Contextualizing large scale signalling networks from expression footprints with CARNIVAL

Enio Gjerga *[1,2] and Panuwat Trairatphisan[2]

[1]RWTH Aachen University, Faculty of Medicine, Joint Research Centre for Computational Biomedicine (JRC-COMBINE), 52074, Aachen, Germany
[2]Heidelberg University, Faculty of Medicine, and Heidelberg University Hospital, Institute of Computational Biomedicine, Bioquant, 69120, Heidelberg, Germany

January 18, 2020

Contents

1 Introduction **1**
1.1 CARNIVAL pipeline . . . 2
1.2 ILP solvers . . . 2
1.3 Citation . . . 3

2 Getting Started **4**
2.1 Prerequisites . . . 4
2.2 Installation . . . 4
2.3 Documentation . . . 4
2.4 CARNIVAL settings . . . 4

3 Running CARNIVAL **6**
3.1 DEG Pre-processing . . . 6
3.2 CARNIVAL pipeline - Example 1 . . . 6
3.3 CARNIVAL pipeline - Example 2 . . . 8
3.4 Toy example with multiple conditions . . . 13

1 Introduction

While gene expression profiling is commonly used to gain an overview of cellular processes, the identification of upstream processes that drive expression changes remains a challenge. To address this issue, we introduce *CARNIVAL* [1], a causal network contextualization tool which derives network architectures from gene expression footprints. *CARNIVAL* (CAusal Reasoning pipeline for Network identification using Integer VALue programming)(see `https://saezlab.github.io/CARNIVAL/`) integrates different sources of prior knowledge including signed and directed protein–protein interactions, transcription factor targets, and pathway signatures.

*enio.gjerga@bioquant.uni-heidelberg.de

1.1 CARNIVAL pipeline

CARNIVAL refines a quantitative objective function for ILP problem by incorporating TF and pathway activities on a continuous scale. In addition, the CARNIVAL framework allows us to contextualize the network with or without known targets of perturbations. The implementation is separated into two pipelines which will be referred henceforth as Standard CARNIVAL *StdCARNIVAL* (with known perturbation targets as an input) and Inverse CARNIVAL *InvCARNIVAL* (without information on targets of perturbation), see Figure **??**. The differential gene expression is used to infer transcription factor (TF) activities with DoRothEA, which are subsequently discretized in order to formulate ILPconstraints. As a result, CARNIVAL derives a family of highest scoring networks which best explain theinferred TF activities. Continuous pathway and TF activities can be additionally considered in the objective function.

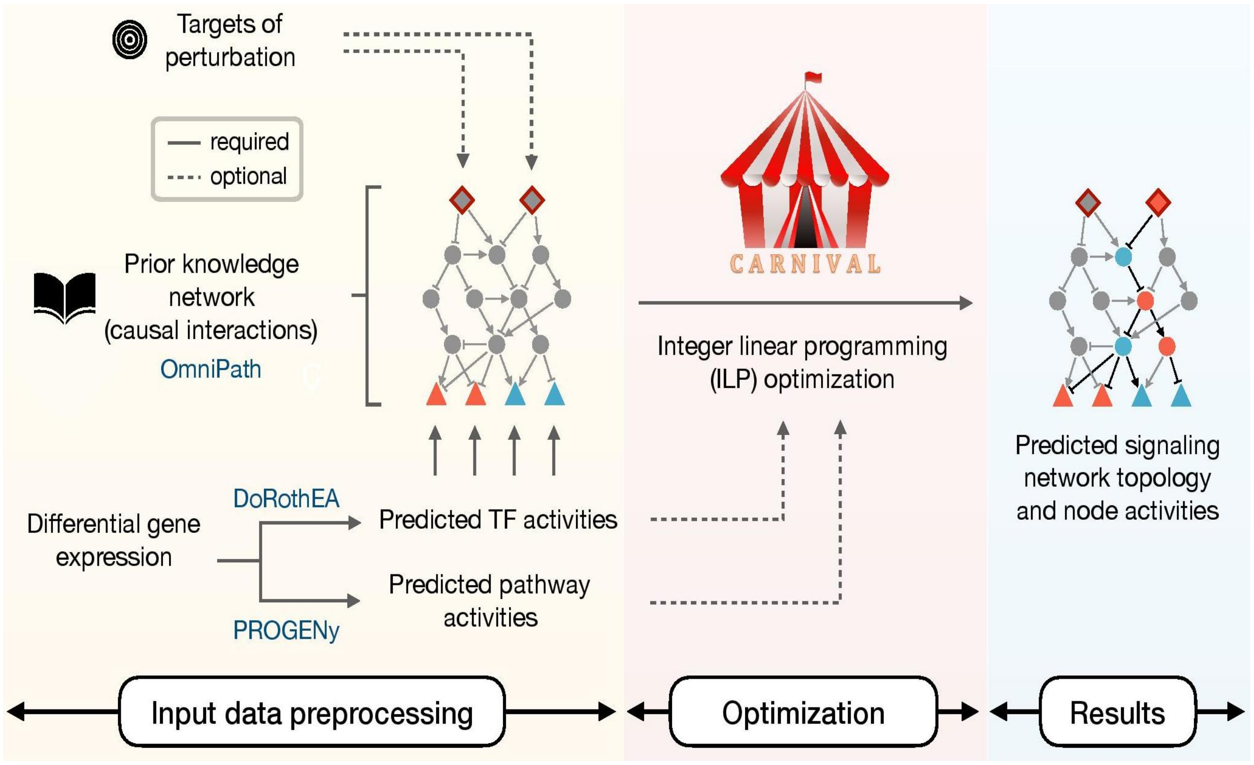

Targets of perturbation
required
optional
Prior knowledge network (causal interactions)
OmniPath
CARNIVAL
Integer linear programming (ILP) optimization
Predicted signaling network topology and node activities
Differential gene expression
DoRothEA
PROGENy
Predicted TF activities
Predicted pathway activities
Input data preprocessing
Optimization
Results

