Building Portfolios for the Protein Structure Prediction Problem

Alejandro Arbelaez
Microsoft-INRIA joint-lab,
Orsay, France
alejandro.arbelaez@inria.fr
, Youssef Hamadi
Microsoft Research,
Cambridge, United Kingdom,
LIX Ecole Polytechnique,
F-91128 Palaiseau, France
youssefh@microsoft.com
and Michele Sebag
Project-team TAO,
INRIA Saclay Ile-de-France,
LRI (UMR CNRS 8623),
Orsay, France
michele.sebag@inria.fr

Abstract

This paper, concerned with the protein structure prediction problem, aims at automatically selecting the Constraint Satisfaction algorithm best suited to the problem instance at hand. The contribution is twofold. Firstly, the selection criterion is the quality (minimal cost) in expectation of the solution found after a fixed amount of time, as opposed to the expected runtime. Secondly, the presented approach, based on supervised Machine Learning algorithms, considers the original description of the protein structure problem, as opposed to the features related to the SAT or CSP encoding of the problem.

1 Introduction

The protein structure prediction problem has been widely studied in the field of bioinformatics, because the 3D conformation of a given protein helps to determine its function. This problem is usually tackled using simplified models such as HP-models in [2] and a constraint logic programming approach in [5], however even considering these abstractions the problem is computationally very difficult and traditional strategies cannot reach a solution within a reasonable time. Also, there has been several attempts to predict the structure and proteins fold using well known machine learning techniques.

In this paper, we propose to use machine learning to automatically select the most promising Constraint Optimization algorithm for the protein structure prediction problem. In this context, proteins are represented as a feature vector in $\mathbb{R}^d$ and the algorithm selection process is based on a well known machine learning technique called *decision tree* which predict the most appropriate variable/value selection strategy used by a branch-and-bound algorithm in order to determine the three dimensional (3D) conformation of a given protein.

Unlike other portfolio-based selection approaches [7] which select the algorithm which minimizes the expected runtime, our work selects the strategy which minimizes the expected cost of the solution found after a fixed amount of time. To the best of our knowledge, this is the first work which performs algorithm selection in an optimization setting. Moreover, and unlike previous works which extract the features exploited during machine learning from the SAT or CSP encoding of the problem, our work uses features directly formulated in the application domain. Again, to the best of our knowledge it is the first time that domain-based features are used to predict the performance of a search algorithm.

The paper is organized as follows. Background material is presented in Section 2. Section 3 presents the general idea of algorithms portfolio. Section 4 shows the features or attributes used to describe proteins. Section 5 reports our experimental validation and Section 6 presents some concluding remarks and future research directions.

2 Background

2.1 Constraint Optimization Problems

A Constraint Optimization Problem (COP) is a tuple (X, D, C, f) where, X represents a set of variables, D a set of associated domains (i.e., possible values for the variables), C a finite set of constraints and $f(X)$ is a function to optimize. Solving a COP involves finding a value for each variable whose $f(X)$ is maximal (or minimal). A backtracking branch-and-bound algorithm is usually used to tackle COPs, at each node in the search tree a variable/value pair is used in cooperation with a look-ahead algorithm which narrows the domain of the variables.

In this paper, we consider six well known variable selection heuristics. The *lexico* heuristic selects the first unassigned variable (from left to right) in the list of decision variables, *mindom* selects the variable with minimal domain, *wdeg* [3] selects the variable which is involved in the highest number of failed constraints, *dom-wdeg* selects the variable which minimizes the ratio $\frac{dom}{wdeg}$, *impacts* [9] selects the variable/value pair which maximizes the reduction of the remaining search space and *domFD* [1] selects the variable that minimizes the ratio $\frac{dom}{FD}$ where *FD* represents the total number of weak functional dependencies of a given variable.

2.2 The protein structure prediction problem

The protein structure prediction problem is well known in computational biology and is currently considered as one of the "grand challenges" in this field. Broadly speaking the problem consists in finding the 3D conformation (so-called tertiary structure) of a protein defined by its primary structure or a sequence of residues $S = \{s_1, s_2, \ldots, s_n\}$ where each residue s_i of the sequence represents one of the 20 amino acids. The ternary structure is often defined by the minimal energy conformation.

