Proceedings of the 2007 IEEE Symposium on Computational
Intelligence in Bioinformatics and Computational Biology (CIBCB 2007)

Optimization of Bacterial Strains with Variable-Sized Evolutionary Algorithms

Miguel Rocha, José P. Pinto
Department of Informatics/ CCTC
Universidade do Minho, Braga, Portugal
Email:mrocha@di.uminho.pt josepedr@gmail.com.

Isabel Rocha and Eugénio Ferreira
IBB-Institute for Biotechnology and Bioengineering
Centre for Biological Engineering
Universidade do Minho, Braga, Portugal
Email: irocha,ecferreira@deb.uminho.pt.

Abstract*—In metabolic engineering it is difficult to identify which set of genetic manipulations will result in a microbial strain that achieves a desired production goal, due to the complexity of the metabolic and regulatory cellular networks and to the lack of appropriate modeling and optimization tools. In this work, *Evolutionary Algorithms (EAs)* are proposed for the optimization of the set of gene deletions to apply to a microorganism, in order to maximize a given objective function. Each mutant strain is evaluated by resorting to the simulation of its phenotype using the *Flux-Balance Analysis* approach, together with the premise that microorganisms have maximized their growth along natural evolution. A new set based representation is used in the *EAs*, using variable size chromosomes, allowing for the automatic discovery of the optimal number of gene deletions. This approach was compared with a traditional binary-based *Genetic Algorithm*. Two case studies are presented considering the production of succinic and lactic acid as the target, with the bacterium *E. coli*. The variable size *EAs*, outperformed the other approaches tested, allowing to reach good results regarding the production of the desired compounds, and additionally presenting low variability among the several runs.

Keywords: Evolutionary Algorithms, Set based representations, Variable size chromosomes, Metabolic engineering, Flux-balance analysis

I. Introduction

Nowadays, there is a growing trend to replace the traditional methods of chemical synthesis by biotechnological processes, in order to produce a number of valuable products, such as pharmaceuticals, fuels or food ingredients. This, however, implies that the microorganisms' metabolism usually needs to be modified to comply to industrial purposes, rather then to follow their natural aims like, for example, the maximization of growth.

In the later years, within the field of Metabolic Engineering, a number of tools have been developed in order to introduce genetic modifications capable of achieving the production of the desired products [15][10]. However, these have still been based mostly on qualitative or intuitive design principles and scarcely on effective mathematical models that can accurately predict cellular behaviour.

A number of attempts have been made to model the whole cell behavior [17], but these models are still incomplete due to the lack of kinetic and regulatory information. Nevertheless, it is possible to predict cellular metabolism, under some assumptions, namely considering steady-state conditions and therefore imposing a number of constraints over the fluxes of all reactions.

This is the way followed by the *Flux Balance Analysis* approach [8], where a particular flux is typically optimized using linear programming, resulting in a value for the fluxes of all reactions in the cell. The most usual application, under this framework, is to define a flux for biomass production and to consider this as the objective function, thus assuming that the microbes have evolved towards optimal growth [7].

Using this technique it is possible to predict the behavior of a microorganism, both in its wild type and mutant forms, under a number of environmental conditions. A bi-level optimization problem can then be formulated, by adding a layer that searches for the best mutant that can be obtained by simply deleting a few genes from the wild type. The idea is to force the microorganisms to produce the desired product by selected gene deletions. Therefore, the underlying optimization problem consists in reaching an optimal subset of gene deletions to optimize an objective function related with the production of a given compound (e.g. yield or productivity).

A first approach to tackle this problem was proposed by the *OptKnock* algorithm [2], where mixed integer linear programming (MILP) is used to reach an optimum solution. This algorithm suffered from two important drawbacks: the impossibility of considering nonlinear objective functions that are of interest in industrial applications and the considerable computation time required for real problems given the large solution spaces to be considered.

An alternative approach was proposed by the *OptGene* algorithm [11] that considers the application of *Evolutionary Algorithms (EAs)* in this scenario. Since *EAs* are a meta-heuristic optimization method, they are capable of providing solutions in a reasonable amount of time, although this solution might not be the optimal one. Furthermore, its application in the context of the yeast *Saccharomyces cerevisiae* allowed for the optimization of a nonlinear objective function in several processes such as the production of succinic acid or vanillin.

