Maximilian Eibl, Martin Gaedke. (Hrsg.): INFORMATIK 2017,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2017 2389

Evolutionary algorithms for finding gen knockout targets

Paul Jähne[1]

Abstract: The organic chemical industry is primarily based on fossil raw materials. An alternative is the use of industrial biotechnology which is based on microorganisms that produce the needed substances as a part of their metabolism from renewable raw materials. Metabolic engineering is a technique to increase the production of the desired products through genetic modifications. The needed modifications are not trivial and no algorithm is currently known. Therefore this paper investigates the use of evolutionary algorithms for finding genetic modifications. The production of polyhydroxybutyrate out of methanol by *Methylobacterium extorquens AM1* is examined as an example. The proposed evolutionary algorithm is able to find a modification with 15 knockouts that increases the production of polyhydroxybutyrate by a factor of 294 at 0.6-fold biomass growth rate.

Keywords: evolutionary algorithm; flux balance analysis; metabolic engineering

1 Introduction

Many basic chemicals used in the organic chemical industry are obtained from fossil raw materials with limited availability [Ve16]. Furthermore there extraction and usage threatens the environment. An alternative to that is the usage of microorganisms which produce the desired substances as part of their metabolism. Such organisms are also called cell factories. They use cheap, renewable raw materials for their metabolism. Cell factories are already used at a low percentage. Their application is restricted by technical and economical factors because the microbes produce the desired substances usually as a side product. Therefore only a small amount of the raw materials is turned into the desired products.

One method to increase the yield of the desired products is *metabolic engineering* (ME). It deals with the targeted modification of the metabolism of an organism. The aim is to influence the production of certain products or to achieve desired cellular characteristics [SAN98, S. 2]. Thereby genes are deleted, regulated, or transferred from other organisms to achieve the modification.

The experimental implementation of such modifications is lengthy and the amount of possible modifications is huge because of the complexity of the genome. Therefore models are needed to estimate the effect and to avoid lengthy experimenting. One possibility is the usage of metabolic network models and methods like the flux balance analysis (FBA). They

[1] Hochschule für Technik, Wirtschaft und Kultur, Fakultät Informatik, Mathematik und Naturwissenschaften, Karl-Liebknecht-Straße 132, 04277 Leipzig, paul.jaehne@gmx.de

doi:10.18420/in2017_244

use few input data in comparison to other models and formulate the task as an optimisation problem. However the size of the search space prevents an exhaustive search and efficient algorithms specialised for this task are not known so far. Hence the use of techniques like meta heuristics is needed to generate promising solution candidates in reasonable time.

Certain conditions have to be met for a practical use of such changes [GL16]. This includes the yield which describes how much raw material is turned into the desired products. This should be as high as possible to utilise the raw materials for the desired purpose. The organism should also still be viable and able to grow. In addition the productivity is also important. It describes the speed at which products are build. This should also be as high as possible to produce a lot of the desired products in a short time.

Examples for the application are [Ja08] and [Zh09]. Therein it is described how genetic modifications in *Escherichia coli* improve the production of succinate out of glucose. Succinate is an important basic chemical which is mainly produced out of petroleum based chemicals. Further examples can be found in [CBN11] and [Fo14].

This paper aims to examine the use of evolutionary algorithms to search for knockout targets to increase the production of desired products. For this purpose the *Methylobacterium extorquens AM1* is examined as an example. This organism uses methyl compounds as the primary source for it is metabolism and produces polyhydroxybutyrate (PHB) as a storage substance under certain environmental conditions. This substance can be used as a biodegradable plastic. Therefore the organism should be used to produce biopolymers out of methanol.

2 Theoretical principles

This section describes the terms and concepts which are needed to understand the paper. These are the modelling of the metabolism with the FBA and evolutionary algorithms (EAs).

2.1 Flux balance analysis

The practical process and verification of genetic changes is time consuming. Because of this models are needed to approximate the possible outcome of certain changes. One such model is the flux balance analysis. It describes the metabolism of the organism as a linear programming (LP) problem. The model needs the stoichiometry of the substances involved in the metabolism. These substances are called metabolites. The metabolism can be seen as a network consisting of reactions and metabolites.

