Automated design of energy functions for protein structure prediction by means of genetic programming and improved structure similarity assessment

Paweł Widera, BSc, MSc

Thesis submitted to the University of Nottingham
for the degree of Doctor of Philosophy

March 2010

Abstract

The process of protein structure prediction is a crucial part of understanding the function of the building blocks of life. It is based on the approximation of a protein free energy that is used to guide the search through the space of protein structures towards the thermodynamic equilibrium of the native state. A function that gives a good approximation of the protein free energy should be able to estimate the structural distance of the evaluated candidate structure to the protein native state. This correlation between the energy and the similarity to the native is the key to high quality predictions.

State-of-the-art protein structure prediction methods use very simple techniques to design such energy functions. The individual components of the energy functions are created by human experts with the use of statistical analysis of common structural patterns that occurs in the known native structures. The energy function itself is then defined as a simple weighted sum of these components. Exact values of the weights are set in the process of maximisation of the correlation between the energy and the similarity to the native measured by a root mean square deviation between coordinates of the protein backbone.

In this dissertation I argue that this process is oversimplified and could be improved on at least two levels. Firstly, a more complex functional combination of the energy components might be able to reflect the similarity more accurately and thus improve the prediction quality. Secondly, a more robust similarity measure that combines different notions of the protein structural similarity might provide a much more realistic baseline for the energy function optimisation.

To test these two hypotheses I have proposed a novel approach to the design of energy functions for protein structure prediction using a genetic programming algorithm to evolve the energy functions and a structural similarity consensus to provide a reference similarity measure. The best evolved energy functions were found to reflect the similarity to the native better than the optimised weighted sum of terms, and therefore opening a new interesting area of research for the machine learning techniques.

Part of the work described in this dissertation has been a topic of the following publications and international talks given by the author:

Journal articles

P. Widera, J.M. Garibaldi, N. Krasnogor, "GP challenge: evolving the energy function for protein structure prediction", Genetic Programming and Evolvable Machines 11(1), p.61-88, 2010

Conference papers

P. Widera, J.M. Garibaldi, N. Krasnogor, "Evolutionary design of the energy function for protein structure prediction", IEEE Congress on Evolutionary Computation, p.1305-1312, Trondheim, Norway, May 2009

Talks

"Structure similarity consensus as a measure of quality in protein structure prediction" at *EURO XXIII*, Bonn, 5–8 July 2009

"Evolutionary design of the energy function for protein structure prediction" at *IEEE Congress on Evolutionary Computation*, Trondheim, 18–21 May 2009

"Evolving energy function for the protein structure prediction" at *Mini EURO Conference on Computational Biology, Bioinformatics and Medicine*, Rome, 15–17 September 2008

"Metaheuristic approach to protein structure comparison" at *EURO XXII*, Prague, 8–11 July 2007

"The Max-CMO problem and its representation" at *5th East Midlands Proteomics Workshop* (rewarded with the Best Short Presentation Award), Nottingham, 2006-11-15

"The Max-CMO problem and its dual representation" at *EURO XXI*, Reykjavík, 2006-07-03

Acknowledgements

I am very grateful to both my supervisors for all the help and support I received from them during my PhD studies in Nottingham. Dr Natalio Krasnogor taught me how to do science, guided me through all the ups and downs and was always ready for another conversation about the life, universe and all the rest. Dr Jonathan Garibaldi dealt faultlessly with all the organisational matters, provided extra funding when needed and gave me many opportunities to present my research to a broader audience. It was a pleasure to work with you both!

Many thanks to my examiners Dr Colin Johnson and Prof. Graham Kendall for the useful comments and a very enjoyable viva experience.

I wish to thank Dr Daniel Barthel, for the fruitful cooperation on his ProCKSI server that made part of my experiments possible and the people maintaining the HPC facility without which my CPU hungry experiments would never end on time.

A big thank you also goes to all the people involved in the free software movement for providing me with great tools to do science, especially to Richard Stallman and the Free Software Foundation who constantly watch over the user freedoms in this fast changing digital world.

In addition, I would like to thank Yang Zhang, the author of I-TASSER, for making the decoys data available online and for explaining the details of I-TASSER energy terms implementation.

I also very much appreciate the funding from the European Community without which this research would not be possible. My work was supported by the Marie Curie Action MEST-CT-2004-7597 under the Sixth Framework Programme of the European Community.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background and Motivation

Proteins are the building blocks of life. There is almost no cell activity in which proteins do not take part. The function of an individual protein depends on its three dimensional structure, or more simply, its shape. The knowledge of a protein structure is therefore crucial in protein research, both in experiment planning and in results interpretation.

For example, the catalysis function of an enzyme is carried out by a small part of the protein chain, called the active site, exposed on the surface of a protein. Interactions with other proteins, nucleic acids or molecules being inhibitors or activators are also restricted to specific protein surface areas. As a result, designing drugs that would interact with a protein surface and modify its undesirable behaviour or localising protein structural mutations responsible for diseases is only possible if the protein structure is known.

Yet, there are 30 000 000 known protein sequences but the structure is known only for 60 000 of them, just 0.2% (as of January 2010 [URL10b]). This disproportion is caused by the expensive and lengthy process of protein structure physical examination by X-ray crystallography or nuclear magnetic resonance (NMR) spectroscopy, as well as by the advances in sequencing techniques which are now much more efficient than what was used in the past (for example to complete the Human Genome Project in 2003) [BDM+08].

As even a rough estimation of a protein structure is usually more useful than no structure at all, a number of protein structure prediction (PSP) methods have been developed in the last two decades. Recent advances are a result of the application of machine learning and pattern matching algorithms, which has led to an increase in prediction accuracy for comparative modelling and fold recognition. New prediction methods are able to determine which kind of fold is most likely to be adopted by a given sequence fragment using relevant homologous structures (similar on a sequence level) from a protein data bank of known structures [Bou03].

However, modelling the structure of a protein with a sequence not similar to any of the proteins with a determined structure, known as *ab initio* or *de novo* structure prediction, still remains very challenging. It requires a different approach based on the physical principles of folding (the protein inter-atomic interactions) and the structural model has to be built from "scratch".

1.2 Overview of the Problem

The most widely accepted hypothesis explaining the process of protein folding was formulated by Christian Anfinsen. In a Nobel prize winning experiment he found that a refolded protein always forms the same native structure [Anf73]. He concluded that a "folding algorithm" has to exist that doesn't use any information, aside from what is already contained in a protein sequence. To explain this phenomena Anfinsen formulated a Thermodynamic Hypothesis that described the native configuration as being in a thermodynamic equilibrium and explained the process of folding as a minimisation of the protein's free energy.

De novo protein structure prediction methods define this energy as a function of structure and use it to distinguish between good and bad candidate structures. In state-of-the-art PSP methods the energy function is expressed as a weighted sum of several energy terms designed by human experts. The weights are optimised so that the energy of a candidate protein structure reflects its structural similarity to the protein's real native state. As a result, the lower the energy of a model, the more accurate it is, i.e. more similar to the native structure. However, in practice the correlation between the energy function and the similarity to the native structure is not significant enough to ensure high quality predictions.

1.3 Aims and Scope

The goal of this dissertation is to examine the computational models of energy functions used in the state-of-the-art *de novo* protein structure prediction methods and discuss and experimentally verify some of the possible improvements. Firstly, this work argues that simple measures of the protein structural similarity are not robust enough and hypothesizes that replacing them with a consensus measure in the process of energy function optimisation may lead to higher accuracy of the predictions made using this energy function (hypothesis **H1**). Secondly, it describes a novel approach to the design of the energy functions that is based on the use of a genetic programming (GP) algorithm to evolve the protein structure energy function and hypothesizes that evolved energy functions may outperform the previously used linear combination of energy terms leading to better energy–similarity correlation and thus improving the prediction quality (hypothesis **H2**).

1.4 Methodological Aspects

To accomplish these goals this dissertation includes four different types of analysis. In Chapter 2 a deep analysis of the literature focused on the internal design of the most successful protein structure predictors is used to identify their weak points and possible improvements in that field. The complexity analysis is used in Chapter 3 to determine the theoretical computational gain of the proposed tabu search based heuristic. In Chapters 4–5 a statistical analysis is used to measure dependency between comparison measures and the energy/similarity correlations. Finally, experimental approaches are used through Chapters 3–6 to investigate the validity of the two proposed hypotheses: applicability of the similarity consensus (**H1**) and the evolvability of the energy functions with GP (**H2**).

1.5 Structure of the Dissertation

The dissertation is organised as follows. The next chapter explains a chemical model of a protein structure and the principles that drive the process of protein folding. The problems of protein structure comparison and prediction are introduced. Computational methods for both are summarised and a critical analysis of the related work is done. A detailed description of relevant structural comparison methods used in Chapter 4 and the energy terms used in Chapter 5 is given. Also an overview of the metaheuristic optimisation and in particular the algorithms used in Chapter 3 (tabu search) and Chapters 5–6 (genetic programming) is provided.

Chapter 3 describes the Max-CMO problem as a robust measure of protein structure similarity and contains a proposition of an efficient tabu search based heuristic algorithm using a novel representation of the problem solution. This measure is later used in constructing the similarity consensus in Chapter 4.

Chapter 4 investigates the relation between a number of structural similarity measures and defines a consensus measure based on them. The structural consensus is later used in the final GP experiments (in Chapter 6) to verify if more robust measure of similarity will lead to an increased prediction quality (hypothesis **H1**). Additionally, a prototype cloud computing application using an alternative (decoy specific) set of similarity measures is presented.

Chapter 5 contains the analysis of energy/similarity correlations for the state-of-the-art protein structure prediction methods. Furthermore, it describes the design of a genetic programming algorithm and the experimental setup used to evolve the protein energy functions. The best results are compared to a baseline random walk experiment.

Chapter 6 provides extended analysis of the evolution process through the population diversity statistic. In addition it contains a comparison of the best evolved functions to a linear combination of terms optimised with a Nelder-Mead downhill simplex method to verify whether the GP algorithm is able to improve over it (hypothesis **H2**). Furthermore it reports the result of the GP based on the consensus similarity, discusses the limitation of the method and suggests possible improvements to the evolvability of energy functions.

Chapter 7 contains concluding remarks and discussion on validity of the proposed hypotheses and possible future work.

1.6 Typographic Conventions

To make reading easier a few typographic conventions are used throughout this thesis. A bold face is used to mark the occurrence of a new term at the place of its definition. The index of all keywords is included at the end of the document. The bold face is also used to place emphasis on names of the methods or algorithm variants.

Vertical bars on the margin (as the one on the left) are used in the next chapter to mark key facts and observations particularly important for understanding the motivation for the presented work and the main principles of the proposed solution. In following chapters it is used to highlight the main contribution and the major findings of a chapter.

1.7 Main Contributions

The main contribution of this dissertation is the introduction of a novel approach to the automated design of energy functions for protein structure prediction. This approach is a result of the critical analysis of the state-of-the-art protein structure predictors and the structural comparison methods and is directly addressing two issues: (1) simplicity of the optimisation methods used to construct the energy functions from the expert designed energy terms and (2) robustness of the structural similarity measure used as a reference in the optimisation.

The proposed solution is composed of three main elements, each described in a separate chapter:

- fast tabu search heuristic for protein structure comparison solving the Max-CMO problem using a novel representation of the problem solution based on line graphs, that led to $O(n)$ performance improvement in the computational cost of the solution evaluation and feasibility check,
- consensus measure of a structural similarity representing a fusion of different similarity aspects contributed by the individual structure comparison methods, which has many uses including design of energy functions, evaluation of protein models or discovery of the structural relations proteins,
- genetic programming algorithm to evolve novel forms of energy function for protein structure prediction that outperform the optimised weighted sum of terms used in state-of-the-art protein structure predictors.

Although the evolutionary algorithms have been used in a number of occasions for protein structure prediction, protein structure comparison and related problems, to the best of the author's knowledge, this is the first time that a machine learning technique based on genetic programming has been used to design the energy functions for the *de novo* protein structure prediction. Similarly, the use of the consensus measure in place of the root mean square deviation in a process of the energy design is an original contribution. Moreover, to the best of the author's knowledge, the use of cloud computing environment to implement a scalable web service for protein structure comparison is also a pioneering work.

Furthermore, the problem of automated design of the energy functions used in protein structure prediction represents a unique opportunity and a challenge for the GP/machine learning community, therefore a special attention has been put to make all the data used in this dissertation publicly available to facilitate a community-wide challenge.

Chapter 2

Background and Related Work

2.1 Proteins

Proteins are polymers, linear chains made of series of 20 different amino acids encoded in the genetic material (DNA or RNA sequence). Each **amino acid** includes an $\alpha - carbon$ (C_α) with bonds to amino (NH) and carboxyl (COOH) groups and a variable **side chain** (different for each type of amino acid). The amino acids in the protein chain are connected with peptide bonds (CO-NH) formed between the amino and carboxyl groups, as shown at the top of Figure 2.1. These bonds are formed in the process of polymerization. Because in this reaction a molecule of water is lost, amino acids in the protein chain are also called **residues**. The linked carbon, oxygen and nitrogen atoms form a **protein backbone**. This backbone forms several repeating local structures such as alpha helices, beta sheets or loops, known as **secondary structure** elements (see Figure 2.1). These elements and their spatial interrelations define the **tertiary structure** of a protein (a fold). An independent part of a protein chain that folds into a distinct structural region is called a **protein domain**. Average length of a domain is around 100 amino acids [WMBB00].

A protein, under physiological conditions, spontaneously folds into a specific shape known as its **native state**. The geometric pattern of the native state determines protein biological activity: macroscopic properties, behaviour and function.

2.1.1 Protein Folding

Anfinsen, in his Nobel price winning experiment, proved that protein ribonuclease can be reversibly denatured/renatured in a test tube [Anf73]. A direct conclusion from this experiment was that if a protein always folds into the same native structure, an algorithm of protein folding has to exist that drives the folding towards the native state of minimal or near to minimal free energy. Moreover, this algorithm doesn't use any information, aside from what is already contained in the protein sequence.

Folding Funnel

The energy landscape of protein folding under the solution condition in a cell may be rough with several local minima traps. Despite of this, the native state of a protein

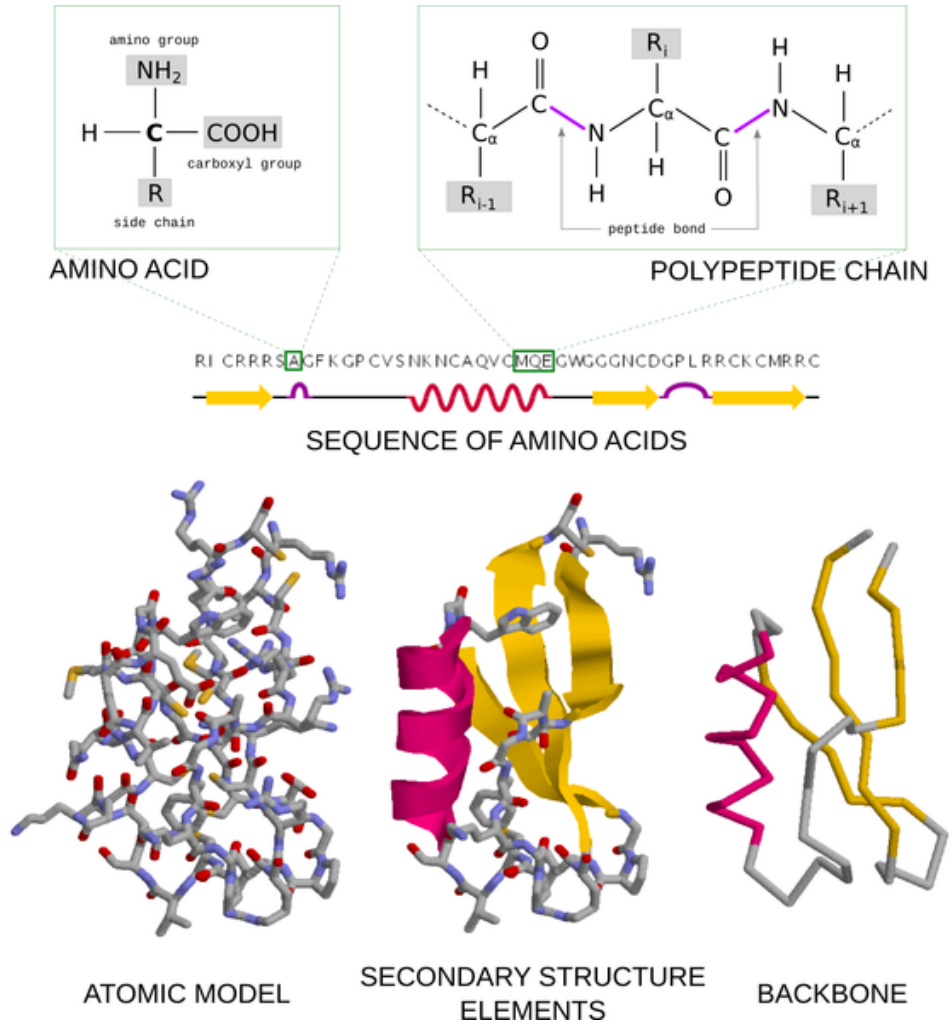

Figure 2.1: The atomic structure of a single amino acid and the chain of amino acids is shown on top of a protein sequence. An all-atom representation of the protein colored by atom type, cartoon representation of secondary structure elements and a simplified representation illustrating the protein backbone colored by secondary structure is shown below.

corresponds to the landscape energetic minimum and the **folding process** is a roll down a free energy hill to the bottom [DC97]. Figure 2.2 presents a variant of the protein energy landscape known as a **folding funnel**, where the well-defined single native state (N) corresponds to the energy minimum surrounded by steep slopes [DSS$^+$00].

Levinthal's Paradox

The process of a protein spontaneous self-assembly into the native state is surprisingly efficient when the number of possible fold configurations is taken into account. This divergence between expected and current folding speed was first shown by Levinthal [Lev69] and is known as Levinthal's paradox.

Let's assume we have a sequence of 101 amino acids and that each bond connecting two of them can have 3 possible states. The number of possible configurations is then equal to $3^{100} = 5 \times 10^{47}$. If we assume a protein sampling rate of 10^{13} configurations per second ($0.1ps$ per configuration) it would be possible to search 3×10^{20} configurations per year.

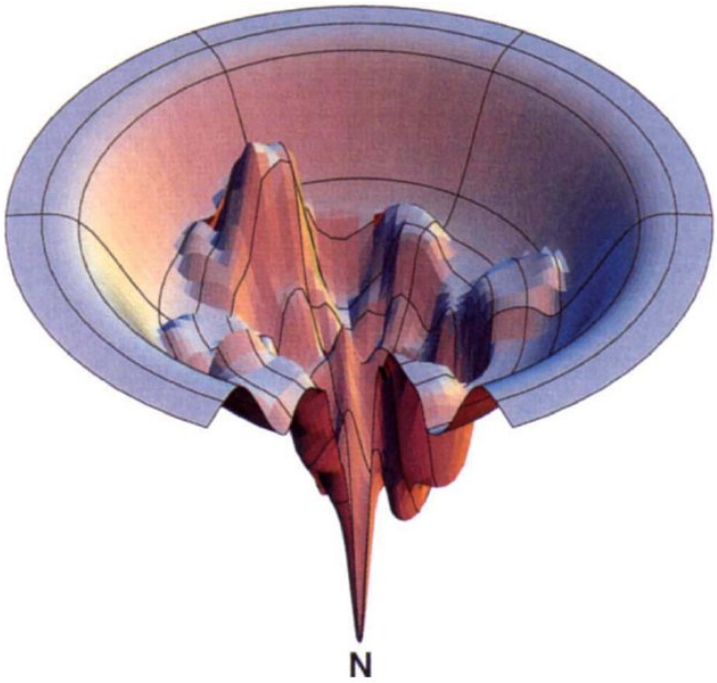

Figure 2.2: Folding funnel [DC97].

Consequently, to try them all, 10^{27} years would be needed, which is much greater than the age of the universe (13.7×10^9). Yet proteins fold in seconds, thus they do not perform an exhaustive search of this space.

The common explanations to this paradox refer either to protein as evolutionary specialised to fold rapidly (other amino acid sequences simply did not survive) or to the lack of stability of some proteins [Hay98]. It has been observed, that the native configuration may not be in a state of the lowest free energy. Moreover, some folds may be erroneous and the faulty proteins are removed from the cell in a process of proteolysis. The absolute stability is apparently not required, a protein has to last only long enough to perform its function.

Molecular Dynamics

There are two groups of forces affecting protein molecules in a folding process: bonded and non-bonded forces. **Bonded forces** arise from existence of covalent bond between two atoms sharing an electron or ionic bond if the electron is transferred between the atoms. **Non-bonded forces** arise from distance interactions between atoms. A function which is a sum of forces for all atoms and describes a potential energy of a whole system of particles is called a **force field**.

Bonded forces are relevant to stretching, bending and rotating. Stretching (see Equation 2.1) and bending potentials (see Equation 2.2) are based on the elastic spring model. They are parametrised with the equilibrium values (l_0 and θ_0) and bond/angle stiffness (k_l and k_θ where $k_\theta \ll k_l$).

$$V(l) = \frac{1}{2} k_l (l - l_0)^2 \tag{2.1}$$

$$V(\theta) = \frac{1}{2} k_\theta (\theta - \theta_0)^2 \tag{2.2}$$

Bond rotation potential is modeled as cosinusoid function of parametrized amplitude (V_ω), periodicity (n) and phase shift (ϕ):

$$V(\omega) = \frac{1}{2} V_\omega [1 + cos(n\omega - \phi)] \tag{2.3}$$

Non-bonded forces are relevant to steric effects, van der Waals' interactions and electrostatic charge. Steric interactions are caused by Pauli short range repulsion (there is an energetic cost of the overlap of electron clouds). Van der Waals' forces are long range attractive forces that arise from polarisation of molecules. They are modelled together with **Lennard-Jones potential**:

$$V(r) = \epsilon \left[\left(\frac{r_{min}}{r} \right)^{12} - 2 \left(\frac{r_{min}}{r} \right)^{6} \right] \tag{2.4}$$

where r_{min} is a distance at the minimum of potential. Electrostatic charge is expressed by Coulomb's law:

$$V(r) = \frac{1}{4\pi\varepsilon_0} \frac{q_1 q_2}{r} \tag{2.5}$$

The full force field that can reproduce the basic features of protein energy landscapes at an atomic level of detail is then represented by the following potential energy function:

$$\begin{aligned} &\sum_{i}^{bonds} V(l_i) + \sum_{i}^{angles} V(\theta_i) + \sum_{i}^{rotations} V(\omega_i) + \\ &+ \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} \left\{ \epsilon_{ij} \left[\left(\frac{r_{min_{ij}}}{r_{ij}} \right)^{12} - 2 \left(\frac{r_{min_{ij}}}{r_{ij}} \right)^{6} \right] + \frac{q_i q_j}{4\pi\varepsilon_0 r_{ij}} \right\} \end{aligned} \tag{2.6}$$

Since early '80 several families of force fields has been developed. According to [PC03] the top three families are: AMBER (Assisted Model Building and Energy Refinement), CHARMM (Chemistry at HARvard Macromolecular Mechanics), OPLS (Optimized Potential for Liquid Simulations). The functional forms of these force fields are very similar to the one shown in Equation 2.6. Each family contains several sets of parameters representing a single force field designed for certain type of molecules. In protein folding simulation AMBER99, CHARMM22 and OPLS-AA are commonly used [Mac04].

Although the molecular dynamics simulation software implementing both AMBER [CCD+05] and CHARMM [BBO+83] is proprietary, there are also several free alternatives: CHARMM based NAMD (NAnoscale Molecular Dynamics) [PBW+05], TINKER [URLb] or GROMACS (Groningen Machine for Chemical Simulations) [VDSLH+05]. The latter two are used by the Folding@home project [SP00].

Since the computational cost of these all-atom energy functions is very high, their practical applicability is limited to the molecular dynamic simulations of short protein chains. For example, massively distributed projects such as Folding@home [PBC+03] or Rosetta@home [DQR+07]. Even though Folding@home is currently the most powerful computing system on Earth (it operates at the 4.5 native peta FLOPS performance level [URL10a]), a simulation of $10\mu s$ of protein folding uses 10 000 CPU days while proteins usually fold in a millisecond timescale [Jac98].

2.2 Protein Structure Prediction

As an exhaustive search for the best fold is not feasible (see Section 2.1.1) and due to a heavy computational cost of all-atom models, in practice some approximation is necessary. One of the earliest such approximations was a hydrophobic-polar (H-P) model proposed by Dill [Dil90].

2.2.1 HP Model

The hydrophobic-polar (HP) model reduces the 20 element amino acid alphabet to 2 letters (H and P). Each amino acid is classified as hydropphobic or polar by its affinity to water. Other properties such as size, shape, electric charge etc. are ignored. The forces acting between amino acids are simplified to a single rule of attraction between H elements (P elements are neutral). The quality of a fold is then measured as a number of H-H contacts. Justification for this model is a hydrophobic collapse hypothesis, based on the observation that native states often contain a hydrophobic core and a hydrophilic (polar) exterior [BKR06].

Despite the reduction of the folding problem complexity, even 2D lattice based H-P folding is proven to be NP-complete by reduction to the Hamiltonian cycle [CGP$^+$98]. Nevertheless several algorithms solving it as has been proposed [KHSP99, LP00, New02, KBHB02], where the problem is stated as a folding of the binary string to a self-avoiding walk on the lattice for which number of adjacent pairs of 1's is maximised.

2.2.2 Residue-level Models

Since the H-P model has been shown too oversimplified to predict the real folds, reduced alphabets of size greater than 2 have also been investigated [BSH$^+$07, BSH$^+$09]. Furthermore, more complex residue-level models has been proposed such as SICHO [KS98], UNRES [LOC$^+$04], CABS [Kol04] or CAS [ZS04c]. Instead of an exact atomic representation these models use a reduced representation, where coordinates of groups of atoms are replaced with a single high level entity (pseudo-atom), e.g. the group center of mass.

The simplified models also use the notion of protein energy differently to compensate for the loss of detail. In these models, the energy function incorporates knowledge-based potentials derived from a statistical analysis of the regularities seen in protein structures. Such energy function does not capture the physical free energy explicitly. Instead, it represents the probability that a given structure is native-like.

2.2.3 CASP

CASP (Critical Assessment of Structure Prediction) is a bi-annual experiment, where for a given set of the target protein sequences structural predictions (3D models of proteins) are submitted by the partcipants. The accuracy of a model is then assessed by the comparison to a real native structure, but the prediction itself is made in a blind manner — the native structure is not known at the moment of submission. The models in CASP are usually created by a mixture of computations and the expert knowledge.

The CASP experiment was initiated by John Moult of the Center for Advanced Research in Biotechnology in 1994 [SF01]. Up till now, eight editions has been run. In each edition a few dozen targets are used. They are classified by the level of difficulty into three categories: comparative modeling, fold recognition/threading and new fold/ab initio prediction. In comparative modeling category the target protein sequence is similar to the sequence of proteins which 3D structure has been already experimentally determined. Fold recognition category contains proteins which have folds that have been already seen in other proteins. In the ab initio or **de novo prediction** there is no relationship between proteins and already known complete structures[MF05].

The models are compared to the target structures using several distance based methods like Global Distance Test (GDT), Z-Score or TM-Score as well as using a human expert visual evaluation [ZVMF99, ZS05].

The younger brother of CASP is CAFASP (Critical Assessment of Fully Automated Structure Prediction), which focuses on evaluation of the fully automated prediction techniques. The models are collected from prediction servers within a limited amount of time (48h) [FB99]. After CAFASP5 it was integrated into the CASP experiment which since then maintains a separate server-only ranking.

Progress in CASP

The progress in prediction quality has been observed in first two CASP experiments and has continued steadily thereafter in more modest manner. Still, the prediction scores for comparative modeling and fold recognition has doubled from CASP1 to CASP6. The ab initio category had a rapid progress period from CASP1 to CASP4, and visible progress for small targets in CASP5/6, though the quality of models for large targets remained very poor[KVFM05].

As the progress for the new fold (NF) targets is slow, it remains the most challenging of the CASP experiment categories. Figure 2.3 presents new fold targets successfully predicted by the individual groups participating in CASP5/6 compared with the success ratio expected by chance. Left-hand graphs show how many groups tried to predict how many targets. Right-hand graphs show how many groups predicted how many targets ranked in the top six. In CASP5 three groups did significantly better than chance, what improved to six groups in CASP6.

With the advance of prediction methods a borderline between CASP categories, especially the distinction between the comparative modeling and fold recognition has been blurred. This change has been reflected in CASP7, where this two categories have been joined under a new name of template-based modeling.

Template-free Prediction

In an attempt to perform successful *de novo* prediction several knowledge-based techniques have been applied including secondary structure prediction, fragments and motifs identification, contact prediction and assembly of folds from fragments. These methods usually are combined with the search methods based on the molecular dynamics, Monte Carlo optimisation or genetic algorithms. A few methods use pure ab initio simulations together with empirical potentials[MF05].