OptGene proposes *EAs* with two alternative representation schemes: binary or integer. The binary representation is closer to the natural evolution of microbial genomes but is more complex and leads to solutions with a large number of knockouts. The integer representation allowed for a more compact genome in the *EA*, encoding only for the gene deletions. However, one of the major limitations of this representation in *OptGene* is the need to define *a priori* the number of gene knockouts, that remains fixed throughout the *EA*'s evolution.

In this work, an *EA* with a set-based representation is proposed in order to extend the previous work. This representation considers two variants, that allow the use of fixed and variable-sized chromosomes to encode sets of genes. The latter allows for the competition of solutions with a different number of knockouts (gene deletions) within the *EAs* population.

The performance of these algorithms was measured by considering its application in two case studies, where the bacterium *Escherichia coli* is the target microorganism. In these cases, the objective function is related to the production of succinic and lactic acid. In the experiments, the proposed set based *EA*s (both with fixed and variable size chromosomes) was compared to a traditional *Genetic Algorithm (GA)* with binary representations.

The paper starts by explaining the algorithms used for the simulation of the metabolic behavior of the microorganisms; then, the *EAs* used for the gene deletion optimization are presented; the description and the results obtained in the case studies follow; the paper closes with the conclusions and further work.

II. Simulation algorithms for the prediction metabolic behavior

Advanced automation techniques in genome sequencing protocols have allowed the number of fully sequenced organisms to increase rapidly in the last few years. One of the many potential applications of the sequenced and annotated genomes of microorganisms is the reconstruction of genome-scale metabolic networks, by combining genomic sequence data with biochemical knowledge. The set of metabolic reactions obtained can therefore be used to simulate the phenotypic behaviour of microorganisms.

One approach may be to write dynamic mass balances for each metabolite in the network, generating a set of ordinary differential equations that may be used to simulate the dynamic behavior of metabolite concentrations. However, there is still insufficient data on kinetic expressions and parameters, and it is therefore only possible to simulate dynamic conditions for a few pathways[4].

A steady state approximation is generally applied, and consequently, for each metabolite in a metabolic network, the sum of all productions and consumptions will be zero, weighted by the stoichiometric coefficients. Thus, for metabolite i, where $i = 1, \dots, M$ (M is the number of metabolites in the network) the following constraint is defined:

$$\sum_{j=1}^{N} S_{ij} v_j = 0 \qquad (1)$$

where S_{ij} is the stoichiometric coefficient for metabolite i in reaction j and v_j is the reaction rate or flux over the reaction j. It is then possible to define matrix S, composed of the S_{ij} values, where $i = 1, \dots, M$ and $j = 1, \dots, N$ (N is the number of reactions). v is defined as the N-dimensional vector of the fluxes of the reactions.

The mass balances are therefore reduced to a set of linear homogeneous equations. The maximum/ minimum values of the fluxes can be set by additional constraints in the form $\alpha_j \leq v_j \leq \beta_j$, that are also used to specify both thermodynamic and environmental conditions (e.g. availability of nutrients).

For most of the metabolic networks, and because the number of fluxes is greater than the number of metabolites, the set of linear equations obtained from the application of Equation 1 to the M metabolites usually leads to an under-determined system, for which there exists an infinite number of feasible flux distributions that satisfy the constraints.

However, if a given flux (or set of fluxes) is chosen to be maximized, and given the linearity of the constraints, it is possible to obtain a single solution by applying standard algorithms (e.g. *simplex*) for linear programming problems using a methodology known as Flux Balance Analysis (FBA) [8].

The combination of this technique with the existence of validated genome-scale stoichiometric models [5][1] allows to simulate the phenotypic behaviour of a microorganism under defined environmental conditions without performing any experiments. The most common flux chosen for maximization is the biomass, based on the premise that microorganisms have maximized their growth along natural evolution, a fact that has been validated by experiments [7].

III. Evolutionary Algorithms

A. Representation schemes and reproduction operators

The optimization problem addressed in this work consists in selecting, from a set of genes in a microbe's genome, a subset to be deleted in order to maximize a given objective function related to the microorganism's metabolism.

The first issue to address, when developing a proper *EA* to tackle this task, is the encoding of the solutions into the chromosome of an *EA*'s individual.

