Evolutionary Symbolic Discovery for Bioinformatics, Systems and Synthetic Biology

P. Widera, J. Bacardit, N. Krasnogor
University of Nottingham, United Kingdom
{plw,jqb,nxk}@cs.nott.ac.uk

C. García-Martínez
University of Córdoba, Spain
cgarcia@uco.es

M. Lozano
University of Granada, Spain
lozano@decsai.ugr.es

ABSTRACT

Symbolic regression and modeling are tightly linked in many Bioinformatics, Systems and Synthetic Biology problems. In this paper we briefly overview two problems, and the approaches we have use to tackle them, that can be deemed to represent this entwining of regression and modeling, namely, the evolutionary discovery of (1) effective energy functions for protein structure prediction and (2) models that capture biological behavior at the gene, signaling and metabolic networks level. These problems are not, strictly speaking, "regression problems" but they do share several characteristics with the latter, namely, a symbolic representation of a solution is sought , this symbolic representation must be human understandable and the results obtained by the symbolic and human interpretable solution must fit the available data without over-learning.

Categories and Subject Descriptors

J.3 [**Life And Medical Sciences**]: Biology and genetics; I.2.6 [**Artificial Intelligence**]: Learning—*knowledge acquisition*

General Terms

Algorithms, Design, Experimentation

Keywords

Evolutionary Search, Modeling, Data Mining, Optimisation, Genetic Programming, P Systems, Protein Structure, Systems Biology, Synthetic Biology

1. INTRODUCTION

Evolutionary Algorithms (EA), more prominently Genetic Programming (GP), have been successfully applied to a variety of symbolic regression problems such as the investigation of molecular docking[5], protein structure energy function inference [46], anticancer therapeutic response prediction [4], modeling the release kinetics of pesticides [1], as well as more traditional – albeit equally challenging – regression problems such as fibonacci, squares and other numerical series [21]. In this paper we briefly overview two problems that can be deemed to represent a challenging "entwining" of symbolic regression and modeling, namely, the evolutionary discovery of (1) effective energy functions for protein structure prediction and (2) models that capture biological behavior at the gene, signaling and metabolic networks level. We describe next each of them in turn. In the first of these two problems we used standard Genetic Programming while for the other problem we employ a combination of metaheuristics that evolve and fine tune executable structures.

GECCO'10, July 7–11, 2010, Portland, Oregon, USA.

2. EVOLVING ENERGY FUNCTION FOR PROTEIN STRUCTURE PREDICTION

The following section sumarises material that originally appeared in our papers [45, 46]. The goal of protein structure prediction is to construct a 3D model of a protein structure, that is to find coordinates of the protein atoms from its 1D description given as an ordered sequence of amino acids. It is believed that this sequence contains enough information to derive from it an exact protein shape. Precise $1D \rightarrow 3D$ transformation is, however, not yet possible as the algorithm of protein folding remains unknown.

The protein structure is generally predicted using a search method that iteratively alters models of the structure to minimise their energy. Whenever possible the initial models are adapted from the native structure of closely related proteins.This procedure is based on the thermodynamic hypothesis [3] which states that the protein in its native folded state is in the thermodynamic equilibrium and thus its free energy is minimal.

In the most challenging template-free category at CASP [6], top ranked predictors, e.g. I-TASSER [47] or Robetta [38], define the energy function as a linear combination of energy terms designed by experts. This terms do not reflect the protein potential energy explicitly but they rather express a probability that the observed structure is native-like based on the statistical distribution of features among the experimentally determined known protein structures.

The weights of specific terms of the energy function are optimised on a training set of computer generated protein models (also called decoys). The goal of the optimisation is to maximise the correlation between the structure energy and its structural similarity to the known native structure. The more dissimilar is the structural model the higher should be its energy. It is also desired that all the decoys

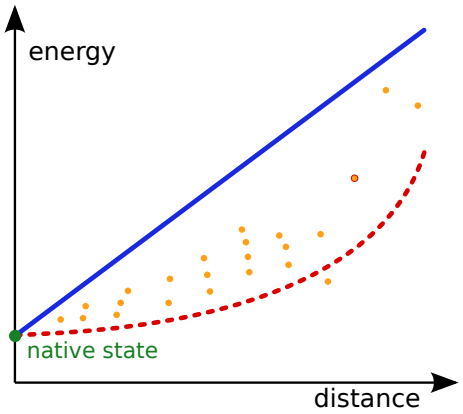

Figure 1: Dependency between the energy and distance to the native structure. Blue solid line shows the ideal case of a perfect linear fit. Red dashed line shows more realistic non-linear dependency where energy discriminates between distant structures more easily. Orange dots are an indicative cartoon of what can be expected in terms of correlation for energy potentials that are currently in use by structure predictors.

