GeneTech

A *Tech*nology mapping tool for **Gene**tic Circuits

A quick start guide v1.0

November 06 2017

Department of Applied Mathematics and Computer Science
Section of Embedded Systems Engineering
Bio-Design Automation Group
Richard Petersens Plads, Lyngby 3200
Technical University of Denmark

http://bda.compute.dtu.dk/tools/genetech/

Contents

About Gene*Tech*

Gene*Tech* (extracted from **Gene**tic *Tech*nology mapping) is a tool which allows a user to generate genetic logic circuits only by specifying the logical function desired to be achieved in a living cell. It performs Boolean optimization, followed by synthesis and technology mapping using a library of genetic logic gates. The genetic logic gates library used in this work has been developed and tested in the laboratory by MIT and Boston University [1]. **Gene***Tech* takes the Boolean expression of a genetic circuit as input, and then first optimize it. Afterwards, it synthesizes the optimized Boolean expression into NOR-NOT form in order to construct the circuit using the real NOR/NOT gates available in the genetic gates library [1]. In the end, **Gene***Tech* performs technology mapping to generate all the feasible circuits, with different genetic gates, to achieve the desired logical behavior.

Gene*Tech* is a java-based tool and is currently available in JAR format. In this short document, you will learn how to use **Gene***Tech* (v1.0) to develop genetic circuits.

Stochastic simulation algorithm (SSA) has been implemented to perform stochastic simulations of SBML models. Furthermore, D-VASim is also capable of simulating the deterministic behavior of a bio model by solving ordinary differential equations.

Latest version of **Gene***Tech* can be downloaded from http://bda.compute.dtu.dk/downloads/genetech/. The sample set of genetic Boolean expressions have also been included in the download package.

Conventions

The following conventions appear in this document:

This icon denotes a tip, which notifies you to advisory information.

This icon denotes an alert, which notifies you to important information.

bold — Bold text denotes items that you must select or click or enter the value in the software, such as providing input Boolean expression.

italic — Italic text denotes the name of a folder or a file path.

bold and italic — Bold and italic text denotes the name of a file.

1. Running Gene*Tech*

In this section, you will learn how to run **Gene***Tech* and construct desired genetic circuits.

To run, **Gene***Tech*, first start the Terminal (in Mac OS) or Command Prompt (in Windows), and then follow the steps given below.

1. Change the current working directory to *../GeneTech/jar/*
2. Execute **java –jar GeneTech.jar** to run ***GeneTech***

Alert – Latest JAVA Runtime Engine (JRE) should be installed on your computer.

Alert – The screen-captured images included in this documentation are taken on Mac OS. The same steps are applicable for another OS.

1.1. Input Interface of Gene*Tech*

Once the tool is started, it first requires a user to provide inputs based on which the **Gene***Tech* perform optimization first. Since the tool first optimizes the Boolean expression based on the Simulated Annealing algorithm [2], it is therefore required to provide the values of **temperature coefficient**, **initial temperature**, and **time to run** (in seconds) the optimization algorithm, as shown in Figure 1.

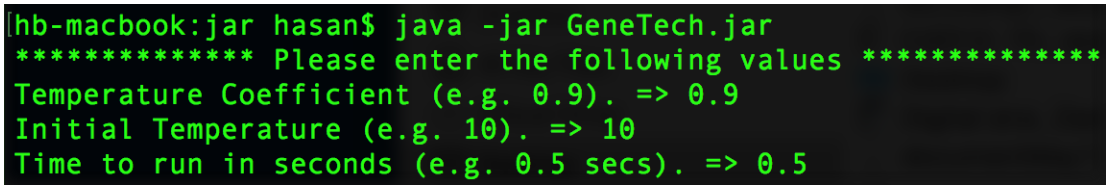

Figure 1. The values required for optimization. This figure shows that the user has already entered the values after the symbol =>.

After entering the values required for logic optimization, a user is then asked to provide the Boolean function in the form of three variables **IPTG**, **aTc**, and **Arabinose**. These variables correspond to the name of external inducers and the logical operation is performed based on their presence/absence inside a living cell. **Gene***Tech* is able to process Boolean expression in SOP form. Figure 2 shows the screen-captured image of an instant when the user has entered a Boolean expression (IPTG'.aTc.Arabinose'+IPTG'.aTc.Arabinose+IPTG.aTc.Arabinose). It should be noted that there is no space in between the characters used in the Boolean expression.

Alert – The inverted terms should be described by " ' "; ANDed terms with " . "; and ORed terms with " + ", without any spaces.

Alert – Names, spelling and the case of input inducers should remain the same i.e., IPTG cannot be iptg; aTc cannot be ATC or atc, etc.