Two alternatives have been used in this regard:

- **Binary representations** - where each possible gene in the microbial strain is directly encoded into a boolean gene in the individual. The deletion of the gene is encoded by the value 0, while its presence implies the value 1. The size of the chromosome is equal to N (the number of genes in the model of the microorganism).
- **Set-based representations** - where only the gene deletions are represented in the *EA*'s chromosome. Each individual consists of a set of integer values representing the genes that will be knocked out. Therefore, if a gene in the *EA*'s individual has the value i, this means the i-th gene in the microbe's genome is deleted. The value of each gene is, therefore, an integer with a value between 1 and N. Two variants of this representation can be defined, considering either fixed or variable sized sets.

The *EA* with a binary representation, which will be denoted by *Genetic Algorithm (GA)*, given its similarities with the classical *GA*, makes use of standard crossover and mutation operators. The *uniform crossover* [16] was chosen, since the relative order of the genes does not have any special biological meaning.

Regarding mutation, the traditional *bit mutation* operator is used. The mutation operator can be applied to a variable number of genes randomly generated from 1 to L (L was set to 3 in the experiments). Both operators are used with equal probabilities to create new solutions.

The set based representation *EA*, which will be denoted by *SBREA* in the following, also uses two reproduction operators, a crossover and a mutation. The crossover operator is inspired on *uniform crossover* and works as follows: the genes that are present in both parent sets are kept in both offspring; the genes that are present in only one of the parents are sent to one of the offspring, selected randomly with equal probabilities. The mutation operator is a *random* mutation, that replaces a gene by a random value in the allowed range.

In *SBREAs*, a minimum and a maximum set size are defined. If these values are equal the search only goes through sets of a given cardinality. The operators comply with that constraint by creating solutions always of the same size. In the case of the *uniform crossover*, this implies that, when selecting the destination of the genes that are present in only one parent, if an offspring reaches the maximum number of elements in the set, the remaining genes go to the other offspring.

If the values of the parameters are different, variable-sized sets can be encoded and compete within the same population. In this case, two additional mutation operators are defined in order to be able to create solutions with a distinct size:

- *Grow*: consists in the introduction of a new gene into the chromosome, whose value is randomly generated in the available range.
- *Shrink*: a randomly selected gene is removed from the genome.

The *Grow* and *Shrink* mutations are each used with a probability of 5% each, meaning that 10% of the new individuals are created in this way. The remaining are bred by the aforementioned crossover and mutation operators with equal probabilities.

In the experiments reported in this work, when a variable size *SBREA* is used, the minimum size is set to 1 and the maximum size is set to the number of genes (N), thus not restricting the possible range of solutions.

It is important to mention that both binary and variable size set based representations are mathematically equivalent, since there is a one-to-one correspondence between the solutions. This means that the underlying search space is the same. However, the reproduction operator and initialization schemes create differences in the way the search space is explored, that can results in distinct outcomes in the end of the optimization process.

B. Decoding and evaluating

The decoding and evaluation process is very similar in both representations. The main principle considered is a correspondence between genes and metabolic reactions, i. e., each gene represented in the *EA* individuals encodes a particular enzyme that catalyzes one or several metabolic reactions.

In binary representations, the individuals are evaluated by taking all genes whose value is 0 and forcing the corresponding fluxes in the model to be also 0, therefore disabling the reactions encoded by those genes. In the set based encoding, the process is similar and works by taking each value in the set and forcing the flux it indexes to the value 0.

In both cases, the process proceeds with the simulation of the mutant using *FBA*. The output of this algorithm is the set of values for the fluxes of all reactions, that are then used to compute the fitness value, given by an appropriate objective function.

In this work, the adopted objective function is the *Biomass-Product Coupled Yield (BPCY)* [11], given by:

$$BPCY = \frac{PG}{S} \tag{2}$$

where P stands for the flux representing the excreted product; G for the organism's growth rate (biomass flux) and S for the substrate intake flux.

This is a good example of a nonlinear function that could not be optimized by algorithms such as *OptKnock*. Besides optimizing for the production of the desired product, this objective function also allows to select for mutants that exhibit high growth rates, i. e., that are likely to exhibit a higher productivity, an important industrial aim.

C. Selection and initial population

Both *EA*s use a selection procedure that consists in converting the fitness value into a linear ranking in the population, and then applying a roulette wheel scheme. In each generation, 50% of the individuals are kept from the previous generation, and 50% are bred by the application of the reproduction operators. An elitism value of 1 is used, allowing the best individual of the population to be always kept.