This problem has been previously studied in [6] using a constraint programming based model. In this model, each amino acid is seen as a single atom unit and two consecutive amino acids in the sequence are separated by a fixed distance also known as a lattice unit. The energy is defined by minimizing the following formula:

$$E(w) = \sum_{1 \leq i < n} \sum_{i+2 \leq j \leq n} contact(w(i), w(j)) \times Pot(s_i, s_j)$$

where, $w(i)$ denotes the current position of the amino acid s_i in the three dimensional space of a given amino acid, *contact* is 1 iff two amino acids are immediate neighbors in the three dimensional cube (or lattice) and not sequential in the primary structure, otherwise contact is set to 0, and *pot* defines the energy contribution of two adjacent residues. It is also important to note that some other lattice models have been proposed such as [2] where each amino acid in the sequence is translated from the 20 symbols alphabet into a two symbols alphabet (i.e., hydrophobic (H) and polar (P)).

2.3 Supervised machine learning

Supervised Machine Learning exploits data labelled by the expert to automatically build hypothesis emulating the expert's decisions. Formally, a learning algorithm processes a training set $\mathcal{E} = \{(x_1, y_1), ..., (x_n, y_n)\}$ where x_i is the example description (e.g., a vector of features, $\Omega = \mathbb{R}^d$) and y_i is the associated output. The output can be a numerical value (i.e., regression) or a class label (i.e., classification).

The learning algorithm outputs a hypothesis $f : \Omega \mapsto Y$ associating to each example description x a desirable output y.

3 Algorithm portfolios

Many portfolio of algorithms have been proposed to select the best technique in order to process a given instance according to its features or descriptors. A classical portfolio is learned by taking into account the overall computational time to solve a set of problem instances. For instance SATzilla [10] a well known portfolio for SAT problems builds a regression model in order to learn the solving time of each constitutive SAT solver. In this way, once an unseen instance arrives SATzilla selects the algorithm with minimal expected run-time.

Solving a constraint optimization problem involves finding the best solution and prove that the solution is the optimal one. Unfortunately, in many cases this process cannot be completed within a reasonable amount of time and the system must provide to the user the best solution found so far. Following this idea, building the portfolio using algorithm's runtime is not an alternative. A solution would be building the portfolio taking into account the quality or cost of the solution found after some fixed amount of computational time (e.g., time-out parameter). In the following, we are going to use this technique to predict the cost of the solution found after 5 minutes for the protein structure prediction problem.

4 Features

The vector of feature was extracted from the extensive machine learning literature on protein fold prediction [8]. In order to build the feature set, every amino acid in the primary structure is replaced by the index 1, 2 or 3 according to the group it belongs to, i.e., Hydrophobicity, Volume, Polarity and Polarizability (see Table 1). For instance, the following sequence RSTVVH is encoded as 122332 based on the hydrophobicity attribute. This encoding is used to compute the following set of descriptors:

- Composition: 3 descriptors representing the percentage of each group in the sequence.
- Transition: 3 descriptors representing the frequency with which a residue from $group_i$ is followed by a residue from $group_j$ (or vice-versa).
- Distribution: 15 descriptors representing the fraction in the sequence where the first residue, 25%, 50%, 75% and 100% of the residues are contained for each encoding in Table 1.

In total the feature set is a composition of 105 descriptors: 84 ((15+3+3)×4) according to Table 1, 20 descriptors which represent the proportion of each amino acid in the sequence, and finally the size of the sequence.

Attribute	Group 1	Group 2	Group 3
Hydrophobicity	R,K,E,D,Q,N	G,A,S,T,P,H,Y	C,V,L,I,M,F,W
Volume	G,A,S,C,T,P,D	N,V,E,Q,I,L	M,H,K,F,R,Y,W
Polarity	L,I,F,W,C,M,V,Y	P,A,T,G,S	H,Q,R,K,N,E,D
Polarizability	G,A,S,D,T	C,P,N,V,E,Q,I,L	K,M,H,F,R,Y,W

Table 1: amino acid feature's group

5 Experiments

In this paper, we used the Gecode model proposed in [4]. All algorithms (see section 2.1) are home-made implementations integrated into the Gecode-2.1.1[1] constraint solver. We experimented with 400 random sequences of sizes ranging from 20 to 99, and performed 10-fold cross validation to evaluate the model. The timeout of each run was set to 5 minutes, which means that the objective of the machine learning part was to predict the strategy which would provide a solution of minimal cost after 5 minutes.