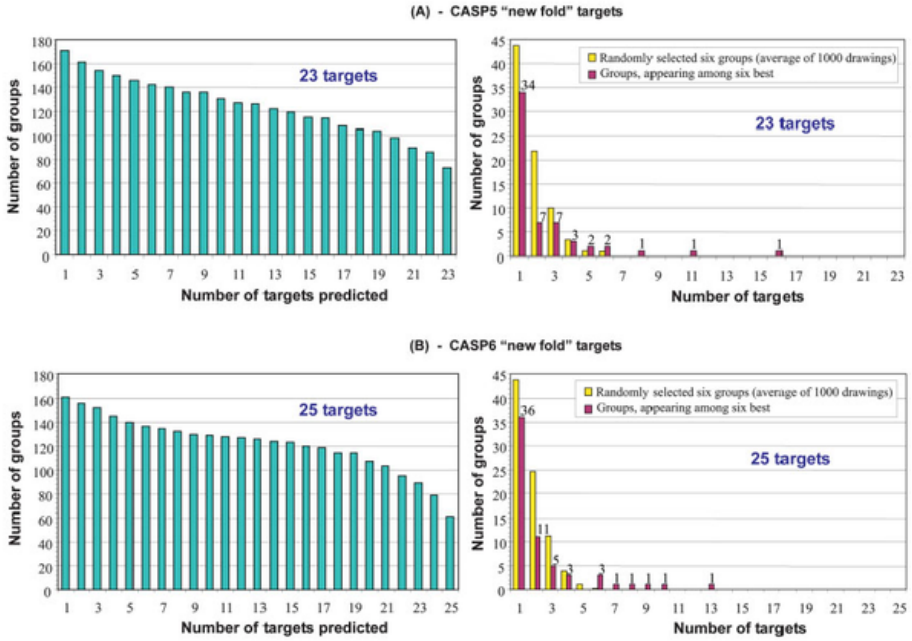

Figure 2.3: Distribution of success in prediction of CASP new fold (NF) targets [KVFM05].

The most influential participants in the template free category of the last few CASPs were David Baker with the Robetta server[CKM+05] using the Rosetta protocol[RSMB04, BSR+02], Yang Zhang and Jeffrey Skolnick with I-TASSER server[ZS04a, ZAS05] and Touchstone[ZKS03], Kevin Karplus with SAM-T0x family[KK05, KK03] and David Jones with FRAGFOLD fragment assembly method[JBC+05].

Even though *de novo* methods demand for computational resources is very large, they have been proven successful on targets where traditional fold recognition fails. Despite of the low quality of models and difficulties in identification of correctly predicted parts, ab initio methods are expected to have biggest impact on structural biology in the future [GGGR05].

Predictor Pipeline

The *de novo* prediction methods are designed to find the native structure without doing the computationally expensive simulation of folding with the molecular dynamics. Therefore, they usually use heuristic algorithms to quickly search the conformational space, instead of exhaustive sampling of the energy landscape. The search is performed by iterative conformational changes and is guided into to the low energy states. The search space is usually additionally limited by either simplification of the protein model to the lattice-based structures [SZA+03] [ZKS03] or by construction of initial candidate structures (also known as **decoys**) through the assembly of fragments of known proteins [BMQ+05, BCM+03, BTR+01].

The progress in the field of *de novo* prediction is attributed to the arise of refinement methods based on clustering of large number of candidate structures obtained by energy optimization. The final models are chosen from the largest clusters what shifts the emphasis from energy values to most common pathways in the energy landscape [GGGR05].

The schema of a *de novo* predictor has been shown in Figure 2.4. Conformations are generated by assembling fragments of protein chains, then conformational search is performed to obtain a set of decoys which is then refined and clustered to select the best model.

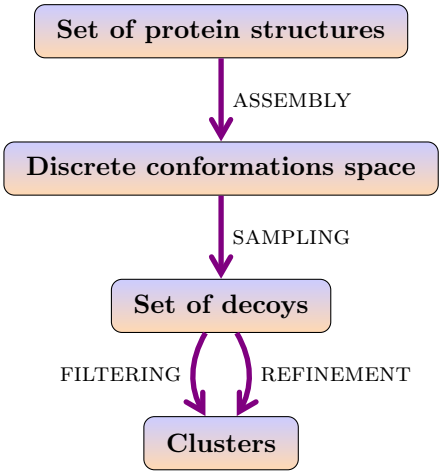

Figure 2.4: General de novo predictor schema.

This statistical approach to improve the accuracy of predictions has been borrowed from meta predictors where diversification of information sources proved to be more successful than relying on a single prediction method [GEFR03, KB03].

A simple meta predictor compares models fetched from various prediction servers and selects the consensus model (most similar to other models). Advanced meta predictor modifies collected models or creates hybrid ones. Servers such as Robetta [CKM+05] or ProtInfo [HNLS05] perform *de novo* prediction guided by structures obtained from other simple meta predictors and because of that are sometimes called meta–meta predictors.

2.3 State-of-the-art De Novo Prediction

The two most successful prediction methods in the "template free" category of the CASP7 experiment [Chi06, Zha06a, BKB+07] are Robetta [RSMB04] and I-TASSER [WSZ07]. Both methods build the initial protein models from short fragments of known structures similar on a sequence level. Small random changes are applied to these models and the Monte Carlo method [ED05] is used to find a structure with the minimal energy. In both methods the energy is formulated as a weighted sum of knowledge-based potentials.

To determine the optimal set of weights both methods generate a set of candidate structures, so called decoys, by applying small random changes to the known native structure. The optimisation objective is to maximise the correlation between the value of energy function and the similarity of decoys to the native structure. Therefore, the energy function is expected to have the lowest value for the decoys that are most similar to the native structure. Similarity is measured as the **root mean square deviation** (RMSD) of euclidean distance between C_α atoms of a decoy and the native structure.

In the weight optimisation process, Robetta used a training set of 21 proteins. For each protein 30 000 decoys were generated and the linear regression against RMSD was used

as an objective function [SRK+99]. I-TASSER used 30 proteins, with 60 000 decoys each, and maximised a complex objective function with correlation to RMSD as its main element [ZKS03]. Both prediction methods are able to distinguish between native-like (RMSD value $< 0.4nm$) and non-native decoys (RMSD value $> 0.8nm$). However, the actual correlation coefficient between the energy and similarity is not too high, eg. Zhang et al. [ZKS03] report it to be 0.54 for naive combination of terms and 0.65 for the optimised weighted sum.

A critical analysis of the approach described above reveals two drawbacks. Firstly, the set of decoys created by randomisation of the native structure is biased towards that structure, resulting in a potential overfitting of the energy function. The process itself is also the exact opposite of what predictors do in practice. In the prediction process the native structure is unknown and the decoys have to be built from scratch.

Secondly, the linear combination of energy terms is a very simple but also potentially very restrictive approach to construct the energy function. It assumes that all energy terms can equally well discriminate between good and bad candidate structures. In practice, however, these terms works well for some proteins and bad for the others. If so why not to learn these cases and apply the energy terms more selectively?

Now, even if the energy terms can distinguish between good and bad candidate structures it is not enough to do it in a discrete manner. There is a need of a certain graduation of values of each energy term across all possible protein structures starting from the dissimilar ones and ending at the native structure itself. Otherwise the energy function being a linear combination of such terms would be unable to provide an effective smooth fitness landscape that could be navigated by a search method. This is why in this dissertation I hypothesise (hypothesis **H2**) that going beyond the weighted sum and towards more complex functional combinations of energy terms will result in improved prediction quality.

2.3.1 Predictor Building Blocks

As the protein structure prediction is a complex multi-step process it takes a long time to build a predictor from scratch. However, apart from complete *de novo* predictors such as (PROTINFO-AB, Robetta or I-TASSER) there are many programs supporting different stages of the prediction. For example 3D Jury and LOMETS are meta-servers for template based modelling incorporating several different methods for fold recognition and threading. FRankenstein and 3D-SHOTGUN performs fragment assembly. Pcons/Pmodeller assesses protein models quality. A novel prediction method could be constructed by combining these existing building blocks.

A serious limitation, though, is the accessibility of these tools. Most of them are provided only as online servers. For the popular high traffic services the waiting times (from data submission to obtaining the results) is counted in months.

Due to that, it is in fact easier to combine the full predictors. The idea of such a novel combination is shown in Figure 2.5. The fragment assembly and *de novo* prediction of initial models could be done with Rosetta software (used by the Robetta predictor) that is free for the academic use. These initial models could become a starting point for further refinement with the use of the I-TASSER energy formulation. Refined models could be

clustered with the use of different similarity measures provided by the ProCKSI server (see Section 2.4.1) with final models selected from the clusters centroids.

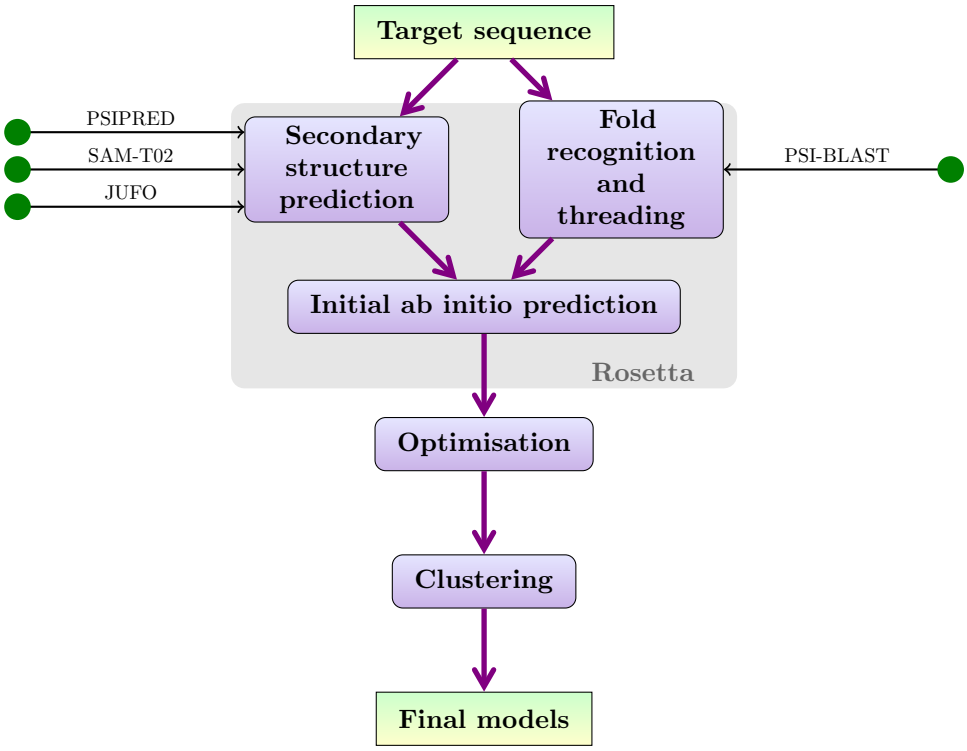

Figure 2.5: An idea of de novo predictor that combines initial models generated by Rosetta with I-TASSER energy optimisation and ProCKSI structural clustering

The use of two different energy formulations, one from Rosetta (to generate initial models) and the other from I-TASSER (for further refinement of decoys), together with our own clustering strategy based on the similarity consensus (see Section 2.4.3) should provide substantial novelty and diversity of the final models.

Although the creation of a protein structure predictor is out of the scope of this dissertation I wanted to demonstrate here a broader vision behind this work and show that extension of the proposed methods into a full working predictor would not be very difficult.

For a better overview of how the components of such a solution could work together, the next sections provide an overall description of the prediction algorithm and energy function used by Rosetta and detailed description of the energy terms used by I-TASSER that are later used in the genetic programming experiments (see Chapter5).

2.3.2 Rosetta Ab Initio Algorithm

The first step of the prediction process is generation of fragments. They are made with the use of PSI-BLAST alignments and up to three secondary structure predictions from PSIPRED, SAM-T02, JUFO or PhD. Similarity between the target sequence and a fragment is a sum of similarity scores for the sequence and secondary structure. This fragment library is used in the next step — the fragment insertion.

The algorithm starts from completely extended chain and incrementally changes fragments of the backbone. Each change starts with random selection of the position in the chain.

One of the fragments starting in that position is selected from the library and backbone torsion angles (ϕ, ψ, ω) from the fragment are applied in that position. Both bond angles and bond lengths are fixed to ideal values, and the side chains are approximated with the centre of their mass.

Fragment insertion is performed in two stages. First 9-residue fragments are used to construct a rough model. Then the skeleton of all the generated decoys is refined with 3-residue fragments.

For each insertion, the choice is made from the top 25 fragments (out of 200 in the library) [RSMB04]. The energy of a structure is evaluated using Rosetta knowledge based potential and the fragment insertion is accepted/rejected according to a Monte Carlo criterion. The number of used potential terms increases gradually with the progress of simulated annealing, starting from steric overlap and finishing with complete potential for the last 25% of iterations of 9-residue stage and for the whole 3-residue one.

The best decoy structures are selected either as a 5% lowest energy subset or cluster analysis is performed to select the representative structures. This becomes a starting point for the full-atom refinement. It is done with several different moves like torsion angles perturbation, non-perturbating fragment insertion (only fragments from library similar to existing fragment in the model), local fragment gradient descent optimisation and optimisation of side chains using a backbone-dependent rotamer library [BCD97, CSD03].

2.3.3 Energy Functions

Although the energy functions used in two most successful CASP7 predictors are based on first principles and try to approximate the value of full-atom force field, the term "energy" is used here in an indicative way only. That is, the components of this functions are knowledge/statistical based terms and do not express the free energy explicitly but rather the probability of "nativeness" of given structure, based on analysis of features of known native structures.

Rosetta Energy Function

Rosetta free energy is based on a Bayesian model of the likelihood of the structure being a native one, given the sequence of amino acids. The components of this model are based on the statistical analysis of common features of the native structures and describe either the energy of structure independent of sequence ($P(structure)$) or energy of sequence given particular structure ($P(sequence|structure)$) [SRK$^+$99].

The all-atom Rosetta energy function (used in models refinement) is a linear combination of Ramachadran torsion preferences, Lennard–Jones potential, solvation and electrostatic effects, hydrogen bonding and backbone dependent rotamer self-energy. In the Lennard-Jones potential, van der Waals radii and well depths from the CHARMM19 are used and repulsion is attenuated to a linear function to compensate for fixed rotamer set. Solvation energy is computed using Lazaridis and Karplus [LK99] complete model. Electrostatic interactions are approximated based on PDB statistics. Hydrogen bonds are included using potentials dependent on secondary structure and orientation.

Simplified, low resolution energy function is used in fragments insertion. In that function the Van der Waals potential is represented by steric repulsion of backbone atoms and

side-chain centroids and rewarding of globally compact structures. Solvation potential is based on statistical data. Hydrogen bonding is not defined explicitly but probabilistic descriptions of geometry of β-strand pairing and β-sheet patterns are given [RSMB04].

I-TASSER Energy Function

The original energy formulation now used in I-TASSER was introduced in TOUCHSTONE-II [ZKS03]. It was based on the simplified on-lattice CABS model (see also [Kol04]) describing the protein as a set of C_α, C_β and the center of mass of the remaining side chain. The bonds between C_α atoms are represented by vectors with integer, lattice based coordinates, with average length of $3.8\mathring{A}$ and angles in 65–165° range seen in real proteins.

In I-TASSER, the CABS model has been replaced with CAS, where position of C_β atoms is no longer set separately but is taken into account as the side chain centroid SG [ZS04c]. The target C_α atoms aligned to a template in threading process are positioned off-lattice, whereas the unaligned fragments are set on simple cubic lattice as in TOUCHSTONE-II algorithm.

In the optimisation process the best on-lattice position of the C_α atoms is looked for. The position of SG is relative to C_α position and is chosen from a set of fixed configurations for each residue type (**rotamer library**).

TOUCHSTONE-II energy function is a weighted sum of short-range correlations, local protein-like stiffness, hydrogen bonds, long-range pairwise interactions, solvent exposure effect, electrostatic interactions, contact environment and a bias towards predicted local distance, contact number and contact order (average sequence separation of residues in contact).

$$E = E_{short} + E_{stiff} + E_{HB} + E_{pair} + E_{solv} + E_{electro} + E_{env} + E_{dist} + E_{COCN} + E_{contact} \tag{2.7}$$

Short-range C_α-C_α and SG-SG interactions are described by statistical potential where correlation of local structure is defined as negative logarithm of the corresponding frequency histogram:

$$E_{short} = \sum_i (w_1 E_{13} + w_2 E_{14} + w_3 E_{15} + w_4 E'_{12} + w_5 E'_{13} + w_6 E'_{14} + w_7 E'_{15}) \tag{2.8}$$

The $E_{i,i+n}$ energy terms represent short-range $C_\alpha - C_\alpha$ interactions of the i-th residue with its n-th next neighbour. Each term measures the correlation of the local structure with the distribution of structural features extracted from the known structures (i.e. the negative logarithm of the frequency histogram derived from PDB database [Ber08]). It depends on the amino acid type, predicted secondary structure and the distance between C_α atoms. Similarly $E'_{i,i+n}$ represents interaction between two side chains in range n. Side chain interactions, however, were used only in TOUCHSTONE-II and dropped out later in I-TASSER.

Stiffness knowlede based potential E_{stiff} represents structural tendency towards a formation of the predicted secondary structure:

$$E_{stiff} = w_8 \sum_i \left(-\lambda \hat{v}_i \cdot \hat{v}_{i+4} - \lambda \left| \hat{b}_i \cdot \hat{b}_{i+2} \right| - \lambda \Theta_1(i) + \Theta_2(i) + \Theta_3(i) \right) \tag{2.9}$$

$\hat{v}_i$ is a normalized (unit) vector between two consecutive carbon atoms $C_{\alpha,i}$ and $C_{\alpha,i+1}$. $\hat{b}_i$ is a unit bisector of the angle between $\hat{v}_{i-1}$ and $\hat{v}_i$ (see Figure 2.6). λ is a stiffness factor that differs for residues inside or outside of the protein **radius of gyration** (i.e. mean square distance from the center of mass). Three Θ functions represent structural bias in favour of predicted secondary structures and penalise irregularities.

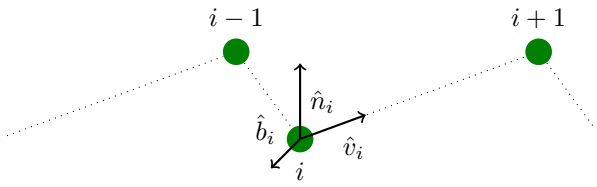

Figure 2.6: Backbone vectors orientation: $C_\alpha - C_\alpha$ vector ($\hat{v}_i$), angle bisector ($\hat{b}_i$), backbone surface normal $\hat{n}_i$.

Due to lack of exact position of bond atoms in the CAS model (it uses only position of C_α atom and the side chain geometrical center of mass), hydrogen bonds are represented as a C_α packing preference supplemented with statistical parameters derived from known structures:

$$E_{HB} = w_9 \sum_{j>i} \lambda' \left(\hat{b}_i \cdot \hat{b}_j \right) |\hat{n}_i \cdot \hat{n}_j| \, \Theta_4(i,j) \tag{2.10}$$

$\hat{n}_i$ is a unit normal vector perpendicular to the plane containing $\hat{v}_i$ and $\hat{b}_i$ ($\vec{n}_i = \hat{b}_i \times \hat{v}_i$). The dot product of vectors specifies a bias towards regular H-bonds. λ' is a stiffness factor depending on predicted secondary structure and Θ_3 represents a knowledge based binary condition whether H-bond between ith and jth residue exists or not.

Long-range interactions are calculated between the side chain centers of mass. The potential is represented here by a hard-core sphere of excluded volume interactions and a soft-core $1/r$ type potential outside the sphere:

$$E_{pair} = w_{10} \sum_{j>i} E_{ij}(s_{ij}) \tag{2.11}$$

E_{ij} is set to 0 outside the soft-core ($s_{ij} > R_{max_{ij}}$), to 4 inside the hard-core ($s_{ij} < R_{min_{ij}}$) or to statistical pairwise potential e_{ij} if between.

Solvent accessibility effects are described by sum of C_α burial/exposure potential and SG hydrophilicity scales:

$$E_{solv} = w_{11} \sum_i \left(E_{C_\alpha,i} + \mu E_{SG,i} \right) \tag{2.12}$$

E_{C_α} is a statistical potential depending on the distance of C_α atom from the protein center in relation to the radius of gyration (for N residues $r_0 \approx 2.2N^{0.38}$). μ is a burial factor representing position of SG in relation to hydrophobic core (center of mass) and E_{SG} is a combination of Kyte-Doolittle and Hopp-Woods hydrophilicity scales.

Electrostatic effects are included using a form of Debye-Hückel equation:

$$E_{electro} = w_{12} \sum_{j>i} \frac{\exp(-k s_{ij})}{s_{ij}} \tag{2.13}$$

k is the inverse Deybe length, where $\frac{1}{k} \sim 15\mathring{A}$ was experimentally chosen by Zhang et al.

Potential describing the contact environment is based on bonds geometry and is derived from the set of PDB structures:

$$E_{env} = w_{13} \sum_{i} V_i \tag{2.14}$$

V_i represents amino acid specific potential and depends on number of residues that are in contact with the ith residue for which their bisectors are parallel ($\hat{b}_i \cdot \hat{b}_j > 0.5$), anti-parallel ($\hat{b}_i \cdot \hat{b}_j < -0.5$) or perpendicular (dot product in $[-0.5, 0.5]$ range). Residues are considered to be in contact if $s_{ij} < R_{min_{ij}}$.

Distance map, contact order, contact number and contacts restraint are based on predictions collected from templates found by threading program PROSPECTOR:

$$E_{dist} = w_{r1} \sum_{j>i} \Theta_5 \left(|r_{ij} - d_{ij}| - \delta_{ij} \right) + w_{r2} \Theta_6 \left(\sum_{j>i} \frac{|r_{ij} - d_{ij}|}{\delta_{ij}} - N_{dp} \right) \tag{2.15}$$

$$E_{contact} = w_{r3} \sum_{j>i \in S_{cp}} \Theta_5 \left(s_{ij} - 6\mathring{A} \right) + w_{r4} \Theta_6 \left(\sum_{j>i \in S_{cp}} \Theta_6 \left(s_{ij} - 6\mathring{A} \right) - N_{cp} \right) \tag{2.16}$$

$$E_{COCN} = w_{14} \left(N_{CO} - N^0_{CO} \right) + w_{15} \left(N_{CN} - N^0_{CN} \right) \tag{2.17}$$

$\Theta_5(x)$ is simplified sign function which returns 0 for negative x and 1 otherwise. $\Theta_6(x)$ is similar but it returns x instead of 1. The first term of Equation 2.15 is a penalty for deviation from predicted distance d_{ij} greater than mean square deviation of prediction δ_{ij}. The second term is a penalty for accumulated normalized deviation greater than number of predictions N_{dp}. The first term of Equation 2.16 is a penalty applied if side chains predicted to be in contact are in distance greater than $6\mathring{A}$. The second term is similar penalty for total violation above the threshold N_{cp}. The last Equation 2.17 penalise deviation from expected contact order N^0_{CO} and contact number N^0_{CN}.

Equation 2.7 has a total of 19 energy terms. The number is so high since short-range interaction and threading prediction based terms are combinations of separately weighted potentials. In overall, the force field uses 13 statistical potentials: E_{13}, E_{14}, E_{15}, E'_{12}, E'_{13}, E'_{14} and E'_{15} for short-range interactions, R_{min}, R_{max} and e for long-range interactions, E_{C_α} and E_{SG} for solvation effects, finally V for contact environment.

The TOUCHSTONE-II energy function has been later improved by using more detailed geometry of the hydrogen bond, where the position of C_α is extended by backbone N and CO groups [ZS04c]. Two variants of this hydrogen potential has been described by Zhang et al. in supplementary materials to [ZHA+06b]. One is using detailed atomic off-lattice model which includes heavy atoms (N,C,O) along with the C_α, the other is a reduced on-lattice CAS model. New potential supplements previous geometrical definition with

statistical parameters derived from known structures:

$$E_{HB_{ij}} = \begin{cases} \lambda_\alpha \frac{\left(1-\left|\hat{b}_i\cdot\hat{b}_j - b_{\alpha 0}\right|\right)(1-|\hat{n}_i\cdot\hat{n}_j - n_{\alpha 0}|)}{(1+||\epsilon\hat{n}_i-\vec{r}|-nr_{\alpha 0}|)(1+||\epsilon\hat{n}_j-\vec{r}|-nr_{\alpha 0}|)} & \text{if } i,j \text{ in } \alpha\text{-helix,} \\ \lambda_\beta \frac{|\hat{n}_i\cdot\hat{n}_j|\left(\hat{b}_i\cdot\hat{b}_j\right)}{\left(1+\frac{1}{2}|\epsilon\hat{n}_i-\vec{r}|\right)\left(1+\frac{1}{2}|\epsilon\hat{n}_j-\vec{r}|\right)} & \text{if } i,j \text{ in } \beta\text{-sheet.} \end{cases} \tag{2.18}$$

The location of hydrogen bonds in a specific secondary structure region is determined based on the contact order and relative orientation of $\hat{v}_i$, $\hat{v}_{i-1}$ and $\hat{v}_j$, $\hat{v}_{j-1}$ backbone vectors. $\hat{n}_i$ is a unit normal vector perpendicular to the plane containing $\hat{v}_i$ and $\hat{b}_i$ ($\vec{n}_i = \hat{b}_i \times \hat{v}_i$). $\lambda_{\alpha/\beta}$ is set to 1 if secondary structure prediction for residues i and j is the same (both α-helix or both β-sheet) and to 0.5 otherwise. *epsilon* is equal to 5Å for α-helix and to 4.6Å for β-sheet. Parameters $b_{\alpha 0}$, $n_{\alpha 0}$, $nr_{\alpha 0}$ are based on statistics derived from a set of 50 α-proteins and 50 β-proteins.

I-TASSER uses a new hydrophobic potential, where accessible surface area (ASA) is predicted by the neural network [CZ05], of almost twice as good accuracy as for Hoop-Woods and Kyte-Doolittle potentials [WSZ07]. In the training process 12 different ASA cutoffs (0.05,0.1,...,0.6) are used to determine if the residue is exposed or buried:

$$E_{ASA} = -\sum_i \left(\frac{x_i^2}{x_0^2} + \frac{y_i^2}{y_0^2} + \frac{z_i^2}{z_0^2} - 2.5\right) \times P(i) \tag{2.19}$$

First term corresponds to μ burial factor from Equation 2.12. Vector (x_i, y_i, z_i) represents ith residue in the center of mass (hydrophobic core) coordinate system and (x_0, y_0, z_0) represents lengths of principal axes of the protein ellipsoid. $P(i)$ is the sum of two state (1=exposure, -1=burial) predictions for all cutoffs.

In both TOUCHSTONE an I-TASSER algorithms the structure energy is optimised using the parallel hyperbolic sampling (PHS), which is extension of the replica exchange Monte Carlo sampling. To flatten the energy landscape it applies inverse hyperbolic sine function to the energy difference [ZKS02].

2.4 Protein Structure Comparison

Structure comparison plays a fundamental role in understanding of the protein function as it is a protein structure, not its sequence that conserves the function. Under the assumption of evolutionary continuity of structure, it is possible to deduce the function of newly discovered proteins by searching for homologous structural fragments and identification of common ancestors. To do that, structures are aligned in way that maximises a similarity score of a common region. Longer alignment usually correspond to a better match [HH09].

Structure comparison is also used in protein structure prediction to evaluate the quality of protein models. In this case the alignment is fixed, as both target native structure and the model represent the same amino acid sequence and differs only in shape. As different superpositions of the compared structures lead to different similarity scores, usually the best score with respect to the rotation is used. In case of the RMSD it could be determined in polynomial time, but in general case it is NP-hard problem and heuristic are employed to solve it.

The comparison in sense of alignment of two different proteins allows to build taxonomies of proteins with hierarchical classes for different protein domains, folds or families.

Currently two large human made classification systems are actively maintained: SCOP [HAB+99, AHC+08] and CATH [PBB+03, GLA+07]. They are used as a "gold standard" and comparison methods are trained and tested against them [KKL05, BHBK07]. In contrast, the comparison in sense of optimal superposition of two structures with the same sequence, as used in prediction, is deficient in such a "standard" reference. In CASP experiment a visual assessment by a human expert is used for selected models [OSO02], however, for obvious reasons this approach does not scale to the case of thousands of decoys as in energy function optimisation and is very error prone.

2.4.1 ProCKSI Meta-server

As there are many biologically meaningful definitions of the protein similarity, there is also a variety of comparison methods, each with its own concept of what "similar" means. Such similarity could depend only on the structure or also rely on the sequence, it could take into account only local best fitting sub-regions or be measured globally for the whole structure, it could focus on the chemical role any physio-chemical properties or rather on the expressed biological function.

This variety causes confusion. It is not clear which method should be applied in which specific case. And the number of methods is growing. In recent years it doubled two times in each decade [HH09].

To address this problem a project to build specialised meta-server for Protein (Structure) Comparison, Knowledge, Similarity, and Information (**ProCKSI**) was started by N. Krasnogor and D. Barthel. The idea behind ProCKSI was to turn around the comparison paradigm from choosing a single "best" method (and excluding the others) to an integrative approach suggested by Camoglu et al. [CCS06] with many different methods included at the same time.

To achieve that, ProCKSI treats the problem of structural comparison as a set of many related, but different structural similarity problems solved with different methods. As the methods measure different aspects of similarity, they complement each other and together create more accurate representation of the protein space.

2.4.2 ProCKSI Comparison Methods

The goal of the ProCKSI meta-server is not only to be able to compare multiple protein structures using multiple similarity measures, but also to provide methods that complement each other. For that reason the core ProCKSI comparison methods where chosen to be able to deal with comparison of both similar and dissimilar structures.

