Using Bees Hill Flux Balance Analysis (BHFBA) for *in silico* Microbial Strain Optimization

Yee Wen Choon[1], Mohd Saberi Bin Mohamad[1], Safaai Deris[1],
Rosli Md. Illias[2], Lian En Chai[1], and Chuii Khim Chong[1]

[1] Artificial Intelligence and Bioinformatics Research Group,
Faculty of Computer Science and Information Systems,
Universiti Teknologi Malaysia,
Skudai, 81310 Johor, Malaysia
ywchoon2@live.utm.my,
{saberi,safaai}@utm.my,
{lechai2,ckchong2}@live.utm.my
[2] Department of Bioprocess Engineering,
Faculty of Chemical Engineering,
Universiti Teknologi Malaysia, Skudai,
81310 Johor, Malaysia
r-rosli@utm.my

Abstract. Microbial strains can be manipulated to improve product yield and improve growth characteristics. Optimization algorithms are developed to identify the effects of gene knockout on the results. However, this process is often faced the problem of being trapped in local minima and slow convergence due to repetitive iterations of algorithm. In this paper, we proposed Bees Hill Flux Balance Analysis (BHFBA) which is a hybrid of Bees Algorithm, Hill Climbing Algorithm and Flux Balance Analysis to solve the problems and improve the performance in predicting optimal sets of gene deletion for maximizing the growth rate and production yield of desired metabolite. *Escherichia coli* is the model organism in this paper. The list of knockout genes, growth rate and production yield after the deletion are the results from the experiments. BHFBA performed better in term of computational time, stability and production yield.

Keywords: Bees Algorithm, Hill Climbing, Flux Balance Analysis, Microbial Strains, Optimization.

1 Introduction

Microbial strains optimization has become popular in genome-scale metabolic networks reconstructions recently as microbial strains can be manipulated to improve product yield on desired metabolites and also improve growth characteristics [1]. Reconstructions of the metabolic networks are found to be very useful in health, environmental and energy issues [2]. The development of computational models for

A. Selamat et al. (Eds.): ACIIDS 2013, Part I, LNAI 7802, pp. 375–384, 2013.

simulating the actual processes inside the cell is growing rapidly due to vast numbers of high-throughput experimental data.

Many algorithms were developed in order to identify the gene knockout strategies for obtaining improved phenotypes. The first rational modeling framework (named OptKnock) for introducing gene knockout leading to the overproduction of a desired metabolite was developed by Burgard *et al.*, 2003 [3]. OptKnock identifies a set of gene (reaction) deletions to maximize the flux of a desired metabolite with the internal flux distribution is still operating such that growth is optimized.

OptKnock is implemented by using mixed integer linear programming (MILP) to formulate a bi-level linear optimization that is very promising to find the global optimal solution. OptGene is an extended approach of OptKnock which formulates the *in silico* design problem by using Genetic Algorithm (GA) [4]. Meta-heuristic methods are capable in producing near-optimal solutions with reasonable computation time, furthermore the objective function that can be optimized is flexible. SA is then implemented to allow the automatic finding of the best number of gene deletions for achieving a given productivity goal [5]. However, the results are not yet satisfactory.

A hybrid of BA and FBA was proposed by Choon *et al.*, 2012 [6], it showed a better performance in predicting optimal gene knockout strategies in term of growth rate and production yield. Pham *et al.*, 2006 [7] introduced Bees Algorithm (BA), is a typical meta-heuristic optimization approach which has been applied to various problems, such as controller formation [8], image analysis [9], and job multi-objective optimization [10]. BA is based on the intelligent behaviours of honeybees. It locates the most promising solutions, and selectively explores their neighbourhoods looking for the global maximum of the objective function. BA is efficient in solving optimization problems according to the previous studies [7, 10]. However, due to the dependency of BA on random search, it is relatively weak in local search activities [11]. Hence, BHFBA is proposed to improve the performance of BAFBA as Hill climbing algorithm is a promising algorithm in finding local optimum. This paper shows that BHFBA is not only capable in solving larger size problems in shorter computational time but also improves the performance in predicting optimal gene knockout strategy than previous works. In this work, we present the results obtained by BHFBA in two case studies where *Escherichia Coli (E.coli)* iJR904 model is the target microorganisms [12]. In addition, we also conduct a benchmarking to test performance of the hybrid of Bee algorithm and Hill climbing algorithm.