Figure 1: *CARNIVAL* pipeline

1.2 ILP solvers

CARNIVAL is an extension of the previously implemented Causal Reasoning method from Melas et al. [2].The network inference process is swiftly performed with an Integer Linear Programming (ILP) formulation of causal reasoning using two solvers: the open-source mixed integer programming solver *Cbc* (Coin-or branch and cut)(see `https://projects.coin-or.org/Cbc`); or the *CPLEX optimizer* from *IBM* (see `https://www.ibm.com/analytics/cplex-optimizer`) which can be obtained for free through the Academic Initiative. To perform the analysis, the users will then need to store the binary *cbc* or *cplex* executables on any directory they wish. The binary files of *cbc* can be found by first downloading one of the optimization suites provided here: `https://www.coin-or.org/download/binary/OptimizationSuite/`, unzip the download and from there save the *cbc* executable (which can be found on the *bin* directory) file on any of the direcotries they wish of their machines. As for the *cplex*, the executable file can be obtained after registration on the *ILOG CPLEX Optimization Studio* here: `https://my15.digitalexperience.ibm.com/b73a5759-c6a6-4033-ab6b-d9d4f9a6d65b/dxsites/151914d1-03d2-48fe-97d9-d21166848e65/technology/data-science`. Similar like before, users will have to find the *cplex* executable binary file and save on a directory of their own wish or keep them on their default installation paths. The path to interactive version of *CPLEX* is differed based on the operating system. The default installation path for each OS is as follows:

For Mac OS:

```
# ~/Applications/IBM/ILOG/CPLEX_Studio129/cplex/bin/x86-64_osx/cplex
```

For Linux:

```
# /opt/ibm/ILOG/CPLEX_Studio129/cplex/bin/x86-64_linux/cplex
```

For Windows:

```
# C:/Program Files/IBM/ILOG/CPLEX_Studio129/cplex/bin/x64_win64/cplex.exe
```

Note that the version of *CPLEX* has to be changed accordingly (the latest version is CPLEX-Studio129).

1.3 Citation

CARNIVAL can be cited as follows:

Liu, A., Trairatphisan, P., Gjerga, E. et al. From expression footprints to causal pathways: contextualizing large signaling networks with CARNIVAL. npj Syst Biol Appl 5, 40 (2019) doi:10.1038/s41540-019-0118-z

2 Getting Started

A tutorial for preparing *CARNIVAL* input files starting from differentially gene expression (DEG) and for running the *CARNIVAL* pipeline are provided as vignettes in R-Markdown, R-script and HTML formats. The wrapper script *runCARNIVAL* was introduced to take input arguments, pre-process input descriptions, run optimisation and export results as network figures. Three built-in *CARNIVAL* examples are also supplied as case studies for users. Moreover processed gene expression values (besides differential values) can also be used as an input to *CARNIVAL* but examples are not provided in this vignette.

2.1 Prerequisites

Besides the above mentioned solvers, users need also to install the following R-package dependencies: *readr*(see https://cran.r-project.org/web/packages/readr/index.html); *igraph* (see https://igraph.org/r/); *readxl*(see https://readxl.tidyverse.org/); *dplyr*(see https://www.rdocumentation.org/packages/dplyr/versions/0.7.8).

In order to visualize the automatically generated *CARNIVAL* networks, users will also need to download and install the Graph Visualization software *graphviz*(see https://www.graphviz.org/).

2.2 Installation

CARNIVAL is currently available for the installation as an R-package from our GitHub page.

```
# Install CARNIVAL from Github using devtools
# install.packages('devtools') # in case devtools hasn't been installed
library(devtools)
install_github('saezlab/CARNIVAL', build_vignettes = TRUE)
# or download the source file from GitHub and install from source
install.packages('path_to_extracted_CARNIVAL_directory',
                 repos = NULL, type="source")
```

2.3 Documentation

After successfully installing *CARNIVAL*, its documentation can be accessed as follows.

```
# Open vignette as pdf file
vignette("CNORode-vignette")
```

2.4 CARNIVAL settings

The main function of the *CARNIVAL* apckage is *runCARNIVAL*. This function runs the pipeline by using the user-provided list of inputs or run CARNIVAL built-in examples. These inputs are:

- *solverPath*: Pointing to the path where the *cplex* or the *cbc* executable files are stored.

- *netObj*: A matrix or data-frame object containing the prior knowledge networks as a list of directed and signed protein interactions. The first column should contain the source node, the second column should contain the sign of the interaction and the third column should contain the target node.

- *measObj*: A matrix or data-frame object containing the measurements (either gene expressions or estimated transcription factor activities).

- *inputObj*: A matrix or data-frame object containing the experimental perturbation targets. The values can either be 1 (for activatory perturbations), -1 (for inhibitory effects of a perturbation) or NA (when we know that a protein is being targeted but we are not sure about the effect - in this case *CARNIVAL* will try to infer the most likely effect). Default set to NULL (in this case we run *InvCARNIVAL*).

- *weightObj*: A matrix or data-frame object containing the pathway activity scores as inferred from *runPROGENy*. Default set to NULL.

- *parallelIdx1*: First index number suitable for parallelisation (numeric). Default set to 1.