An initial population is randomly created and the termination criterion is based on a fixed number of generations. In the binary representations, the probability of a gene deletion is set to 1% in the initial population, in order to reduce the number of knockouts. In the variable size *SBREAs*, the size of the individual is randomly created in the range [1,10].

D. Pre-processing and post-processing

Typically, in microbial genome-scale models the number of variables (fluxes over metabolic reactions) is quite high (hundreds or even a few thousands) and therefore the search space is very hard to address by the optimization algorithms. Thus, every operation that gives a contribution to reduce this number, without compromising the quality of the solutions, greatly improves the convergence of the *EA*.

In this work, a number of operations is implemented in order to reduce the search space, namely:

- Detection of variables (fluxes) that, given the constraints of the linear programming problem, cannot exhibit values different from 0. For every variable (flux) in the model, two linear programming problems are solved running the *simplex algorithm*. The first is defined by setting the current variable as the maximization target, while for the second the same variable is minimized. If both have an optimum solution with a value of 0 for the objective function, the variable is removed from the model.

- Detection of equivalent variables, i.e. pairs of variables that are constrained to have the same value by the linear programming model. These are directly identified from the S matrix coefficients. Each group of equivalent variables is replaced by a single variable.
- Discovery of essential genes that can not be deleted from the microorganism genome. As these genes should not be considered as targets for deletion, the search space for optimization is reduced. For each gene in the microbe's genome, a linear programming problem instance is defined, setting the corresponding flux to 0, while maximing the biomass flux as usual. After running the simplex algorithm, if the resulting biomass flux is zero (or near zero) the gene is marked as essential. The biological meaning of this fact is that the microbe is unable to survive when the gene is absent. It should be noted that, unlike the previous ones, this process does not imply any changes in the model, but produces information that is useful for the optimization algorithms.

The list of essential genes can be manually edited to include genes that are known to be essential, although that information can not be reached from the mathematical model. This feature also allows the definition of a set of genes that can not be deleted by the optimization algorithm in a given experiment.

Aditionally, at the end of each *EA*'s run, the best solution goes through a simplification process. This is achieved by identifying all gene deletions that contribute to the fitness of the solution, removing all deletions that keep the objective function unaltered. The aim of this step is to keep only the necessary knockouts, given that the practical implementation of a gene deletion is both time consuming and costly.

E. Implementation issues

The implementation of the proposed *EA*s was based on a general purpose package, developed by the authors in the *Java* programming language, which allowed the development of the new features introduced in this work.

In the implementation of the *FBA* algorithm, the *GNU linear programming package (GLPK)*[1] was used to run the *simplex* algorithm.

IV. Case studies and results

A. Experimental setup

Two case studies were used to test the aforementioned algorithms. Both consider the microorganism *Escherichia coli* and the aim is to produce succinic and lactic acid (case studies I and II, respectively), with glucose as the substrate. The genome-scale model for this microorganism used in the simulations was developed by Reed et al [12].

This model considers the whole *E. coli* metabolic network, including a total of $N = 1075$ fluxes and $M = 761$ metabolites. After the pre-processing stages, the simplified model remains with $N = 550$ and $M = 332$ metabolites. Furthermore, 227 essential genes are identified, which leaves 323 variables to consider by the optimization algorithms.

Both the binary *GAs* and the *SBREAs* described above were applied in both case studies. The latter was implemented in its fixed and variable size versions. In the first case, the cardinality of the set (k) took the following values: 2,3,4,5,6,8,10 and 12.

In all cases, the following values were used for the *EA* parameters: the population size was set to 100 and the number of generations was set to 1000. This resulted in about 50000 fitness evaluations in each run. For each experimental setup the process was repeated for 30 runs and the mean and 95% confidence intervals were calculated.

B. Case study I: Succinic acid

Succinic acid is one of the key intermediates in cellular metabolism and therefore an important case study for metabolic engineering strategies[9]. In fact, the knockout solutions that lead to an improved phenotype regarding the production of succinic acid are not straightforward to identify since they involve a considerable number of interacting reactions. Aditionally, it is a chemical used as feedstock for the synthesis of a wide range of other chemicals with several industrial applications (e.g. food industry).

Succinic acid and its derivatives have been used as common chemicals to synthesize polymers, as additives and flavoring agents in foods, supplements for pharmaceuticals, or surfactants. Currently, it is produced through petrochemical processes that can be expensive and have significant environmental impacts.

In this case, the process of aerobic production of succinic acid from *E. coli* was approached. The aerobic condition was forced by adding the fluxes related to oxygen consumption to the file of essential genes, thus preventing its deletion.