Such a universality is very difficult to achieve with a single measure and ProCKSI includes a different measure for each case: USM for dissimilar structures, and Max-CMO for the similar ones. Both these measures where proposed to address the weak point of popular comparison methods based on heuristic structural superpositions where optimality of the score cannot be guaranteed [LCWI01, KP04].

Contact Maps

The **contact map** is a representation of the protein 3D structure based on proximity relation between residues. The map is a square binary matrix C of size n equal to the number of protein residues. For each pair of residues i and j, C_{ij} is equal to 1 if i and j are in contact, and 0 otherwise. Two residues are in contact if the Euclidean distance between them is less than a given threshold α. An example of protein tertiary structure and its contact map is shown in Figure 2.7.

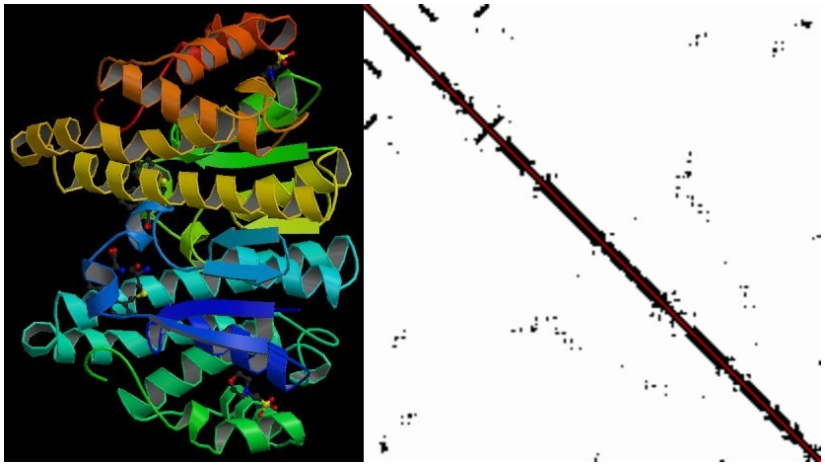

Figure 2.7: Protein and its contact map [BHBK07].

USM

The Universal Similarity Metric (USM) is a very general similarity measure that compares the information content of protein contact maps C_1 and C_2 by approximation of the Kolmogorov complexity K using a compression algorithm [KP04]. Kolmogorov complexity $K(C_1)$ of a contact map C_1 is approximated by the length of compressed map $|zip(C_1)|$. Conditional complexity $K(C_1|C_2)$ of contact map C_1 when $K(C_2)$ is known is approximated by compression of the sum of objects $|zip(C_1 + C_2)|$.

Based on that the similarity between two structures is defined as universal, problem domain independent Normalised Compression Distance (see Equation 2.20). According to Krasnogor and Pelta [KP04] NCD is particularly effective with distantly related structures.

$$NCD(C_1, C_2) = \frac{max[K(C_1|C_2), K(C_2|C_1)]}{max[K(C_1), K(C_2)]} \tag{2.20}$$

Max-CMO

As the USM is a very general metric, for fine grained comparison the **Contact Map Overlap** (CMO) measure [GSK93] was implemented in ProCKSI as a part of this work (see Chapter 3 for details). The CMO measure compares two contact maps in a form of graphs and the value of CMO corresponds to the number of overlapping (matching) edges of these graphs.

Let's define the contact map of a protein in terms of relation R between two residues r_i and r_j. Let $r_i R r_j$ be true if and only if the Euclidean distance between residues $\delta(r_i, r_j)$

is less or equal to the given threshold α. Now, relation R could be depicted as a **graph** (see Figure 2.8), where each residue is represented by a node and two nodes r_i and r_j are connected with an edge if $r_i R r_j$ is true (i.e. corresponding residues are in contact).

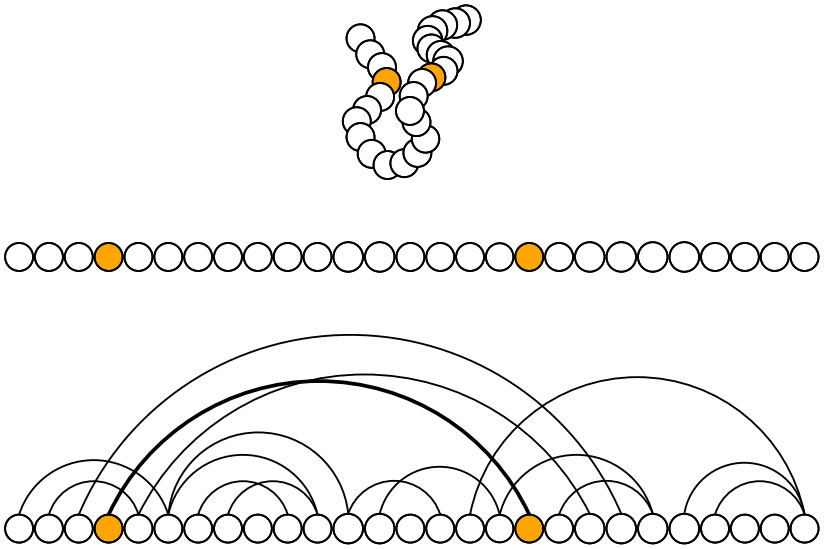

Figure 2.8: Construction of a contact map graph. Residues are mapped to nodes of the graph and the ones being in contact, i.e. in distance less than a threshold α (here marked with color), are connected with an edge.

As for the CMO measure exact upper and lower bounds can be computed and compared, many authors consider it as one of the most reliable and robust measures of protein structure similarity [CL02, KP04].

Other Methods

In addition to the two contact map based methods described above, ProCKSI includes other comparison methods:

1. Combinatorial Extension of the optimal path (CE)[SB98],
2. Distance Matrix Alignment (Dali) [HP00],
3. FAST (FAST Alignment and Search Tool)[ZW05],
4. TM-align [ZS05],
5. Vorolign (aligning Voronoi contacts) [BGCZ07],
6. hybrid URMS-RMS (Unit-vector Root Mean Squared) [YK05].

2.4.3 ProCKSI Consensus

The ultimate goal of ProCKSI is to move the integration of comparison methods beyond the multi-comparison towards intelligent merging of similarity aspects contained in each of the individual methods. The limitations of this approach are not known so far as only two very simple consensus methods are currently implemented in ProCKSI: average and

median of all the similarity values (see Section 4.2.1 for detailed description of consensus construction).

However, the first application of the ProCKSI consensus measure to the problem of protein classification revealed very promising results. The simple mean of the similarity values classified the proteins on the SCOP fold level as accurately as the best of them. A hand picked subset of the methods worked even better revealing the synergy effect and higher classification accuracy than the best method included in the consensus[BHBK07].

2.4.4 Decoy Comparison Measures

It is not very clear how well the protein structure comparison methods included in the ProCKSI server will perform when used to compare decoys. There is evidence that for specific targets the Max-CMO measure and the USM/Max-CMO consensus reflected the CASP6 assessments quite accurately and even was able to detect better models[BHBK07]. However, this might not be a common behaviour across all methods.

Although in this dissertation the comparison is limited to the ProCKSI methods, I am aware that using decoy specific measures to construct a similarity consensus is an interesting alternative approach. For that reason I discuss the most popular decoy comparison measures below. Additionally, later in Section 4.2.3 I describe a prototype of a ProCKSI-like cloud computing application using decoy comparison measures. In the future this might be extended with a consensus construction method.

Root Mean Square Deviation

The oldest and *de facto* standard measure of similarity between a target native structure and a predicted decoy structure is the **RMSD**. It is calculated as a root mean square deviation of the distance between corresponding C_α atoms of two structures (see Equation 2.21). The optimal rotation that minimises the RMSD between two sets of coordinates is found by the Kabsch algorithm [Kab78, CSD04]. Despite its popularity it is known to have a number of deficiencies [CZF+01, WFB03, Zem03]. First, due to the quadratic nature of the RMSD, a small local perturbation is penalised with a large deviation indicating incorrectly that two structures are different. Second, the absolute value of RMSD is meaningless as it depends on the length of the compared structures (it tends to increase with protein size). Finally, as it is sensitive to outliers it typically reports how distant are the most incorrect parts of a model instead of what is usually much more interesting, namely, which regions of a model are closest to the native structure.

$$RMSD = \sqrt{\frac{1}{N}\sum_{i=1}^{i=N}\delta_i^2} \tag{2.21}$$

Global Distance Test

To address the first issue a **global distance test** total score (GDT_TS) was proposed [Zem03]. It takes into account both local and global superpositions by using 4 different distance cutoffs. At each cutoff a largest superimposable set of residues is found. The value of GDT_i is the size of such a set for the cutoff distance i (see Equation 2.22).

Effectively GDT_TS express an average coverage of the target structure. In addition to being size depended similarly to the RMSD, it is also more crude as it focuses only on the size of the substructures within a given distance threshold ignoring the exact total distance. It is currently a baseline measure in the CASP experiment [BDNBP+09].

$$GDT\_TS = \frac{\sum_{i \in \{1,2,4,8\}} GDT_i}{4} \tag{2.22}$$

Relative RMSD

To avoid the size dependency a normalised variant of RMSD was proposed called the relative RMSD (see Equation 2.23). The denominator is the average RMSD of two random proteins of equivalent radius of gyration.

$$rRMSD(A,B) = \frac{RMSD(A,B)}{avgRMSD(Rg_A, Rg_B, pairs)} \tag{2.23}$$

TM-score

The same concept is used a bit differently in **TM-score** [ZS04b]. An average distance d_0 between corresponding residue pairs of two randomly related proteins is calculated based on the power-law dependency between the radius of gyration and the length of a protein and used to normalize the match difference to [0;1] range (see Equation 2.4.4).

$$\begin{aligned} d_0(L_\mathrm{n}) &= 1.24\sqrt[3]{L_\mathrm{n} - 15} - 1.8 \\ \text{TM-score} &= \max\left[\frac{1}{L_\mathrm{n}} \sum_{i}^{L_\mathrm{a}} \frac{1}{1 + \left(\frac{d_i}{d_0(L_\mathrm{n})}\right)}\right] \end{aligned} \tag{2.24}$$

2.5 Metaheuristic optimisation

This section provides a brief overview of the key concepts used in metaheuristic optimisation and describes two metaheuristics used in this work: tabu search algorithm (see Chapter 3) and genetic programming (see Chapter 5). The explanation on why these specific algorithms were chosen and the discussion of alternatives are given in the introductions to the aforementioned chapters.

2.5.1 Heuristics

The word **heuristic** comes from the Greek root "heuriskein" which means "to find" (from the same root comes the word "eureka"). A heuristic for a given problem is a method of finding a solution. Although heuristics do not always achieve the best possible solution (the so called **global optimum**), they can dramatically reduce the time required to solve a problem, by eliminating the need to consider unlikely possibilities or irrelevant states. In general heuristics could be divided in two classes: **constructive algorithms**, which build the solution from parts and **local search algorithms** based on a concept of traversing the solutions in some neighbourhood [GP05]. The **neighbourhood** of a

solution is a subset of all possible solutions (which are known as a **search space**) that are possible to reach in a "single step" from some fixed point [BR03].

2.5.2 Metaheuristics

A **metaheuristic** is a set of strategies which could be used to define a heuristic method applicable to a wide range of different problems. A metaheuristic is a high level algorithmic framework which is general enough to be used to solve different optimization problems without a time-consuming adaptation to the specific problem requirements.

A variety of different metaheuristics have been proposed so far [GK03, BK05, Luk09] including: simulated annealing, tabu search, greedy randomized adaptive search procedure (GRASP), scatter search, variable neighbourhood search (VNS), evolutionary algorithms (e.g. genetic algorithms, genetic programming, memetic algorithms), swarm intelligence (e.g. ant colony optimization, particle swarm optimization, honey bees algorithm). Interestingly, inspiration for most of these algorithms was drawn from nature: behaviour of ant or bee colonies, the process of evolution, process of steal annealing or operation of a human memory.

The most important elements of all metaheuristics are concepts of diversification and intensification. The idea behind the **diversification** is to allow a move resulting in worse than current solution quality (so called uphill move) in order to escape from a local minimum. The **intensification** does the reverse to assure the search convergence. It decreases the probability of selecting an uphill move shifting the bias from exploration of the search space towards solution improvement. The core of every metaheuristic is a strategy to manage the intensification/diversification balance [BR03].

2.5.3 Tabu Search

The **tabu search** (TS) method was formalised by Fred Glover in [Glo89] and [Glo90]. It applies the idea of iterative improvement combined with a concept of memory and the observation how humans use it in problem solving. In each step the algorithm moves from a current solution x_i to a solution x_{i+1} in the neighbourhood of x_i. After each move the new solution x_{i+1} is stored in the **long-term memory**, usually implemented as a hash table. By keeping a track of all visited states the TS algorithm can avoid revisiting them and explore new regions of search space instead.

Additionally selected attributes of x_{x+1} related to a recent move $x_i \rightarrow x_{i+1}$ are remembered using a short-term memory mechanism called **tabu list**. The size of a tabu list, also known as **tabu tenure**, is limited to a few recent solutions and can either be fixed from the start (static tabu list) or it can vary during the search (dynamic tabu list).

Whenever a move $x_i \rightarrow x_{i+1}$ alters the **tabu-active** attributes, that is the ones stored in the tabu list, it might be considered tabu. A tabu status of a move is decided using tabu activation rules. For example, after a move that drops element a from the solution, a becomes a tabu-active attribute and activation rule will make a move that adds a again tabu in the next $tabu\_tenure$ iterations.

The purpose of the tabu list is to prohibit the moves which affect recently altered attributes of the current solution. Effectively that restricts the neighbourhood but prevents move reversal and cycling.

However, under specific conditions called the **aspiration criteria**, a tabu move might be allowed. For example when all moves are tabu or when the resulting solution is better than an aspiration threshold related to the value of the current best solution.

```
current = initial_solution()
for i in range(1, max_iterations):
    N = neighbourhood(current)
    while 0 < N.size():
        move = N.pop_best_move()
        tabu = tabu_list.is_tabu(move)
        if not tabu or (tabu and aspiration(move)):
            candidate = current.apply(move)
            if candidate not in visited:
                current = candidate
                break

    tabu_list.add(move)
    visited.add(current)
    best = min(best, current)
```

Listing 2.1: Tabu Search metaheuristic pseudocode

2.5.4 Genetic Programming

The concept of **genetic programming** (GP) was popularised by John Koza with his book on the "programming of computers by means of natural selection" [Koz92]. It draw its inspiration from the biological evolution and is a specialisation of genetic algorithms to the case of computer programs. Essentially, the genetic programming breeds a population of computer programs to solve a problem. It starts with randomly generated programs and iteratively evolves them using mechanisms modeled upon Darwinian principle of survival of the fittest and genetic recombination in order to obtain better programs.

In the classic GP algorithm a computer program is represented as a syntax tree. Internal nodes of a tree correspond to operator functions, terminal nodes correspond to operands. This allows the use of a prefix notation to efficiently evaluate a tree. For example the tree P_1 from Figure 2.9 can be written as +A(*BC).

Initial population of trees is usually generated using the **full method**, which results in a full tree of given depth with terminals selected only at the lowest level, or the **grow method** where at all levels of a tree nodes are selected from both operators and terminals. A combination of these methods where each is applied with 0.5 probability in known as **half-and-half initialization**.

The initial population is then improved in the iterative cycle of selection, breeding and replacement of individuals. One or two individual programs are selected from the current population with probability based on their fitness. Selected individuals are next used to create new individuals by applying the genetic operations: reproduction, crossover or mutation, each with a specific probability given as a parameter of the GP algorithm. Generated offspring is then used to form a new population (see Listing 2.2).

```
population = init(init_op)
population.evaluate()
for i in range(1, max_generations):
    best = population.select(selection_op)
    offspring = best.breed(crossover_op, mutation_op)
    offspring.evaluate()
    population.replace(offspring)
```

Listing 2.2: Genetic Programming metaheuristic pseudocode

Two most popular selection methods are roulette wheel selection, where probability of selection is proportional to the fitness of an individual and **tournament selection**, where a winner (the one with best fitness) of tournament between randomly chosen k individuals is selected. The bigger the k, the smaller is the chance that a weak individual will be selected.

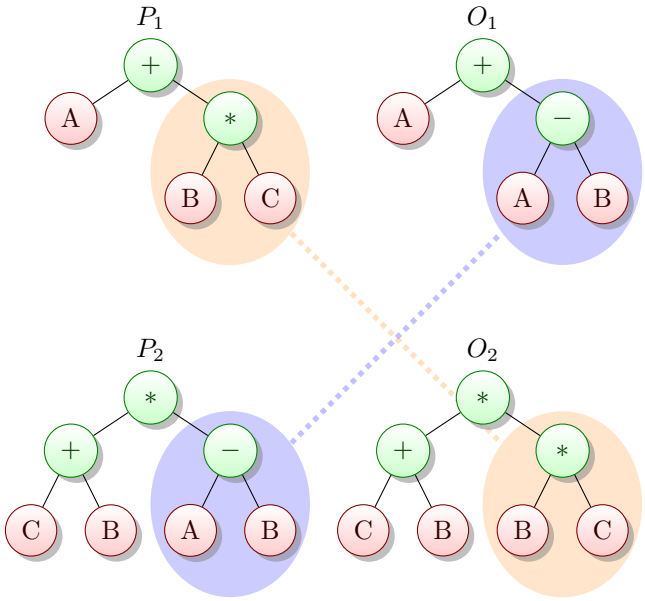

Figure 2.9: Crossover operation generates offspring trees O_1 and O_2 from parents P_1 and P_2.

Most of the new individuals are created using a **crossover** operator that recombines randomly chosen parts of two parent trees to generate the offspring (see an example in Figure 2.9). A small fraction of the offspring is created by the **mutation** operator which introduces a random change to a randomly chosen part of a parent tree. The most common way is to replace a sub-tree with a random tree, but other variants are possible such as sub-tree swap or tree shrink where a node is replaced by one of its child nodes (see an example in Figure 2.10). Finally, the **reproduction** operator copies a few individuals without any changes.

The generated offspring could be included in the new population using different replacement strategies. The simplest one is the **generational replacement** where individuals bred in a current generation simply replace the old population. In the alternative **steady-state replacement** strategy old individuals are replaced by offspring "in place", that is least fit individuals from the previous generation are deleted one by one and newly created individuals take their place in the population. Both these strategies are often extended

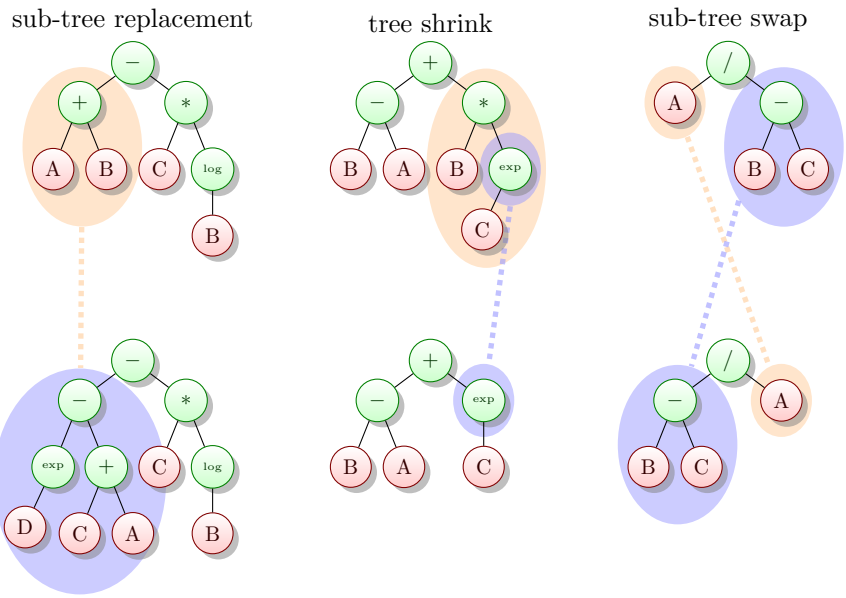

Figure 2.10: Three different mutation operators. Trees in the top row are mutated to generate the trees in the bottom row.

with the **elitism** mechanism, where a small fixed-size elite set of best individuals is never replaced by the offspring.

Genetic programming was shown to be able to generate human competitive results in a number of different areas such as electronic design or game playing. There are several examples of GP based innovations that duplicate functionality of patented solutions or even are novel enough to be patentable themselves [Koz].

2.6 Conclusions

Protein structure prediction is a complex multi-step process that incorporates a range of different methods from sequence and structure comparison, through geometrical features prediction (secondary structures, contact maps, solvent accessibility), to the structure optimisation and clustering. This dissertation touches two aspects of this process: structure optimisation and selection of most common models through clustering (which needs a measure of similarity). However, this work is not about optimisation of the protein structures per se, but about the design of the objective functions (here called energy functions as they tend to represent the potential energy of a protein structure) used in that optimisation process.

The analysis of the state-of-the-art prediction methods has revealed several weak points. First, the decoys used as a training sets in design of the energy functions where biased against the native structure and very different in their nature from what the predictors have to deal with in practice. Second, the RMSD was used as a reference similarity measure despite its many inaccuracies such as a large score penalty for local errors, meaningless absolute value (dependency on the protein length) and sensitivity to outliers that results in emphasis on dissimilar regions. Finally, a very simple, yet restrictive linear combination of energy terms was used to find the best energy function.

In the following chapters this issues will be addressed one by one. First, a tabu search based heuristic solving the Max-CMO problem will be proposed and then added to the ProCKSI server to complement the USM method already implemented there. Then, the ProCKSI server would be modified to enable a large scale comparison of N decoys against a single native structure needed to compute the consensus similarity. After that the information content of the consensus will be compared against the individual measures to verify if there is evidence that it contains the sum of knowledge of individual measures (supporting the **H1** hypothesis). Finally, a genetic programming algorithm will be used to design the energy functions as complex free-form functional combinations of the energy terms (validating the **H2** hypothesis) and the influence of the similarity consensus on the evolvability of the energy functions will be tested (validating the **H1** hypothesis).

Chapter 3

Metaheuristic for Max-CMO

3.1 Introduction

To have a balance set of the comparison methods integrated into the ProCKSI meta-server, and thereby into the similarity consensus, a robust structure comparison method good in discriminating between closely related structures was needed, to complement the already implemented USM method, which is known for its effectiveness in comparing distantly related structures. For these purpose a tabu search based metaheuristic algorithm solving the Max-CMO problem was developed.

As an implementation of this algorithm was not the main objective of the dissertation but an intermediate step to obtain the similarity consensus, the choice of the method was motivated mainly by the implementation convenience. I have already used tabu search in my MSc thesis and thus I was able to develop the algorithm much faster. However, the key elements of the algorithm that contribute to the performance improvement, that is the representation of a solution and the definition of a neighbourhood, are very general and could be applied with the same result in any metaheuristic framework based on the neighbourhood search.

3.1.1 Max-CMO Problem Definition

Maximum contact map overlap is an alignment of two proteins that maximises the structural similarity. This similarity is measured as number of edges that could be matched between two contact map graphs. Figure 3.1) presents an example alignment with optimal CMO value equal to 3. The overlapping (matched) edges are shown in bold.

However, not every edge to edge matching is allowed. The overlap can be formed only if the order of nodes in graph G_1 is also preserved for assigned nodes of graph G_2 (consistent ordering constraint). In other words, for assignments $n_x^{G_1} \rightarrow n_x^{G_2}$ and $n_y^{G_1} \rightarrow n_y^{G_2}$ either $n_x^{G_1} < n_y^{G_1}$ and $n_x^{G_2} < n_y^{G_2}$ or $n_x^{G_1} > n_y^{G_1}$ and $n_x^{G_2} > n_y^{G_2}$ must hold. This constraint is also called non-crossing property of an overlap [XS06].

The Max-CMO problem has been proven to be NP-hard [GPI99] and because of the size of the search space exact algorithms trying to solve it are exponential in the worst case.

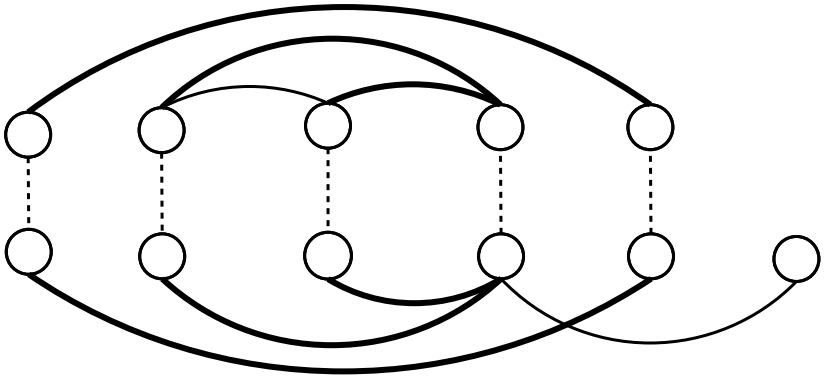

Figure 3.1: Optimal solution of the Max-CMO problem example. Overlapping (matched) edges are in bold.

3.1.2 Search Space

The size of a search space (upper bound) for given two contact maps graphs G_1 and G_2, each of n nodes for simplicity, corresponds to the number of possible k element node assignments between G_1 and G_2, where $k \in \{1 \dots n\}$. Since assignment source nodes (from graph G_1) as well as assignment destination nodes (from graph G_2) could be selected in number of ways equal to:

$$\sum_{k=0}^{n} \binom{n}{k} = 2^n \tag{3.1}$$

the number of all possible assignments is equal to:

$$\sum_{k=0}^{n} \binom{n}{k}^2 = \binom{2n}{n} \tag{3.2}$$

As typical size of a protein is 100 to 500 residues, in most practical cases it is not possible, as expected, to solve Max-CMO problem to optimality by exhaustive search. This become even more relevant in case of protein structure prediction where usually at least several hundreds if not thousands of structures are being compared.

3.1.3 Related Work

To solve the Max-CMO problem many different methods have been applied. Foundations for exact methods was led by Lancia et al. with reduction to maximum independent set problem (non-crossing map) and the integer programming (IP) formulation [LCWI01]. This IP problem was solved by branch and cut algorithm using an upper bound found by linear programming (LP) relaxation and the lower bound found by a local search and a simple genetic algorithm. A year later this method was improved by the use of lagrangian relaxation [CL02]. Also a reduction to the maximum clique problem and a pruning algorithm was proposed in [SBS05].

Another technique of pruning pairs of nodes during search (called reduction by domination) has been recently described in [XS06]. Although it solves to optimality many previously unsolved instances and seems to be the best of exact methods proposed so far, it still

does not cope well with dissimilar proteins and with proteins larger than 60 residues, let alone with the simultaneous comparison of potentially thousands of decoys.

A polynomial time algorithm for crossing contact maps (simplified version of a problem) was proposed by Gramm[Gra04]. This method was not intended to work with graphs with nested edges (edges can only precede each other or cross). Additionally the cross was defined as disjoint (two edges can't share a common node). The author claimed that a single pattern matching could be performed in $O(n^6m^2)$, where n is a size of CM and m is a size of a pattern. However, there are at most 3^m possible patterns to check, so the algorithm is in fact $O(3^m n^6 m^2)$. Later on Li et al. show a flaw in this algorithm and proved the simplified problem to be also NP-hard [LL06].

A metaheuristic approach using memetic evolutionary algorithm was proposed by Krasnogor et al. in [Kra04] and [CHK+02]. Additionally, a comparison of this approach with the lagrangian relaxation was described in [KLZ+03]. Although the quality of results for small and similar proteins was not as good as for the exact method, the memetic algorithm proved to be quite robust method of comparing proteins of different families.

Interesting fuzzy generalisation of the contact map concept and the variable neighbourhood search (VNS) heuristic for solving the generalised fuzzy CMO problem has been proposed in [PKBC+05].

3.2 Tabu Search for Max-CMO

3.2.1 Representation

Heuristic methods described in literature represents the solution of Max-CMO problem as an alignment of two proteins [LCWI01, PKBC+05, Kra04]. Such alignment is constructed by node to node assignments between two contact map graphs and reflects the biological interpretation of the solution. However, it is possible to propose an alternative representation based on the graph isomorphism problem.

To do that, I used a line graph of the original contact maps graph. Now, when a node to node assignment between two line graphs is made, it corresponds to the assignment of edges between original contact map graphs (see Figure 3.2). The problem can be then restated as order restricted maximum common subgraph isomorphism.

Let's assume two contact map graphs G_1 and G_2. In ideal case, nodes of graph G_1 would be assigned to nodes of G_2 and whenever there is an edge between two assigned nodes of G_1, there would be also an edge between corresponding nodes of G_2. This would then mean that a subgraph of G_1 is isomorphic to a subgraph of G_2.

Not any common subgraph, however, satisfy the consistent ordering constraint. Moreover, it is not straightforward to detect a violation of this constraint while operating on the line graphs instead of the original contact map graphs (see page 45 in next section). The advantage of operating on subgraphs is the ability to restate the problem in terms of isomorphism of vertex-induced subgraphs.

A vertex-induced subgraph $G_s(V_s, E_s)$ consists of subset V_s of the nodes of the original graph G and all the edges E_s connecting nodes from V_s in G. This means that whenever $n_x^{G_1} \rightarrow n_y^{G_2}$ node to node assignment is made, all assigned nodes adjacent to $n_x^{G_1}$ needs to have their G_2 correspondents also adjacent to $n_y^{G_2}$ to make the assignment feasible.