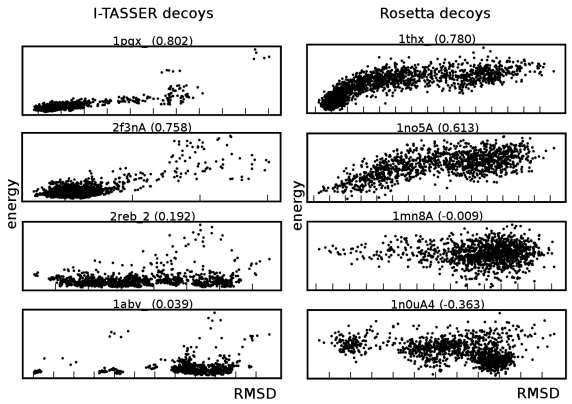

Figure 2: Examples of correlation between energy and distance to the native structure measured as root mean square deviation (RMSD) for I-TASSER and Rosetta generated decoys. Each panel represents a single protein chain (name and the correlation coefficient is given on top), and each dot in a panel represents a single decoy.

have a higher energy than the real native structure. An energy function with such properties is then able to distinguish between near-native and dissimilar structures (see Figure 1). In the best case the decoys sorted by the values of energy function would be in the same order as when sorting is done by the structural similarity to the target native structure. Therefore, in general, the more information about the structural similarity is captured by the energy function the better.

2.1 Methods

In the simplest case, where the energy function is just a weighted sum of energy terms, it could be fitted to reflect the structural similarity by the linear regression. This approach is used to design the Rosetta's energy function [42]. More complex approach is used in I-TASSER where apart from linear regression also a correlation coefficient and a separation from the native are used in the objective function [48]. However, in general case the energy function could be expressed by any functional combination of the energy terms without limiting it to the linear combination.

The goal of the optimisation is to have an energy function that reflects the structural similarity to the target native structure and is alone enough to navigate the search process in the protein structure prediction. This differs from standard regression as the function doesn't have to be fit to the data in the strict sense. It is enough to obtain a good approximation of decoys relative quality, that is an energy based ranking that would be similar to the ranking based on structural similarity. For a more high level assessment of how fit the function is, even a simple correlation coefficient might be sufficient. Examples of a correlation between the energy end the similarity to the native for decoys generated by both I-TASSER and Rosetta prediction methods are shown in Figure 2.

When the energy function is a general functional combination of energy terms, not just a weighted sum, it could capture the information about the structural similarity more precisely. Moreover, as it features a range of basic algebraic operators and/or transcendental functions operating on a fixed set of symbols (energy terms) it is meant to be human-interpretable and have a knowledge discovery potential. Going a step further, with more atomic energy components, being for example low level characteristics of the selected protein atoms (instead of high level statistical aggregates), the symbolic regression could generate much more fine-tuned energy functions and discover which properties are most useful for different types of proteins.

2.2 Results

The first symbolic regression approach to the design of energy functions that could be useful in the process of protein structure prediction was made with a use of the genetic programming [46]. The key element of the GP algorithm design in that work was the definition of a fitness. Its definition was based on the distance between two rankings of decoys: the reference ranking R_R generated using the root mean square deviation (see below) as a reference measure and the evolved ranking R_E generated using the evolved energy function. In case of the ideal energy function the evolved ranking would be identical to the reference one, as the energy function would perfectly reflect the structural similarity to the native state.

The exact procedure to compute the total fitness was as follows:

1. construction of the reference ranking $\mathbf{R_R}$ (decoys sorted by the similarity to native)

2. rank decoys using evolved energy function $\mathbf{R_E}$ (decoys sorted by energy)

3. comparison of rankings — R_R vs. R_E
4. total fitness = average distance for m proteins
$$F = \frac{1}{m}\sum_{i=1}^{m} d(R_{Ri}, R_{Ei})$$

The reference ranking was constructed using RMSD, the oldest and *de facto* standard measure of similarity between protein structures. It is calculated as a root mean square deviation between 3D coordinates of C_α atoms of two structures minimised with respect to the rotation using Kabsch algorithm [26][11].

$$RMSD = \sqrt{\frac{1}{N}\sum_{i=1}^{i=N}\delta_i^2} \quad (1)$$

The ties in the ranking construction were solved in two ways. First, a tie was called when RMSD values were the same up to the first two decimal places, what corresponds to a precision of 1 picometer (in terms of inter-atomic distance). Then a second sorting criterion was used, being the original I-TASSER energy of the decoys being compared. As this method was not very efficient (I-TASSER energy alone does not reflect well the structural similarity), another approach was assigning an averaged rank to all decoys with the same RMSD value. The drawback of this approach was that only the Spearman distance (see below) could be used to compare the ranks as other measures require a permutation as an input.

To compute the distance between the rankings several different measures were used:

- *Levenshtein edit distance* [29], a popular string metric where distance is given by the minimum number of operations (insertion, deletion or substitution of a character) needed to transform one string into the other,
$$L(a,b) = d_{n,n}$$
$$d_{i,0} = d_{0,i} = i \text{ for } i = 0 \dots n$$
$$d_{i,j} = min\{d_{i-1,j}+1, d_{i,j-1}+1, d_{i-1,j-1}+c(i,j)\}$$
$$c(i,j) = \begin{cases} 0 & \text{if } a(i) = b(i) \\ 1 & \text{if } a(i) \neq b(i) \end{cases}$$
- *Kendall tau distance* [28], the number of inversions between two permutations also known as the bubble-sort distance,
$$K(a,b) = |\{(i,j) : i < j \wedge a(i) < a(j) \wedge b(i) > b(j)\}|$$
- *Spearman edit distance* [13], the sum of differences between the ranks of elements.
$$S(a,b) = \sum_{i}^{n} |a(i) - b(i)|$$

These measures differs in computation cost, e.g. for the Levenshtein distance a dynamic programming algorithm has a complexity of $O(n^2)$ and the Spearman distance can be calculated in linear time. They also differ in the range of possible distance values, i.e. maximum distance for the Kendall measure is $\frac{n(n-1)}{2}$, for the Spearman measure it is $\frac{1}{2}n^2$, while the Levenshtein distance, similarly to many other editing distance metrics on permutations such as Hamming metric, Cayley distance or Ulam metric, is bounded by the $O(n)$.