In Table I, the results for the all the *EAs* in this case study are given. The results show the mean, the 95% confidence interval and the maximum value of the objective function (BPCY) for each *EA* configuration (specifying its representation and maximum number of knockouts k when applicable). In the last column, the mean of the number of gene deletions in the solution (after the post-processing simplification process) are shown.

By examining Table I it is possible to observe that, when using set-based representations with fixed size chromosomes, the results improve with the increase on the number of gene deletions and the best results are obtained for a maximum of 12 knockouts. The improvement obtained when increasing from 6 to 12 gene deletions is essentially visible in the increase of the mean, since the best solution suffers minor improvements. Furthermore, there is less variability in the results, given by the smaller confidence intervals,

[1]http://www.gnu.org/software/glpk/

TABLE I
RESULTS OBTAINED BY THE *EA* FOR THE CASE STUDY I - PRODUCTION OF SUCCINIC ACID.

EA	k	Mean	Conf. int.	Best	Knockouts
Binary	-	0.3220	±0.0263	0.3578	42.9
SBREA	2	0.0532	±0.0092	0.0752	2.0
SBREA	3	0.0889	±0.0101	0.1459	3.0
SBREA	4	0.1877	±0.0377	0.3366	3.8
SBREA	5	0.2756	±0.0352	0.3511	5.0
SBREA	6	0.2978	±0.0345	0.3573	5.7
SBREA	8	0.3428	±0.0165	0.3577	7.5
SBREA	10	0.3490	±0.0132	0.3578	8.5
SBREA	12	0.3565	±0.0009	0.3578	9.4
SBREA	VS	0.3574	±0.0005	0.3579	12.7

TABLE II
RESULTS OBTAINED BY THE *EA* FOR THE CASE STUDY II - PRODUCTION OF LACTIC ACID.

EA	k	Mean	Conf. int.	Best	Knockouts
Binary	-	0.2535	±0.0030	0.2553	11.4
SBREA	2	0.0019	±0.0010	0.0031	1.6
SBREA	3	0.2547	±0.0000	0.2547	3.0
SBREA	4	0.2553	±0.0000	0.2553	4.0
SBREA	5	0.2553	±0.0000	0.2553	4.0
SBREA	6	0.2553	±0.0000	0.2553	4.0
SBREA	VS	0.2553	±0.0000	0.2553	4.0

On the other hand, the binary representation is unable to reach the level of results of its set based counterpart, specially when looking at the mean value of the solutions. This denotes a high variability in the results, indicating that the algorithm fails to find high quality solutions in a considerable number of runs, which is confirmed by the relatively large confidence interval. Furthermore, it created solutions with a large number of gene deletions, an undesirable feature.

It should be noted that the number of knockouts given in the last columns of the table refers to the simplified solutions and that the number of knockouts in the original solutions generated by em GAs is typically even quite higher.

The variable size alternative produces the best overall results, both in the mean and in the best solution obtained and it seems to be able to automatically find the appropriate number of gene deletions. It also presents a very low variability, given the small confidence interval.

This fact is also confirmed by the fact that 7 genes appear in more than 90% of the best solutions reached in each run and a total of 10 genes appear in more than 80%. This denotes a high consistency in the results obtained.

C. Case study II - Lactic acid

Lactic acid and its derivatives have been used in a wide range of food-processing and industrial applications like meat preservation, cosmetics, oral and health care products and baked goods. Additionally, and because lactate can be easily converted to readily biodegradable polyesters, it is emerging as a potential material for producing environmentally friendly plastics from sugars [6].

Several microorganisms have been used to commercially produce lactic acid, such as *Lactobacillus* strains. However, those bacteria also have undesirable traits, such as a requirement for complex nutrients which complicates acid recovery.

On the other hand, *Escherichia coli* has many advantageous characteristics as a production host, such as rapid growth under aerobic and anaerobic conditions and simple nutritional requirements. Moreover, well-established protocols for genetic manipulation and a large physiological knowledge base enable the development of *E. coli* as a host for production of optically pure D- or L-lactate by metabolic engineering [3].

In Table II, the results for the *EAs* in case study II are given. The first conclusion that can be drawn is that this case study seems to be less challenging than the one presented on the previous section. In fact, 4 gene deletions seem to be enough to obtain the best solution, and therefore the results with k larger than 6 are not shown.