2 Bees-Hill Flux Balance Analysis (BHFBA)

In this paper, we proposed BHFBA in which BAFBA is only applied to identify optimal gene knockout strategies recently. Fig. 1 shows the flow of BAFBA while Fig. 2 shows our proposed BHFBA. The important steps are explained in the following subsections.

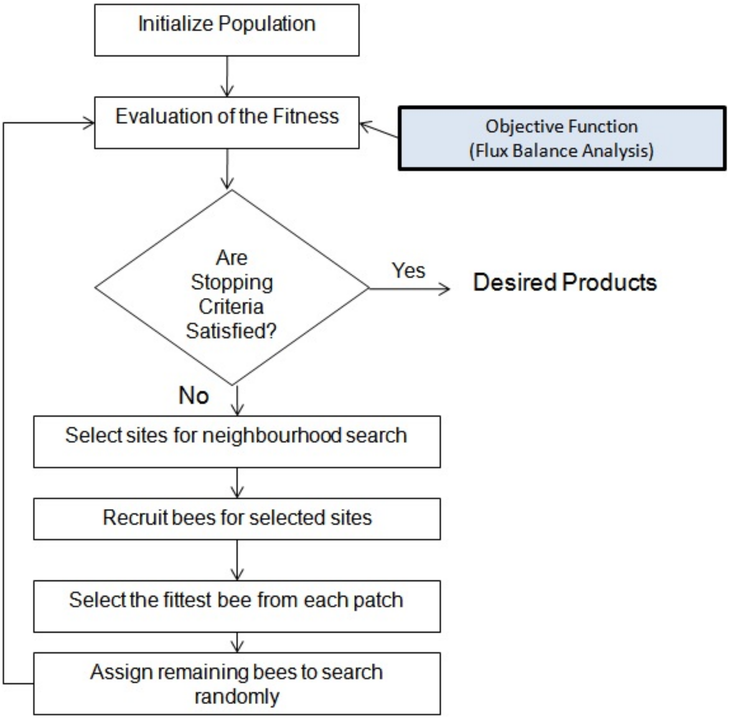

Initialize Population
Evaluation of the Fitness
Objective Function
(Flux Balance Analysis)
Are
Stopping
Criteria
Satisfied?
Yes
Desired Products
No
Select sites for neighbourhood search
Recruit bees for selected sites
Select the fittest bee from each patch
Assign remaining bees to search
randomly

Fig. 1. BAFBA Flowchart

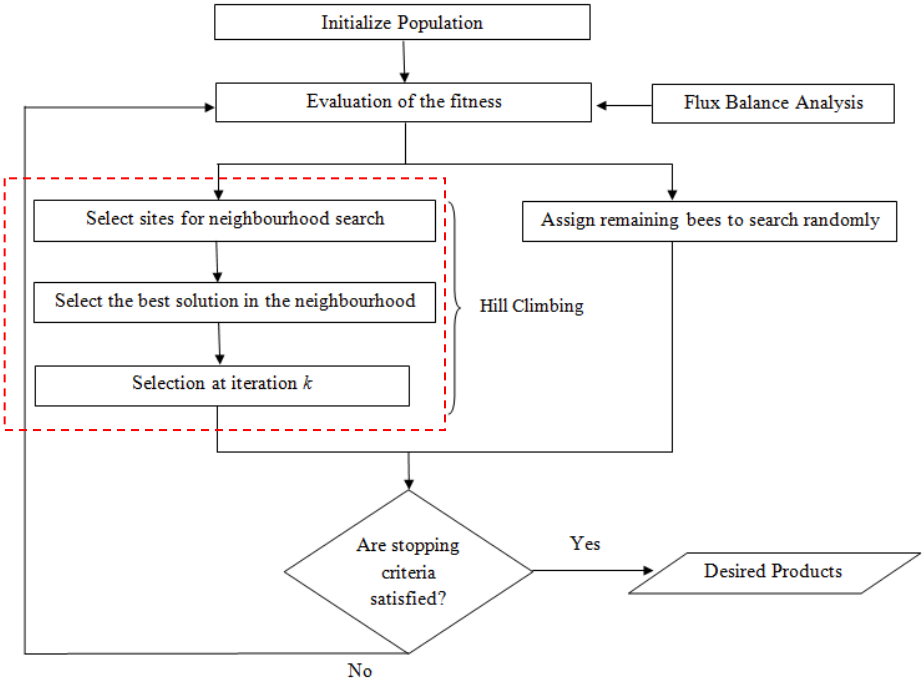
2.1 Model Pre-processing

The model is pre-processed through several steps based on biology assumptions as well as computational approaches to reduce the search space as while as increase the accuracy. Lethal reactions such as the genes that are found to be lethal *in vivo*, but not *in silico*, should be removed to improve the quality of the results. The results are invalid if a lethal reaction is deleted. The following are the details of computational pre-processing steps to the model [5].