The presentation as a metabolic network is not suitable for the FBA. Instead the stoichiometric matrix $\mathbf{S}$ represents the reactions and metabolites. An example is shown in equation 1. Thereby each column represents one reaction and each row one metabolite. For example the

reaction $R_{biomass}$ needs one time A, two times C, and two times D to produce one $biomass$. This results in the stoichiometric equation $A + 2C + 2D \rightarrow biomass$. The metabolites on the left side are taken up to produce the metabolites on the right side. Therefore the metabolites on the left side have a negative sign and on the right a positive sign.

$$\mathbf{S} = \begin{array}{c} \\ A \\ B \\ C \\ D \\ biomass \end{array} \begin{array}{c} \begin{array}{cccccccccc} R_1 & R_2 & R_3 & R_4 & R_5 & R_6 & R_7 & R_8 & R_9 & R_{biomass} \end{array} \\ \begin{pmatrix} 1 & -2 & 0 & -2 & 0 & 0 & 0 & 0 & 0 & -1 \\ 0 & 1 & -1 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 & -1 & 1 & 0 & 0 & 0 & -2 \\ 0 & 0 & 1 & 0 & 1 & -1 & 1 & 0 & 0 & -2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 \end{pmatrix} \end{array} \tag{1}$$

The model assumes that the reactions are in a balanced state. This means that production and uptake for each metabolite is equal. Another assumption is that the biomass reaction and therefore the growth is to be maximised. With these two assumptions it is possible to calculate the reaction rates. They are called fluxes and are denoted by the vector $\mathbf{v}$. The values in the vector $\mathbf{c}$ are set to zero except for the position of the biomass flux which is one as this is the optimisation target. Additionally, each reaction rate has an upper and lower bound, $\mathbf{u}$ and $\mathbf{l}$ respectively, which are in place because of reaction speeds and transport processes. This results in a LP problem like in equation 2. The results are the reaction rates $\mathbf{v}$ for the belonging reactions which allow for the biggest growth.

$$\max \{\mathbf{c} \cdot \mathbf{v} \,|\, \mathbf{S} \cdot \mathbf{v} = \mathbf{0}, \mathbf{l} \leq \mathbf{v} \leq \mathbf{u}\} \tag{2}$$

The FBA compared to other models has the advantage of making quantitative predictions with few and comperatively easily obtainable data [LNP12]. Nevertheless there are unrecognised factors like gen regulation which influences the activity of genes and therefore the production of their products. Hence the accuracy of the predictions is reduced.

The deletion of reactions is called knockout. They are being integrated in the FBA by setting the upper and lower bounds for the reaction rates to 0. This is the only genetic modification which is used in this paper.

2.2 Evolutionary algorithm

EAs are powerful, domain independent search methods which are inspired by the evolutionary theory. They create new solution candidates by selection, combination and random changes of existing solution candidates. The above named operators are based on stochastic methods which is why there is no guarantee to find the optimal solution, but good solutions are found in general. This type of algorithm is called metaheuristic. They are useful if no exact

solution algorithm is known or if an exact solution would take to much time and when approximate solutions are sufficient.

The course of a general EA is as follows. An initial population is created at the start. It consists of an amount of solution candidates which are called individuals. Afterwards each individual is evaluated with the fitness function. Then a new generation is created by the application of the operators selection, crossover, and mutation on the population of the previous generation. Finally, the newly created generation is evaluated with the fitness function. New generations are created until a termination criteria is reached. Based on this principal different subtypes of EA evolved. These differ in the representation of the individuals and operators used.