As before, the binary *GA* has a worse performance, with a slightly lower mean and reaching solutions with a larger number of knockouts. On the other hand, the variable size *SBREA* confirms its merits once it is again able to find the best solution in all runs, and automatically finds the adequate number of knockouts.

D. Discussion

The comparison of the several *EA* alternatives in both case studies, gives some clear indications regarding their relative performance. In fact, the binary *GAs* seem to have some trouble in findind good solutions in a consistent way and tend to provide solutions with a large number of gene deletions, a feature that is undesirable in this task given the costs involved in implementing knockouts in the lab.

On the other hand, the *SBREAs* with variable size genomes emerge as the best algorithm available in this study, as they seem able to obtain the best results, reaching good quality solutions and presenting low variability. Additionally, they are able of automatically finding the adequate number of gene deletions.

As it is impossible for a metabolic engineer to know, *priori*, the number of knockouts that will lead to a high-performance microbial strain, this feature is of great relevance for the application of any algorithm that designs microbial strains *in silico*. To the best of the authors knowledge, the proposed approach is the first one, apart form the binary *GAs* firstly introduced in [11] (which suffer from the drawbacks mentioned here), that is able to pinpoint, in a single run, the number of genes to be eliminated from a microbial genome in order to achieve an optimal production of a desired compound.

Two features that are important when comparing meta-heuristic optimization algorithms are the computational effort required and the convergence of the algorithm to a good solution. The computational burden of all the *EAs* is approximately the same, since the major computational effort is devoted to fitness evaluation and the same number of solutions is evaluated in every case. A typical run of an *EA* for the case studies presented will take approximately two or three hours in a regular PC.

Regarding the convergence of the algorithms, a plot of the evolution of the objective function along the generations of the *EAs* is given in Figure 1 (the mean of the 30 runs is plotted in each case). An illustrative subset of the algorithms was selected to allow a better visualization. It is clear from this plot that the variable size *SBREA* converges a bit slower than the case of $k = 12$, but the results become similar at about half of the evolution.

This slightly slower convergence is probably due to the fact that solutions with a distinct size have to compete and the adequate cardinality has to emerge during evolution. If a good value for k is not known *a priori* this overhead is, however, much smaller than the one that would be necessary to run a number of *EAs* with different values of k.

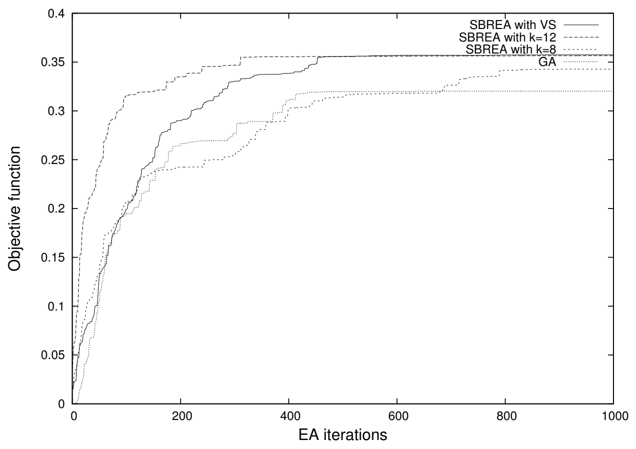

Fig. 1. Convergence plots for the best algorithms in case study I.

It is important to refer that a biological validation of the results was conducted. The best solutions obtained by the algorithms were analyzed by experts in *Biotechnology* and their biological significance was validated with the help of Bioinformatics databases (e.g. *EcoCyc*[2]).

V. Conclusions and further work

The development of efficient and accurate modeling and optimization methods in *Metabolic Engineering* has a considerable impact in *Biotechnology*, leading to substantial economical gains in areas such as the production of pharmaceuticals, fuels and food ingredients.

In this work, a contribution to this arena was provided by the development of *Evolutionary Algorithms* that are able of reaching an optimal (or near optimal) set of gene deletions in a microbial strain, in order to maximize the production of a given product.

The main contribution of this work was the introduction of a novel set-based representation, that made use of variable size chromosomes. These were tested in two case studies that dealed with the production of succinic and lactic acid by the *E. coli* bacterium.