- *parallelIdx2*: Second index number suitable for parallelisation (numeric). Default set to 1.

- *nodeID*: Define the input format of nodes in the network (either 'uniprot' or 'gene' symbol). Default set to 'uniprot'. Leave it to default if you do not wish to map proteins or if you using other symbols besides gene/uniprot ID's.

- *UP2GS*: For plotting: define if Uniprot ID will be converted to gene symbols for better readability (logical T/F).

- *DOTfig*: For plotting: define if DOT figure will be exported in the result folder (logical T/F). Default set to FALSE

- *timelimit*: Optimization parameter: Time limit of optimisation (in seconds)

- *mipGAP*: CPLEX parameter: Allowed gap of accepted solution comparing to the best solution (fraction; default: 0.05 = 5 percents).

- *poolrelGAP*: Optimization parameter: Allowed relative gap of accepted solution comparing within the pool of accepted solution (fraction; Default set to 0.0001)

- *limitPop*: CPLEX parameter: Allowed number of solutions to be generated (Default set to 500)

- *poolCap*: CPLEX parameter: Allowed number of solution to be kept in the pool of solution (Default set to 100)

- *poolIntensity*: CPLEX parameter: Intensity of solution searching (0,1,2,3,4 - Default set to 4)

- *poolReplace*: CPLEX parameter: Replacement strategy of solutions in the pool (0,1,2 - Default set to 2 = most diversified solutions)

- *alphaWeight*: Objective function: weight for mismatch penalty (Default set to 1 - will only be applied once measurement file only contains discrete values).

- *betaWeight*: Objective function: weight for node penalty (Default set to 0.2)

- *dir name*: Directory name where the users wishes to save the plotted network as a .DOT figure (in case if DOTfig=TRUE). By default it will be stored on a DOTfigure folder within the current working directory.

- *solver*: Solver to use: *cplex* or *cbc* (Default set to *cplex*).

- *experimental conditions*: In the case when the user wishes to combine information from multiple experimental conditions, it has to specify which experimental conditions wishes to use. Default set to NULL - in which case *CARNIVAL* will generate as many network solutions as there are rows in the *measObj*, *inputObj* and/or *weightObj*.

While *Cbc* is open-source and can be used from any user, the *CPLEX* solver is more computationally efficient and is able to provide multiple equivalent solutions which are then combined. The *mipGAP*, *limitPop*, *poolCap*, *poolIntensity* and *poolReplace* make sense only if *CPLEX* solver is used to train the networks.

3 Running CARNIVAL

In the CARNIVAL package, built-in examples are available as the test cases as follows:

1. SBVimprover species translational dataset with EGF as the perturbator
2. TG-GATEs dataset with paracetamol (APAP) as the perturbator
3. Toy Model of two crosstalk pathways

3.1 DEG Pre-processing

Users can generate the measurement file which contains calculated transcription factor (TF) normalised enrichment scores by applying the function *runDoRothEA*. An example of the pipeline to generate a measurement file for the Example 1 (SBVimprover-EGF) is shown as follows:

```
library(CARNIVAL) # load CARNIVAL library
library(dplyr) # load dplyr library
library(readr) # load readr library
library(igraph) # load igraph library
library(readxl) # load readxl library
file.copy(from=system.file("SBV_EGF_tvalues.csv",package="CARNIVAL"),
           to=getwd(),overwrite=TRUE)
file.copy(from=system.file("dorothea_TF_mapping.csv",package="CARNIVAL"),
           to=getwd(),overwrite=TRUE)
load(file = system.file("BEST_viperRegulon.rdata",package="CARNIVAL"))
df<-read.csv2("SBV_EGF_tvalues.csv", row.names = 'GeneName')
map<-read.csv("dorothea_TF_mapping.csv")
#Run DoRothEA and convert from gene symbol to uniprot ID
TF_genesymbol<-runDoRothEA(df, regulon=viper_regulon,
                              confidence_level=c('A','B','C'))
TF_uniprot<-GeneSymbol2Uniprot(TF_genesymbol, map, 1, 2)
#Generate measurement files in CARNIVAL input format
generate_measfile(measurements=TF_uniprot, topnumber=50,
                   write2folder="measurements")
```

Also, users can generate the additional weight input file which contains calculated pathway scores by applying the function *runPROGENy*. An example of the pipeline to generate a weight object file for Example 1 (SBVimprover-EGF) is shown as follows:

```
## Loading weight matrix of PROGENy
file.copy(from=system.file("model_NatComm+14_human.csv",package="CARNIVAL"),
           to=getwd(),overwrite=TRUE)
weight_matrix<-read.csv("model_NatComm+14_human.csv")
df_genenames<-data.frame('gene'=rownames(df),df)
## Run PROGENy
pathway_scores<-runPROGENy(df_genenames,weight_matrix, z_scores = F)
```

The results from DEG-preprocessing pipeline are saved in the directory ”measurements” in the current working directory.

3.2 CARNIVAL pipeline - Example 1

To run the CARNIVAL pipeline, users fist have to define the path to interactive CPLEX on the variable *solverPath* in the runCARNIVAL function. The path to the interactive version of *Cbc* is where the user has copied this file The path to interactive version of CPLEX is differed based on the operating system. The default installation path for each OS is as described in *Section 1.2*.