The best evolved energy functions has been shown to have over 10% higher fitness than the functions found by the random walk with the same number of fitness evaluations. The fitness was as well higher than in case of the naive sum of all energy terms $E_{naive} = \sum_i T_i$.

Also the linear combination of of terms $E_{linear} = \sum_i w_i T_i$ with a vector of weights $\vec{w}$ optimised using the Nelder-Mead downhill simplex method [33][32] was outperformed by GP with best fitness being over 10% higher.

Moreover, the GP algorithm was able to discover the most and the least useful energy terms without knowing the correlation of these terms to the RMSD. The most frequently used energy terms in the best evolved functions had the highest correlation to RMSD and the least frequently used energy terms were the ones with the correlation to RMSD closest to zero.

3. INFOBIOTICS MODEL EVOLUTION IN SYSTEMS AND SYNTHETIC BIOLOGY

The following section sumarises material that originally appeared in our papers [39, 9, 15]. Living cells are complex systems that arise from a rich array of interrelated biomolecular processes. In order to understand, manipulate and even coerce a cellular system into producing a target phenotype, the development of good models is a critical steppingstone. Thus sibling disciplines systems [2, 27] and synthetic [7] biology depend crucially on the availability of sophisticated and expressive modeling methodologies and tools. *Infobiotics* approaches [40, 25], also called *Executable biology* [14] and *Algorithmic Systems Biology*[35] are gaining momentum as alternative ways of modeling large biological complex systems. Infobiotics is based on models that are built not by specifying differential equations but rather mechanistically by defining algorithms (under a variety of possible formalisms) whose execution mimics the causal relation behind change and time/space dynamics in biological systems. Existent executable biology methodologies are rigorous and mathematically sound modeling techniques. Executable biology models are deemed to be expressive as they seamlessly capture the modularity behind many biological systems. Infobiotics models have been successfully used to model a variety of biological systems [37, 12, 18]. A fundamentally appealing property of Infobiotics models is that being intrinsically executable structures, they are appealing from an evolutionary computing point of view. These models represent large collections of relatively short algorithms that execute in a stochastic fashion to produce an emergen behaviour. Hence, they bring forth several interesting challenges to the evolutionary computing community. Moreover, evolving models to fit data (see below) is similar but not identical to standard symbolic regression as here the model is itself a (biological) computer program that must be run (i.e. simulated) and from this execution an emergent behaviour will be compared to the data to be fitted.

3.1 Methods

The approach discussed here uses a computational, modular and discrete-stochastic modelling approach based on P systems[36]. More specifically, a variant called stochastic P systems developed for the specification and simulation of cellular systems is used [40].

A stochastic P system is a construct

$$\Pi = (O, L, \mu, M_{l_1}, M_{l_2}, \ldots, M_{l_n}, R_{l_1}, \ldots, R_{l_n})$$

where O is a finite alphabet of objects representing molecules. $L = \{l_1, \ldots, l_n\}$ is a finite set of labels identifying compartment types. μ is a membrane structure containing $n \geq 1$ membranes defining compartments arranged in a hierarchical manner. Each membrane is identified in a one to one manner with labels in L which determines its type. M_{l_i} for each $1 \leq i \leq n$, is the initial configuration of membrane i consisting of a multiset of objects over O initially placed inside the compartment defined by membrane with label l_i. $R_{l_i} = \{r_1^{l_i}, \ldots, r_{k_{l_i}}^{l_i}\}$, for each $1 \leq i \leq n$, is a finite set of rewriting rules associated with the compartment with label $l_i \in L$ and of the following general form:

$$o_1[o_2]_l \xrightarrow{c} o_1'[o_2']_l \tag{2}$$

with o_1, o_2, o_1', o_2' multisets of objects over O (potentially empty) and $l \in L$ a label. These multiset rewriting rules affect both the inside and outside of membranes. An application of a rule of this form replaces simultaneously a multiset o_1 outside membrane l and a multiset o_2 inside membrane l by multisets o_1' and o_2', respectively. A stochastic constant c is associated specifically with each rule in order to compute its propensity according to Gillespie's theory of stochastic kinetics [19]. More specifically, rewriting rules are selected according to an extension of Gillespie's well known *Stochastic Simulation Algorithm* (SSA) [19] to the multicompartmental structure of P system models [34].