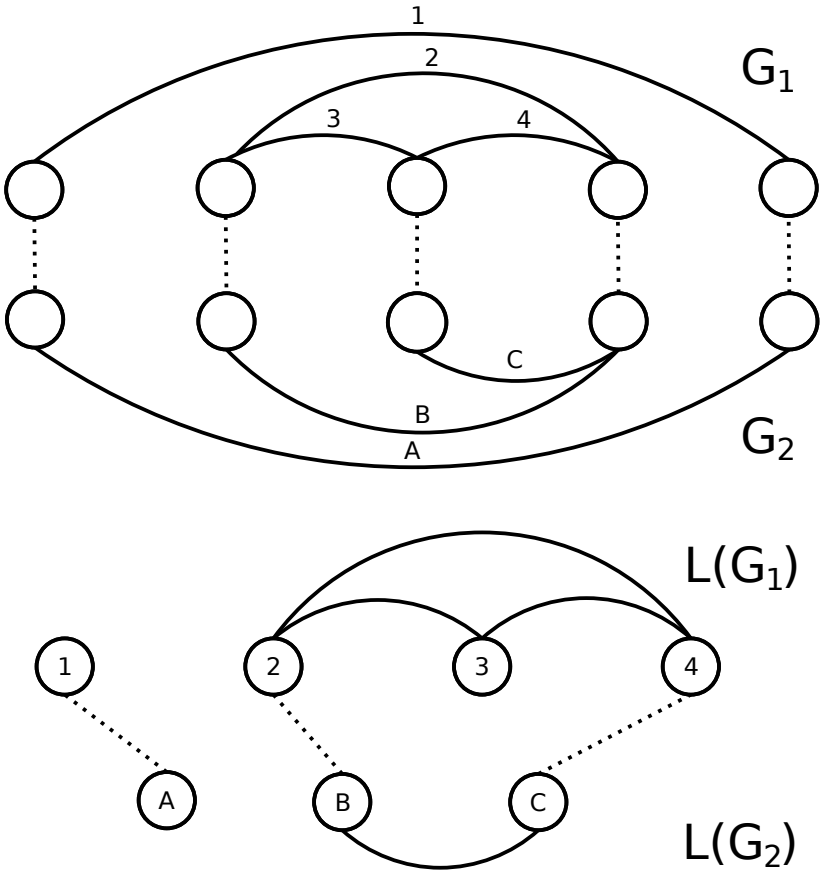

Figure 3.2: Original Max-CMO problem (top) and the corresponding line graph representation (bottom).

The maximum common subgraph problem has been proven to be NP-hard [GJ79] and as its order restricted version is not easier it is also NP-hard.

3.2.2 Search Efficiency Improvements

All metaheuristic search methods are based on common framework of iterative exploration of the neighbourhood of current solution. The neighbourhood is in general a set of moves — operations on attributes of a solution that moves the search process from current solution to the neighbouring one.

```
for i in range(1, max_iterations):
    ...
    N = neighbourhood(current)
    best_move = N.get_best_move()
    ...
    current.apply_move(best_move)
    ...
```

Listing 3.1: Main loop

The most computationally expensive part of the method is the generation of the neighbourhood, as for each candidate move that belongs there its feasibility and the cost/gain value needs to be evaluated.

```
for move in moves_set:
    if move is feasible:
        value = evaluate(current, move)
        N.add(move, value)
```

Listing 3.2: Inner loop

To evaluate a Max-CMO problem solution the number of matching edges needs to be found. In alignment based representation for each node of graph G_1 all adjacent nodes and their G_2 counterparts needs to be check, which gives computational complexity of $O(n * d(V))$. If we assume an iterative change, then the same check needs to be applied only to nodes being a part of a new assignment, which lowers the complexity to $O(d(V))$. In line graph representation each node to node assignment corresponds directly to a match between two edges. This not only make each assignment meaningful (in alignment based representation not every assignment changes the solution value), but also lowers the complexity to $O(m)$ in general case and $O(1)$ if we assume an iterative change.

To decide if a move is valid the ordering constraint violation needs to be examine. In alignment based representation the new assignment $n_x^{G_1} \to n_y^{G_2}$ needs to be compared against all existing $G_1 \to G_2$ assignments in the solution and dismissed as infeasible whenever $(n_x^{G_1} - n_i^{G_1}) * (n_y^{G_2} - n_j^{G_2}) < 0$ for any $i, j \in \{1 \dots n\}$ for which $n_i^{G_1} \to n_j^{G_2}$ assignment exists. This process needs $O(n)$ operations. In line graph representation similar examination is not possible as the information about residues order is lost as shown in Figure 3.3). It is not possible to distinguish between $1 \to B$ and $2 \to B$ assignments and decide that the latter is infeasible.

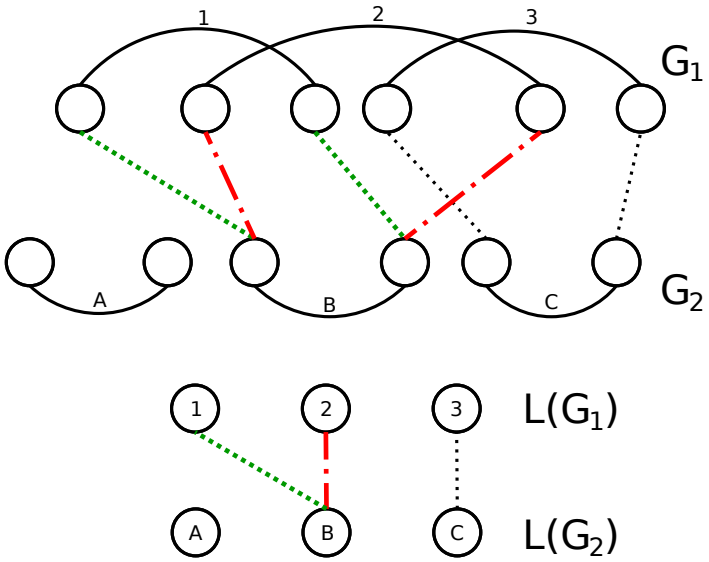

Figure 3.3: Move feasibility. Existing assignments are marked with black thin dotted line, feasible moves are marked with thick dotted green lines and the infeasible moves with thick red dash dotted lines.

As a result, to decide if a move is feasible, a reference to the alignment based representation is needed. However, it is possible to decide feasibility in $O(1)$ time at the cost of updating an extra memory structure with assignments constraints. Since the move is infeasible as soon as a single ordering constraint is violated, it is enough to compare it against the neighbouring assignment. If ordering condition is not violated there, it is also not violated

for further neighbouring assignments. An example of a new move and it's neighbouring assignments is presented in Figure 3.4. The dashed red lines are marking infeasible assignments.

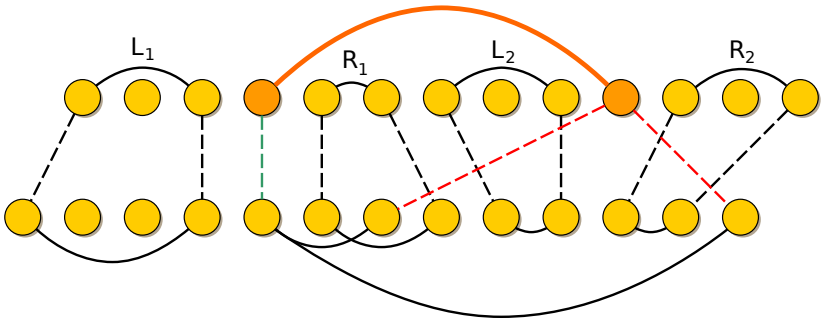

Figure 3.4: Move feasibility constraints.

For each node of the graph G_1 information about its neighbouring assignments is stored. This structure is updated once per iteration when new current solution is chosen (see Listing 3.1) which takes $O(n)$ time.

Summing up the proposed improvements, the total efficiency gain per iteration is:

- evaluation of solution: $O(m^2 * O(1))$ vs. $O(n^2 * O(d(V)))$
- examination of feasibility: $O(m^2 * O(1) + O(n))$ vs. $O(n^2 * n))$

Knowing that the total sum of vertex degrees in a graph is equal to twice number of edges, we can state the relation between m and n as below:

$$\begin{aligned} d(V) &= \frac{\sum_{i=1}^{n} d_i(V)}{n} = \frac{2m}{n} \\ n * d(V) &= n * \frac{2m}{n} = 2m \end{aligned} \tag{3.3}$$

If we take into account that for line graph representation a larger neighbourhood is searched ($O(m^2) = \frac{1}{4}n^2 * d(V)^2 \geq n^2$ for $d(V) \geq 2$), the overall efficiency gain is equal to $O(n + d(V) - 1)$ or simply $O(n)$.

3.2.3 Tabu Search Implementation

I have implemented improvements described in previous section using the tabu search [Glo89, Glo90, HTdW95] metaheuristic framework. As a neighbourhood operator I used simple 1-opt node to node assignment. To disallow state revisiting in the process of search I have implemented a hash based long term memory (LTM). For efficiency reasons short term memory (tabu list) was constructed with two separate data structures: a hash set (for $O(1)$ search) and a priority queue (for $O(1)$ update).

To examine the new method of testing the move feasibility an empirical approach was used. For each iteration the feasibility of all moves in the neighbourhood was decided with both methods: naive ($O(n)$) and the new constraint based $O(1)$. Then the decision was compared to make sure that new approach yields exactly the same results as the old one.

The experiment was run on reference protein data sets used before by Lancia [LCWI01], Skolnick [LCWI01], Sokol [URLa], Chew and Kedem [CK02] and Leluk, Konieczny and Roterman [LKR03]. Each pairwise comparison was repeated 5 times with different random seed, since the search process was starting from a randomly constructed solution.

3.3 Results

The tabu search based algorithm solving the Max-CMO problem was tested on 5 test sets (see Section 3.2.3). A statistical summary of proteins in each test set is given in Table 3.1. The length of protein (number of nodes in the contact map graph) is denoted by n and number of contacts in the protein (number of edges in the contact map graph) is denoted by m.

		min		**max**		**mean**	
test set	**size**	**n**	**m**	**n**	**m**	**n**	**m**
Sokol	9	21	45	62	168	52	139
Leluk-Konieczny-Roterman	6	337	990	421	1307	383	1155
Chew-Kedem	35	90	230	497	1508	179	526
Skolnick	40	66	181	255	815	155	460
Lancia	269	64	139	72	243	68	179

Table 3.1: Statistics of the test sets. n is the number of nodes, and m is the number of edges in the contact map graphs from the set.

3.3.1 Algorithm Performance

To measure the improvement in the algorithm performance introduced by the constraint based feasibility check (see Section 3.2.2), new version of the algorithm has been compared to the naive $O(n)$ check. For each pairwise comparison from single test set the execution time is a mean of 5 single runs of the algorithm. For each run different random seed was used (Mersenne's primes $(2^n - 1)$ for $n \in \{7, 13, 17, 19, 31\}$). The initial solution was chosen randomly. Two stop conditions were used: empty list of moves (all tabu) or 100 iterations without any improvement. Tabu tenure factor has been set to 10% of the number of contacts in the shorter protein.

Blue and red dots in Figure 3.5 represent respectively naive and constraint based check. Numbers along the X axis correspond to the size of instance (i.e. the product of number of contacts of proteins being compared), where Y axis shows the execution time.

For test sets where the range of protein sizes is limited (e.g. Sokol set), it is hard to observe how the difference in performance changes with the instance size. It is clear, however, if we analyze the plot for Skolnick set (enlarged). For the smallest instances (48k) the new algorithm is about 100 times faster. For 10 times greater instances (560k) the ratio increases to about 1000 indicating linear dependency on the instance size. This confirms theoretical performance gain derived from complexity analysis.

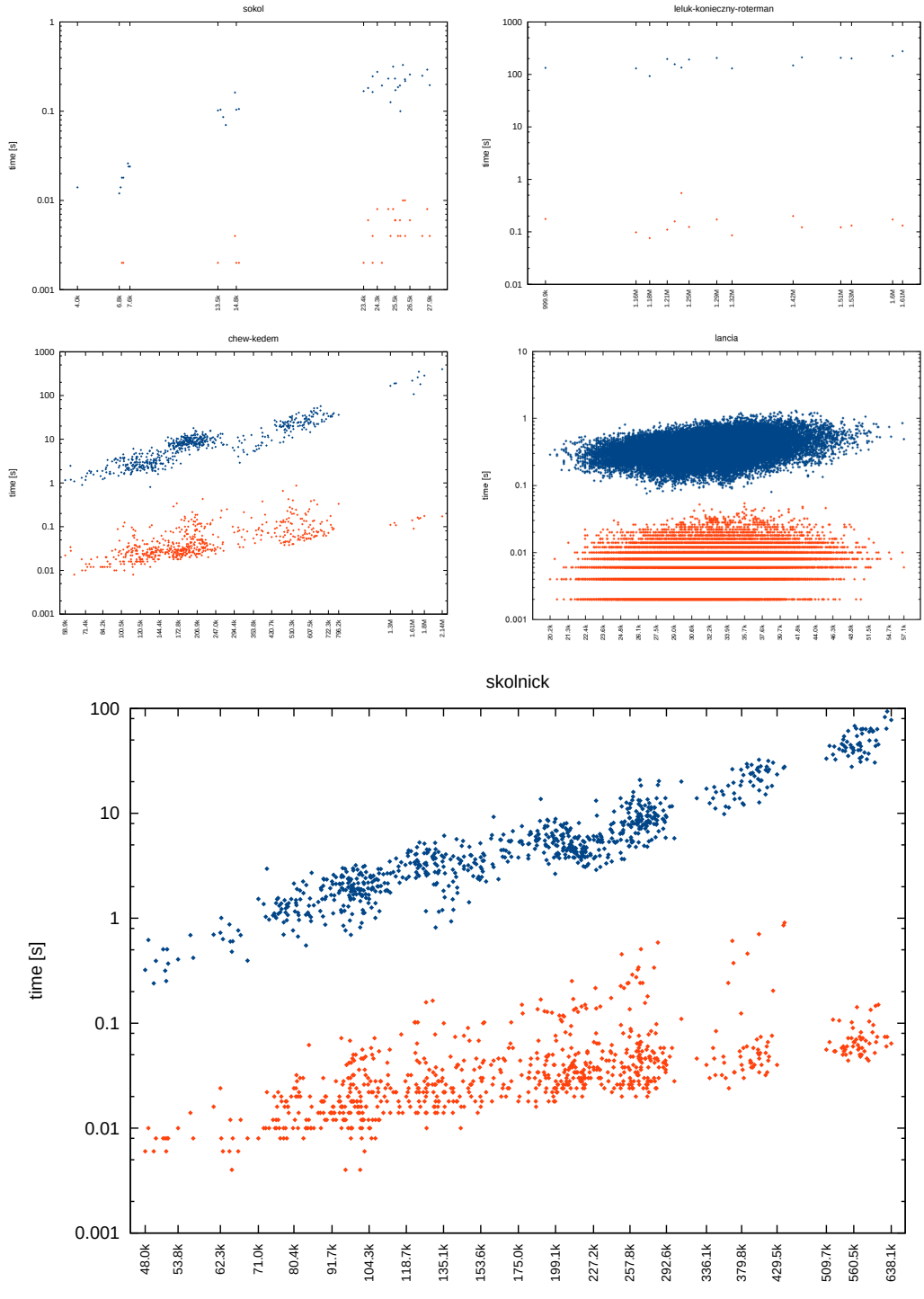

Figure 3.5: Max-CMO heuristic performance comparison. Each plot shows execution time vs. the instance size. Blue dots represents the naive $O(n)$ feasibility check, the red dots represent the new constrained based check.

3.3.2 Quality of Solutions

Choosing the Best Move

The line graph representation introduces a new shape of solution space, where a small change in current state results in a small change in the value of objective function (low ruggedness) and improvements in part of the solution do not compromise past achievements. Although these are the features of a good landscape [WA96], still searching it is not easy.

Main difficulty in the search process guidance, as described in Section 3.2.3, was the choice of the best move in a current state, when many moves with the same value of objective function exist. This feature of the landscape is shown in Figure 3.6. All green states improve the value of the current orange state by the same amount. Analogously all red states equally decrease the value of solution.

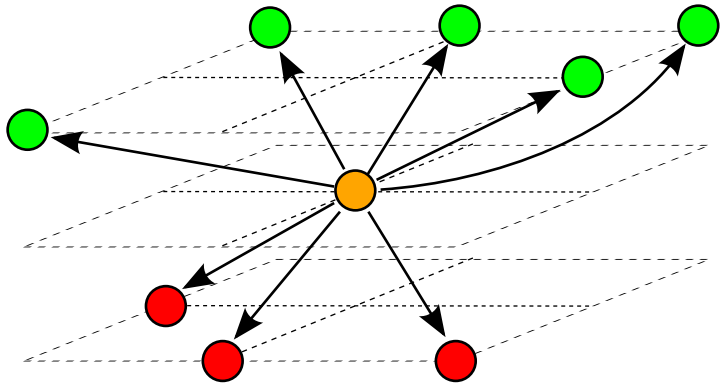

Figure 3.6: For a current state marked with orange there is a number of uphill (green) and downhill (red) moves with the same value of objective function (+1 and -1 respectively) and the choice of the best move is not obvious.

To solve this problem an additional preference has been applied to the group of moves with equal value of objective function. Since the preference is simply a secondary sorting criterion, it does not change the value of the original objective function, so the definition of similarity measure also does not change.

The goal of each additional preference is to bias the choice towards more promising moves using problem specific knowledge. Several different methods of preferring one solution to the other has been proposed and tested:

- **length** - similarity of edge lengths
- **degree** - similarity of degrees of edge nodes
- **position** - similarity of relative position of edge nodes in the graph
- **min_distance** - distance to a closest assigned edge

The idea behind the first three methods is to compare local features of the graphs and choose the move that adds assignment of most similar edges or removes assignment between most dissimilar ones. The last method increases local density of assignments.

Figures 3.7–3.11 show the comparison of different preference methods for all 5 test sets. The top scatter plot on each figure show the value of solution (overlap) for each pairwise

comparison, sorted by instance size. The bottom bar plot present the same data as a percentage histogram of ranks obtained by each methods across all pairwise comparisons.

As in performance comparison each overlap value on the plot is the mean of 5 runs. The stop condition was 100 000 iterations without an improvement. When list of moves was empty, the search was restarted from a new random solution. Tabu tenure factor has been set to 30% of the number of contacts in the shorter protein.

The use of restarts increased the quality of best solution about 2-3 times compared to the single run. Increased number of iterations only slightly improved the quality of solution and no influence was observed if set above 100k. Method denoted on plots as **restart** is a reference implementation where no preference is used. The choice of move though, is not random but deterministic and depends on the sorting algorithm.

An important observation here is that the instance difficulty is not strictly related to the instance size but also to the protein similarity. The clusters of similar proteins with higher values of the overlap are most clearly visible for the Chew-Kedem and Skolnick test sets. It is also worth to notice that for some comparisons the use of a preference did not improve the quality (none of the methods was strictly better than **restart**).

Clearly **degree** was the worst method. In fact it introduces a negative bias, since for 80% of comparisons quality is lower than for **restart**. The best method was **length** dominating the first two ranks with **min_distance** following it closely, except the Leluk-Konieczny-Roterman set where it was ranked 4th. This together with part of the Chew-Kedem plot for big instances shows that **min_distance** strategy of dense assignments does not lead to good solutions for long proteins.

The behaviour of **position** method was very interesting. It was frequently ranked last as well as first. There were several comparisons in each test set where overlap value for **position** was significantly higher than those obtained by all the other methods. This happened for proteins of similar length and could be observed on the Lancia plot. The cluster of high overlap values emerges there around the mean instance size.

Changing the Tabu Attributes

To further improve the quality of solutions several variants of tabu search algorithm were tested. First modification was a change in definition of the tabu list. The coarse grained list from previous experiments uses the source edge of an assignment as the tabu attribute. This way the source edge with previously removed assignment could not be reassigned to any target as long as it is tabu. New fine grained list is using both source and target edge as a tabu attribute. This way the source edge with removed assignment cannot be reassigned only to the previously assigned target edge (not to any as before).

The fine grained approach increases the number of possible moves and let the search process explore more solutions in a single run. However, it becomes more difficult to choose the size of the list (tabu tenure factor) that would guide the search effectively for all instances.

On Figures 3.12–3.16 the modified tabu list method is denoted as **tabu**. Two methods **tabu_length1** and **tabu_length2** combine **tabu** method with length based preference. First one uses the additional preference only for new assignments (add moves) and the other, as in all previous experiments, uses it for both add and remove moves. The

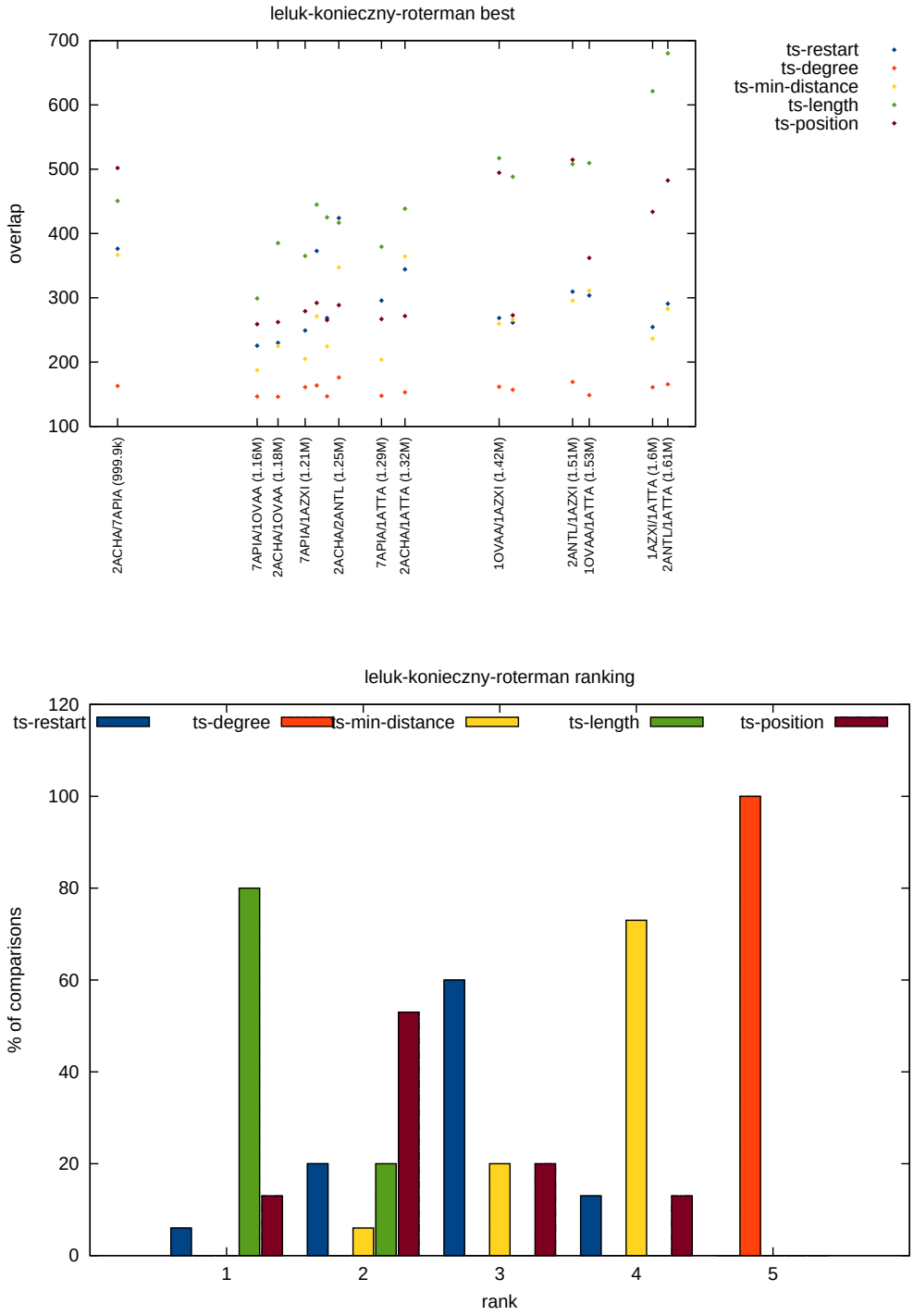

Figure 3.7: Comparison of preference methods on the **Leluk-Konieczny-Roterman** set. The overlap values for different instance sizes are shown on top. Histogram of ranks across all comparison instances for each method is shown on the bottom.

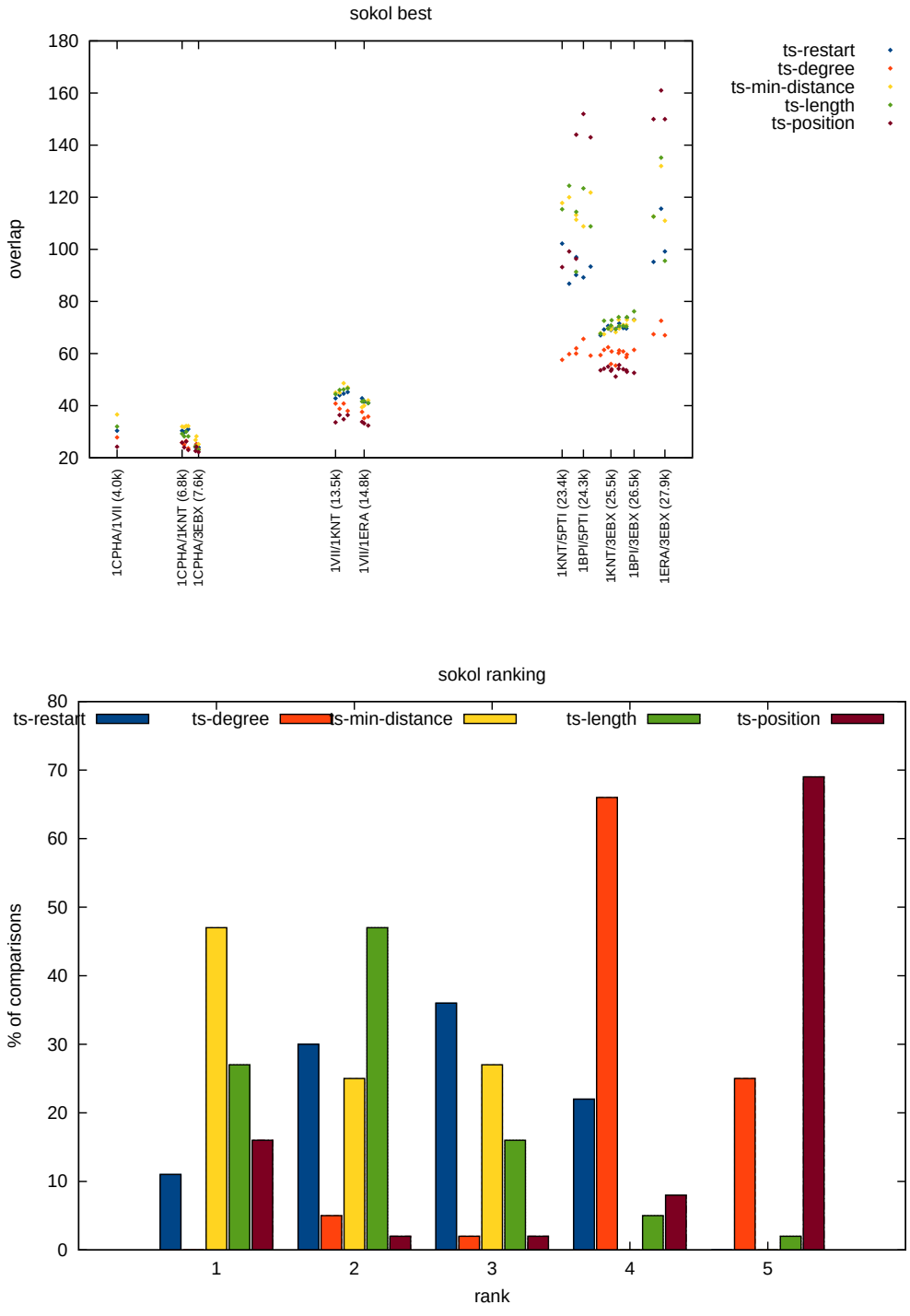

Figure 3.8: Comparison of preference methods on the **Sokol** set. The overlap values for different instance sizes are shown on top. Histogram of ranks across all comparison instances for each method is shown on the bottom.