For this example, we can set:

```
solverPath = "~/Applications/IBM/ILOG/CPLEX_Studio1281/cplex/bin/x86-64_osx/cplex"
```

Next, users can load the *CARNIVAL* inputs for Example 1. The *PROGENy* weights can then be assigned to representative nodes (via *progenyMembers.RData*) of each of the pathways through *assign-PROGENyScores* function. On this case:

```
##
# Example 1
load(file = system.file("Ex2_inputs_SBV_EGF.RData",
                         package="CARNIVAL"))
load(file = system.file("Ex2_measurements_SBV_EGF.RData",
                         package="CARNIVAL"))
load(file = system.file("Ex2_network_SBV_Omnipath.RData",
                         package="CARNIVAL"))
load(file = system.file("progenyMembers.RData",package="CARNIVAL"))
## pathway_scores as generated as described in section 3.1
weightObj <- assignPROGENyScores(progeny = t(pathway_scores),
                                  progenyMembers = progenyMembers,
                                  id = "uniprot")
```

The *measurements* object is the same as table generated from the *generate measfile* function after estimating the Transcription Factor (TF) activites via *DoRothEA* [3] analysis (see *runDoRothEA* function) as described above. The *weights* are the normalized estimated pathway activities through *PROGENy* [4] (sign indicating the direction of the regulation) (see *runPROGENy*). The *inputs* represents the target of the perturbation (activation of EGF on this case). The *network* object represents the signed and directed prior knowledge network. The proteins on this case are mapped as *Uniprot ID's*.

As a next step, we run the *runCARNIVAL* function to perform the analysis as shown below. We allow a maximal of 1 hour timelimit to perform the analysis, although it will only take few minutes. Also we ask the solver to give us 100 (*poolCap*) of the most diverse (*poolReplace*) 500 (*limitPop*) solutions inferred:

```
## Without PROGENy weights
result1 = runCARNIVAL(solverPath=solverPath,
                      netObj = Ex2_network_SBV_Omnipath,
                      measObj = Ex2_measurements_SBV_EGF,
                      inputObj = Ex2_inputs_SBV_EGF,
                      weightObj = NULL,
                      timelimit = 3600,
                      poolCap = 100,
                      poolReplace = 2,
                      limitPop = 500,
                      nodeID = "uniprot",
                      UP2GS = TRUE,
                      DOTfig = TRUE,
                      solver = "cplex")
## With PROGENy weights
result2 = runCARNIVAL(solverPath=solverPath,
                      netObj = Ex2_network_SBV_Omnipath,
                      measObj = Ex2_measurements_SBV_EGF,
                      inputObj = Ex2_inputs_SBV_EGF,
                      weightObj = weightObj,
                      timelimit = 3600,
                      poolCap = 100,
                      poolReplace = 2,
                      limitPop = 500,
                      nodeID = "uniprot",
                      UP2GS = TRUE,
                      DOTfig = TRUE,
                      solver = "cplex")
```

Here we use *CPLEX* solver for running the analysis and it takes up to one hour Once the analysis is finished, the returned *result1* and *result2* object contains the resulting weighted network (*weightedSIF* – combined network solution with weights of interactions telling how often an interaction appears in the pool of solutions); the attributes of each node (*nodesAttributes* – indicating the average inferred protein activities across the multiple solutions); all the single network solutions inferred (*sifAll*);

result1 corresponds to the *CARNIVAL* results without considering *PROGENy* weights, while *result2* corresponds to the *CARNIVAL* when considering them.

Since we set *DOTfig=TRUE*, the analysis will generate the *.dot* file as visualized in Figure 3 and 2 of the network within the default *DOTfig* directory (please use *dir name* variable if you wish to save the network solution at a different directory and on a different name). This figure represents the combined solution network as in the *weightedSIF*. Since the *UP2GS=TRUE*, the generated network will have Gene ID's, otherwise it can be set to *FALSE*. Otherwise if the user is not interested to generate any *.dot* figure, it can as well keep the default *DOTfig=FALSE* for the analysis.

3.3 CARNIVAL pipeline - Example 2

In case the target(s) of perturbation is unknown (e.g. in the case of multi-factorial diseases or broad-spectrum drugs), users can choose the Inverse CARNIVAL pipeline to exclusively build a sub-network without input's target(s) information (i.e. inputFile = NULL, inverseCR=T). Note that the pathway scores from PROGENy is highly recommended to be included once running the Inverse CARNIVAL pipeline. Here we present Example 3 as a case study for the Inverse CARNIVAL pipeline (to investigate downstream toxicity effects of paracetamol/APAP on cellular processes). An example of R-code by is provided below:

Training with *CPLEX* (100 combined solutions showed in Figure 4):

```
library(CARNIVAL) # load CARNIVAL library
library(dplyr) # load dplyr library
library(readr) # load readr library
library(igraph) # load igraph library
library(readxl) # load readxl library
load(file = system.file("Ex3_measurement_APAP_TGG_24hrHighDose.RData",
                         package="CARNIVAL"))
load(file = system.file("Ex3_network_APAP_TGG_Omnipath.RData",
                         package="CARNIVAL"))
# cplex solver
result_cplex = runCARNIVAL(solverPath=solverPath,
                            netObj = Ex3_network_APAP_TGG_Omnipath,
                            measObj = Ex3_measurement_APAP_TGG_24hrHighDose,
                            timelimit = 3600,
                            poolCap = 100,
```