Initial experiments suggested that *lexico* is a powerful heuristic for the protein structure prediction problem, therefore we explored an extension of traditional variable selection algorithms, this novel version is presented as follows:

1. Select the first unassigned variable X_i if and only if X_{i+1} is assigned.
2. If the previous step cannot be satisfied, then select the variable according to a given heuristic criteria (e.g., *dom-wdeg*, *domFD*, etc)

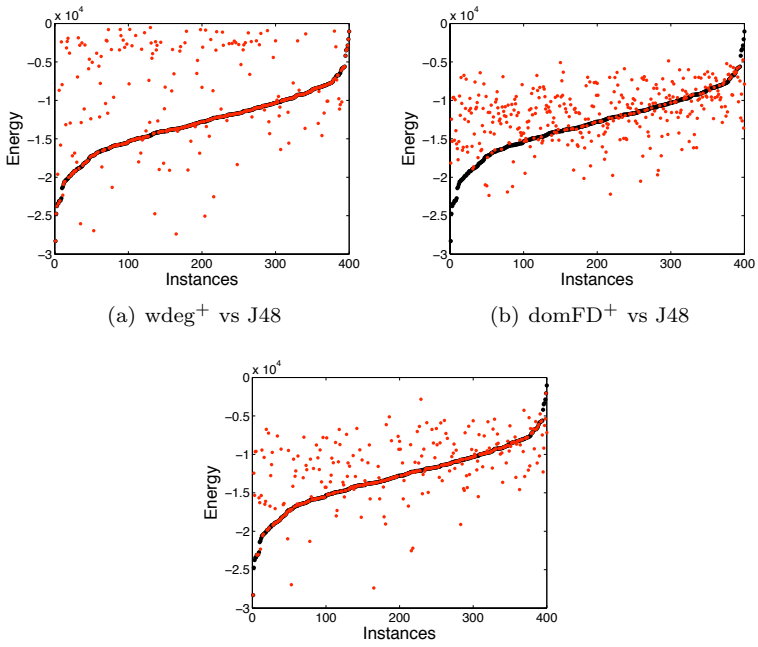
The algorithms which follow the strategy mentioned above would be named as: *dom-wdeg*$^+$, *wdeg*$^+$, *domFD*$^+$ and *impacts*$^+$. Overall, we are considering a set of 10 variable selection heuristics $\mathcal{H}_{var}$ = { *lexico*, *mindom*, *dom-wdeg*, *wdeg*, *domFD*, *impacts*,*dom-wdeg*$^+$, *wdeg*$^+$, *domFD*$^+$, *impacts*$^+$ } and 2 value selection algorithms $\mathcal{H}_{val}$ = { *min-val*, *med-val* } would lead to 18 heuristics candidates ($8 \times 2 + 2$) (notice that *impacts* and *impacts*$^+$ are variable-value selection techniques). Nevertheless the majority of the candidates are low-quality heuristics that were almost always dominated by the top heuristics. In this way, after eliminating these weak ones, the portfolio is build on top of the following heuristic set $\mathcal{H}$ = { $\langle$*lexico*,*min-val*$\rangle$, $\langle$*domFD*$^+$,*med-val*$\rangle$, $\langle$*wdeg*,*med-val*$\rangle$, $\langle$*wdeg*$^+$,*med-val*$\rangle$ }. A fixed time limit of 5 minutes was used for each experiment.

In this paper, we experimented with the following learning schemes:

- J48[2]: weka implementation of the C4.5 algorithm. Although J48 supports continuous features values we experimentally found that including a feature discretization step improved the accuracy of the machine learning algorithm.
- SATzilla based approach: We used the code proposed in [7] to build the linear regression model, it includes an important feature pre-processing phase (i.e., pairwise feature composition and forward feature selection).