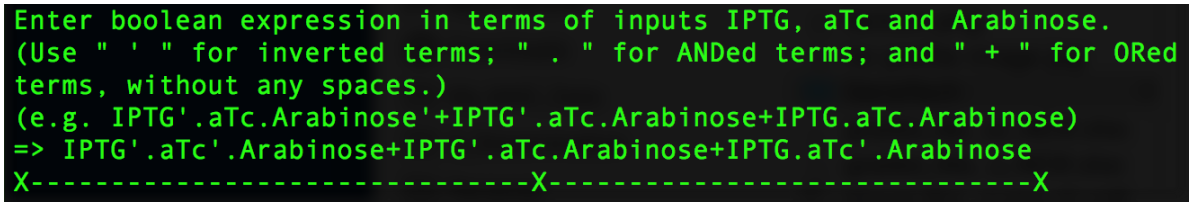

Figure 2. Image showing that the user has entered a Boolean expression after the symbol => according to the defined rules.

1.2. Optimization and Synthesis by Gene*Tech*

When a correct logic function is typed in, as shown in Figure 2, the tool first estimates the initial cost of a Boolean expression in terms of literals; where the name of each input inducer is considered as a separate literal. For example, the number of literals, in the entered Boolean expression are 9. Therefore, the cost of this logical function is 9 as shown to be estimated by the tool in Figure 3.

Gene*Tech* then reduces the input Boolean expression and calculate its new cost, as shown in Figure 3. Once the input expression is reduced, it is then synthesized into NOR/NOT form, as depicted in the same figure.

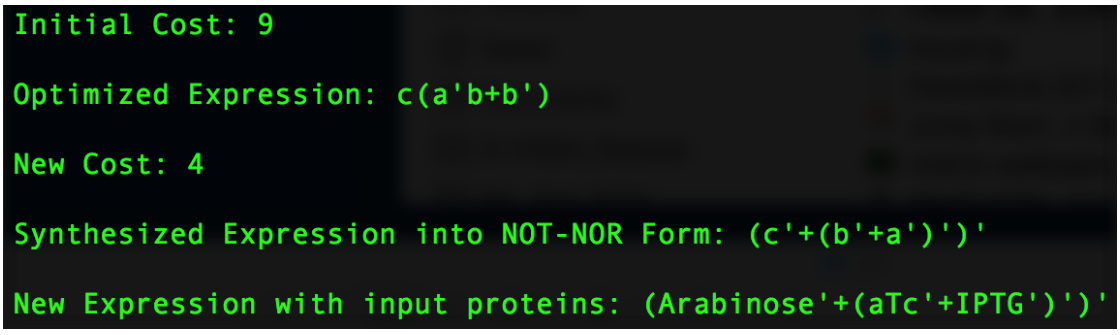

Figure 3. The optimization and synthesis output of **Gene***Tech* tool.

1.2. Technology Mapping by Gene*Tech*

Before performing technology mapping, the optimized expression is synthesized into NOR-NOT form because the genetic gates library used by **Gene***Tech* tool consists of NOR and NOT genetic gates. If **Gene***Tech* found compatible gates, it lists down all the possible genetic circuits as shown in Figure 4. This figure shows that the tool suggests four possible different genetic circuits to obtain the logic expressed as a Boolean expression shown in Figure 2. **Gene***Tech* generates the circuit in the form of a text string which is similar to SBOL visual notations [3]. For example, the genetic circuit 1 shown in Figure 4 indicates that the protein, AmtR, generated (shown in the third line of circuit 1) by the promoter PTac represses its corresponding promoter PAmtR (shown in the first line of circuit 1).

```
************************************ Genetic Circuit 1 ************************************

PTet-> (AmeR)----|PAmeR-> PAmtR-> (Betl)----|PBetl-> PSrpR-> (PhlF)----|PPhlF-> (YFP)
PBad-> (SrpR)-----------------------------------------T
PTac-> (AmtR)--------------T

*******************************************************************************************

************************************ Genetic Circuit 2 ************************************

PTet-> (AmeR)----|PAmeR-> PHlYllR-> (Betl)----|PBetl-> PSrpR-> (PhlF)----|PPhlF-> (YFP)
PBad-> (SrpR)-----------------------------------------T
PTac-> (HlYllR)------------T

*******************************************************************************************

************************************ Genetic Circuit 3 ************************************

PTet-> (AmtR)----|PAmtR-> PHlYllR-> (Betl)----|PBetl-> PSrpR-> (PhlF)----|PPhlF-> (YFP)
PBad-> (SrpR)-----------------------------------------T
PTac-> (HlYllR)------------T

*******************************************************************************************

************************************ Genetic Circuit 4 ************************************

PTet-> (AmtR)----|PAmtR-> PPhlF-> (HlYllR)----|PHlYllR-> PSrpR-> (BM3R1)----|PBM3R1-> (YFP)
PBad-> (SrpR)-----------------------------------------T
PTac-> (PhlF)--------------T

*******************************************************************************************
```

Figure 4. The technology mapping results produced by **Gene***Tech*.