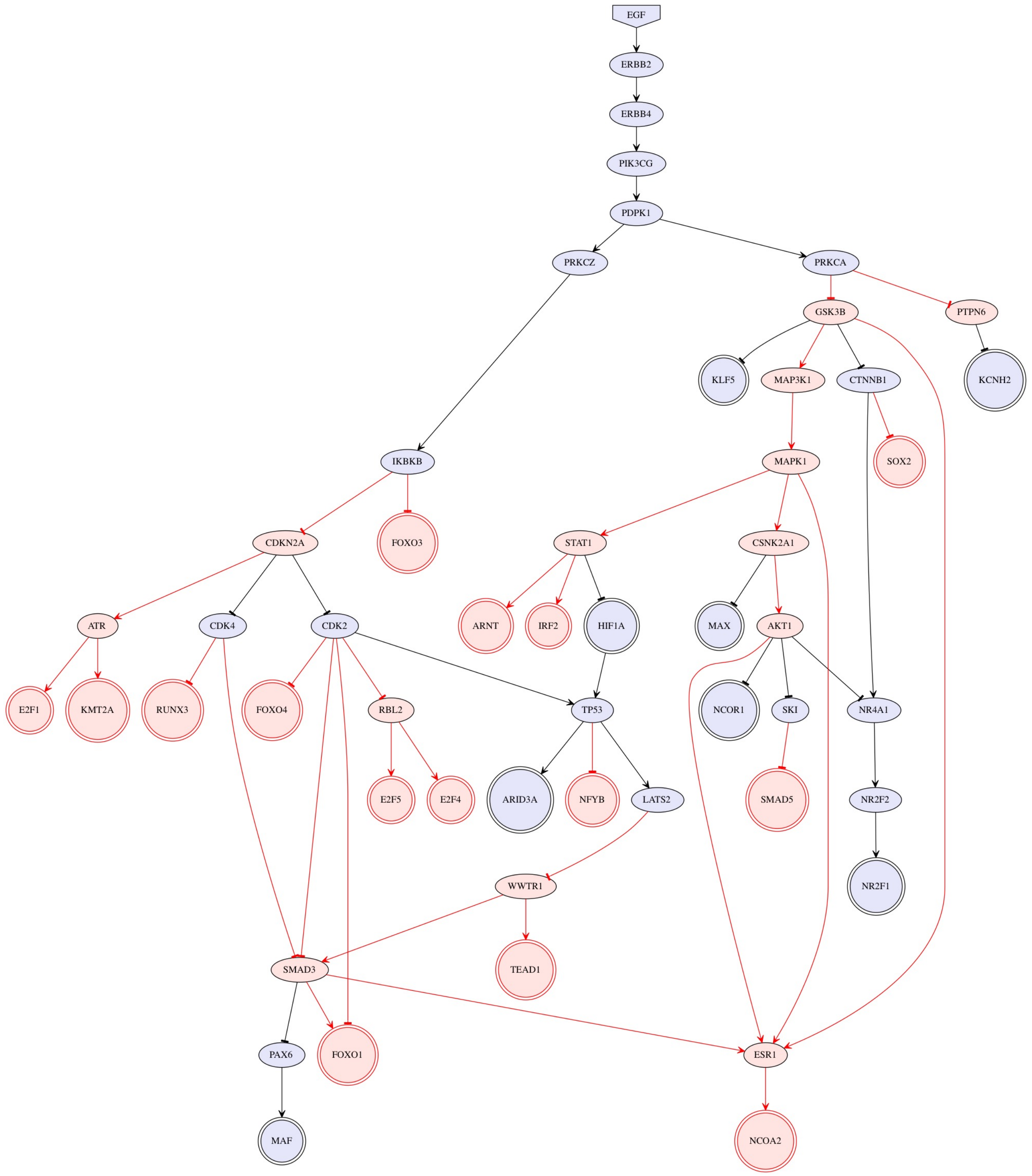
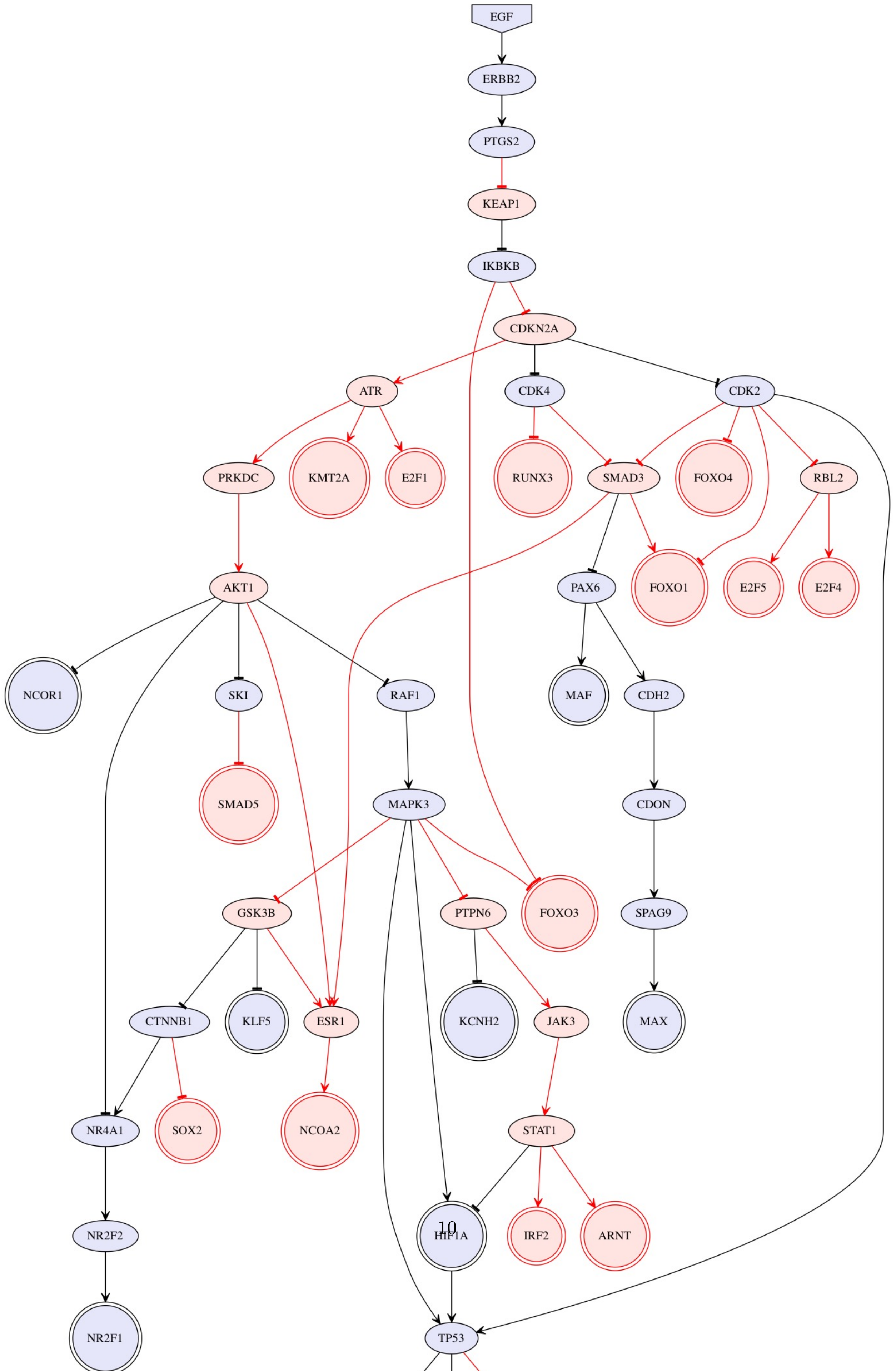