A biological module is a separable discrete entity that defines a particular biological function [22]. A *P system module* is a set of rewriting rules (such as those in 2) where some of the objects, kinetic constants or labels, might be variables. This increases reusability because large models can be constructed by combining common modules instantiated with specific values. At the end, the result is a set or rewriting rules representing a particular cellular model. Formally, a module M is specified as $M(V, C, L)$ where V constitutes object variables, which can be instantiated with specific objects, C are variables for the kinetic constants, and L are variables for the labels of the compartments appearing in the rules. For example the module *Constitutive or Unregulated expression* states how gene, gX, is firstly transcribed into its mRNA, rX, without any transcriptional regulatory factor; and then, the mRNA, rX, is translated into the corresponding protein pX. Cell machinery can also degrade the mRNA and protein. These processes take place in compartment l according to the kinetic constants $c_1, \ldots, c_4$.

$$UnReg(\{X\}, \{c_1, c_2, c_3, c_4\}, \{l\}) = \\ = \left\{ \begin{array}{l} r_1 : [gX]_l \xrightarrow{c_1} [gX + rX]_l \\ r_2 : [rX]_l \xrightarrow{c_2} [rX + pX]_l \\ r_3 : [rX]_l \xrightarrow{c_3} [\,]_l \\ r_4 : [pX]_l \xrightarrow{c_4} [\,]_l \end{array} \right\}$$

X is a variable that can be instantiated with a concrete gene. Kinetic constants may take specific values representing different transcription, translation and degradation rates. Further details and examples of other modules are available in [39, 9, 15]. In the automated generation of systems or synthetic biology models one is given (a) a set of experimental data, e.g., microarray datasets, phenotype arrays, optical density maps, etc, (see Fig. 3) and (b) a library of model modules and is asked to evolve a model, which might be "bootstrapped" with modules from the library, that when simulated reproduces the given data. [39, 9, 15] dealt with the problem of evolving P systems model structures and parameters. A variety of fitness functions were analysed as well as several metaheuristic algorithms.

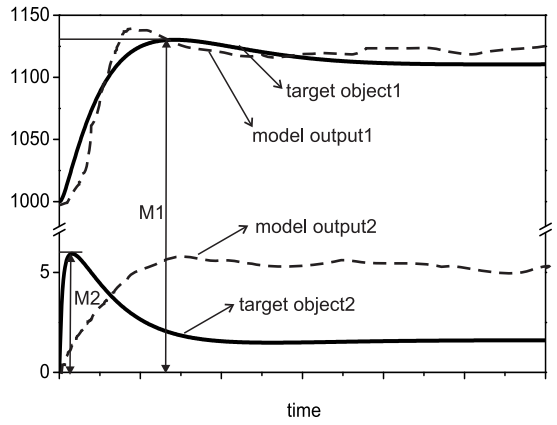

Figure 3: A number of target time series is given and the models must match the profiles of all of them simultaneously. The time series are noisy, have different scales ($M1, M2$) and differ in their key features, e.g., response times, half-lives, etc.

Suppose we have N target time series $(X_1, X_2, \ldots, X_N)$, each representing a specific protein, gene, rna, etc and where $X_j = (x_j^1, x_j^2, \ldots, x_j^M)^T$, that is, each time series has up to M data points. Each candidate stochastic model is run multiple times and an average model output obtained for each of the N time series: $(\hat{X}_1, \hat{X}_2, \ldots, \hat{X}_N)$ where $\hat{X}_j = (\hat{x}_j^1, \hat{x}_j^2, \ldots, \hat{x}_j^M)^T$, $j = 1, \ldots, N$. These output time series are then used to calculate fitness as follows:

In the *Equally Weighted Sum Method (F1)*, the fitness calculation formula for this method is:

$$Fitness\ (F1) = \sum_{j=1}^{N} \sum_{i=1}^{M} (|\hat{x}_j^i - x_j^i|)$$

This is the most commonly used method [31] in which all the error items from different objects are considered to have the same significance. As we depict in Fig. 3, using this method the fitness function can be dominated by the errors of the time series with large values, neglecting the errors of objects with small values. This can prevent the algorithm from finding a good compromise model for all the objects.

Data normalization is an important data preprocessing technology for many applications. Sola and Sevilla [43] systematically studied the importance of input data normalization for the application of neural networks to complex industrial problems by experimenting with five different data normalization procedures on the training data set. In essence, data normalization consists in the transformation of the original data into the range [0,1] in order to make the data comparable at the same level. There are many such transformations, for example, the *Normalisation Method (F2)*

$$\hat{f}_i(x) = \frac{f_i(x) - min\{f_i(x)\}}{max\{f_i(x)\} - min\{f_i(x)\}} \quad (3)$$

used in [10].

normalises the absolute error for each data point. Hence the formula to calculate the fitness using the F2 method is as follows:

$$Fitness\ (F2) = \sum_{j=1}^{N}\sum_{i=1}^{M} \frac{(|\hat{x}_j^i - x_j^i|) - min(|\hat{x}_j^i - x_j^i|)}{max(|\hat{x}_j^i - x_j^i|) - min(|\hat{x}_j^i - x_j^i|)}$$

The above normalization method removes the saliency of large absolute errors and brings all the time series and their misfit values into an equal footing. On the other hand, by compressing all the data into a $[0, 1]$ interval some of the time series subtleties might be lost.

Another possible fitness function could be the *Randomly Weighted Sum Method (F3)*. This method is similar to F1 but instead of assuming equal contribution from all the errors, here they are adjusted according to a normalized weight vector generated randomly.