a. Fluxes that are not associated with any genes, such as the fluxes related to external metabolites and exchange fluxes that represent transport reaction should not be involved in the process. These fluxes do not have a biological meaning thus they should not be knocked out.
b. Essential genes that cannot be deleted from the microorganism's genome need to be removed. The search space for optimization is reduced due to that these genes should not be considered as targets for deletion. A linear programming problem is defined by setting the corresponding flux to 0, while maximizing the biomass flux for each gene in the microorganism's genome. If the biomass flux result from the Linear Programming algorithm is zero (or near zero) then the gene is marked as essential. This biological meaning of this fact is that the microorganism is unable to

survive without this gene. This process does not suggest any changes to the model like the previous one, but provides favorable information for the optimization algorithms. With the help of biologists, the list of essential genes can be manually edited to include genes that are known to be essential *in vivo*, but not *in silico*.

c. Given the constraints of the linear programming problem, the fluxes need to be removed if the fluxes cannot exhibit values different from 0. Two linear programming are solved for every reaction in the model: the first is to define the flux over that reaction as the maximization target, while the second is to set the same variable as minimization target. If the objective function is 0 for both problems, then the variable is removed from the model.

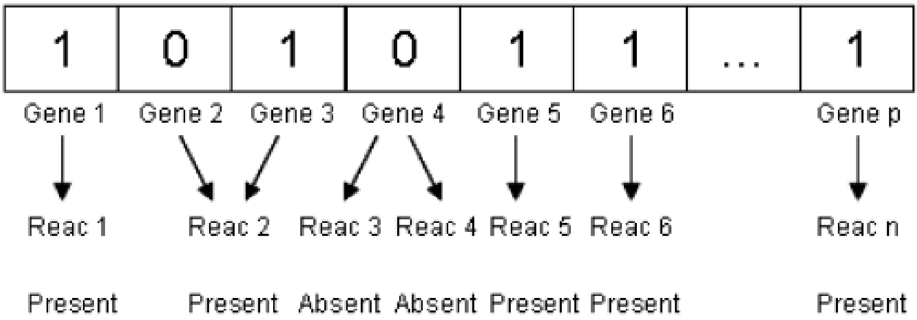

Initialize Population
Evaluation of the fitness
Flux Balance Analysis
Select sites for neighbourhood search
Assign remaining bees to search randomly
Select the best solution in the neighbourhood
Hill Climbing
Selection at iteration *k*
Are stopping criteria satisfied?
Yes
Desired Products
No

Note: Red-dotted box is Hill Climbing algorithm.

Fig. 2. BHFBA Flowchart

2.2 Bee Representation of Metabolic Genotype

One or more genes can be discovered in each reaction in a metabolic model. In this paper, each of those genes is represented by a binary variable indicating its absence or presence (0 or 1), these variables form a 'bee' representing a specific mutant that lacks some metabolic reactions when compared with the wild type (Fig. 3.)



Note: Reac represents reaction.

Fig. 3. Bee representation of metabolic genotype

2.3 Initialization of the Population

The algorithm starts with an initial population of n scout bees. Each bee is initialized as follows: assume that a reaction with n genes. Bees in the population are initialized by setting present or absent status to each gene randomly. Initialization of the population is done randomly so that all bees in the population have an equal chance of being selected. The result might not truly reflect the population if it is done with bias setting.

2.4 Scoring Fitness of Individuals

Each site is given a fitness score that determines whether to recruit more bees or should be abandoned. In this work, we used FBA to calculate the fitness score for each site and the equation is as follow:

Maximize Z, where

$$\mathrm{Z} = \sum c_i v_i = \mathbf{c.v} \tag{1}$$

where c = a vector that defines the weights for of each flux.