Figure 2: Solution network - Example 1 - Analysis without PROGENy weights


EGF
ERBB2
PTGS2
KEAP1
IKBKB
CDKN2A
ATR
CDK4
CDK2
PRKDC
KMT2A
E2F1
RUNX3
SMAD3
FOXO4
RBL2
AKT1
PAX6
FOXO1
E2F5
E2F4
NCOR1
SKI
RAF1
MAF
CDH2
SMAD5
MAPK3
CDON
GSK3B
PTPN6
FOXO3
SPAG9
CTNNB1
KLF5
ESR1
KCNH2
JAK3
MAX
NR4A1
SOX2
NCOA2
STAT1
NR2F2
HIF1A
IRF2
ARNT
NR2F1
TP53

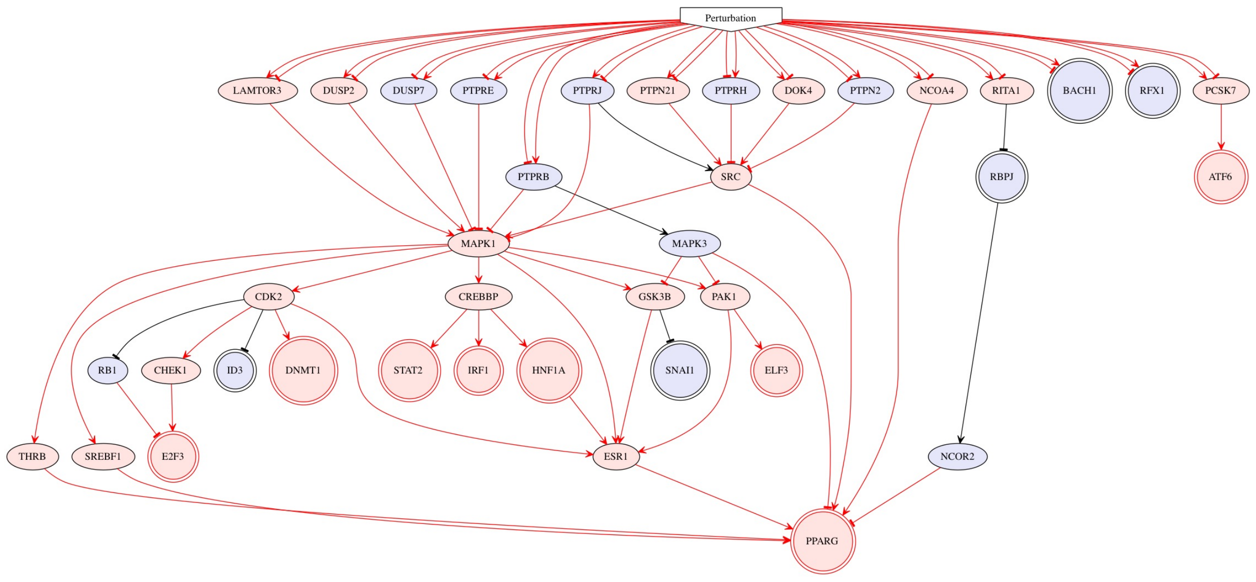


Figure 4: Solution network - Example 2 - Result with *CPLEX* solver

```
                             poolReplace = 2,
                             limitPop = 100,
                             nodeID = "uniprot",
                             UP2GS = TRUE,
                             DOTfig = TRUE,
                             solver = "cplex")
```

Training with *Cbc* (1 single solution obtained and showed in Figure 5):

```
library(CARNIVAL) # load CARNIVAL library
library(dplyr) # load dplyr library
library(readr) # load readr library
library(igraph) # load igraph library
library(readxl) # load readxl library
load(file = system.file("Ex3_measurement_APAP_TGG_24hrHighDose.RData",
                        package="CARNIVAL"))
load(file = system.file("Ex3_network_APAP_TGG_Omnipath.RData",
                        package="CARNIVAL"))
# cbc solver
result_cbc = runCARNIVAL(solverPath=solverPath,
                          netObj = Ex3_network_APAP_TGG_Omnipath,
                          measObj = Ex3_measurement_APAP_TGG_24hrHighDose,
                          timelimit = 3600,
                          nodeID = "uniprot",
                          UP2GS = TRUE,
                          DOTfig = TRUE,
                          solver = "cbc")
```