A weight vector $(w_1, w_2, \ldots, w_N)$ is called normalized if it meets the following condition:

$$\forall j \quad w_j \geq 0 \ and \ \sum_{j=1}^{N} w_j = 1$$

This method was proposed by Ishibuchi and Murata [23] to deal with the case of fitness functions that are composed of a weighted sum of partial objectives. As this approach does not assure uniform sampling of the normalized weight vectors, Jaszkiewicz proposed the following algorithm in [24] to ensure that the weights vectors are drawn with uniform probability distribution:

$$\lambda_1 = 1 - \sqrt[J-1]{rand()}$$
$$\ldots$$
$$\lambda_j = (1 - \sum_{l=1}^{j-1} \lambda_l)(1 - \sqrt[J-1-j]{rand()})$$
$$\ldots$$
$$\lambda_J = 1 - \sum_{l=1}^{J-1} \lambda_l$$

where function $rand()$ returns a random value within the range (0,1) with uniform probability distribution. The algorithm above is used to randomly generate a normalized weight vector and then the weighted sum of all errors is used as the fitness value. This procedure is repeated K times and the average is used as the final fitness. Thus, the fitness calculation formula for this method is as follows:

$$Fitness\ (F3) = \frac{\sum_{n=1}^{K}\sum_{j=1}^{N}(w_j^n \sum_{i=1}^{M}(|\hat{x}_j^i - x_j^i|))}{K}$$

where w_j^n is the random weight for the jth target time series generated at the nth time.

Finally, we have also tested the *Equally Weighted Product Method (F4)*. This fitness method is obtained by multiplying all the error items for each target time series and the fitness calculation formula is:

$$Fitness\ (F4) = \prod_{j=1}^{N}\sum_{i=1}^{M}(|\hat{x}_j^i - x_j^i|)$$

GA for model structure optimization

```
BEGIN
  t = 0;
  read the imported model library lib;
  randomly generate an initial population of
    valid P system models P(0)based on lib;
  calculate the fitness of the individuals
    in P(0);
  while(t < SOMAXGENO)
  {
    do the parameter optimization for each
      individual in P(t) by GA (see Fig. 3);
    recombine P(t) by tournament selection and
      elitism strategy;
    randomly choose two parents from P(t) to do
      genetic operations (crossover, mutation)
        based on fitness;
    calculate fitness of the individuals in P(t);
    keep the best individual to next generation;
    t = t + 1;
  }
  output the best individual in current population
    as the final model;
END
```

Figure 4: GA for P system model structure optimization.

Bridgman [8] was the first author to refer to this approach and later Gerasimov and Repko [17] successfully applied this method to the multi-objective optimization of a truss. These fitness functions were plugged into a nested Evolutionary Algorithm to evolve P system models that could match a biological phenotype that is specified through a collection of time series representing molecular concentrations of various species. The EA's first layer searches for model structures using a *genetic algorithm* (GA); while the inner layer, also implemented as a GA, acts as a local search for the continuous parameters of the model. The pseudo code of the main GAs is shown in Fig. 4.

In [15] the internal parameter optimisation algorithm was improved by replacing the GA by a Variable Neighborhood Search with Evolutionary Components. In [30], Lozano and García-Martínez introduced a novel methodology to design hybrid algorithms, which invoked EAs as inner components. The idea was to replace classic components of well-known optimisation methods by specialised EA approaches performing the same tasks, but more satisfactorily. They called these EA approaches *evolutionary components*. Later, they presented a model based on *Variable Neighbourhood Search* method (VNS) [20] where the original three components were replaced by three evolutionary components [16]: *Generation:* At the beginning, CMA-ES is executed to generate an initial solution s^c. The applied parameter values were the ones suggested in the C code available at the webpage of the author[1]. *Improvement:* Afterwards, *Continuous Local EA* [16] locally optimises the current solution, attempting to accurately reach a local optimum. An important remark of Continuous Local EA is that it employs a specific replacement strategy and mating scheme to gather up and exploit valuable information from the search space. *Shaking: Mirco Cross-generational elitist selection, Heterogeneous recombination, and Cataclysmic mutation* [16] perturbs the best solution so far trying to get out of the valley where it lies. Following the idea in the original VNS, the intensity of per-

[1] `http://www.lri.fr/~hansen/cmaes_inmatlab.html`

turbation performed by this component (N_k) is strengthen when the previous improvement process could not improve the best solution so far, and weaken otherwise. The new solution is given to Continuous Local EA to be optimised. The second and third steps are then repeated until a stoping condition is fulfilled.

3.2 Results

Using the methods F1, F3 or F4 the algorithm always finds good models that can accurately reproduce the dynamical behavior of the target cellular system. F2, usually reports worse results than the alternative fitness functions. Specifically, for simple cases all these methods consistently find a single model structure, *i.e.* the target one, nevertheless the diversity of the models found by our algorithm using the methods increases significantly with the complexity of the target data set. More interestingly, some of these models are simpler than the target one. This result is very encouraging as it could help biologists to design new experiments to discriminate among competing hypothesis (models) and then only engineered in the lab the one that has been proven as the best. This is a potentially very useful feature to help close the loop between modeling and experimentation in both synthetic and systems biology. Generally speaking, if some target output objects of the predesigned cellular system have very different orders of magnitude in their time series, F3 and F4 work better than F1 when trying to obtain a good compromise solution. Moreover, we also run five different EAs specifically in what relate to the optimisation of the models parameters, i.e., their structures were assumed to be known. We analysed their scalability in terms of the numbers of dimensions to search. For relatively low dimensional problems, population based methods worked better while as the number of dimensions increased procedures whose search processes are focused on just one solution, such as VNS-ECs described above become more profitable.