2. Genetic Gates Library

The genetic gates library used by **Gene***Tech* is available as a text file – ***GatesLib.txt***, and is shown in Figure 5. The gates are arranged in this file as Ext, Int or Semi-Ext-, based on if the inputs are external inputs i.e. IPTG (PTac) or aTc (PTet) or Arabinose (PBad); internal inputs; or semi-external inputs.

Alert – The ***GatesLib.txt*** file should not be removed from its default location inside the download folder, otherwise the tool will not function properly.

For example, in Figure 5, the first gate in the list of internal inverters, <Int-Inverters>, has an input promoter PBM3R1 which suppresses the output promoter PSrpR. Similarly, in the list of semi-external NOR gates, <Semi-Ext-NorGates>, the first input of the first NOR gate is one external input promoter i.e. PTac (or IPTG), the second one is intermediate promoter, PAmer, and they both suppress the output promoter PBetl.

Similarly, other repression-based gates can be added in the gates library of **Gene***Tech*.

GatesLib — Edited

```
<Ext-Inverters>
Ext-Input       In      Out
a'      PTac    PAmtR END
a'      PTac    PHlYllR END
a'      PTac    PSrpR END
a'      PTac    PPhlF END
b'      PTet    PAmeR END
b'      PTet    PAmtR END
c'      PBad    PAmtR END
c'      PBad    PSrpR END
c'      PBad    PBM3R1 END
</Ext-Inverters>
<Int-Inverters>
PBM3R1  PSrpR END
PBetl   PHlYllR END
PPhlF   PAmtR END
PBM3R1  PAmtR END
PSrpR   PAmtR END
PPhlF   PBM3R1 END
PHlYllR PBetl END
PPhlF   PHlYllR END
</Int-Inverters>
<Ext-NorGates>
In1     In2     Out
PTac    PTet    PSrpR END
PTac    PTet    PAmer END
PTac    PTet    PHlYllR END
PTac    PTet    PPhlF END
PTac    PTet    PBetl END
PBad    PTet    PHlYllR END
PBad    PTet    PSrpR END
PBad    PTet    PBM3R1 END
</Ext-NorGates>
<Semi-Ext-NorGates>
PTac    PAmeR   PBetl END
PTac    PAmtR   PHlYllR END
PTac    PAmtR   PPhlF END
PTac    PBetl   PPhlF END
PTac    PHlYllR PPhlF END
PTet    PHlYllR PBetl END
PTet    PSrpR   PPhlF END
PTet    PPhlF   PSrpR END
PTet    PBM3R1  PSrpR END
PTet    PAmtR   PBetl END
PTet    PPhlF   PHlYllR END
PTet    PHlYllR PBM3R1 END
PBad    PHlYllR PBM3R1 END
PBad    PHlYllR PAmtR END
PBad    PAmtR   PPhlF END
PBad    PHlYllR PSrpR END
</Semi-Ext-NorGates>
<Int-NorGates>
PSrpR   PAmtR   PPhlF END
PHlYllR PAmeR   PBetl END
PSrpR   PBetl   PPhlF END
PPhlF   PHlYllR PBM3R1 END
PAmtR   PAmeR   PBetl END
PBetl   PHlYllR PPhlF END
PPhlF   PAmtR   PBetl END
PSrpR   PHlYllR PBM3R1 END
PSrpR   PHlYllR PPhlF END
PPhlF   PAmtR   PHlYllR END
PAmtR   PHlYllR PBetl END
</Int-NorGates>
```

Figure 5. The genetic gates library used by GeneTech. The gates are categorized bases on if the inputs to them are external, internal or semi-external.

References

[1] A. a K. Nielsen, B. S. Der, J. Shin, P. Vaidyanathan, V. Paralanov, E. a Strychalski, D. Ross, D. Densmore, and C. a. Voigt, "Genetic circuit design automation.," *Science*, vol. 352, no. 6281, p. aac7341, 2016.

[2] S. Kirkpatrick, C. Gelatt Jr, and M. Vecchi, "Optimization by Simulated Annealing," *Science (80-.).*, vol. 220, no. 4598, pp. 671–680, 1983.

[3] J. Y. Quinn, R. S. Cox, A. Adler, J. Beal, S. Bhatia, Y. Cai, J. Chen, K. Clancy, M. Galdzicki, N. J. Hillson, N. Le Nov??re, A. J. Maheshwari, J. A. McLaughlin, C. J. Myers, P. Umesh, M. Pocock, C. Rodriguez, L. Soldatova, G. B. V Stan, N. Swainston, A. Wipat, and H. M. Sauro, "SBOL Visual: A Graphical Language for Genetic Designs," *PLoS Biol.*, vol. 13, no. 12, 2015.