Cellular growth is defined as the objective function Z, vector *c* is used to select a linear combination of metabolic fluxes to include in the objective function, *v* is the flux map and *i* is the index variable (*1, 2, 3, ..., n*). After optimizing the cellular growth, mutant with growth rate more than 0.1 continues the process by minimizing and maximizing the desired product flux at fixed optimal cellular growth value. Hence, we can enhance yield of our desired products at fixed optimal cellular growth. Production yield is the maximum amount of product that can be generated per unit of substrate. The following shows the calculation for production yield:

$$\text{Production yield= (production rate}_{\text{production}}\text{)/(consumption rate}_{\text{substrate}}\text{)} \\ \text{(mmol/mmol)(gm/gm)} \tag{2}$$

where mmol = millimole and gm is gram.

We used Biomass-product coupled yield (BPCY) as the fitness score in this work, the calculation for BPCY is as follow:

$$\text{BPCY} = \text{product yield} * \text{growth rate } (\text{mmol}(\text{mmol}*\text{hr})^{-1})(\text{gm } (\text{gm} * \text{hr})^{-1}) \qquad (3)$$

where mmol is millimole, hr is hour and gm is gram.

2.5 Neighbourhood Search (Hill Climbing Algorithm)

The algorithm carries out neighbourhood searches in the favored sites (*m*) by using Hill climbing algorithm. Hill climbing is an iterative algorithm that starts with an arbitrary solution to a problem, then attempts to find a better solution by incrementally changing a single element of the solution. In this paper, the initial solution is the *m* favored sites from the population initialized by BA. The algorithm starts with the solution and makes small improvements to it by adding or reducing a bee to the sites. User defined the value of initial size of patches (ngh) and uses the value to update site (*m*) which is declared in the previous step to search in neighbourhood area. In this paper, *m* is equal to 15 and ngh is equal to 30, the values are obtained by conducting a small number of trials with the range of 10 to 25 and 20 to 35 respectively. This step is important as there might be better solutions than the original solution in the neighbourhood area.

2.6 Randomly Assigned and Termination

The remaining bees in the population are sent randomly around the search space to scout for new feasible solutions. This step is done randomly to avoid overlooking the potential results that are not in the range. These steps are repeated until either the maximum loop value is met or the fitness function has converged. At the end, the colony generates two parts to its new population – representatives from each selected patch and other scout bees assigned to perform random searches.

3 Results and Discussion

In this paper, *E. coli i*JR904 is used to test on the operation of BAFBA [12]. The model contains 904 genes, 931 unique biochemical reactions, and 761 metabolites. The model is pre-processed through several steps based on biology assumptions and computational approaches before BHFBA is applied. This resulted in the size of the model is reduced to 667 reactions. Lethal reactions such as the genes that are found to be lethal *in vivo*, but not *in silico*, are not included as the possible targets in BHFBA. The reason to remove lethal reaction is that the microorganism is unable to survive without this reaction. The *E.coli* simulations are performed for aerobic minimal media conditions. The glucose uptake rate are fixed to 10 mmol/gDW/hr while a set non-growth associated maintenance of 7.6 mmol ATP/gDW/hr. The experiments are carried out by using a 2.3 GHz Intel Core i7 processor and 8 GB DDR3 RAM computer.

Table 1 and Table 2 summarize the results obtained from BHFBA for succinic acid and lactic acid production from *E.coli*. As shown from the results, this method has produced better results to the previous works in term of growth rate and BPCY meanwhile potential genes which can be removed are identified [3][4][5].

Table 1. Comparison between different methods for production of Succinic acid in *E.coli*

Method	Growth Rate (1/hr)	BPCY	List of knockout genes
BHFBA	0.7988	0.93656	PTAr**, RPE, SUCD1i
BAFBA [10]	0.62404	0.66306	FUM, PTAr**, TPI**
SA + FBA [5]	N/A	0.39850	ACLD19*, DRPA, GLYCDx, F6PA, TPI**, LDH_D2, EDA, TKT2, LDH_D-
OptKnock [3]	0.28	N/A	ACKr, PTAr**, ACALD*

Note: The shaded column represents the best result. N/A – Not Applicable. * Common genes for all methods. ** Common genes in either 2 methods. BPCY is in gram (gram-glucose.hour)$^{-1}$.