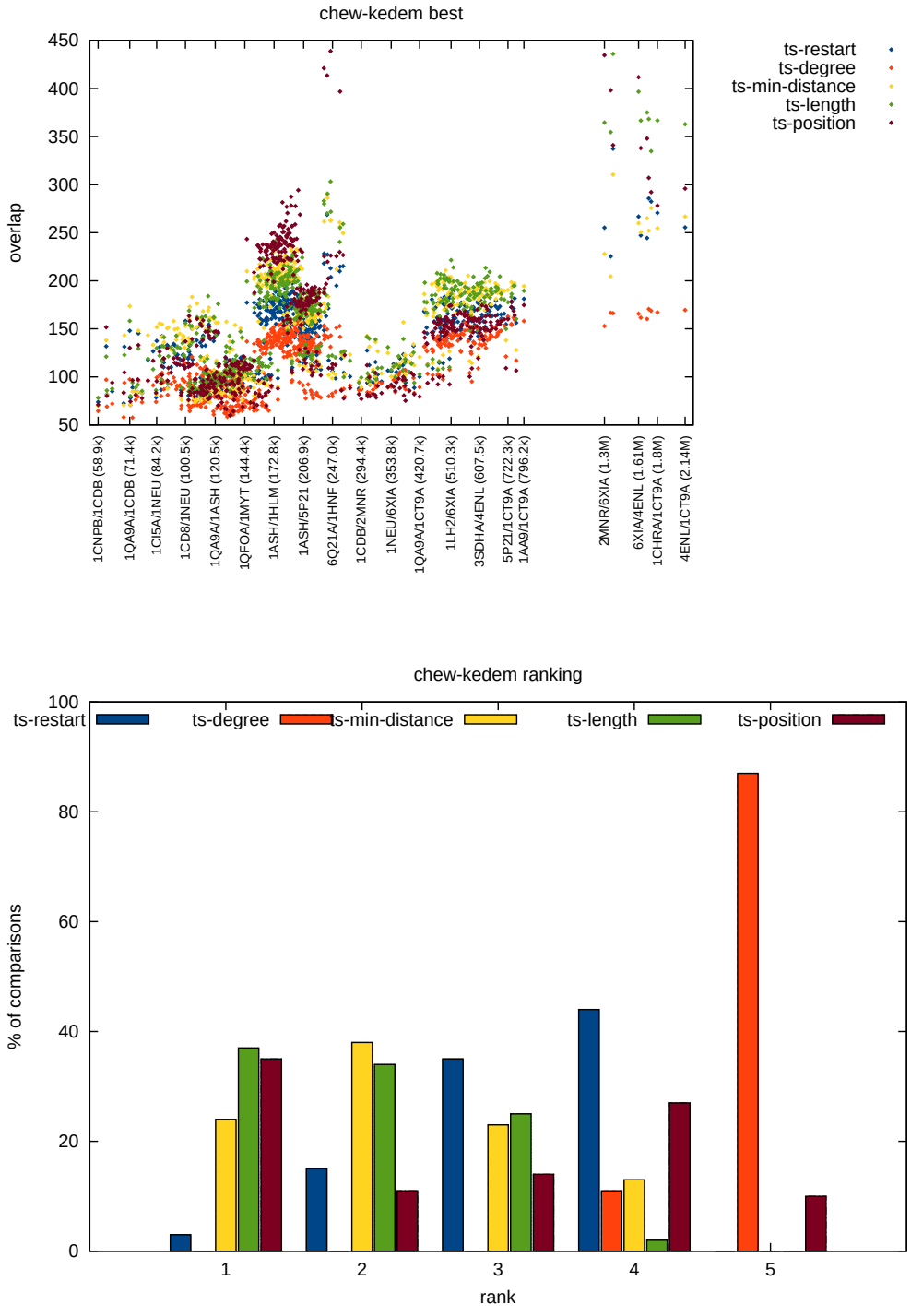

Figure 3.9: Comparison of preference methods on the **Chew-Kedem** set. The overlap values for different instance sizes are shown on top. Histogram of ranks across all comparison instances for each method is shown on the bottom.

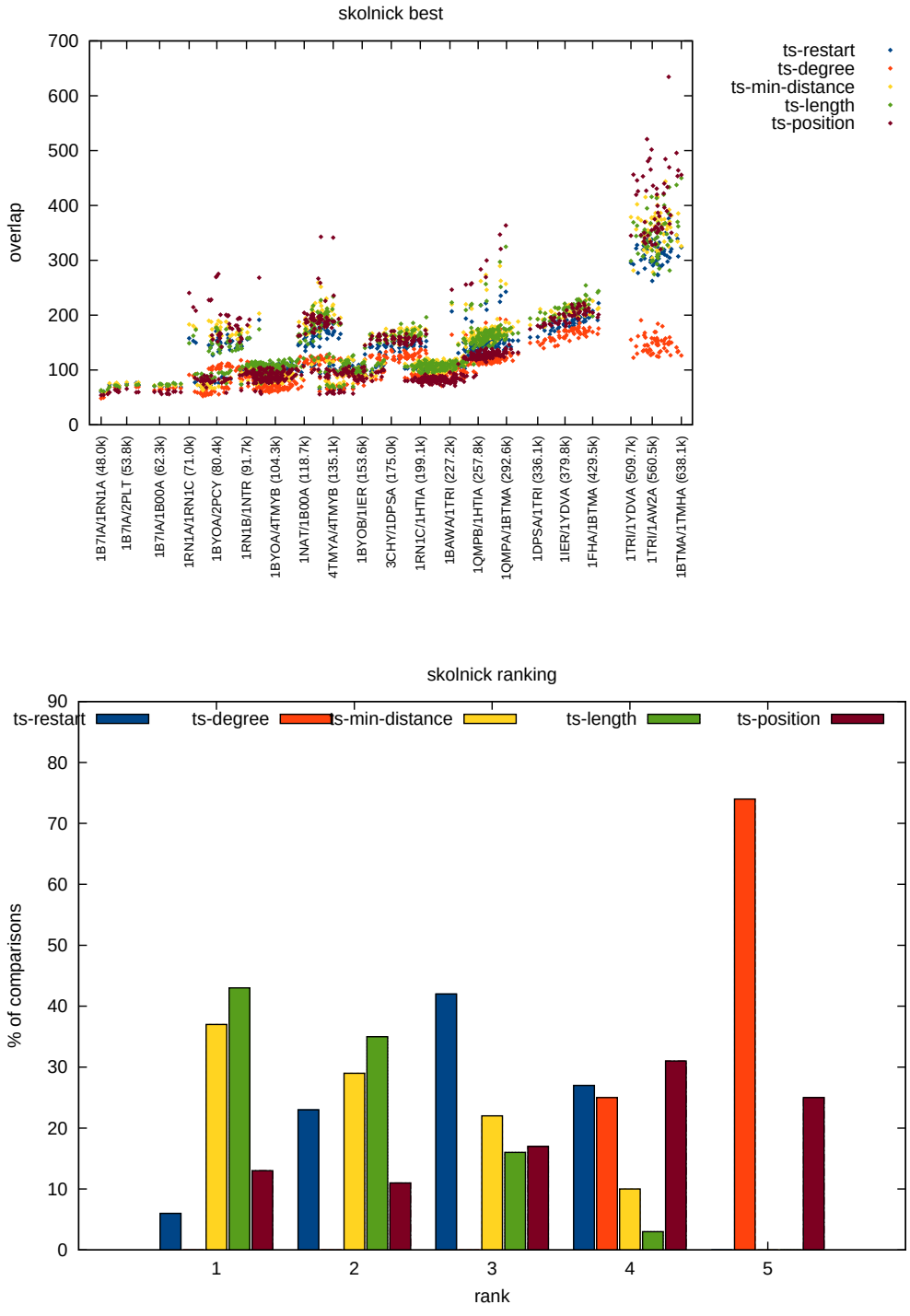

Figure 3.10: Comparison of preference methods on the **Skolnick** set. The overlap values for different instance sizes are shown on top. Histogram of ranks across all comparison instances for each method is shown on the bottom.

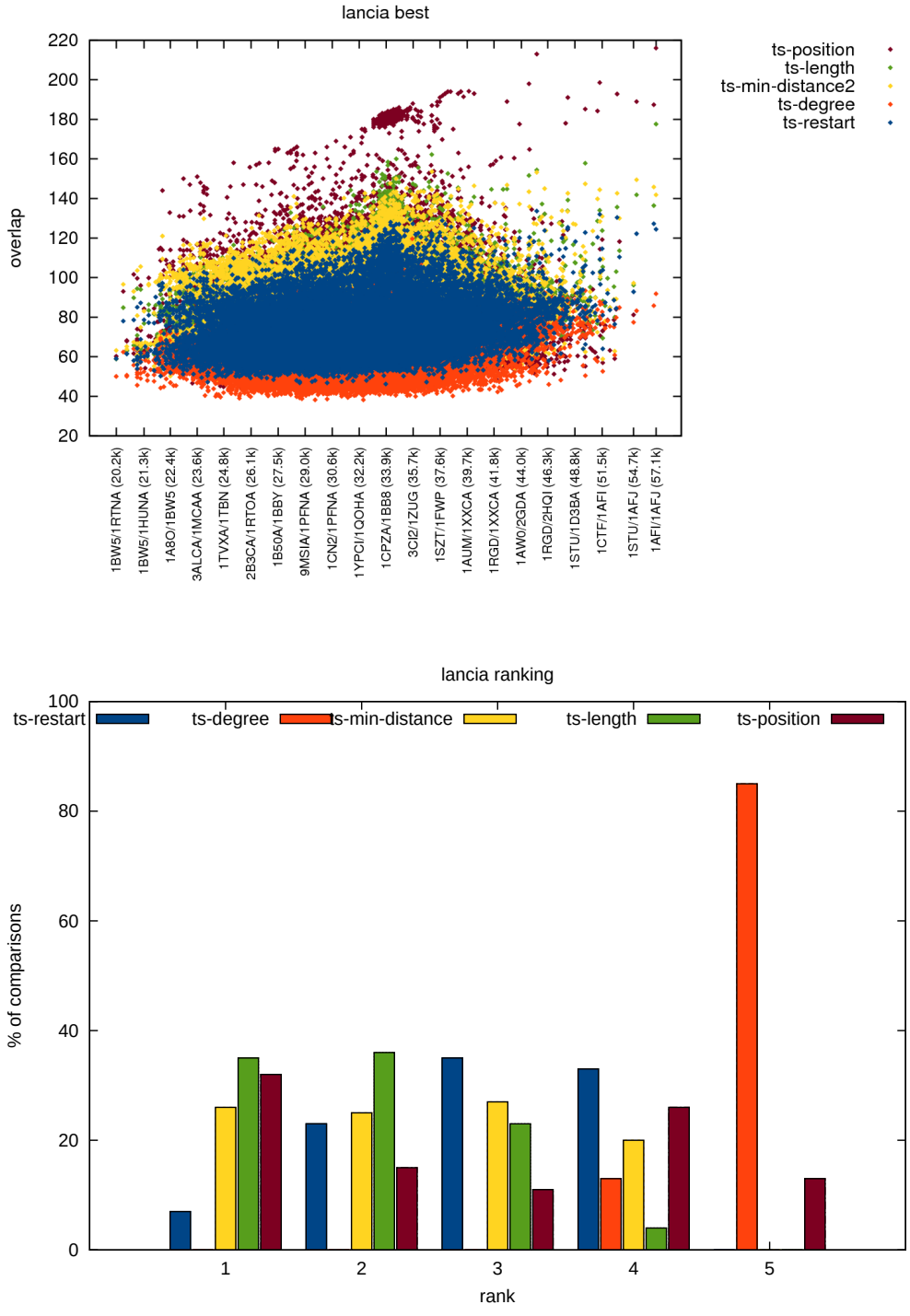

Figure 3.11: Comparison of preference methods on the **Lancia** set. The overlap values for different instance sizes are shown on top. Histogram of ranks across all comparison instances for each method is shown on the bottom.

experiment set up was the same as in previous experiment except of the tabu tenure factor, which was set to 50% of the number of contacts, and the tabu list being now cleared after each restart.

Beside the comparisons in the Leluk-Konieczny-Roterman set involving long proteins, the tabu modification alone was able to improve the quality of solutions and was as good or even better than the **length** method. For some instances, however, the quality decreased significantly below **restart** method level. Clear clusters of such instances are visible on Sokol (1BPI/5PTI or 1ERA/3EBX) and Skolnick plots (1TRI/1YDVA or 1BTMA/1TMHA). These groups of proteins have equal or similar length and very similar contacts. It seems that the arbitrary chosen tabu tenure is not selected well for these instances.

Inclusion of the length based preference further improved the quality. For all test sets except again Leluk-Konieczny-Roterman, **tabu_length1** and **tabu_length2** methods were clearly much better than **length** and **tabu**, with **tabu_length1** slightly ahead of **tabu_length2**.

Using 2-opt Operators

Further modification were made to neighbourhood operators. Standard add and remove moves were extended with 2-opt look ahead and change operators.

The look ahead operator checks how many new feasible assignments (add moves) exist after making a particular move. Although every new assignment increases the value of solution, it also introduces new restrictions to solution feasibility. As a result, the most beneficial assignment would be the one least limiting the possibility of further improvements. Furthermore, when no improvement is possible the removal of most restrictive assignment would maximise the probability of improvement in next move.

The change operator is used when no add moves are possible and is simply a reassignment of an existing source edge to a different target. It may be seen as effective version of combined 2-opt remove-then-add operator. It should allow to move across the solution space without the solution improvement in an attempt to drift away from the local optimum.

Figures 3.17–3.20 present results of four new methods based on the **tabu** method from previous experiments. The look ahead operator was used in **tabu_lookahead1** and **tabu_lookahead2** methods. In the first method, the operator is applied only to the removal of assignments. In the second method it is applied to both add and remove moves. As the neighbourhood searched by the look ahead operator is large (square of 1-opt neighbourhood size $\approx O(m^4)$), the **tabu_lookahead2** method is computationally expensive. The method denoted as **tabu_length_lookahead** is a combination of the **tabu_lookahead1** and **tabu_length1** from previous experiment. Change operator was used in **tabu_change** method.

Surprisingly, the use of the change operator did not improve the quality of solutions. For all sets the **tabu_change** method was clearly the worst in the group. The look ahead operator was not very successful either. The **tabu_lookahead1** method was not significantly better than **tabu**. Except Sokol set, it got higher ranks more often but still its most frequent rank was lower. The difference for Sokol plot appears here because of

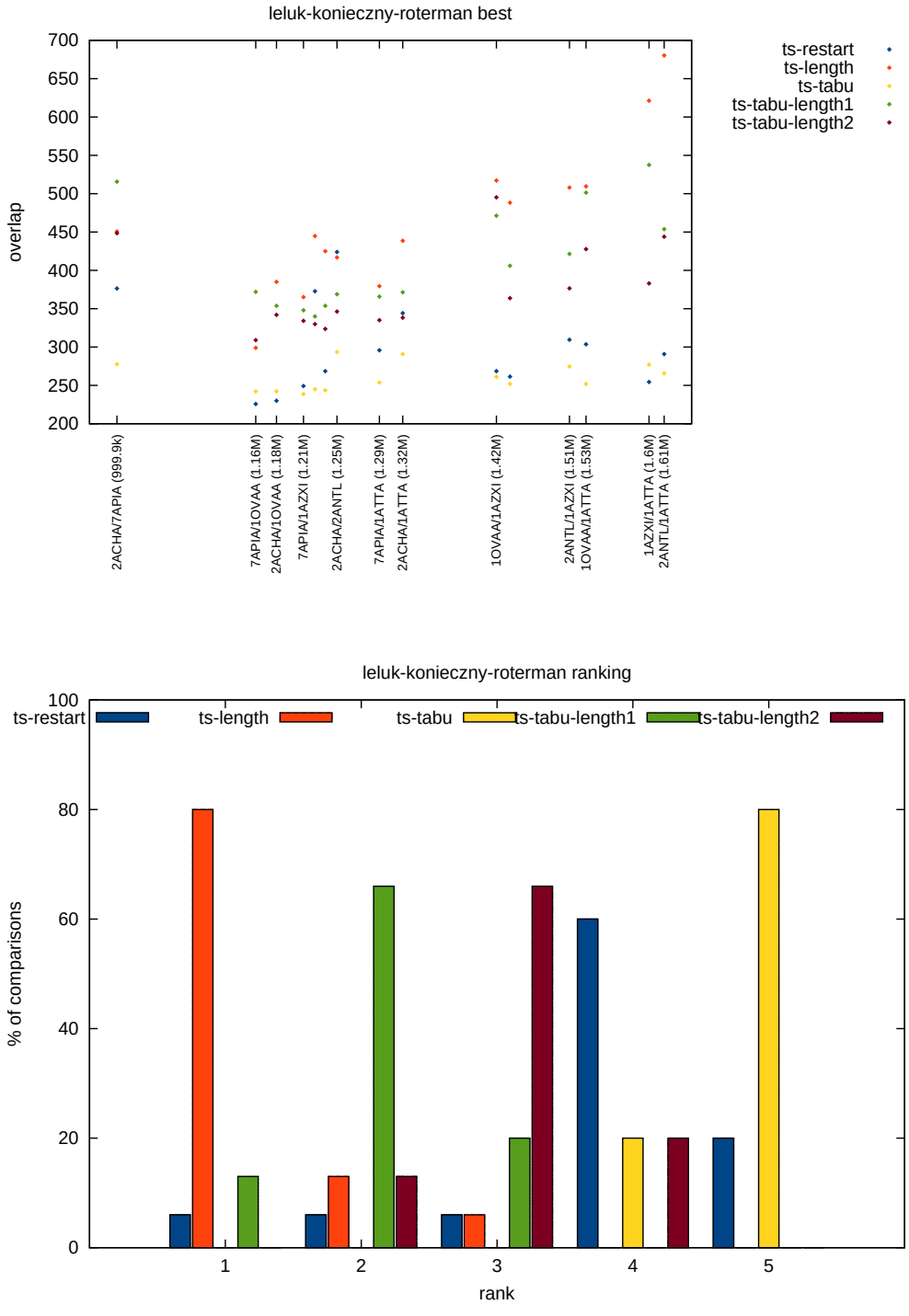

Figure 3.12: Comparison of modified tabu attributes on the **Leluk-Konieczny-Roterman** set. The overlap values for different instance sizes are shown on top. Histogram of ranks across all comparison instances for each method is shown on the bottom.

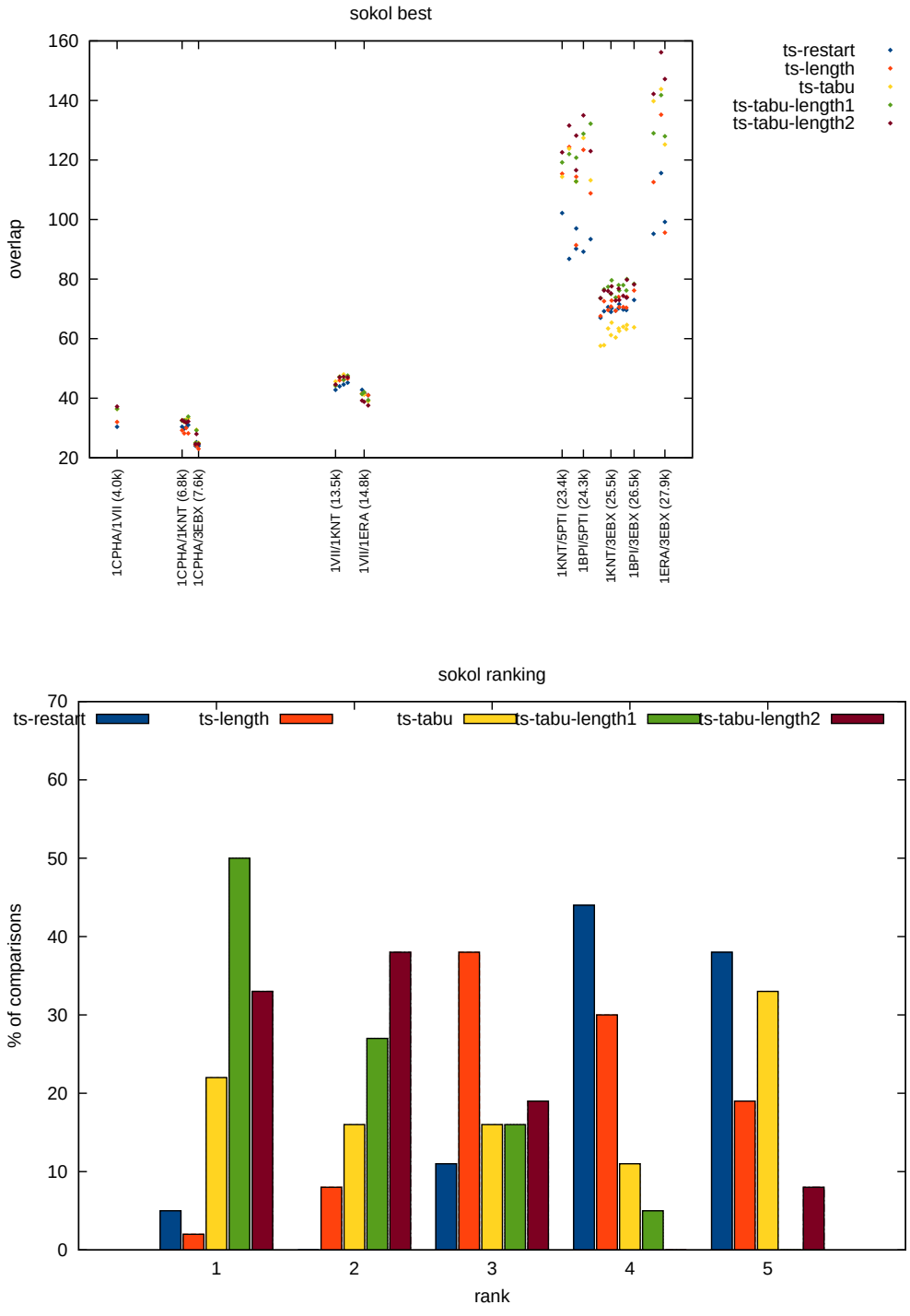

Figure 3.13: Comparison of modified tabu attributes on the **Sokol** set. The overlap values for different instance sizes are shown on top. Histogram of ranks across all comparison instances for each method is shown on the bottom.

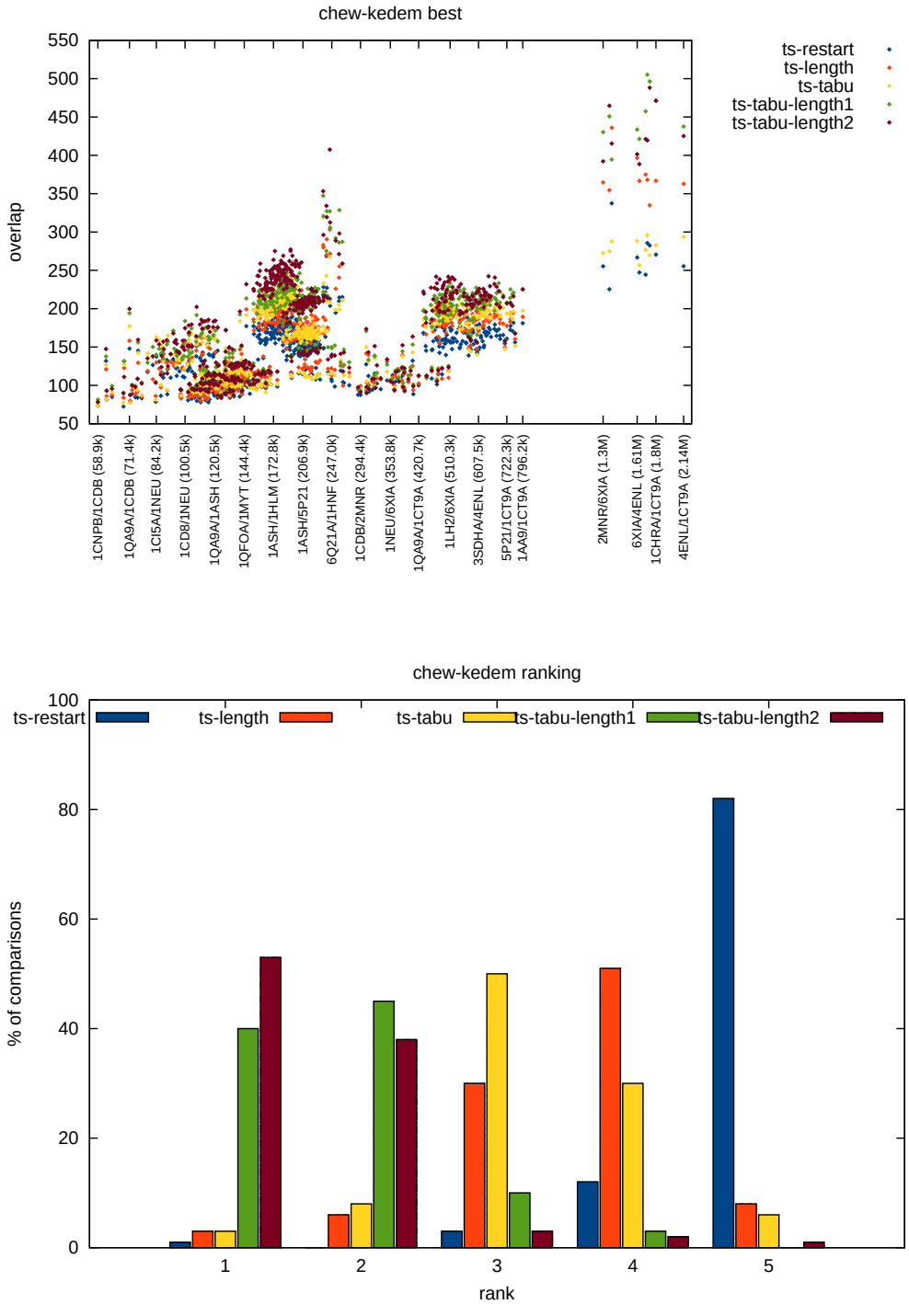

Figure 3.14: Comparison of modified tabu attributes on the **Chew-Kedem** set. The overlap values for different instance sizes are shown on top. Histogram of ranks across all comparison instances for each method is shown on the bottom.

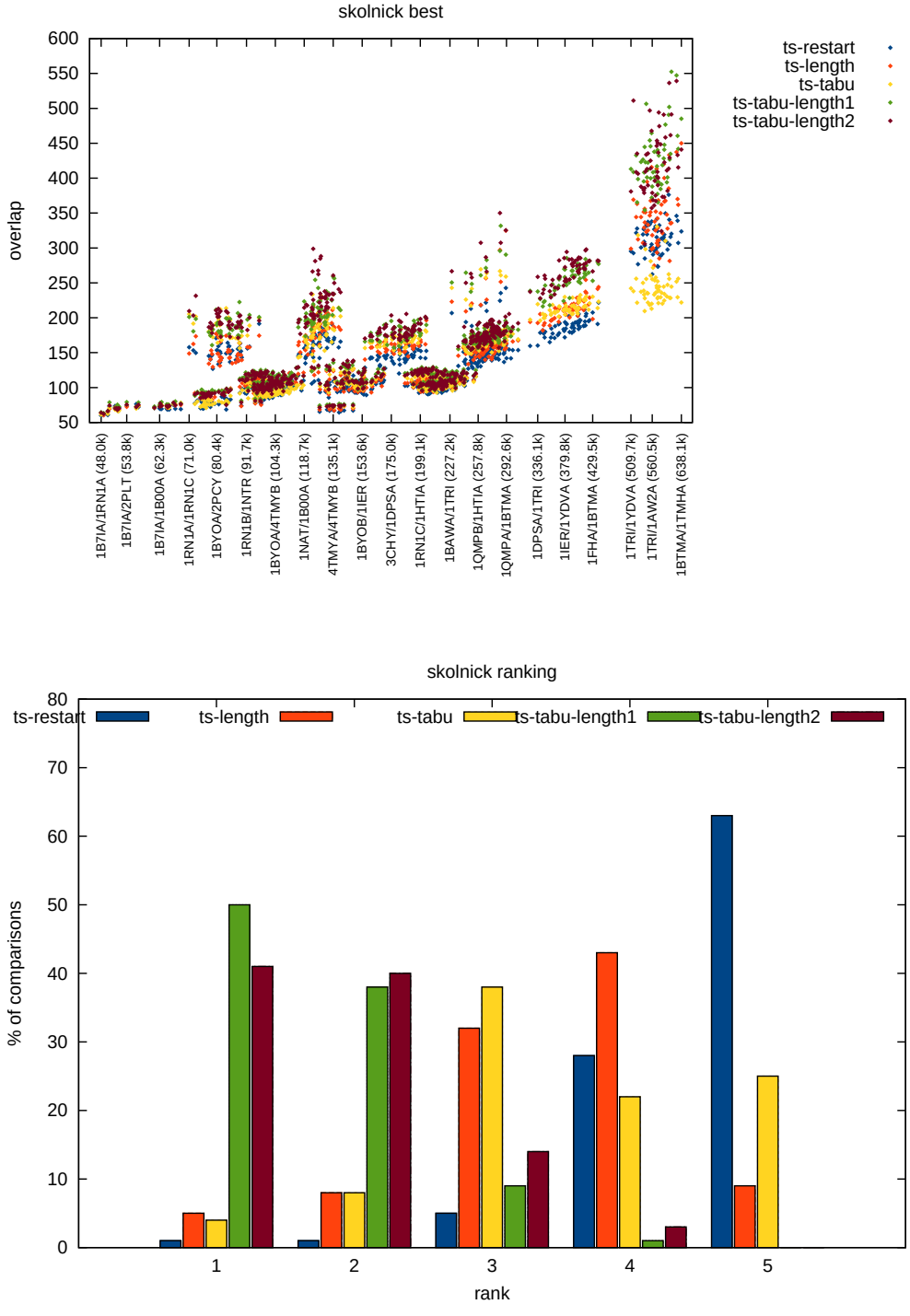

Figure 3.15: Comparison of modified tabu attributes on the **Skolnick** set. The overlap values for different instance sizes are shown on top. Histogram of ranks across all comparison instances for each method is shown on the bottom.