4. CHALLENGES

Both of the problems and the methods described in the previous sections are a complex mixture of regression and modeling in which one is interested not only on fitting data but also on being able to generalize and explain. There are several challenges that remain unsolved in these two problems.

4.1 Design of Effective Fitness Functions

The design of effective fitness functions for these problems is a critical and yet unmet milestone. In the case of the Genetic Programming Evolution of a Protein Structure energy function, a more optimal ranking construction and comparison and a better choice of robust reference measures of structural similarity are needed. In addition, the selection of operators and low-level attributes that would enable more complex computations (including conditional clauses or loops) while keeping the evolved functions compact and human-interpretable would be desirable. As it is, in principle, more difficult to evolve a single general function that would be able to capture the knowledge about structural similarity of many different kinds of proteins, the set of GP operators could be extended with conditional clauses operating on low-level protein attributes (e.g. length) or predicted features (e.g. secondary structure or recursive convex hull

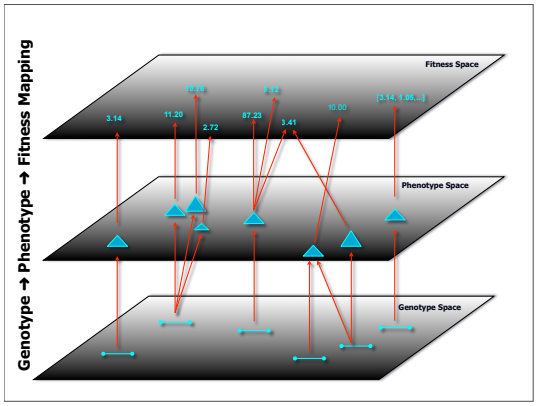

Figure 5: Symbolic regression and modeling under complex genotype → phenotype → fitness mappings

[44]), to automatically tune the evolving functions to different protein types. Moreover, it is not yet understood how general the evolved functions are and whether the simultaneous use of different set of proteins or decoys generated by different predictors will not reveal any over-training we might have missed. In terms of the evolution of systems and synthetic biology models, the design of effective fitness functions remains elusive. Researchers have naively assume that minimising the sum of squared errors would always be the best strategy. We have shown that, as expected, there is no single best fitness function and that depending on the type of input data and problem at hand different fitnesses must be used. Moreover, in automated model building, additional fitness criteria might be considered [9] such as parameters sensitivity, model robustness and parsimony, etc. The question of whether a meta-algorithm could auto-tune the fitness function remains open.

4.2 What Space is Being Searched and How?

The two combined symbolic regression and modeling problems described in this paper show a complex genotype → phenotype → fitness mapping (see Fig. 5). In both cases, highly structured genotypes are interpreted by a series of algorithms and data sets that introduce both nonlineariries and noise into the mapping.

In turn, these phenotypes (mathematical functions in one case and computer models in the other) are interpreted / executed and a fitness is assigned to these executional traces. We discussed above the perils of the last stage of this process, namely, that of fitness assignment yet the entire mapping must be taken into account as to be able to develop robust search algorithms. For example, the fitness function used in the protein structure energy function case must be sufficiently fine-grained as to be able to distinguished between similar, yet not identical, rankings (i.e. phenotypes). Currently, the distances between any two rankings are limited to a range of $O(n^2)$, where n is the protein structure size, and this makes distinguishing between candidate energy functions difficult. In the case of the search for alternative systems/synthetic biology models one faces the prospect of trying to fit just a few data samples with a high dimensional model that might produce good fit to the existing data but not be able to generalise to new, yet unseen, samples. Distinguishing amongst model structures (i.e. phenotypes) and

their parameters before simulating them (e.g. by partially checking the models) could be a promising way of better understanding the search space that is being explored. Thus developing effective techniques for avoiding resampling, optimising signal to noise ratios, detecting multimodality, etc are some important open challenges.

4.3 CPU hungry problems

Ideally, when evolving the energy functions for protein structure prediction one would use Turing-complete GP constructs, multiple decoys and data sets as input as well as multiple structure comparison techniques. Clearly, these requirements would make an already expensive problem even more cpu-hungry. In the case of the evolution of models for systems and synthetic biology, a quite immediate concern is that the computational bottleneck lies with the simulation of the candidate models. As biological systems are stochastic and noisy, a judicious selection of the number of replicas could substantially speed-up the evaluation stages. Moreover, a multi-stage algorithm that uses first surrogate fitness functions to zoom-in into promising model structures and then shifts to full simulations for the fine tuning of model parameters seems to be a promising avenue. Interestingly, the possibility of partial evaluation of P systems (and other executable biology) models could prove to be a promising avenue for achieving substantial speed ups. Finally, in both problems, state of the art distributed and parallel code would be critical contributors to future progress.