Table 1 shows that BHFBA performed better than the previous works with growth rate 0.7988 and BPCY 0.93656. Knocking out succinate dehydrogenase (SUCD1i) interrupts the formation from succinic acid to fumarate. Without the conversion from succinic acid to fumarate, production yield of succinic is improved. Next, phosphotransacetylase (PTAr) is removed, according to Burgard *et al.*, 2003[3], the mutants can grow anaerobically on glucose by producing lactate. In the next step, ribulose-5-phosphate-3-epimerase (RPE) is suggested to knockout. This knockout involves the inflow reaction of ammonium. As stated in Bohl *et al.*, 2010 [13], the utilization of nitrate as electron acceptor and ammonium source under anaerobic conditions can improve succinate production.

Table 2 shows the results of BHFBA and previous works. BHFBA obtained a better BPCY that is 5.2241 than the previous work. However, the growth rate is slightly lower than BAFBA due to BHFBA finds a solution with higher BPCY with the condition that the growth rate is more than 0.1. The deletion of fructose bisphosphatase and phosphoglycerate kinase decreased the efficiency of gluconeogenesis which resulted in the increment concentration of phosphoenolpyruvate. Phosphoenolpyruvate is then converted into pyruvate and continues to convert into lactic acid. Knocking out acetaldehyde dehydrogenase which catalyze the conversion of acetaldehyde into acetic acid eliminated the competiting product, acetic acid. In consequence the yield of lactic acid is improved.

Table 3 shows the computational time comparison between BHFBA and BAFBA for 1000 iterations. The average computational time of BHFBA improved 69% of the BAFBA result for 1000 iterations.

Table 2. Comparison between different methods for production of Lactic acid in *E.coli*

Method	Growth Rate (1/hr)	BPCY	List of knockout genes
BHFBA	0.62501	5.2241	FBP, PGK, ACALD*
BAFBA [10]	0.86381	2.1677	ACALD*, ACKr**, GLUN, GND, ME1
SA + FBA [5]	N/A	0.39850	ACLD19*, DRPA, GLYCDx, F6PA, TPI, LDH_D2, EDA, TKT2, LDH_D-
OptKnock [3]	0.28	N/A	ACKr**, PTAr, ACALD*

Note: The shaded column represents the best result. N/A – Not Applicable. * Common genes for all methods. ** Common genes in either 2 methods. BPCY is in gram (gram-glucose.hour)$^{-1}$.

Table 3. Comparison between average computational time of BHFBA and BAFBA for 1000 iterations

Method	Computation Time (s)
BHFBA	3223
BAFBA	10253

In addition, since BA and Hill Climbing algorithm is a new hybrid algorithm. Hence, we conducted a benchmarking to test performance of a hybrid of BA and Hill Climbing algorithm (BH). As BA is looking for the maximum, the functions are inverted before the algorithm is applied. The De Jong and Martin & Gaddy functions are used in this paper. Table 4 shows the mathematical representation of the functions. Table 5 shows mean and standard deviation (STD) of the both functions, De Jong and Martin & Gaddy, tested on both original BA and BH. As seen from the results, both BHFBA and BH performed better than other algorithms. It can be concluded that the capability of Hill Climbing algorithm in finding local optimum improved the performance of the original BA. The original BA with the problem of repetitive iterations of the algorithm in local search where each bee keep searching until the best possible answer is reached. Our proposed BHFBA solved the problem by implementing Hill Climbing algorithm into the local search part. Hill Climbing algorithm is a powerful local search algorithm which attempts to find a better solution by incrementally changing a single element of the solution until no further improvements can be found, the search process is recorded so the process is not repeated. Furthermore, one of the advantages of Hill Climbing algorithm is it can return a valid solution even if it is interrupted at any time before it ends.

Table 4. Mathematical representation of De Jong and Beale functions

Name	Mathematical representation
De Jong	$\max F = (3905.93) - 100(x_1^2 - x_2)^2 - (1 - x_1)^2$
Martin & Gaddy	$\min F = (x_1 - x_2)^2 + ((x_1 + x_2 - 10) / 3)^2$

Table 5. Obtained fitness value of both De Jong and Beale functions

Function	Mean		STD	
	BA	BH	BA	BH
De Jong	3.91e+03	3.90e+03	0.000504	4.79e-13
Martin & Gaddy	11.1083	11.1111	0.002797	0

4 Conclusion and Future Works

In this paper, BHFBA is proposed to predict optimal sets of gene deletion to maximize the production of desired metabolite. Experimental results on *E. coli i*JR904 model obtained from literature showed that BHFBA is effective in generating optimal solutions to the gene knockout prediction, and is therefore a useful tool in Metabolic Engineering [12]. We are interested in applying other fitness functions in BHFBA such as minimization of metabolic adjustment (MOMA) and regulatory on/off minimization (ROOM) to further improve the performance of BHFBA. Besides that, BA employs many tunable parameters which are difficult for the users to determine so it is important to find ways to help the users choose suitable parameters.