3 Related work

Patil et al. present in [Pa05] an evolutionary algorithm to search for knockout targets. For this purpose they use a binary and real encoding of the solution candidates. The binary encoding assigns each reaction to one bit which encodes the presence of this reaction. So the individual is a list of as much bits as there are reactions. An individual in real encoding is a list of knockouts. The amount of knockouts is thereby determined by the size of the individual. The fitness is evaluated with the *biomass product coupled yield* (BPCY) with values from the FBA. This value describes the quality of a modification and consists of the product of biomass growth rate and the yield. Both are important factors for a successful practical implementation as discussed in section 1. As operators in the EA they use roulette wheel selection, one-point, two-point, and uniform crossover. The proposed algorithm is tested against three examples for which good solution candidates are found within 1000 generations. The solution quality did not increase when more iterations like 5000 were allowed. The fitness develops in steps which is explained by the possibility that the solution space might be discrete. For the parameters of the EA it is mentioned that increasing the population size beyond 125 individuals does not improve the solution quality significantly. The different crossover strategies and encoding schemes resulted in similar solution quality.

Rocha et al. compare evolutionary algorithm and simulated annealing for the search for knockout targets in [Ro08]. Simulated annealing is also a metaheuristic. It generates new solution candidates out of previous ones by a kind of mutation. Better solutions are always accepted and worse with a certain probability. The probability decreases in each iteration which should resemble cooling down of a system, hence the name. Both algorithms use the set-based, real encoding. There the solution candidates consist of a variable sized list of knockouts. The solution candidates are initialised with with ten random knockouts. The operators for the EA are roulette wheel selection, uniform crossover, and single-point mutation. Additionally, the operators grow and shrink are added to change the size of the individuals. One half of a new generation is created by selection and the other half by the other operators. The solution candidates are evaluated with the BPCY with values from the FBA. Both algorithms are tested on four examples. It is shown that simulated

annealing reaches better solutions faster for a given number of solution candidates and the best solutions are slightly better than the ones found by the EA. The EA is additionally tested without the crossover. The results are similar indicating that the crossover is of no use in this case.

4 Implemented algorithm

This section describes the implemented algorithm which consists of the evolutionary algorithm and postprocessing. At first the mapping of the problem to the EA is explained, then the implementation of the operators, and finally the postprocessing.

4.1 Mapping to the evolutionary algorithm

The implemented EA uses the island model. It separates the whole population in independent subpopulations. The only interaction between these is the migration. It swaps individuals in set intervals between the islands which is described in greater detail in section 4.2. Through the separation it is possible that subpopulations examine different parts of the search space. Thereby the probability of the algorithm getting stuck at a local optimum might decrease.

The individuals encode the reactions in the metabolic network. Each reaction is represented by one bit. They correspond to the columns in the coefficient matrix of the LP problem. A 1 shows the presence of a reaction and a 0 it is absence. The i-th bit encodes the i-th reaction. Thus the individuals consist of a list of bits. They are represented as arrays of 32-bit unsigned integers in the algorithm. The encoding is shown in figure 1.

variable	1					2					...
bit	0	1	2	...	31	0	1	2	...	31	
value	1	1	1	...	1	1	0	1	...	1	
reaction	1	2	3	...	32	33	34	35	...	64	

Fig. 1: Encoding of the problem into individuals of the EA.

To evaluate the individuals they have to be decoded into a coefficient matrix of a LP problem. One possible implementation would be to do a matrix-matrix multiplication with the original coefficient matrix and the list of bits of the individual as a row matrix. But this wastes a lot of computational resources. So the matrix would be in large parts a copy of the original one. Therefore the original matrix is copied and the values in the columns corresponding to a 0 in the individual are set to 0.

4.2 Evolutionary algorithm operators

The initial population is not created with completely random values, as this would mean that on average half the reactions would be set to zero and therefore absent. This would result in

large parts of the metabolism being disabled and likely breaking it completely. This would lead to a fitness value of 0 and thus no search direction would exist. This consideration is also supported by publications on metabolic modifications. They have at most a low two-digit number of modifications (see section 1). An alternative is to start with all reactions as present and mutating each individual. This creates diversity in the population without destroying the metabolism.