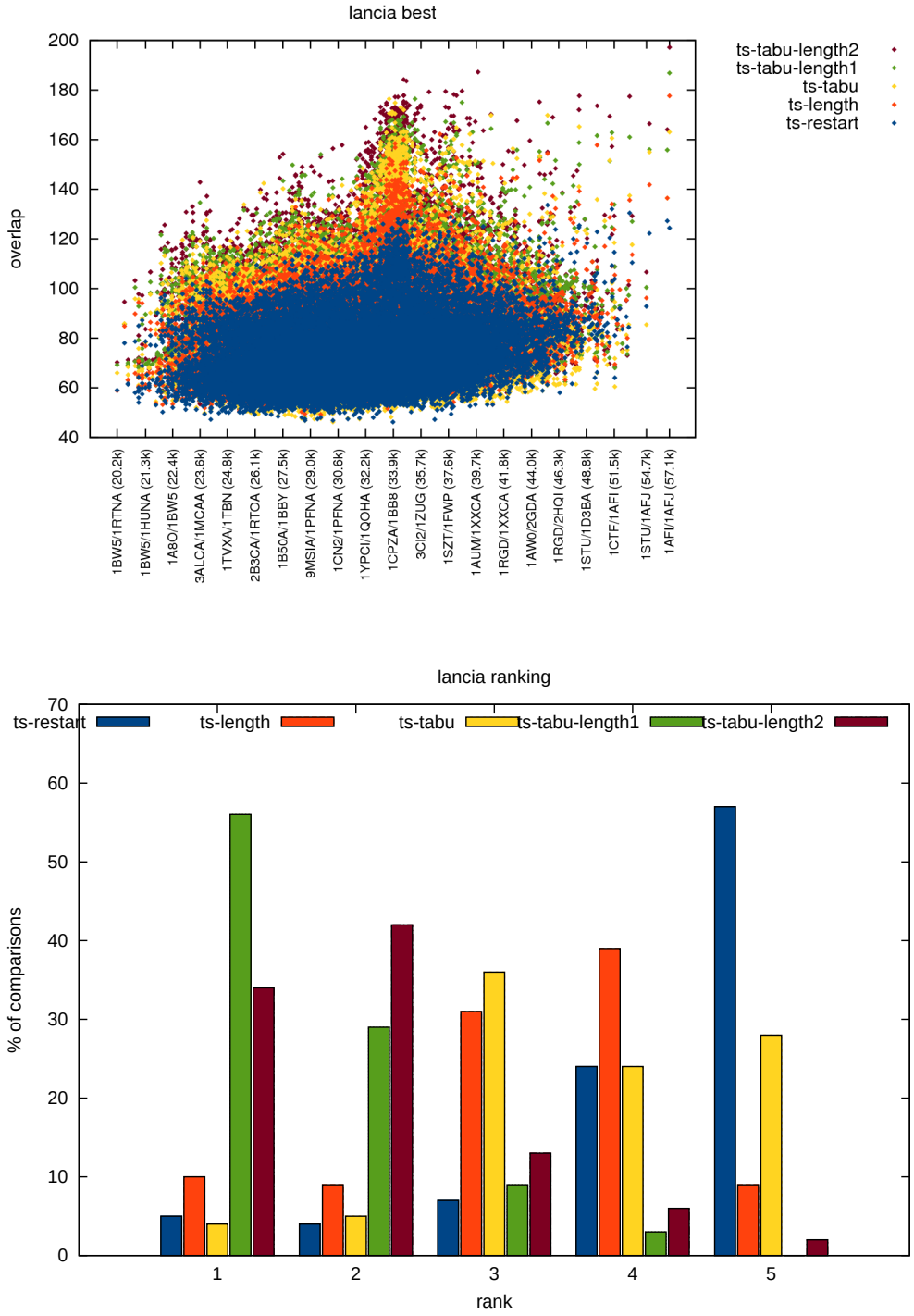

Figure 3.16: Comparison of modified tabu attributes on the **Lancia** set. The overlap values for different instance sizes are shown on top. Histogram of ranks across all comparison instances for each method is shown on the bottom.

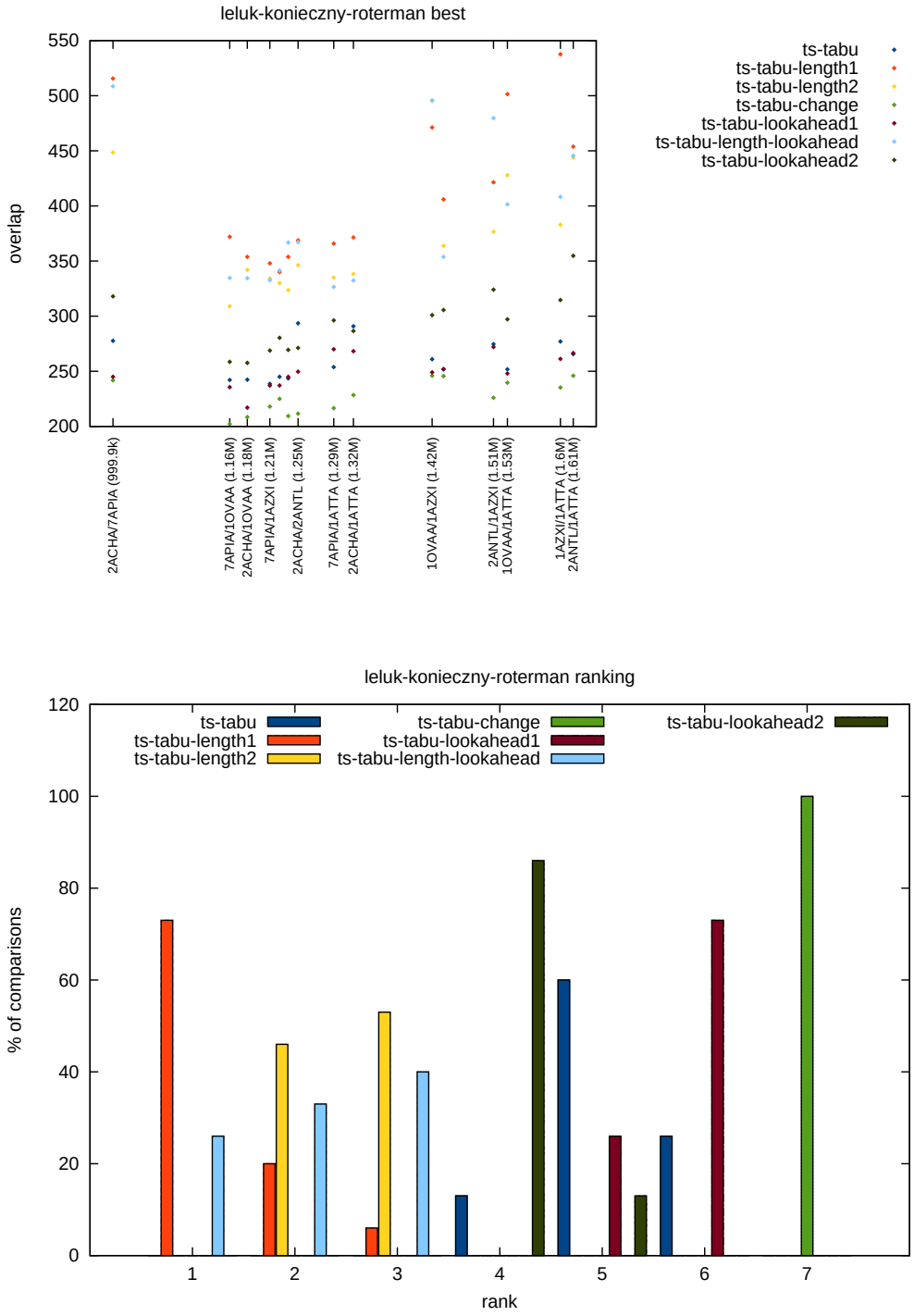

Figure 3.17: Comparison of 2-opt operators on the **Leluk-Konieczny-Roterman** set. The overlap values for different instance sizes are shown on top. Histogram of ranks across all comparison instances for each method is shown on the bottom.

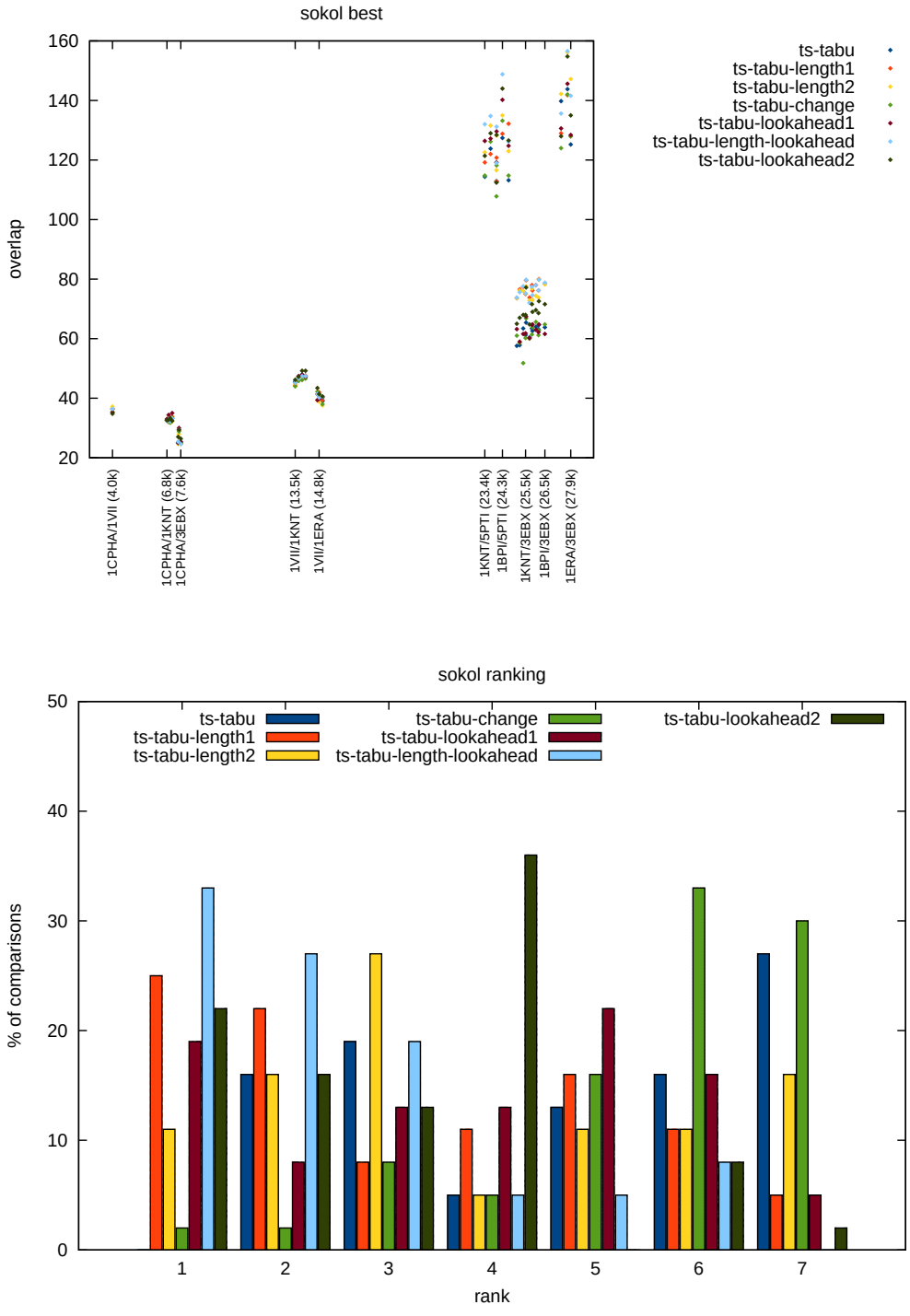

Figure 3.18: Comparison of 2-opt operators on the **Sokol** set. The overlap values for different instance sizes are shown on top. Histogram of ranks across all comparison instances for each method is shown on the bottom.

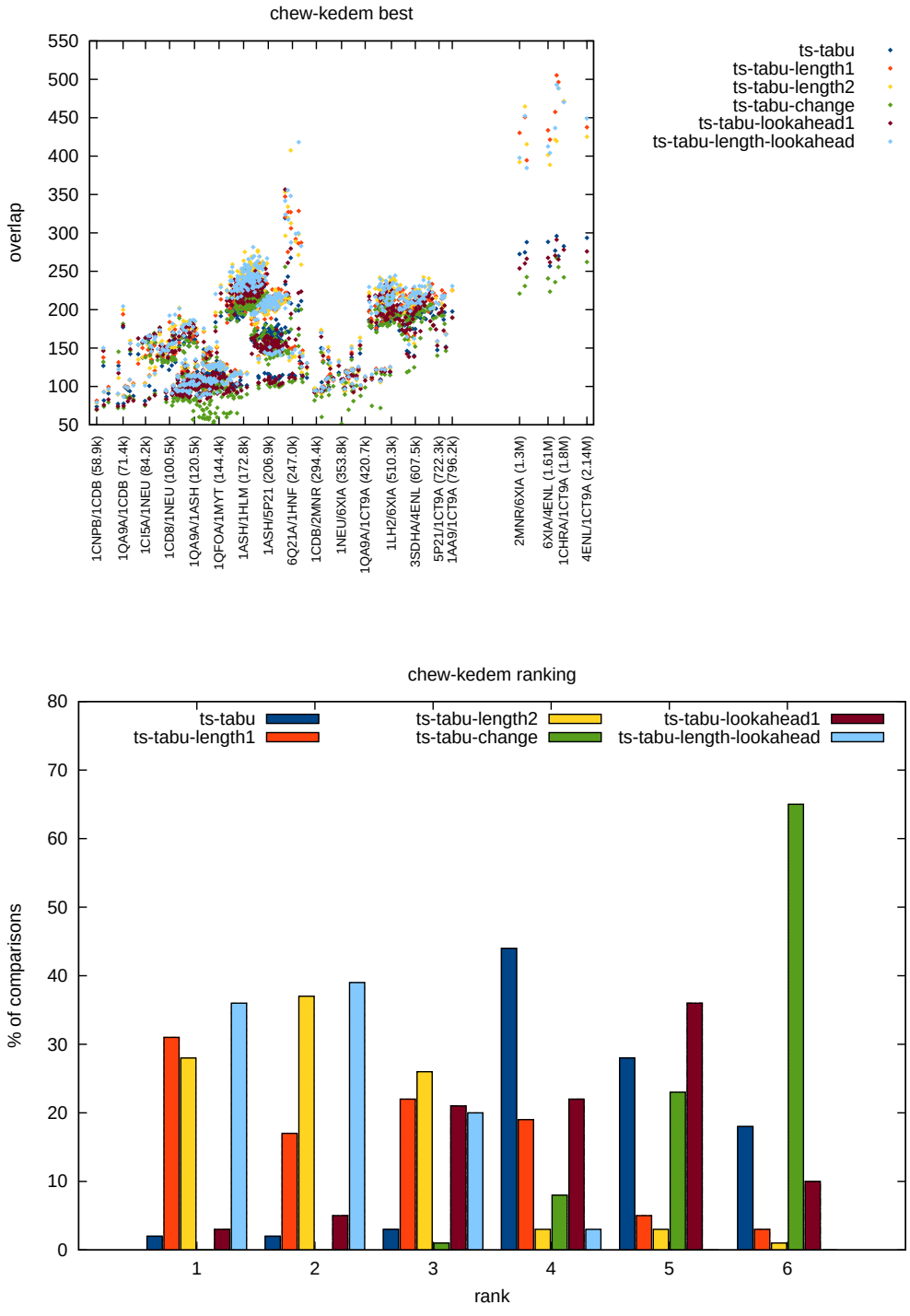

Figure 3.19: Comparison of 2-opt operators on the **Chew-Kedem** set. The overlap values for different instance sizes are shown on top. Histogram of ranks across all comparison instances for each method is shown on the bottom.

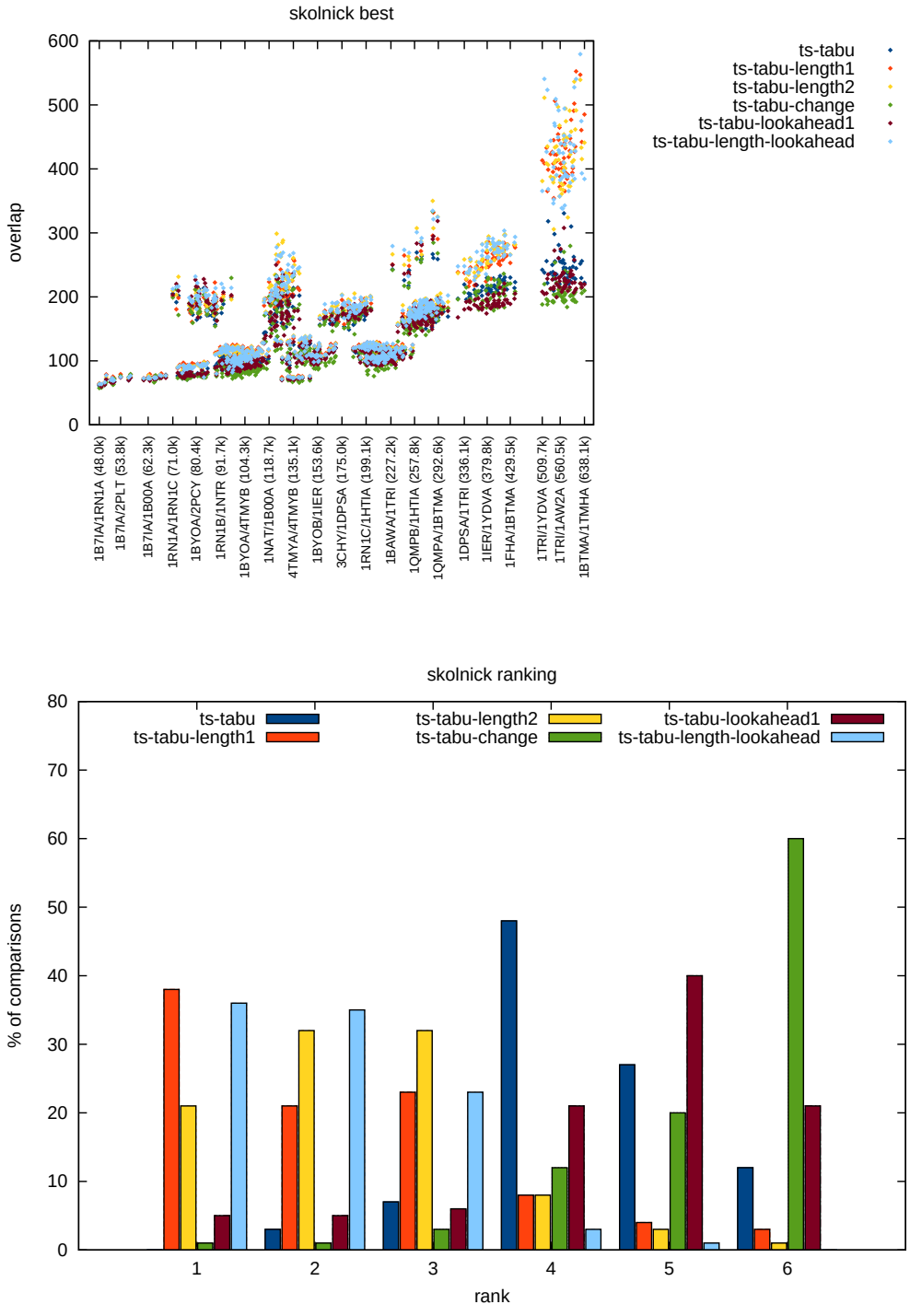

Figure 3.20: Comparison of 2-opt operators on the **Skolnick** set. The overlap values for different instance sizes are shown on top. Histogram of ranks across all comparison instances for each method is shown on the bottom.

very tight clusters for smallest proteins. Several methods were sharing the same rank there and the separation is not so clear as for other tests. The **tabu_lookahead2** method was tested only for the two smallest sets, because of the long computations. It was ranked higher than **tabu** but still lower than the best methods.

The addition of length based preference made **tabu_length_lookahead** method the best in the group. It was clearly better than **tabu_lookahead2** and except the Leluk-Konieczny-Roterman set, better than both **tabu_length1** and **tabu_length2** methods.

Distance from Optima

To compare the quality of the algorithm I measured the distance to the optimal solutions found by Xie at el. [XS07] for Skolnick test set. For each comparison the best solution (not mean) returned by all the methods is compared to the known optimum. The distance is measured as the percentage normalised difference $\frac{o-b}{o}$ between optimum o and the value of best solution found b.

Figure 3.21 shows that only a few optimal solutions were found and for many instances the best result is 20% lesser than optimum. This is not a satisfactory result and further algorithm improvements need to be considered.

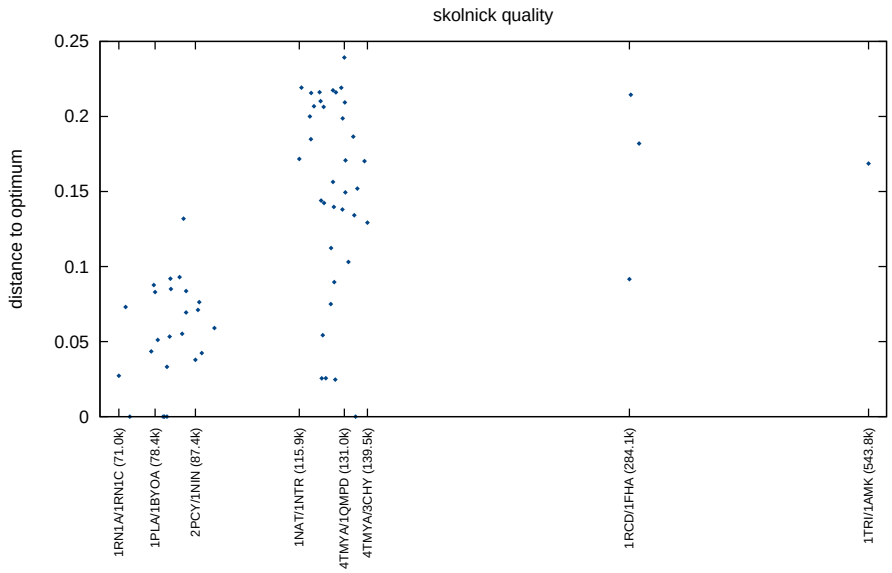

Figure 3.21: Distance of the best solutions to the optima.

3.4 Discussion

Although the complexity improvement in evaluation of the solution and the feasibility check are obvious selling points of the proposed algorithm it is not free from limitations. Despite increased number of solutions evaluated in the unit of time the algorithm has difficulties in finding the good quality solutions. While a few more promising variations of the preference could be test (see below) to possibly further improve the intensification

phase, the biggest drawback of the algorithm seems to lie in oversimplified diversification that is not able to determine correctly which assignments should be removed.

As the case of **tabu_length_lookahead** method has shown better results for combination of methods, several additional experiments could be performed. Instead of single preference method a weighed combination of **length** and **position** or even **length** and **degree** could be used. Also **tabu_change** could be combined with preferences or even look ahead operator. The results for double change operator might be also worth testing.

However, as all the methods are still far from reaching optimal solutions, the biggest improvement may depend on other factors. It might be a more strict condition for edges assigning (for example only equal length), more advanced intensification and diversification with extra learning (reactive tabu search with dynamic tabu tenure factor [SS05, BB05] or additional long term memory with Elite List [LMC99]) or early abandoning of not promising solutions based on upper bound estimation.

3.4.1 Algorithm Behaviour Analysis

For selected successful and unsuccessful runs of the algorithm I analysed in more detail how the search process is carried out and how much does the final solution differ from the starting one. In many cases the random choice of the initial solution was not a good idea. Because the number of feasible assignments is decreasing with each move first choices are very important. When made randomly they often led to solutions that block further improvement.

Two examples are shown on Figure 3.22. Because of the starting shift or long cross assignments there is no way to improve that solutions other than removing all the assignments first and then adding them from the scratch. The search process in most cases was not able to do that.

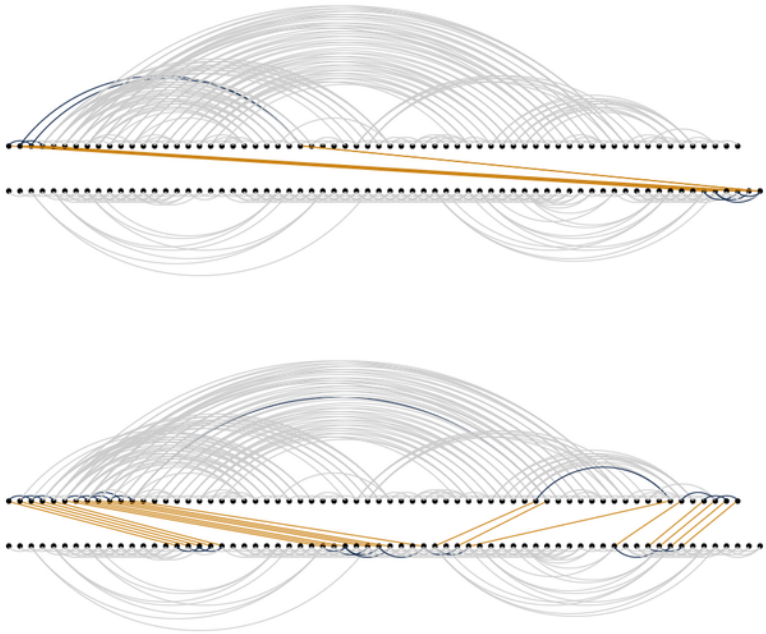

Figure 3.22: Two examples of “dead end” starting solutions.

There are, however, cases where the search process works well. Figure 3.23 shows an example of a starting and final solution (optimum for this instance). The initial solution is establishing a framework for further improvements, which allows the search procedure to overcome the shallow local traps (cross assignments) and find the optimal solution.

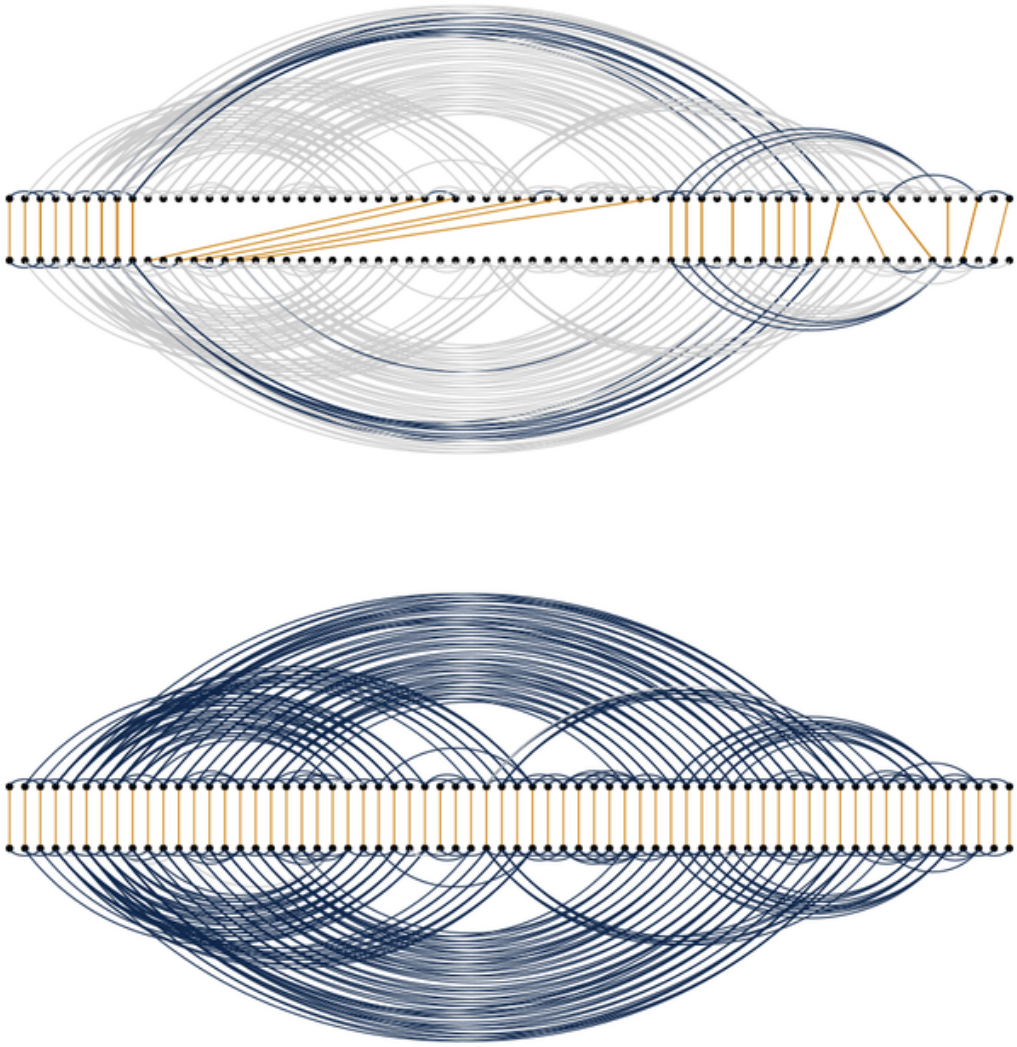

Figure 3.23: Starting solution for which the optimum was found.

As for randomly generated starting solutions most of the iterations of the algorithm are wasted on the search in the unpromising regions of the search space, the next step in algorithm improvement should be an experiment with more intelligent method of choosing initial solution.