Acknowledgement. Institutional Scholarship MyPhD provided by the Ministry of Higher Education of Malaysia finances this work. We also would like to thank Universiti Teknologi Malaysia for supporting this research by the UTM GUP research grants (Vot numbers: QJ130000.7107.01H29 and QJ130000.7123.00H67).

References

1. Feist, A.M., Herrgård, M.J., Thiele, I., Reed, J.L., Palsson, B.Ø.: Reconstruction of biochemical networks in microorganisms. Nat. Rev. Microbiol. 72, 129–143 (2009)
2. Chandran, D., Copeland, W.B., Sleight, S.C., Sauro, H.M.: Mathematical modeling and synthetic biology. Drug Discovery Today Disease Models 5(4), 299–309 (2008)
3. Burgard, A.P., Pharkya, P., Maranas, C.D.: OptKnock: A bilevel programming framework for identifying gene knockout strategies for microbial strains optimization. Biotechnol. Bioeng. 84, 647–657 (2003)
4. Patil, K.R., Rocha, I., Förster, J., Nielsen, J.: Evolutionary programming as a platform for in silico metabolic engineering. BMC Bioinformatics 6, 308 (2005)
5. Rocha, M., Maia, P., Mendes, R., Pinto, J.P., Ferreira, E.C., Nielsen, J., Patil, K.R., Rocha, I.: Natural computation meta-heuristics for the in silico optimization of microbial strains. BMC Bioinformatics 9, 499 (2008)
6. Pham, D.T., Ghanbarzadeh, A., Koç, E., Otri, S., Zaidi, M.: The bees algorithm – a novel tool for complex optimization problems. In: Proceedings of the Second International Virtual Conference on Intelligent Production Machines and Systems, July 3-14 (2006)
7. Choon, Y.W., Mohamad, M.S., Deris, S., Chong, C.K., Chai, L.E., Ibrahim, Z., Omatu, S.: Identifying Gene Knockout Strategies Using a Hybrid of Bees Algorithm and Flux Balance Analysis For in silico Optimization of Microbial Strains. In: The 9th International Symposium on Distributed Computing and Artificial Intelligence, DCAI 2012. University of Salamanca, Spain (2012)

8. Pham, D.T., Darwish, A.H., Eldukhri, E.E.: Optimisation of a fuzzy logic controller using the bees algorithm. International Journal of Computer Aided Engineering and Technology 1(2), 250–264 (2006)
9. Olague, G., Puente, C.: The Honeybee Search Algorithm for Three-Dimensional Reconstruction. In: Rothlauf, F., Branke, J., Cagnoni, S., Costa, E., Cotta, C., Drechsler, R., Lutton, E., Machado, P., Moore, J.H., Romero, J., Smith, G.D., Squillero, G., Takagi, H. (eds.) EvoWorkshops 2006. LNCS, vol. 3907, pp. 427–437. Springer, Heidelberg (2006)
10. Pham, D.T., Ghanbarzadeh, A.: Multi-objective optimisation using the bees algorithm. Paper Proceedings of the Third International Virtual Conference on Intelligent Production Machines and Systems, July 2-13 (2007)
11. Cheng, M.Y., Lien, L.C.: A Hybrid Swarm Intelligence Based Particle Bee Algorithm For Benchmark Functions And Construction Site Layout Optimization. In: Proceedings of the 28th ISARC, Seoul, pp. 898–904 (2011)
12. Reed, J.L., Vo, T.D., Schilling, C.H., Palsson, B.O.: An expanded genome-scale model of Escherichia coli K-12 (iJR904 GSM/GPR). Genome Biology 4, 54 (2003)
13. Bohl, K., Figueiredo, L.F.: d., Hadicke O., Klamt S., Kost C., Schuster S., Kaleta C: CASOP GS: Computing intervention strategies targeted at production improvement in genome-scale metabolic networks. In: The 5th German Conference on Bioinformatics 2010, September 20-22 (2010)