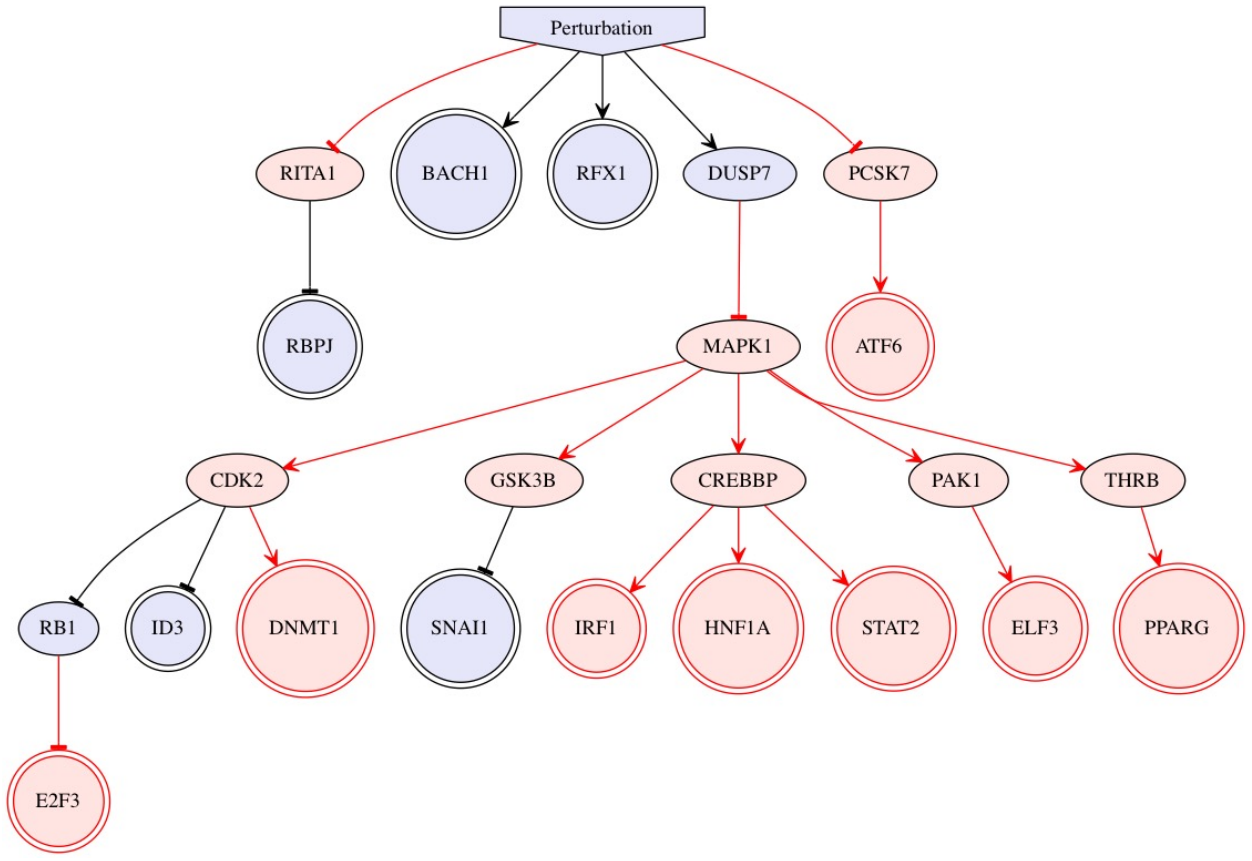
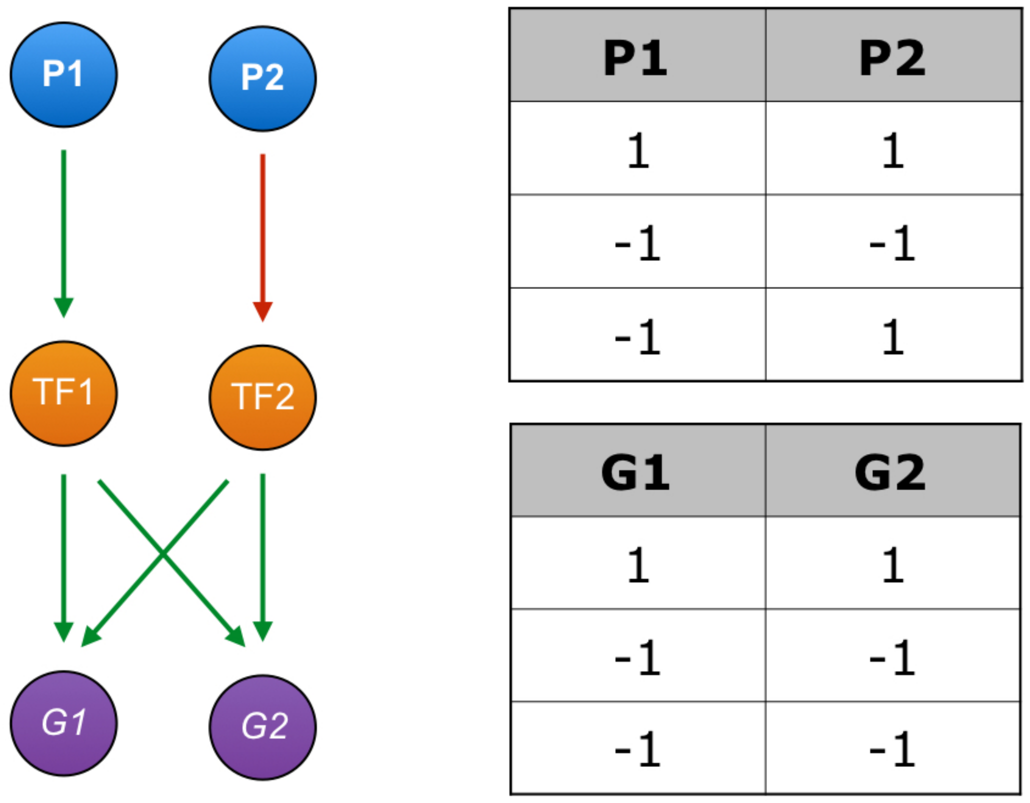