4.4 Human understandability and biological plausibility

In both problems, it is of critical importantance that the evolved entities are human understandable and biologically plausible. In terms of the evolution of energy functions for the protein structure prediction problem, one could refine the building blocks we have used and include other physically realistic constraints, e.g. similar to those used in [41], in the search process. In the case of the evolvability of systems biology models, our methods promote understandability through the use of clearly defined and intuitive P systems modules than can be combined into larger models. However, ensuring biological plausibility still remains a challenge. Different avenues could be considered in solving this problem, e.g., collecting a database of model composition constraints, adding a filtering stage of text mining to the evolutionary problem whereby biological papers abstracts could be scanned for the co-ocurrence of modules within papers content, and the biasing of the search process itself based on collected statistics on the co-location of biological modules and network motifs across species.

5. ACKNOWLEDGEMENTS

We would like to acknowledge projects EP/E017215/1, EP/H016597/1, BB/F01855X/1 and BB/D019613/1.

6. REFERENCES

[1] E. Alfaro-Cid, A. Esparcia-Alcazar, P. Moya, J. J. Merelo, B. Femenia-Ferrer, K. Sharman, and J. Primo. Multiobjective genetic programming approach for a smooth modeling of the release kinetics of a pheromone dispenser. In A. I. E. et. al., editor, *GECCO-2009 Symbolic regression and modeling workshop (SRM)*, pages 2225–2230, Montreal, 8-12 July 2009. ACM.

[2] U. Alon. *An Introduction to Systems Biology (Mathmatical and Computational Biology Series)*. Chapman & Hall/Crc., 2006.

[3] C. Anfinsen. Principles that Govern the Folding of Protein Chains. *Science*, 181(4096):223–30, July 1973.

[4] F. Archetti, I. Giordani, and L. Vanneschi. Genetic programming for anticancer therapeutic response prediction using the NCI-60 dataset. *Computers & Operations Research*, 37(8):1395–1405, 2010. Operations Research and Data Mining in Biological Systems.

[5] F. Archetti, I. Giordani, and L. Vanneschi. Genetic programming for QSAR investigation of docking energy. *Applied Soft Computing*, 10(1):170–182, Jan. 2010.

[6] J. N. D. Battey, J. Kopp, L. Bordoli, R. J. Read, N. D. Clarke, and T. Schwede. Automated server predictions in CASP7. *Proteins: Structure, Function, and Bioinformatics*, 69(S8):68–82, 2007.

[7] S. A. Benner and A. M. Sismour. Synthetic biology. *Nature Review, Genetics*, 6:533–543, 2005.

[8] P. W. Bridgman. *Dimensional Analysis*. New Haven: Yale University Press, 1922.

[9] H. Cao, F. Romero-Campero, S. Heeb, M. Camara, and N. Krasnogor. Evolving cell models for systems and synthetic biology. *Systems and Synthetic Biology (Springer)*, 4(1):55–84, 2009.

[10] C. A. C. Coello, D. A. Vanveldhuizen, and G. B. Lamont. *Evolutionary Algorithms for Solving Multi-Objective Problems*. Kluwer Academic Publishers, 2002.

[11] E. A. Coutsias, C. Seok, and K. A. Dill. Using quaternions to calculate RMSD. *Journal of Computational Chemistry*, 25(15):1849–1857, 2004.

[12] C. D. Errampalli and P. Quaglia. A formal language for computational systems biology. *OMICS A Journal of Integrative Biology*, 8:370–380, 2004.

[13] C. Dwork, R. Kumar, M. Naor, and D. Sivakumar. Rank aggregation methods for the Web. In *Proceedings of the 10th international conference on World Wide Web*, pages 613–622, Hong Kong, 2001. ACM.

[14] J. Fisher and T. Henzinger. Executable cell biology. *Nature Biotechnology*, 25:1239–1249, 2007.

[15] C. Garcia-Martinez, C. Lima, J. Twycross, and N. Krasnogor. P system model optimisation by means of evolutionary based search algorithms. In *Proceedings of the 2010 Genetic and Evolutionary Computation Conference (GECCO 2010)*, volume (to appear). ACM, 2010.

[16] C. García-Martínez and M. Lozano. A continuous variable neighbourhood search based on specialised eas: application to the noiseless bbo-benchmark 2009. In *GECCO '09: Proceedings of the 11th Annual Conference Companion on Genetic and Evolutionary Computation Conference*, pages 2287–2294. ACM, 2009.

[17] E. N. Gerasimov and V. N. Repko. Multicriterial optimization. *Sov. Appl. Mech.*, 14:1179–1184, 1978.

[18] M. Gheorghe, N. Krasnogor, and M. Camara. P

systems applications to systems biology. *Biosystems*, 91:435–437, 2008.

[19] D. T. Gillespie. Stochastic simulation of chemical kinetics. *Annu. Rev. Phys. Chem.*, 58:35–55, 2007.

[20] P. Hansen and N. Mladenovic. Variable neighborhood search for the $p-$median. *Location Sciences*, 5(4):207–226, 1998.

[21] S. Harding, J. Miller, and W. Banzhaf. Self modifying cartesian genetic programming: Fibonacci, squares, regression and summing. In L. Vanneschi, S. Gustafson, A. Moraglio, I. De Falco, and M. Ebner, editors, *Proceedings of the 12th European Conference on Genetic Programming, EuroGP 2009*, volume 5481 of *LNCS*, pages 133–144, Tuebingen, Apr. 15-17 2009. Springer.

[22] L. Hartwell, J. Hopfield, S. Leibler, and A. Murray. From molecular to modular cell biology. *Nature*, 402:c42–c52, 1999.

