPARAM_Preprocessing_NumPass                   50
CPXPARAM_Output_CloneLog                         -1
CPXPARAM_Preprocessing_CoeffReduce               2
CPXPARAM_Preprocessing_BoundStrength             1
CPXPARAM_MIP_Strategy_HeuristicFreq              1000
CPXPARAM_Preprocessing_Relax                     1
CPXPARAM_Emphasis_MIP                            4
CPXPARAM_Preprocessing_Symmetry                  1
CPXPARAM_MIP_Strategy_RINSHeur                   50
CPXPARAM_MIP_Pool_RelGap                         0.10000000000000001
CPXPARAM_MIP_Limits_Populate                     40

Populate: phase I
Tried aggregator 12 times.
MIP Presolve eliminated 9344 rows and 10469 columns.
MIP Presolve modified 9523 coefficients.
Aggregator did 3520 substitutions.
Reduced MIP has 7520 rows, 7817 columns, and 46898 nonzeros.
```

The results obtained are the following:

```
gMCSs{1}
```

```
ans = 2×1 cell array
    '2987'
    '6240'
```

```
gMCSs{2}
```

```
ans = 3×1 cell array
    '1716'
    '6240'
    '7296'
```

```
gMCSs{3}
```

```
ans = 3×1 cell array
    '1716'
    '50484'
    '6240'
```

```
gMCSs{4}
```

```
ans = 4×1 cell array
    '1633'
    '1635'
    '6240'
    '7296'
```

```
gMCSs{5}
```

```
ans = 4×1 cell array
    '26289'
    '51727'
    '6240'
    '7296'
```

```
gMCSs{6}
```

```
ans = 5×1 cell array
    '2766'
    '3251'
    '51292'
    '6240'
    '8833'
```

6 gMCSs have been calculated involving 2, 3, 3, 4, 4 and 5 genes, respectively. It is important to note that no more gMCSs have been found as a consequence of setting the number of gMCSs to be calculated to 6. If we increase the aforementiones input, more gMCSs involving RRM1 will be calculated. It is important to note that the calculation of a minimal intervention strategy involving 5 genes with an exhaustive combinatorial strategy is actually inviable.

TIMING

1. Equipment Setup: ~25 sec.

2. Toy Example: ~5 sec.
3. gMCSs in large metabolic networks: ~5 min.

Note that both *calculateMCS()* and *calculateGeneMCS()* functions return MCS_time and gMCS_time, respectively, which contain the timing for all the solving steps.

REFERENCES

1. von Kamp, A. & Klamt, S. Enumeration of smallest intervention strategies in genome-scale metabolic networks. *PLoS computational biology*, *10*(1), e1003378 (2014).

2. Apaolaza, I., *et al*. An in-silico approach to predict and exploit synthetic lethality in cancer metabolism. *Nature Communications*, *8*(1), 459 (2017 (a)).

3. Apaolaza, I., José-Eneriz, E. S., Agirre, X., Prósper, F. & Planes, F. J. COBRA methods and metabolic drug targets in cancer. *Molecular & Cellular Oncology*, (just-accepted), 00-00 (2017 (b)).

4. Thiele, I., *et al*. A community-driven global reconstruction of human metabolism. *Nature biotechnology*, *31*(5), 419-425 (2013).