The tournament selection is used as the selection operator. It randomly chooses an amount of individuals given by the tournament size, where multiple selection is allowed. The individual with the best fitness of the selected individuals is taken to the temporary population for the next generation. So it is possible that worse individuals get into the next generation and therefore the diversity is bigger compared to a best selection. There are as many tournaments as set by the selection rate.

The uniform crossover is used as the crossover operator. The bits are randomly chosen from the parents based on a random bitmask. The parents are randomly chosen from the previous generation and multiple selection is possible. There are as many crossover operations as needed to fill up the population with new individuals to it is original size after the selection.

The one-point mutation is used as mutation operator. There randomly chosen bits are inverted. The amount of inversions is given by the mutation strength. The mutation rate determines the amount of individuals to mutate. They are randomly chosen from the temporary population and multiple selection is possible.

The migration operator swaps individuals between the island. The amount is determined by the migration size. The swap is done in a ring topology. For this the first individuals are chosen which fitness are above the average of the island. They then replace the first individuals of the neighbouring island which are below the average fitness of this island. The migration interval sets the amount of iterations after that the migration is performed.

The product of the biomass growth rate and the yield (BPCY) is used for the fitness evaluation. These are two important factors for a successful practical application [GL16]. The cells have to be able to grow and also produce a high yield. One is not possible without the other and a smaller value in one factor can be compensated by increase in the other factor. Therefore this value is a indicator for practical use. The BPCY is given as $\mathrm{mmol}_{product}$ $\mathrm{mmol}^{-1}_{substrate}$ h^{-1}. The *GNU Linear Programming Kit* (GLPK) library is used to solve the LP problem to determine the values for the fitness calculation.

4.3 Postprocessing

The proposed solutions of the EA have a large number of knockouts. This would lead to a big effort for implementing these changes. But the number of knockouts can be reduced because not all have influence on the result. Because of this an additional program is created which searches the knockouts for superfluous elements. For this it is checked if removing a

knockout changes the result. If so, the knockout is necessary and stays in the list. Otherwise, the knockout is not needed and removed from the list. The list of knockouts is run through repeatedly until no further change is possible. The so created reduced list of knockouts is not unique. Depending on the way the list is traversed there can be multiple different reductions.

5 Experiments

This section describes the results of the parameter study. For this purpose the model of the used organism is described. Then the environment for the parameter study is introduced. The most notable results are explained and then combined into a parameter set. This set is afterwards tested and the best knockout is given.

5.1 Test example

Methylobacterium extorquens AM1 is examined as an example. This organism mainly takes up methanol and produces polyhydroxybutyrate as a storage substance under lack of nitrogen. Methanol is available as a cheap raw material and PHB is a biodegradable plastic. Therefore this process is interesting for various applications. The stoichiometric matrix used is from the model by Peyraud et al. in [Pe11]. It consists of 1075 rows, 1142 columns, and 4574 non-zero elements. The FBA results in a biomass growth rate of $2.13 \cdot 10^{-1}$ h^{-1}, $5.91 \cdot 10^{-2}$ $mmol_{PHB}$ $g^{-1}_{dryweight}$ h^{-1} for the production rate of PHB, and $1.50 \cdot 10^{1}$ $mmol_{methanol}$ $g^{-1}_{dryweight}$ h^{-1} for the uptake of methanol. This results in a BPCY of $8.39 \cdot 10^{-4}$ $mmol_{PHB}$ $mmol^{-1}_{methanol}$ h^{-1}.

5.2 Parameter study

The effect of the parameters is analysed below. For this the individual parameters are varied based on a common set of parameters. The common values used are listed in table 1. All given values are calculated as the average of the best fitness values of each iteration and from 100 runs. Only the most significant parameters are presented below.

parameter	value	parameter	value	parameter	value
island amount	10	migration size	5	population size	100
iteration amount	1000	mutation rate	0.3	selection rate	0.3
migration interval	10	mutation strength	1	tournament size	5

Tab. 1: Common values for the parameter study.