The easiest way to improve the quality of starting solutions is to construct them heuristically. One of the ideas is to use the knowledge about frequency of edges length in the shorter graph and assign most frequent longest edges first. If the estimation of an upper bound would be known, instead of random selection of feasible moves, exact number of estimated assignment could be made with the feasibility of solution restored afterwards.

3.5 Conclusions

In this chapter I have summarised the Max-CMO problem as a robust protein structure similarity measure. Using a novel representation of the problem based on a line graph I have demonstrated performance improvements resulting in lower computational cost of solution evaluation and feasibility check with a overall gain of $O(n)$.

An algorithm using the new representation of the solution was implemented as a tabu search metaheuristic and several different types of neighbourhoods/moves and tabu list definitions were tested. Running in a fraction of time (seconds) used by the exact solutions (days) the algorithm was able to solve a few instances of Skolnick set to optimality and achieving 76% of the optimum in the worst case.

The main limitation of the presented metaheuristic is its sensitivity to inital solution. This could be possibly eliminated by replacing the random choice of the initial solution with a hand-crafted heuristic or by introducing more differentiating destruction/diversification phase.

Chapter 4

Consensus Similarity

4.1 Introduction

The protein structure comparison process is an important element of protein structure prediction. It has many uses including model evaluation (measure of distance between the model and the target native structure), optimisation of parameters of energy functions used in prediction (order of decoys by a distance to the native structure) or selection of best decoys (clustering of a decoy to decoy distance matrix).

As there is no agreement upon which protein similarity measure is the best and because of so many different comparison methods with more and more new ones appearing each year, the choice is difficult. Instead of choosing a single best measure a number of individual measures is combined, with each measure contributing its own notion of similarity, into a single similarity consensus measure. Through a positive synergy of these individual similarities the consensus similarity might be more accurate than the best of the individual measures.

The approach to construct the similarity consensus described in this chapter is based on the comparison methods provided by the ProCKSI meta-server (see Section 2.4.1. The server software was modified to allow to perform the consensus computations between a large number of decoys and the native target structure. In contrast to the previous study on the consensus with the use of ProCKSI [BHBK07] which compared different proteins, a focus of the research presented here is on the decoy-native comparison. In particular, on verification if the consensus similarity computed for decoys indeed contains the knowledge of individual methods as suggested by the previous study. If the evidence for that is found, it will support the **H1** hypothesis, that the consensus measure is more robust and better suited as a reference measure in the process of energy function optimisation (this is verified later in Chapter 6).

However, in case of the protein structure prediction it could be more accurate to construct the consensus using scoring functions designed for decoys evaluation such as GDT_TS or TM-Score, instead of protein-protein comparison methods used in ProCKSI. Having in mind the scalability limitations of the infrastructure available for the ProCKSI server, I decided not to introduce these methods there. Instead, I used a cloud computing environment to build a prototype of a scalable web application for decoy comparison. The details of its implementation and the results of the performance test are a parallel

thread of this chapter. Although this new web application was not yet ready to be used to compute the alternative (decoy specific) consensus, it provided a strong basis for further work.

4.2 Methods

4.2.1 ProCKSI Consensus

To make this large scale comparison experiment possible the ProCKSI meta-server was equipped with a new N:1 comparison mode in which all decoy structures are compared against a single native. This work was done in collaboration with Daniel Barthel who was the original ProCKSI developer.

For the time of the experiment the ProCKSI server had to be taken offline, so that no outside interference was possible as this would very likely led to a crash of the system under such a heavy load. It took 14 days to perform all the experiments reported in the next section using all 4 nodes of the ProCKSI cluster with a cumulative of 12 CPU cores.

Consensus Construction

To construct the consensus I used 6 out of 8 methods available in ProCKSI, excluding Vorolign and Dali. Vorolign was found to be the most CPU hungry method and was excluded to speed up the computations. As Dali needs to detect significant similarities between the structures to compute the score, it was not well suited for the decoy-target comparison where most of the decoys are dissimilar to the native structure.

Out of these six methods only the USM measures the distance (being 0 for identical structures and increasing for dissimilar ones). The other methods measure the similarity (being 1 for identical structures and decreasing for dissimilar ones). Additionally some methods also report RMSD of the chosen alignment (see Table 4.1).

name	metric type	RMSD
CE	similarity	yes
FAST	similarity	yes
Max-CMO	similarity	no
TM-align	similarity	yes
URMS	similarity	yes
USM	distance	no

Table 4.1: Measures used in the consensus construction.

For all the proteins each decoy was compared against the target native structure and against itself. The similarity values s were converted to distances d with the use of the self-similarity values $s(i,i)$ and then normalised to the [0;1] range as shown in Equation 4.2.1. If a similarity value was missing due to an error in a method, it was ignored.

$$\begin{aligned} d\left(i,t\right) &= \max\left[s(i,i),\ s(t,t)\right] - s(i,t) \\ norm\left[d\left(i,t\right)\right] &= \frac{d\left(i,t\right) - \min\left[d\left(i,i\right),\ d\left(t,t\right)\right]}{\max_j\left[d\left(j,t\right) - \min\left[d\left(j,j\right),\ d\left(t,t\right)\right]\right]} \end{aligned} \tag{4.1}$$

The consensus similarity was calculated in two variants: mean or median of all measures. Two different methods of aggregation were used: mean/median across all 6 similarity values (similarity consensus) or mean/median 4 RMSD values (RMSD consensus). As a result 4 different types of consensus were computed.

4.2.2 Mutual Information

To measure how large is the mutual dependence of different similarity measures and the 4 variants of the consensus measure a universal metric based on mutual information was used. Let's define entropy $H(X)$ as a measure of the uncertainty associated with a random variable X:

$$H(X) = \sum_{x \in X} p_x log_2(p_x) \tag{4.2}$$

The joint entropy of two variables X and Y is a measure of uncertainty associated with a joint system XY:

$$H(XY) = H(X \cup Y) = \sum_{x,y} p_{x,y} log_2(p_{x,y}) \tag{4.3}$$

Then the mutual information $I(X;Y)$ is a measure of reduction in uncertainty of X knowing Y (the amount of information Y contains about X):

$$I(X;Y) = H(X) + H(Y) - H(XY) \tag{4.4}$$

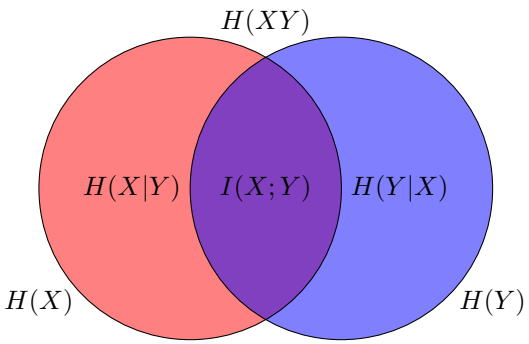

Figure 4.1: Mutual information between random variables X and Y. H(X) is the entropy of variable X, H(XY) is the joint entropy of X and Y, H(X—Y) is the conditional entropy of variable X when Y is known.

Estimation of the mutual information for the finite set of data points results in a small bias related to the size of the set. For that reason it was not used directly but a universal metric $D(X,Y)$ was used instead:

$$D(X,Y) = 1 - \frac{H(X \cap Y)}{H(X \cup Y)} = 1 - \frac{I(X;Y)}{H(XY)} \tag{4.5}$$

The metric $D(X,Y)$ scales with the total information, is normalised to range [0;1] and is universal. Essentially this means that if $X \approx Y$ according to any non-trivial distance measure, then $X \approx Y$ also according to D [KSAG03].

To further improve the estimation for the mutual information and ensure that individual distributions of similarity values would not blur the results, the compared data sets

were normalised to an identical reference distribution using the adaptive partitioning [SKD+02]. Instead of dividing the range of N similarity values into M bins with fixed intervals, the width of a bin is determined by the local density of the data and each bin contains approximately N/M data points.

To test the hypothesis that two sets of similarity values X and Y are independent k random permutations of the original set Y were created by shuffling the order of its data points. The probability that the original set Y has the smallest mutual information distance within the whole ensemble of shuffled sets by chance is then equal to $\alpha = \frac{1}{k+1}$. The null hypothesis could therefore be rejected with confidence of $1 - \alpha$, where α is the probability of a false rejection [SKD+02].

4.2.3 Protein Decoys Comparator

As the current architecture of ProCKSI is limited to a maximum of 2500 concurrent comparisons (due to limited local cluster infrastructure) in its current state it can't provide a stable base for further development. Because of that I decided to run a prototype application using the Google App Engine (GAE)[1], a recently introduced web application platform designed for scalability. GEA operates as a cloud computing environment providing Platform as a Service (PaaS). It removes the need to consider physical resources as they are automatically scaled up as/when required. It also does not require any setup or maintenance of the hardware infrastructure.

GAE offers two runtime environments based on Python or Java, both well documented and frequently updated with new features. A limited amount of GAE resources is provided for free and is enough to run a small application. There are no set-up costs and all payments are based on the daily amount of resources (storage, bandwidth, CPU time) used above the free levels.

The Protein Decoys Comparator functionality is very simple. It allows a user to upload a number of decoys and run the comparison experiment in one of two modes: comparison of all structures against a known native structure (N:1) or comparison of all structures against each other (N:N). The user can currently choose from four comparison measure: RMSD, LCS, GDT_TS[Zem03] and TM-score [ZS04b]. When the experiment is finished, an e-mail notification is send. For a quick visual assessment, the results are shown as the frequency histograms of the resulting distance vector/matrix. Also a raw data file is provided for download and further analysis. The application website is:
`http://pd-cmp.appspot.com/`.

Architecture

The user interface (UI) and most of the application logic was implemented in Python using web2py framework[2]. This makes the application portable as it can also run outside of the GAE infrastructure without any code changes. The histograms were made using Google Charts API[3], so that they do not use extra CPU time. The comparison engine was implemented in Groovy using Gaelyk[4], a small lightweight web framework designed

[1] `http://code.google.com/appengine/`
[2] `http://www.web2py.com/`
[3] `http://code.google.com/apis/chart/`
[4] `http://gaelyk.appspot.com`

for GAE. It runs in Java Virtual Machine (JVM) environment and interfaces with the BioShell java library [GK08] that implements all four structure comparison methods.

The comparison experiment is started by an HTTP request to the comparison engine from the UI module (see Figure 4.2). Then the comparison module splits the experiment into a number of structure vs. structure comparison tasks and executes them as parallel background processes. Each task reads the structures written to the datastore by the UI module and stores back the results. Finally the UI module reads the results and generates the histograms.

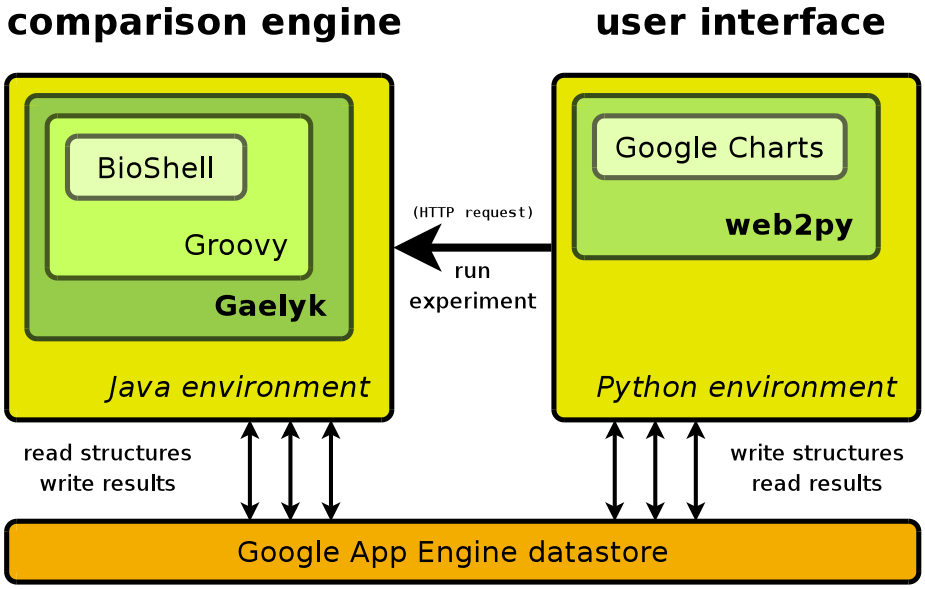

Figure 4.2: Protein Decoys Comparator architecture on GAE.

Distributed Computing Model

The response time to each HTTP request on GAE is limited to 30s. While this seems at first to be quite restrictive, in combination with a task queue mechanism it is an efficient distributed computation service. The work could be organised into small tasks and done in the background, outside of the user request. Each task is put into a queue and later automatically dispatched according to defined scheduling criteria. The only problem is the efficiency of Python and Java being interpreted languages. A common practice of binding these languages with fast modules written in C/C++ is not possible on GAE, as it cannot run arbitrary native code. For that reason we implemented the computational engine on JVM using BioShell.

File Size Limitations

There is a hard 1MB limit on the size of stored data structures (entities) and hence common practice of uploading data in a single compressed file cannot be applied to GAE. Even small archives can be problematic as GAE lacks direct file system access which makes decompression more complex. We decided to upload the files one by one as single structure files are usually much smaller than 1MB. To make the upload process easy

and capable of handling hundreds of files, we used Uploadify library[5] which combines Javascript and Flash to provide a multi-files upload and progress tracking.

4.3 Results

In experiments reported in this section I have used decoys generated by two different protein structure predictors: I-TASSER and Rosetta. I-TASSER decoys were generated during the Monte Carlo based structure optimisation process [ZKS02] and are available online [WSZ07]. For 55 small non-homologous proteins (i.e. with pairwise sequence identity $< 30\%$) I took a 10% sample of the generated decoys (one decoy every 10 I-TASSER iterations). This resulted in set of 1250–2000 decoys per protein. For the same set of proteins I ran the Rosetta ab initio prediction [RSMB04] (version 2.2.0) to obtain the same number of decoys as in the I-TASSER case. Table 4.2 shows the average length of these proteins and a distribution of secondary structure types in the set.

SS type			**length**		
$\boldsymbol{\alpha}$	$\boldsymbol{\beta}$	$\boldsymbol{\alpha\beta}$	**min**	**max**	**avg**
16	14	26	47	118	81.07

Table 4.2: Secondary structure type and length of chains used in the consensus experiments.

4.3.1 Relation to the RMSD

To quickly evaluate the difference between the four variants of the consensus and the RMSD the respective correlations were calculated (see Table 4.3). The RMSD consensus was surprisingly less correlated on average to the RMSD than the similarity consensus. However, if we consider that the protein comparison methods calculate the RMSD only for the best aligned fragment of the structure and that the value of RMSD depends on the fragment length, the lower correlation is justified. It is also in line with the observation made by Barthel et. al [BHBK07] on poor performance of the RMSD consensus in the protein classification case.

	aggregated measure			
	similarity		**RMSD**	
decoys	**mean**	**stddev**	**mean**	**stddev**
I-TASSER	0.625	0.199	0.454	0.319
Rosetta	0.687	0.187	0.354	0.287

Table 4.3: Average correlation between the consensus and RMSD.

[5] `http://www.uploadify.com/`

4.3.2 Dependency Between Measures

The mutual information distance between the similarity measures was in most of the cases greater than 0.9. Besides the obvious smaller distance between the mean and median variants of both similarity and RMSD consensuses, not many similarities were found (see Tables 4.4 and 4.5). TM-align and URMS and were slightly closer to the mean similarity consensus than the other three measures.

However, when the statistical independence was tested, the dependency between measures became clear. For the **mean similarity consensus** the null hypothesis that measures are independent was rejected for all measures and in case of all proteins (see Tables 4.6 and 4.7). This is a strong indication that the consensus indeed combines the information from all the measures. The rejection of the null hypothesis is less frequent in case of the RMSD consensus. For the Max-CMO and USM measures which were not part of the RMSD consensus the rejection ratio is the lowest, reaching about 70% of proteins. This shows that for about 30% of proteins these contact map based measures rank the decoys different than the other four measures. That means that inclusion of the contact map based methods in the consensus was a reasonable idea as they add new information to the consensus.

4.3.3 Consensus Disagreement

Following the observation from the measure independence test, I decided to examine the level of disagreement amongst the similarity methods used in the consensus with respect to the consensus average value. The following scatter plots are showing a correlation between the energy of decoys computed by the predictor and the consensus similarity. Each panel shows decoys for a single protein. The number on top of each panel is the correlation coefficient. The disagreement is visualised with color and it is represented by the standard deviation amongst similarity measures for a given decoy. Saturated red is the color of the decoys with the lowest disagreement and saturated blue — the highest. Pale blue/red color shows disagreement in between this two extrema.

In Figures 4.3–4.6 most of the decoys are red. Especially in the Rosetta case only a few saturated blue decoys with high level of disagreement are visible. They tend to be located on the right side of the plots, that is where the distance to the native is the largest. Similarly saturated red color is more often marking decoys similar to the native structure. The plots are fairly similar for both aggregation methods.

For the similarity consensus the disagreement distribution is very different. In Figures 4.7–4.9 the decoys with highest level of disagreement are the ones most similar to the native structure, whereas the lowest level of disagreement is observed for the most distant decoys. There seems to be a clear correlation between the level of disagreement and the distance to native structure. Despite a discretisation of a dense regions into bins the median similarity consensus does not differ from the mean one.

The coefficient of the correlation between the **mean similarity consensus** and its standard deviation is equal to -0.76 for Rosetta decoys and -0.46 for I-TASSER. The more dissimilar to the native the decoys are the smaller is the similarity consensus deviation. Analogically, the similarity measures disagree the most for the near native decoys.

	CE	FAST	Max-CMO	TM-align	URMS	USM	mean	median	rmsd-mean	rmsd-median
CE	0.000 ± 0.000	0.946 ± 0.042	0.960 ± 0.026	0.883 ± 0.063	0.904 ± 0.059	0.950 ± 0.037	0.860 ± 0.056	0.891 ± 0.056	0.933 ± 0.055	0.940 ± 0.054
FAST	0.946 ± 0.042	0.000 ± 0.000	0.950 ± 0.024	0.939 ± 0.047	0.944 ± 0.045	0.958 ± 0.030	0.923 ± 0.055	0.935 ± 0.046	0.951 ± 0.042	0.956 ± 0.037
Max-CMO	0.960 ± 0.026	0.950 ± 0.024	0.000 ± 0.000	0.957 ± 0.029	0.959 ± 0.028	0.952 ± 0.035	0.917 ± 0.036	0.931 ± 0.046	0.966 ± 0.024	0.967 ± 0.024
TM-align	0.883 ± 0.063	0.939 ± 0.047	0.957 ± 0.029	0.000 ± 0.000	0.837 ± 0.070	0.943 ± 0.042	0.831 ± 0.072	0.786 ± 0.076	0.898 ± 0.080	0.910 ± 0.076
URMS	0.904 ± 0.059	0.944 ± 0.045	0.959 ± 0.028	0.837 ± 0.070	0.000 ± 0.000	0.948 ± 0.037	0.833 ± 0.070	0.787 ± 0.072	0.912 ± 0.067	0.917 ± 0.073
USM	0.950 ± 0.037	0.958 ± 0.030	0.952 ± 0.035	0.943 ± 0.042	0.948 ± 0.037	0.000 ± 0.000	0.925 ± 0.054	0.926 ± 0.039	0.959 ± 0.032	0.960 ± 0.031
mean	0.860 ± 0.056	0.923 ± 0.055	0.917 ± 0.036	0.831 ± 0.072	0.833 ± 0.070	0.925 ± 0.054	0.000 ± 0.000	0.754 ± 0.090	0.920 ± 0.062	0.928 ± 0.060
median	0.891 ± 0.056	0.935 ± 0.046	0.931 ± 0.046	0.786 ± 0.076	0.787 ± 0.072	0.926 ± 0.039	0.754 ± 0.090	0.000 ± 0.000	0.917 ± 0.057	0.926 ± 0.055
rmsd-mean	0.933 ± 0.055	0.951 ± 0.042	0.966 ± 0.024	0.898 ± 0.080	0.912 ± 0.067	0.959 ± 0.032	0.920 ± 0.062	0.917 ± 0.057	0.000 ± 0.000	0.784 ± 0.105
rmsd-median	0.940 ± 0.054	0.956 ± 0.037	0.967 ± 0.024	0.910 ± 0.076	0.917 ± 0.073	0.960 ± 0.031	0.928 ± 0.060	0.926 ± 0.055	0.784 ± 0.105	0.000 ± 0.000

Table 4.4: Mutual information distance between measures for **Rosetta** decoys. Average and standard deviation over 55 proteins are shown.

	CE	FAST	Max-CMO	TM-align	URMS	USM	mean	median	rmsd-mean	rmsd-median
CE	0.000 ± 0.000	0.940 ± 0.040	0.973 ± 0.015	0.867 ± 0.058	0.947 ± 0.035	0.970 ± 0.026	0.916 ± 0.039	0.904 ± 0.059	0.947 ± 0.035	0.950 ± 0.032
FAST	0.940 ± 0.040	0.000 ± 0.000	0.964 ± 0.018	0.920 ± 0.048	0.944 ± 0.042	0.967 ± 0.027	0.907 ± 0.053	0.923 ± 0.052	0.941 ± 0.044	0.945 ± 0.040
Max-CMO	0.973 ± 0.015	0.964 ± 0.018	-0.000 ± 0.000	0.972 ± 0.015	0.979 ± 0.012	0.972 ± 0.018	0.928 ± 0.025	0.929 ± 0.054	0.978 ± 0.013	0.979 ± 0.012
TM-align	0.867 ± 0.058	0.920 ± 0.048	0.972 ± 0.015	-0.000 ± 0.000	0.903 ± 0.054	0.966 ± 0.027	0.897 ± 0.055	0.820 ± 0.112	0.876 ± 0.074	0.885 ± 0.070
URMS	0.947 ± 0.035	0.944 ± 0.042	0.979 ± 0.012	0.903 ± 0.054	-0.000 ± 0.000	0.974 ± 0.022	0.899 ± 0.049	0.884 ± 0.068	0.910 ± 0.079	0.913 ± 0.079
USM	0.970 ± 0.026	0.967 ± 0.027	0.972 ± 0.018	0.966 ± 0.027	0.974 ± 0.022	0.000 ± 0.000	0.951 ± 0.033	0.949 ± 0.039	0.972 ± 0.025	0.973 ± 0.023
mean	0.916 ± 0.039	0.907 ± 0.053	0.928 ± 0.025	0.897 ± 0.055	0.899 ± 0.049	0.951 ± 0.033	0.000 ± 0.000	0.827 ± 0.062	0.935 ± 0.045	0.940 ± 0.041
median	0.904 ± 0.059	0.923 ± 0.052	0.929 ± 0.054	0.820 ± 0.112	0.884 ± 0.068	0.949 ± 0.039	0.827 ± 0.062	-0.000 ± 0.000	0.923 ± 0.055	0.931 ± 0.050
rmsd-mean	0.947 ± 0.035	0.941 ± 0.044	0.978 ± 0.013	0.876 ± 0.074	0.910 ± 0.079	0.972 ± 0.025	0.935 ± 0.045	0.923 ± 0.055	0.000 ± 0.000	0.728 ± 0.121
rmsd-median	0.950 ± 0.032	0.945 ± 0.040	0.979 ± 0.012	0.885 ± 0.070	0.913 ± 0.079	0.973 ± 0.023	0.940 ± 0.041	0.931 ± 0.050	0.728 ± 0.121	0.000 ± 0.000

Table 4.5: Mutual information distance between measures for **I-TASSER** decoys. Average and standard deviation over 55 proteins are shown.

	single measures						consensus measures			
	CE	**FAST**	**Max-CMO**	**TM-align**	**URMS**	**USM**	**mean**	**median**	**rmsd-mean**	**rmsd-median**
CE	55	51	50	55	55	51	55	55	51	48
FAST	46	55	55	48	47	48	55	51	45	44
Max-CMO	49	55	55	51	47	53	55	55	41	39
TM-align	55	49	51	55	55	50	55	55	55	53
URMS	55	45	49	55	55	50	55	55	55	54
USM	51	49	52	50	49	55	55	55	46	43
mean	55	55	55	55	55	55	55	55	52	51
median	55	51	55	55	55	55	55	55	54	54
rmsd-mean	51	44	**42**	55	55	**44**	55	55	55	55
rmsd-median	47	43	**40**	52	54	**42**	50	53	55	55

Table 4.6: Measures independence test for **Rosetta** decoys. For each pair of measures and each protein the hypothesis that measures are independent was tested. The table shows for how many proteins (out of 55) this hypothesis was rejected at significance level $\alpha = 0.05$.

	single measures						consensus measures			
	CE	**FAST**	**Max-CMO**	**TM-align**	**URMS**	**USM**	**mean**	**median**	**rmsd-mean**	**rmsd-median**
CE	55	55	45	55	53	49	55	55	55	53
FAST	55	55	53	55	54	51	55	55	54	55
Max-CMO	41	53	55	50	40	46	55	54	40	38
TM-align	55	55	50	55	54	53	55	55	55	55
URMS	54	54	36	54	55	41	55	54	54	53
USM	44	52	46	53	42	55	55	53	44	47
mean	55	55	55	55	54	55	55	55	55	55
median	55	55	53	55	54	53	55	55	53	55
rmsd-mean	54	53	**37**	55	54	**44**	55	54	55	55
rmsd-median	53	54	**35**	55	53	**45**	55	54	55	55

Table 4.7: Measures independence test for **I-TASSER** decoys. For each pair of measures and each protein the hypothesis that measures are independent was tested. The table shows for how many proteins (out of 55) this hypothesis was rejected at significance level $\alpha = 0.05$.

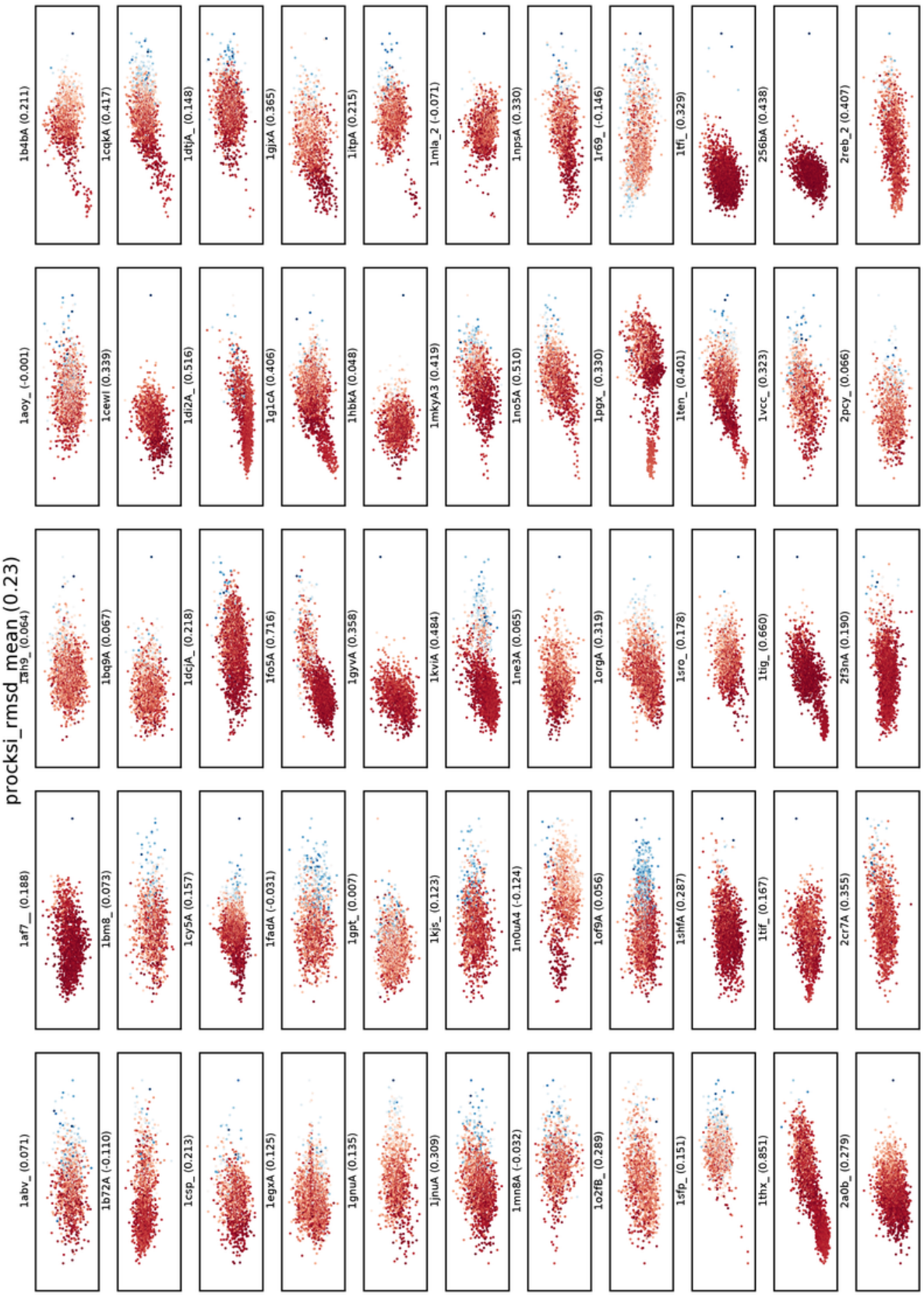

Figure 4.3: Correlation of **Rosetta** generated decoys original energy and **RMSD mean consensus**. Standard deviation for each consensus value is marked with color on a scale from saturated red (lowest standard deviation) to saturated blue (highest standard deviation).