Fig 1 shows our overall experimental results. Each black point represents the performance of J48-portfolio for a given instance and each red point represents the performance of each

[1] www.gecode.org
[2] www.cs.waikato.ac.nz/ml/weka



(a) wdeg$^+$ vs J48

(b) domFD$^+$ vs J48

(c) SATzilla vs J48

Figure 1: Overall evaluation

comparative algorithm (i.e., *wdeg*$^+$, *domFD*$^+$ and SATzilla-based portfolio) for a given instance. For analysis purposes, data have been sorted according to the performance of J48-portfolio. Notice that since the optimization goal is to find the minimal energy configuration, red points above the black ones indicate that J48-portfolio is better.

Fig 1(a) shows the performance of *wdeg*$^+$ against the portfolio, in this figure J48-portfolio is better than *wdeg*$^+$ in 110 instances (resp. worse in 43 instances). Fig 1(b) shows the performance of *domFD*$^+$ against J48-portfolio, here J48-portfolio is better in 231 instances and worse in 127 instances, and finally Fig 1(c) shows the performance of J48-portfolio against the SATzilla based portfolio, in this case J48-portfolio is better than SATzilla-portfolio in 127 instances and worse in 66 instances.

6 Conclusions and future work

In this paper, we have studied the application of Machine learning techniques to build algorithms portfolios in the context of the protein structure prediction problem, we have shown that using machine learning might help to select promising algorithms improving the overall performance of the system.

Currently, we manually select the algorithms of the portfolio. Part of our future work consists in automatically selecting algorithms using racing techniques (i.e., F-RACE) during the training phase, the idea would be to remove the algorithms that are statistically worse than the rest.

References

[1] Alejandro Arbelaez and Youssef Hamadi. Exploiting weak dependencies in tree-based search. In *ACM Symposium on Applied Computing (SAC)*, pages 1385–1391, Honolulu, Hawaii, USA, March 2009. ACM.

[2] Rolf Backofen and Sebastian Will. Fast, constraint-based threading of HP-sequences to hydrophobic cores. In *CP*, volume 2239 of *LNCS*, pages 494–508, Paphos, Cyprus, Nov 2001. Springer.

[3] Frederic Boussemart, Fred Hemery, Christophe Lecoutre, and Lakhdar Sais. Boosting systematic search by weighting constraints. In *ECAI*, pages 146–150, Valencia, Spain, Aug 2004. IOS Press.

[4] Raffaele Cipriano, Alessandro Dal Palù, and Agostino Dovier. A hybrid approach mixing local search and constraint programming applied to the protein structure prediction problem. In *Workshop on Constraint Based Methods for Bioinformatics (WCB)*, Paris, France, 2008.

[5] Alessandro Dal Palù, Agostino Dovier, and Federico Fogolari. Protein folding in clp(fd) with empirical contact energies. In *CSCLP*, volume 3010 of *LNCS*, pages 250–265, Budapest, Hungary, June 2003. Springer.

[6] Alessandro Dal Palù, Agostino Dovier, and Enrico Pontelli. A constraint solver for discrete lattices, its parallelization, and application to protein structure prediction. *Softw. Pract. Exper.*, 37(13):1405–1449, 2007.

[7] Frank Hutter, Youssef Hamadi, Holger H. Hoos, and Kevin Leyton-Brown. Performance prediction and automated tuning of randomized and parametric algorithms. In *CP*, volume 4204 of *LNCS*, pages 213–228, Nantes, France, Sept 2006. Springer.

[8] Nikhil R. Pal and Debrup Chakraborty. Some new features for protein fold prediction. In *ICANN*, volume 2714, pages 1176–1183, Istanbul, Turkey, June 2003. Springer.

[9] Philippe Refalo. Impact-based search strategies for constraint programming. In *CP*, volume 2004 of *LNCS*, pages 557–571, Toronto, Canada, Sept 2004. Springer.

[10] Lin Xu, Frank Hutter, Holger H. Hoos, and Kevin Leyton-Brown. The design and analysis of an algorithm portfolio for sat. In *CP*, volume 4741 of *LNCS*, pages 712–727, Providence, RI, USA, Sept 2007. Springer.