The work reported here is ongoing and there are still a number of features that need to be introduced/ improved. These include other algorithms for simulation and distinct objective functions. Regarding the former, an alternative algorithm for simulating mutants' phenotype is the *MOMA* algorithm, that was proposed by Segre et al [13], where it is assumed that knockout metabolic fluxes undergo a minimal redistribution with respect to the flux configuration of the wild type. This implies solving a quadratic programming problem, whose aim is to minimize the differences between the fluxes in the mutant and the ones in the wild type. One other alternative is the *ROOM* algorithm [14].

It would also be very interesting to consider an objective function capable of taking into account the number of knockouts of a given solution and the cost of its experimental implementation. Some improvements are also expected in the visualization tools guided by the aim of making easier the analysis of the solutions by researchers from *Biotechnology*.

VI. Acknowledgments

The authors wish to thank the Portuguese Foundation for Science and Technology for their support through project ref. POSC/EIA/59899/2004, partially funded by FEDER.

References

[1] I. Borodina and J. Nielsen. From genomes to in silico cells via metabolic networks. *Current Opinion in Biotechnology*, 16(3):350–355, 2005.
[2] A.P. Burgard, P. Pharya, and C.D. Maranas. Optknock: A bilevel programming framework for identifying gene knockout strategies for microbial strain optimization. *Biotechnol Bioeng*, 84:647–657, 2003.
[3] D.E. Chang, S. Shin, J. Rhee, and J. Pan. Homofermentative production of d- or l-lactate in metabolically engineered escherichia coli rr1. *Appl. Environ. Microbiol*, 65:1384–1389, 1999.
[4] C. Chassagnole, N. Noisommit-Rizzi, J.W. Schmid, K. Mauch, and M. Reuss. Dynamic modeling of the central carbon metabolism of escherichia coli. *Biotechnology and Bioengineering*, 79(1):53–73, 2002.
[5] M.W. Covert, C.H. Schilling, I. Famili, J.S. Edwards, I.I. Goryanin, E. Selkov, and B.O. Palsson. Metabolic modeling of microbial strains in silico. *Trends in Biochemical Sciences*, 26(3):179–186, 2001.
[6] K. Hofvendahl and B. Hahn-Hagerdal. Factors affecting the fermentative lactic acid production from renewable resources. *Enzyme Microbial Technology*, 26:87–107, 2000.
[7] R.U. Ibarra, J.S. Edwards, and B.G. Palsson. Escherichia coli k-12 undergoes adaptive evolution to achieve in silico predicted optimal growth. *Nature*, 420:186–189, 2002.
[8] K.J. Kauffman, P. Prakash, and J.S. Edwards. Advances in flux balance analysis. *Curr Opin Biotechnol*, 14:491–496, 2003.
[9] S.Y. Lee, S.H. Hong, and S.Y. Moon. In silico metabolic pathway analysis and design: succinic acid production by metabolically engineered escherichia coli as an example. *Genome Informatics*, 13:214–223, 2002.
[10] J. Nielsen. Metabolic engineering. *Appl Microbiol Biotechnol*, 55:263–283, 2001.
[11] K. Patil, I. Rocha, J. Forster, and J. Nielsen. Evolutionary programming as a platform for in silico metabolic engineering. *BMC Bioinformatics*, 6(308), 2005.
[12] J.L. Reed, T.D. Vo, C.H. Schilling, and B.O. Palsson. An expanded genome-scale model of escherichia coli k-12 (ijr904 gsm/gpr). *Genome Biology*, 4(9):R54.1–R54.12, 2003.
[13] D. Segre, D. Vitkup, and G.M. Church. Analysis of optimality in natural and perturbed metabolic networks. *Proc National Acad Sciences USA*, 99:15112–15117, 2002.
[14] T. Shlomi, O. Berkman, and E. Ruppin. Regulatory on/off minimization of metabolic flux changes after genetic perturbations. *Proc National Acad Sciences USA*, 102:7695–7700, 2005.
[15] G. Stephanopoulos, A.A. Aristidou, and J. Nielsen. *Metabolic engineering principles and methodologies*. Academic Press, San Diego, 1998.

[2]http://www.ecocyc.org

[16] Gilbert Syswerda. Uniform Crossover in Genetic Algorithms. In J. Schaffer, editor, *Proc. of the 3rd Intern. Conference on Genetic Algorithms*, pages 2–9, San Mateo, CA, 1991. Morgan Kaufmann.

[17] M. Tomita. Whole-cell simulation: a grand challenge for the 21st century. *Trends Biotechnol*, 19:205–210, 2001.