[23] H. Ishibuchi and T. Murata. Multi-objective genetic local search algorithm and its application to flowshop scheduling. *IEEE Trans. Syst. Man Cybern. C*, 28:392–403, 1998.

[24] A. Jaszkiewicz. On the performance of multiple-objective genetic local search on the 0/1 kanpsack problem-a comparative experiment. *IEEE Trans. Evol. Comput.*, EC-6:402–412, August, 2002.

[25] J.Twycross, L. Band, M. J. Bennett, J. King, and N. Krasnogor. Stochastic and deterministic multiscale models for systems biology: an auxin-transport case study. *BMC Systems Biology*, 4(:34), March 2010.

[26] W. Kabsch. A discussion of the solution for the best rotation to relate two sets of vectors. *Acta Crystallographica Section A*, 34(5):827–828, Sep 1978.

[27] E. Klipp, R. Herwig, A. Kowald, C. Wierling, and H. Lehrach. *Systems Biology in Practice: Concepts, Implementation and Application*. Wiley-VCH, Weinheim, 2005.

[28] W. R. Knight. A Computer Method for Calculating Kendall's Tau with Ungrouped Data. *Journal of the American Statistical Association*, 61(314):436–439, June 1966.

[29] V. I. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Dokl.*, 10(8):707–710, Feb. 1966.

[30] M. Lozano and C. Garcia-Martinez. Hybrid metaheuristics with evolutionary algorithms specialising in intensification and diversification: overview and progress report. *Comput. Oper Res.*, 37:481–497, 2010.

[31] R. T. Marler and J. S. Arora. Survey of multi-objective optimization methods for engineering. *Struct Mutidisc Optim*, 26:369–395, 2004.

[32] K. I. M. McKinnon. Convergence of the Nelder-Mead simplex method to a nonstationary point. *SIAM Journal on Optimization*, 9:148–158, 1999.

[33] J. Nelder and R. Mead. A simplex method for function minimization. *The Computer Journal*, 7:308–313, 1964.

[34] M. J. Pérez-Jiménez and F. J. Romero-Campero. P systems, a new computational modelling tool for systems biology. *Transactions on Computational Systems Biology VI*, pages 176–197, 2006.

[35] C. Priami. Algorithmic systems biology. *Communications of the ACM*, 52(5):80–88, 2009.

[36] G. Păun. *Membrane Computing: An Introduction*. Springer-Verlag, Berlin, 2002.

[37] A. Regev, W. Silverman, and E.Shapiro. Representation and simulation of biochemical processes using the pi-calculus process algebra. *Proceedings of the Pacific Symposium on Biocomputation*, pages 459–470, 2001.

[38] C. A. Rohl, C. E. M. Strauss, K. M. S. Misura, and D. Baker. Protein Structure Prediction Using Rosetta. In L. Brand and M. L. Johnson, editors, *Numerical Computer Methods, Part D*, volume Volume 383 of *Methods in Enzymology*, pages 66–93. Academic Press, Jan. 2004.

[39] F. Romero-Campero, H.Cao, M. Camara, and N. Krasnogor. Structure and parameter estimation for cell systems biology models. In M. K. et.al, editor, *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2008)*, pages 331–338. ACM Publisher, 2008. This paper won the Best Paper award at the Bioinformatics track.

[40] F. J. Romero-Campero, J. Twycross, M. Camara, M. Bennett, M. Gheorghe, and N. Krasnogor. Modular assembly of cell systems biology models using p systems. *International Journal of Foundations of Computer Science*, 20:427–442, 2009.

[41] M. Schmidt and H. Lipson. Distilling free-form natural laws from experimental data. *Science*, 324:81–85, 2009.

[42] K. T. Simons, I. Ruczinski, C. Kooperberg, B. A. Fox, C. Bystroff, and D. Baker. Improved recognition of native-like protein structures using a combination of sequence-dependent and sequence-independent features of proteins. *Proteins: Structure, Function, and Genetics*, 34(1):82–95, 1999.

[43] J. Sola and J. Sevilla. Importance of input data normalization for the applicatiob of neural networks to complex industrial problems. *IEEE Transactions on Nuclear Science*, 44(3):1464–1468, 1997.

[44] M. Stout, J. Bacardit, J. D. Hirst, and N. Krasnogor. Prediction of recursive convex hull class assignments for protein residues. *Bioinformatics*, 24(7):916–923, 2008.

[45] P. Widera, J. Garibaldi, and N. Krasnogor. Evolutionary design of the energy function for protein structure prediction. In *IEEE Congress on Evolutionary Computation 2009*, pages 1305–1312, Trondheim, Norway, May 2009.

[46] P. Widera, J. Garibaldi, and N. Krasnogor. GP challenge: evolving energy function for protein structure prediction. *Genetic Programming and Evolvable Machines*, 11(1):61–88, March 2010.

[47] S. Wu, J. Skolnick, and Y. Zhang. Ab initio modeling of small proteins by iterative TASSER simulations. *BMC Biol*, 5(1):17, May 2007.

[48] Y. Zhang, A. Kolinski, and J. Skolnick. TOUCHSTONE II: A New Approach to Ab Initio Protein Structure Prediction. *Biophys. J.*, 85(2):1145–1164, Aug. 2003.