To draw conclusions from the values it does not suffice to compare the position of the graphs because they only show the average but nothing about the scatter. Therefore a statistical analysis of the values is needed. For this purpose the independent two-sample t-test is used.

A value of 0.05 is used as the significance level. The probability calculated by the t-test is given behind the statements. Except when otherwise noted these values refer to the end result after the last iteration.

Island amount vs. population size To find out whether more islands or bigger islands are advantageous the population size and island amount are varied at a constant amount of 1000 individuals. For this 1, 2, 4, 8, 10, 20, 40, 50, and 100 islands and the corresponding population size are tested. The results are shown in figure 2. Few large islands find in the beginning better solutions. After 400 iterations the smaller islands achieve better values. The best values are obtained with 40 islands and with a population size of 25. But there is no statistically relevant difference up to 100 islands (0,250), this is the case starting from 20 islands (0,003). The single island stagnates after the steep increase at the beginning. Multiple islands are less prone to getting stuck at a local optimum and increase the solution over a longer period. The development of fitness values shows that the island model improves the solutions.

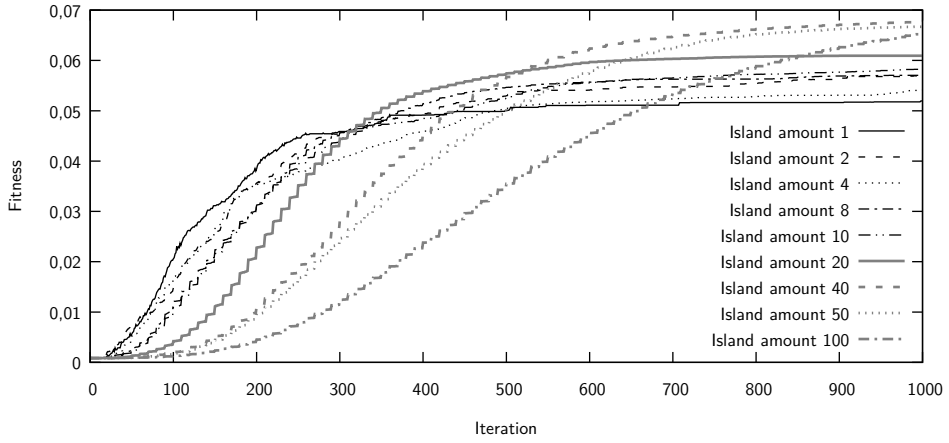

Fig. 2: Impact of island amount and population size on the fitness development. The population size is the result of $1000/islandamount$.

Mutation rate The effect of different mutation rate is shown in figure 3. For this values between 0.1 and 1.0 are tested in steps of 0.1. Higher mutation rates lead to higher fitness values over the whole course. The best fitness value is reached at a mutation rate of 1.0. This value is statistically not distinguishable from 0.9 (0.385). A statistically significant difference can be seen at a mutation rate of 0.6 (0.043). The best fitness values overall were reached during this experiment. Therefore this parameter is particularly important to improve the solution quality.

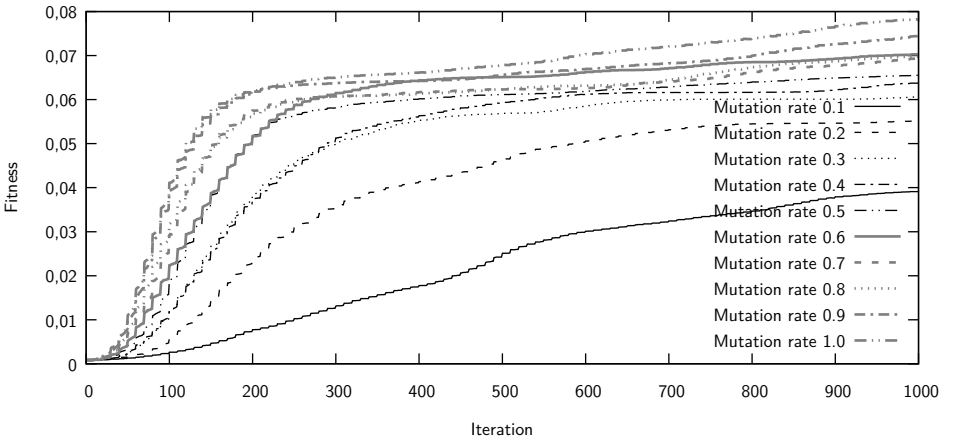

Fig. 3: Impact of the mutation rate on fitness development.