Figure 5: Solution network - Example 2 - Result with *Cbc* solver



P1	P2
1	1
-1	-1
-1	1

G1	G2
1	1
-1	-1
-1	-1

Figure 6: Toy example with multiple experimental conditions. *Left panel:* Prior knowledge network. *Right upper panel:* Inputs. *Right lower panel:* Measurements.

3.4 Toy example with multiple conditions

In the context of multiple perturbation experiments, users would wish to obtain one consensus model fitting best the combined experiments rather than obtaining separate solutions for each experimental conditions. *CARNIVAL* has been implemented in such a way to handle these cases and we show one such application over a small toy example as shown in Figure 6:

The toy example shows the effects of perturbing nodes *P1* and *P2* over *G1* and *G2*. On the left panel we have the prior knowledge network showing how inputs connect to the measurements through the activatory interactions.

We train the network with *CPLEX* as below:

```
library(CARNIVAL) # load CARNIVAL library
library(dplyr) # load dplyr library
library(readr) # load readr library
library(igraph) # load igraph library
library(readxl) # load readxl library
load(file = system.file("Ex1_inputs_Toy_mult.RData",
                         package="CARNIVAL"))
```

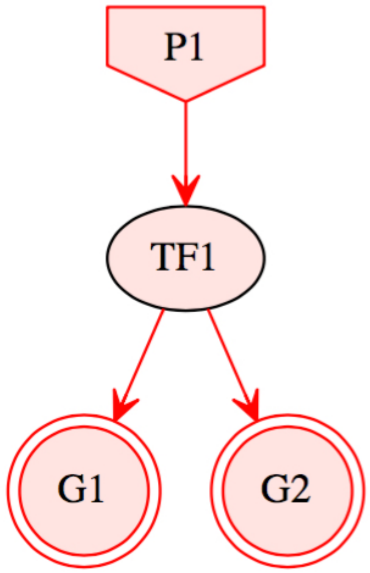


Figure 7: Consensus network solution of the Toy Example.

```
load(file = system.file("Ex1_measurements_Toy_mult.RData",
                        package="CARNIVAL"))
load(file = system.file("Ex1_network_Toy_mult.RData",
                        package="CARNIVAL"))
# cplex solver
result1 = runCARNIVAL(solverPath = "~/Documents/cplex",
                      netObj = Ex1_network_Toy_mult,
                      measObj = Ex1_measurements_Toy_mult,
                      inputObj = Ex1_inputs_Toy_mult,
                      mipGAP = 0,
                      poolrelGAP = 0,
                      DOTfig = TRUE,
                      solver = "cplex",
                      experimental_conditions = c(1, 2, 3))
```

The consensus network is as shown in Figure 7.

We can easily notice that this is the desired network although measurement *G2* is not perfectly fitted well on the third experimental condition. This is because if by connecting *P2* to *G2* through *TF2*, we would obtain a good fit of *G2* on the third experiment, however this would come at the expense of mis-fiting other measurements for the other two experimental conditions.

When running *CARNIVAL* for multiple experimental conditions we only obtain the network solution and not the list of the node activities since they might differ from one experimental condition to the other.

References

[1] A. Liu, P. Trairatphisan, E. Gjerga, A. Didangelos, J. Barratt and J. Saez-Rodriguez. From expression footprints to causal pathways: contextualizing large signaling networks with CARNIVAL. npj Syst Biol Appl 5, 40 (2019) doi:10.1038/s41540-019-0118-z.

[2] I.N. Melas, T. Sakellaropoulos, F. Iorio, L.G. Alexopoulos, W. Loh, D.A. Lauffenburger, J. Saez-Rodriguez and J.P.F. Bai Identification of drug-specific pathways based on gene expression data: application to drug induced lung injury. Integr Biol (Camb). 2015 Aug;7(8):904-20. doi: 10.1039/c4ib00294f.

[3] L. Garcia-Alonso, F. Iorio, A. Matchan, N. Fonseca, P. Jaaks, G. Peat, M. Pignatelli, F. Falcone, C.H. Benes, I. Dunham, G.R. Bignell, S. McDade, M.J. Garnett and J. Saez-Rodriguez Transcription Factor Activities Enhance Markers of Drug Sensitivity in Cancer. Cancer Research, 78(3), 769–780.

[4] M. Schubert, B. Klinger, M. Klunemann, A. Sieber, F. Uhlitz, S. Sauer, M.J. Garnett, N. Bluthgen and J. Saez-Rodriguez Perturbation-response genes reveal signaling footprints in cancer gene expression Nat Commun. 2018;9(1):20.