5.3 Combined parameter set

From the results of the parameter study a combined parameter set is created. This is put together from the best values of the individual parameters, except for population size and island amount. These values are taken from the parameter study for island amount vs. populations size, so that the amount of individuals is identical to the parameter study to achieve comparability. The parameter set is listed in figure 2.

parameter	value	parameter	value	parameter	value
island amount	40	migration size	10	population size	25
iteration amount	1000	mutation rate	1.0	selection rate	0.2
migration interval	10	mutation strength	1	tournament size	4

Tab. 2: Combined parameter set.

With these parameters there were also 100 runs conducted. The average development of fitness can be seen in figure 4. Additionally, the standard deviation is given. The combined parameter set finds better results than the individual experiments of the parameter study, for example when compared to the mutation rate of 1.0 (0.034). The fitness development flattens after 300 iterations. The distribution of the values increases during the course of the iterations.

The parameter study did not consider the crossover because it is coupled with the selection rate. There are as many individuals created by crossover as there are needed to fill up the population to its original size before the selection. Because of this the crossover is looked at separately. To check whether the crossover influences the fitness development it is removed entirely. The mutation is then used to generate new individuals to fill up the population. Therefore the mutation rate is now coupled with the selection rate. The fitness development

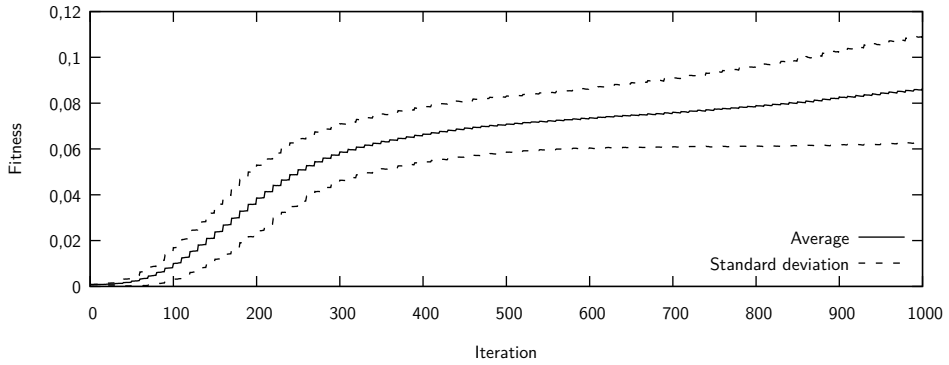

Fig. 4: Average of the fitness development and corresponding deviation with the combined parameter set.

of the algorithm without crossover compared to the previous one is shown in figure 5. At the beginning the algorithm without crossover reaches better solutions faster. The fitness development flattens after 200 iterations and the algorithm with crossover reaches better solutions at about 350 iterations. Also the standard deviation of the values is higher without recombination. The difference in solution quality is also statistically relevant (0.014). The crossover therefore contributes to better solutions.

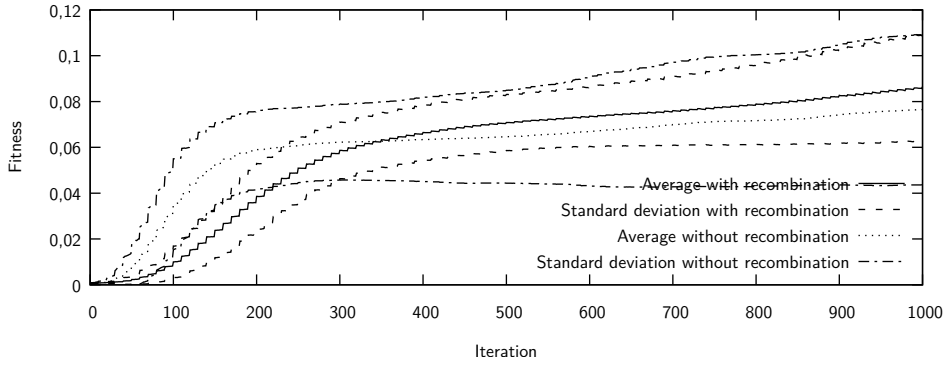

Fig. 5: Average of the fitness development and corresponding deviation with the combined parameter set with and without recombination.

The BPCY of the wild type for PHB production is $8.39 \cdot 10^{-4}$ mmol_{PHB} $\text{mmol}^{-1}_{methanol}$ h^{-1} and is composed of a biomass growth rate of $2.13 \cdot 10^{-1}$ h^{-1}, a production rate for PHB of $5.91 \cdot 10^{-2}$ mmol_{PHB} $\text{g}^{-1}_{dryweight}$ h^{-1}, and an uptake rate for methanol of $1.50 \cdot 10^{1}$ $\text{mmol}_{methanol}$ $\text{g}^{-1}_{dryweight}$ h^{-1}. The run with the highest BPCY achieved $1.48 \cdot 10^{-1}$ mmol_{PHB} $\text{mmol}^{-1}_{methanol}$ h^{-1}. This consists of a biomass growth rate of $1.28 \cdot 10^{-1}$ h^{-1}, a production rate for PHB of $1.74 \cdot 10^{1}$ mmol_{PHB} $\text{g}^{-1}_{dryweight}$ h^{-1}, and an uptake rate for

methanol of $1.50 \cdot 10^1$ $\text{mmol}_{methanol}$ $\text{g}^{-1}_{dryweight}$ h^{-1}. Thus the production rate of PHB increased nearly 300-fold which is a substantial improvement. The growth rate is above half of the wild type and the organism is therefore still capable of surviving. The values are also shown in table 3 for easier comparison.

property	wild type	best modification	factor
biomass growth rate [h^{-1}]	$2.13 \cdot 10^{-1}$	$1.28 \cdot 10^{-1}$	0.60
PHB production [mmol_{PHB} $\text{g}^{-1}_{dryweight}$ h^{-1}]	$5.91 \cdot 10^{-2}$	$1.74 \cdot 10^1$	294
methanol uptake [$\text{mmol}_{methanol}$ $\text{g}^{-1}_{dryweight}$ h^{-1}]	$1.50 \cdot 10^1$	$1.50 \cdot 10^1$	1.00
BPCY [mmol_{PHB} $\text{mmol}^{-1}_{methanol}$ h^{-1}]	$8.39 \cdot 10^{-4}$	$1.48 \cdot 10^{-1}$	176

Tab. 3: Comparison of the wild type and the run with the highest BPCY.

The corresponding knockouts consist of 221 reactions which would lead to a big effort for a practical implementation. Because of this the list of knockouts is reduced as shown in section 4.3. The knockouts are thereby reduced to 15 essential ones. These are with their names from the *Systems Biology Markup Language* (SBML) file of the appendix to [Pe11]: R-0005, R-0019, R-0031, R-0044, R-0069, R-0202, R-0252, R-0256, R-0347, R-0351, R-0412, R-0425, R-0879, R-1009, R-1026. SBML is a common exchange format for models of biological processes. The influence of a single reaction can not be specified as these only achieve the results together.

6 Summary

This paper describes an implementation of an EA for the search of knockout targets with FBA. It describes the theory behind it. Afterwards a overview of related work is shown. Then a description of the implemented algorithm is given. With this implementation a parameter study is conducted with *Methylobacterium extorquens AM1* as an example. Based on this a parameter set is created and knockouts are proposed.

The implemented EA uses the island model which divides the population in sub populations. This improves the results whereby a larger island amount achieves better results. The most influential parameter is the mutation rate whereby a higher mutation rate leads to better results. The crossover also improves the solution candidates.

After parametrisation the EA finds promising solution candidates for the production of PHB. The amount of knockouts resulting from the EA is large, but can be reduced through postprocessing. For example the best solution candidate has 221 knockouts which are reduced to 15 essential once via postpreocessing. This knockout suggestion increases the BPCY from the wild type with $8.39 \cdot 10^{-4}$ mmol_{PHB} $\text{mmol}^{-1}_{methanol}$ $\text{h}^{-1}$to $1.48 \cdot 10^{-1}$ mmol_{PHB} $\text{mmol}^{-1}_{methanol}$ h^{-1}. This correspond to an increased production rate of PHB by the factor 294 at 0.6 fold biomass growth rate.

The proposed knockout strategy could not be verified experimentally within this paper. It would therefore be interesting if it can be implemented practically and whether the predicted values from the FBA are achieved in practice. Additionally, the transferability of the parameter set to other organisms should be examined.

References

[CBN11] Cvijovic, Marija; Bordel, Sergio; Nielsen, Jens: Mathematical models of cell factories: moving towards the core of industrial biotechnology. Microbial Biotechnology, 4(5):572–584, 2011.

[Fo14] Fong, Stephen S.: Computational approaches to metabolic engineering utilizing systems biology and synthetic biology. Computational and Structural Biotechnology Journal, 11(18):28–34, 2014.

[GL16] Gustavsson, Martin; Lee, Sang Yup: Prospects of microbial cell factories developed through systems metabolic engineering. Microbial Biotechnology, pp. n/a–n/a, 2016.

[Ja08] Jantama, Kaemwich; Zhang, Xueli; Moore, Jonathan C.; Shanmugam, Keelnatham T.; Svoronos, S. A.; Ingram, Lonnie O.: Eliminating side products and increasing succinate yields in engineered strains of Escherichia coli C. Biotechnology and Bioengineering, 101(5):881–893, 2008.

[LNP12] Lewis, Nathan E.; Nagarajan, Harish; Palsson, Bernhard O.: Constraining the metabolic genotype–phenotype relationship using a phylogeny of in silico methods. Nature Reviews Microbiology, 10(4):291–305, 2012.

[Pa05] Patil, Kiran Raosaheb; Rocha, Isabel; Förster, Jochen; Nielsen, Jens: Evolutionary programming as a platform for in silico metabolic engineering. BMC Bioinformatics, 6(1):1–12, 2005.

[Pe11] Peyraud, Rémi; Schneider, Kathrin; Kiefer, Patrick; Massou, Stéphane; Vorholt, Julia A.; Portais, Jean-Charles: Genome-scale reconstruction and system level investigation of the metabolic network of Methylobacterium extorquensAM1. BMC Systems Biology, 5(1):1–22, 2011.

[Ro08] Rocha, Miguel; Maia, Paulo; Mendes, Rui; Pinto, José P.; Ferreira, Eugénio C.; Nielsen, Jens; Patil, Kiran Raosaheb; Rocha, Isabel: Natural computation meta-heuristics for the in silico optimization of microbial strains. BMC Bioinformatics, 9(1):1–16, 2008.

[SAN98] Stephanopoulos, George; Aristidou, Aristos A.; Nielsen, Jens: Metabolic Engineering: Principles and Methodologies. Elsevier Science, 1998.

[Ve16] Verband der Chemischen Industrie e.V.: , Rohstoffbasis der chemischen Industrie - Daten und Fakten. https://www.vci.de/vci/downloads-vci/top-thema/daten-fakten-rohstoffbasis-der-chemischen-industrie-de.pdf, July 2016.

[Zh09] Zhang, Xueli; Jantama, Kaemwich; Moore, Jonathan C.; Jarboe, Laura R.; Shanmugam, Keelnatham T.; Ingram, Lonnie O.: Metabolic evolution of energy-conserving pathways for succinate production in Escherichia coli. Proceedings of the National Academy of Sciences, 106(48):20180–20185, 2009.