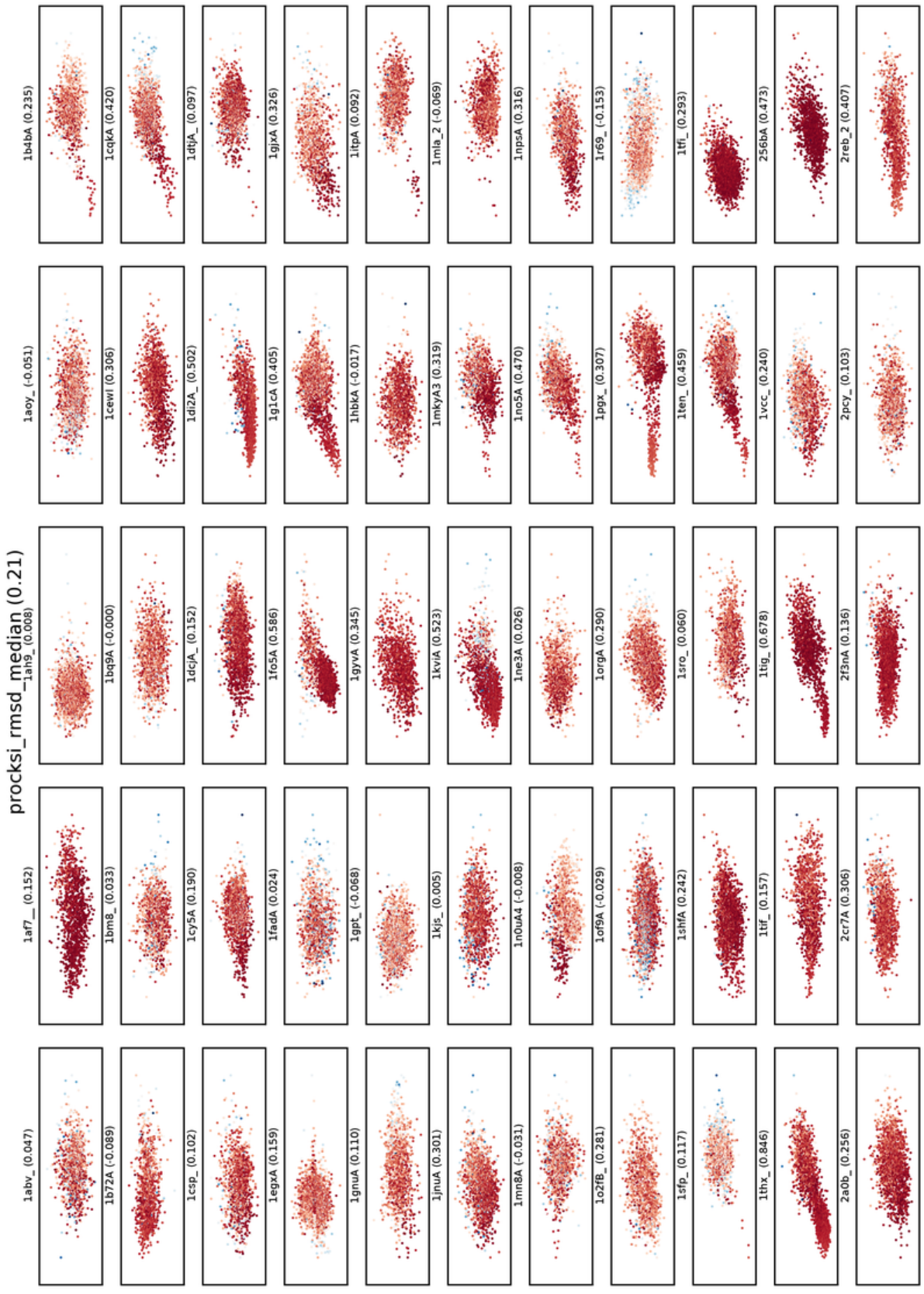

Figure 4.4: Correlation of **Rosetta** generated decoys original energy and **RMSD median consensus**. Standard deviation for each consensus value is marked with color on a scale from saturated red (lowest standard deviation) to saturated blue (highest standard deviation).

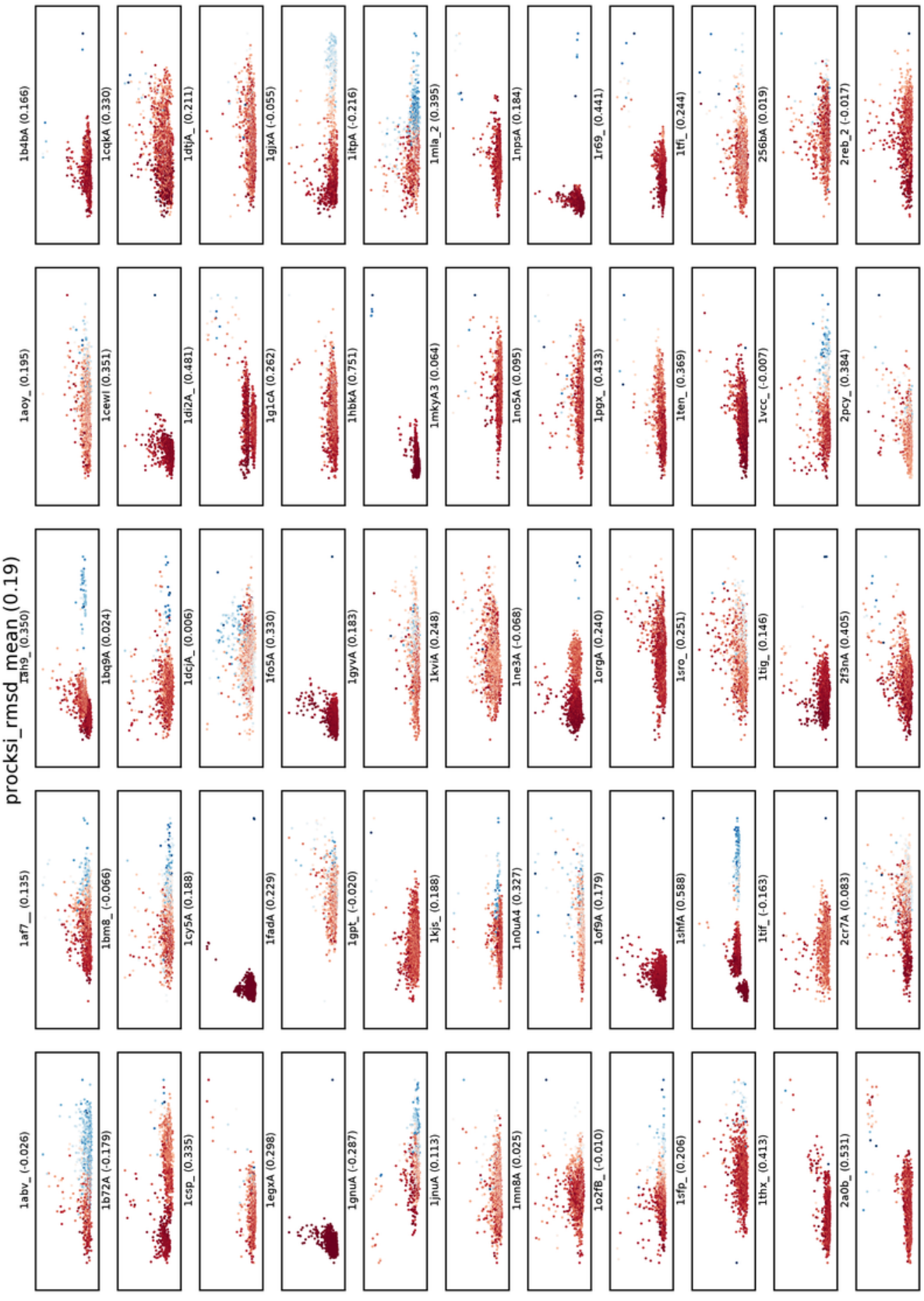

Figure 4.5: Correlation of **I-TASSER** generated decoys original energy and **RMSD mean consensus**. Standard deviation for each consensus value is marked with color on a scale from saturated red (lowest standard deviation) to saturated blue (highest standard deviation).

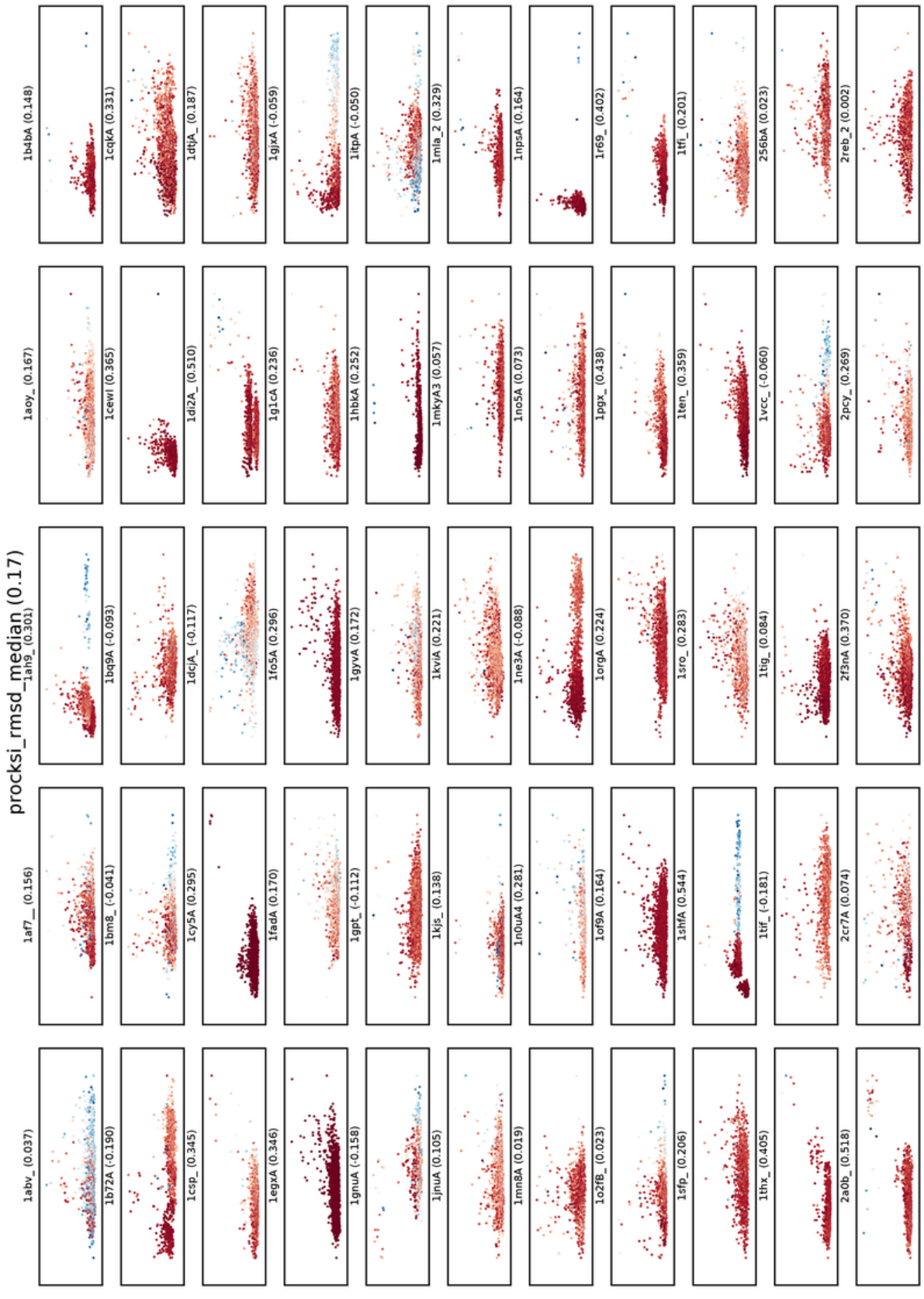

Figure 4.6: Correlation of **I-TASSER** generated decoys original energy and **RMSD median consensus**. Standard deviation for each consensus value is marked with color on a scale from saturated red (lowest standard deviation) to saturated blue (highest standard deviation).

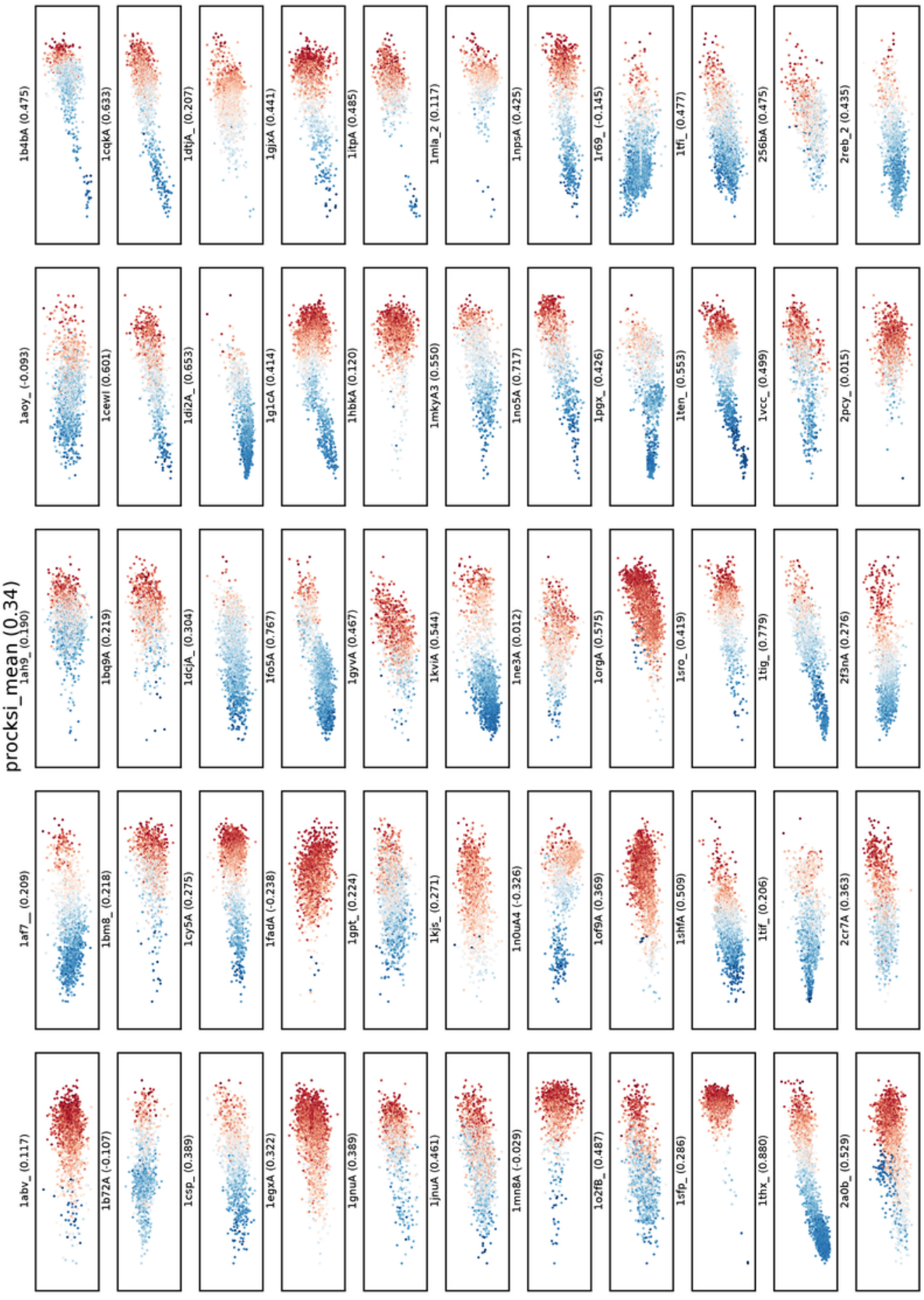

Figure 4.7: Correlation of **Rosetta** generated decoys original energy and **similarity mean consensus**. Standard deviation for each consensus value is marked with color on a scale from saturated red (lowest standard deviation) to saturated blue (highest standard deviation).

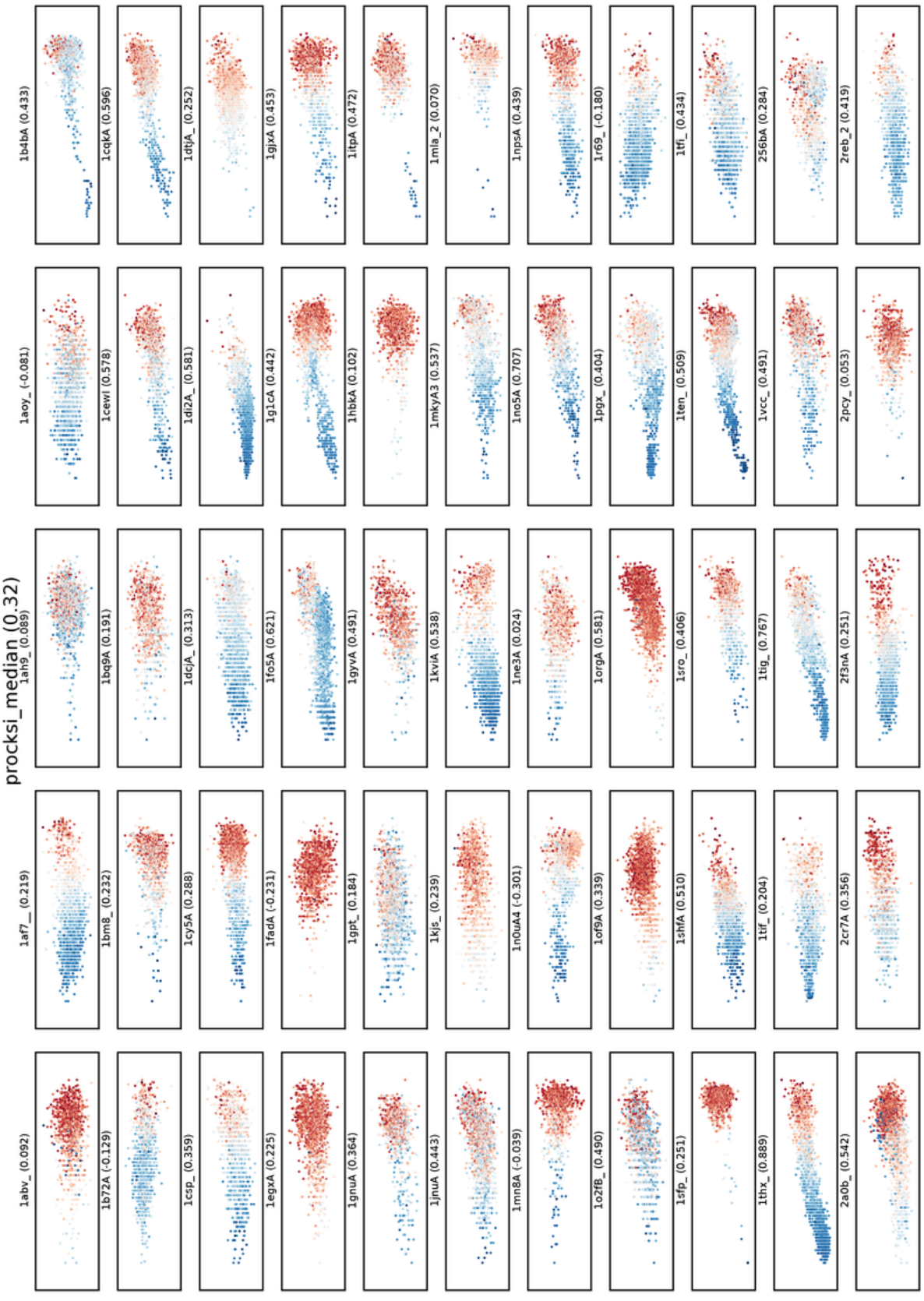

Figure 4.8: Correlation of **Rosetta** generated decoys original energy and **similarity median consensus**. Standard deviation for each consensus value is marked with color on a scale from saturated red (lowest standard deviation) to saturated blue (highest standard deviation).

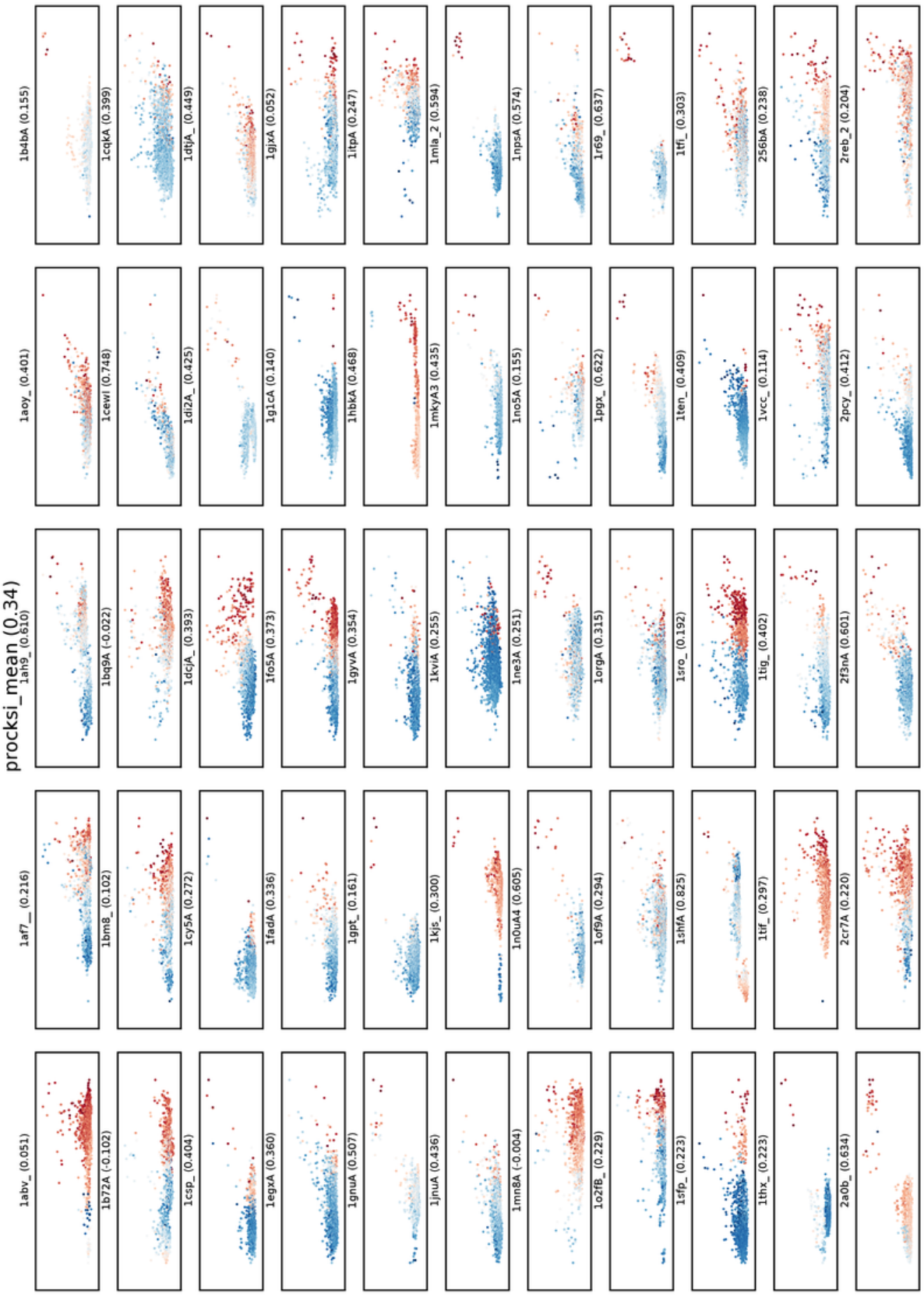

Figure 4.9: Correlation of **I-TASSER** generated decoys original energy and **similarity mean consensus**. Standard deviation for each consensus value is marked with color on a scale from saturated red (lowest standard deviation) to saturated blue (highest standard deviation).

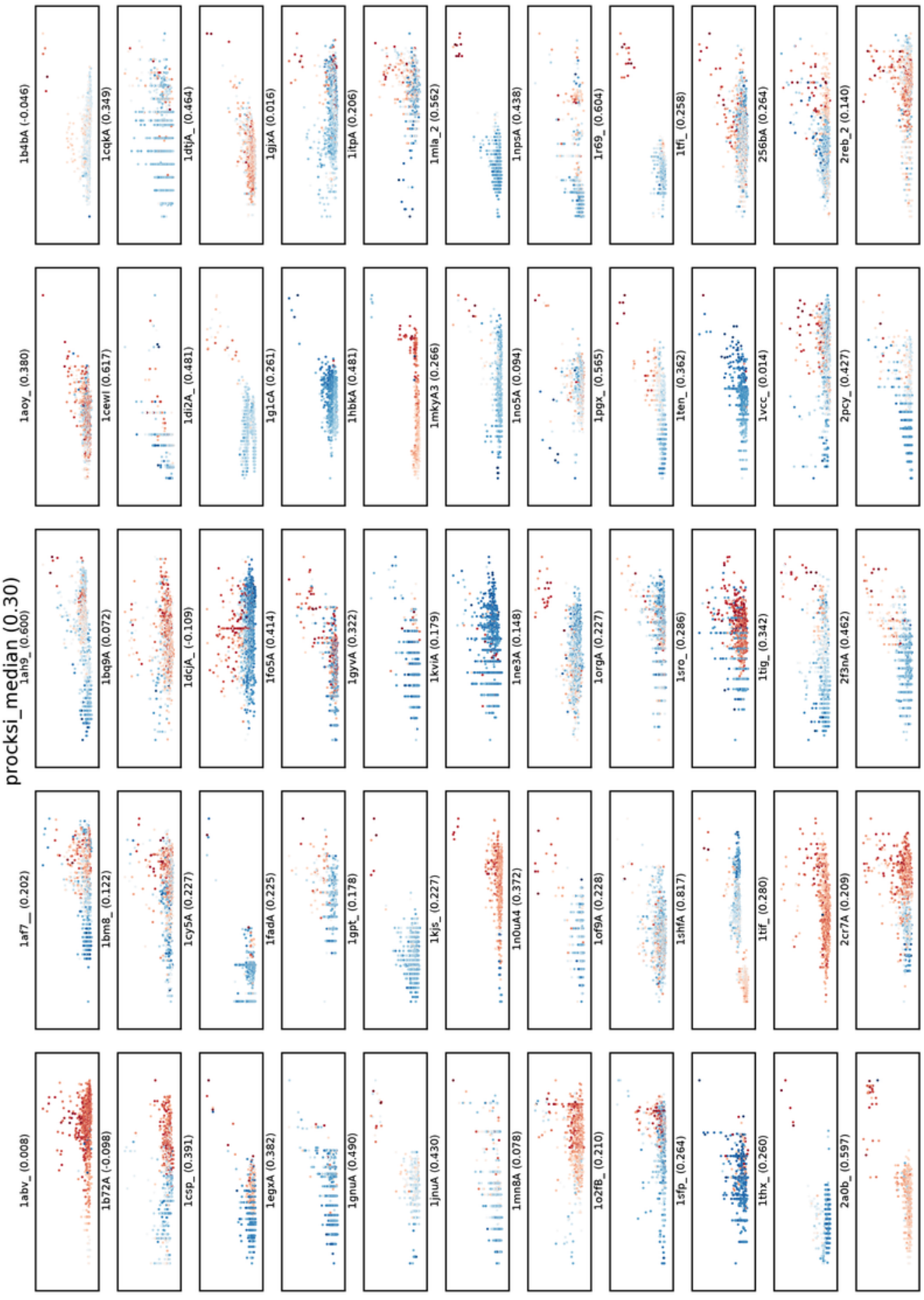

Figure 4.10: Correlation of **I-TASSER** generated decoys original energy and **similarity median consensus**. Standard deviation for each consensus value is marked with color on a scale from saturated red (lowest standard deviation) to saturated blue (highest standard deviation).

4.3.4 Google App Engine Benchmark

To test the performance of the proposed architecture I run a 48h test in which a group of beta testers run multiple experiments in parallel at different times of a day. I observed a very consistent behaviour of the service, with a relative absolute median deviation smaller than 10%. The values reported in Table 4.8 show the statistics for 15 runs per each of the 4 decoy sets.

		processing time[s]				
protein	**decoys**	**length**	**median**	**mad**	**min**	**max**
1b72A	350	49	178	17	108	272
1egxA	300	115	195	17	125	274
1kviaA	500	68	236	16	203	406
1fo5A	800	85	369	33	307	459

Table 4.8: Performance of the Protein Decoy Comparator. **mad** (median absolute deviation) = $median_i(|x_i - median(X)|)$.

4.4 Discussion

A high variance of the similarity consensus for decoys similar to the native indicates the difficulty of assessing the similarity of such structures. The closer the decoys are the less is the relative error margin for the similarity score. Dissimilar decoys are easier to assess as the same absolute error is much smaller in relation to the distance.

As the negative correlation between the variance and the mean value of the consensus was relatively high it would be possible to even use the consensus disagreement itself as a measure of structural similarity or identification of the near native structures.

Such a good agreement of the RMSD consensus for the majority of decoys indicates that the alignments constructed by different similarity measures were similar. It seems to be more straightforward to align the best fragments correctly than to find a superposition that would be optimal for the whole structure. And this in particular makes the decoy to native comparison more challenging than comparison of two different proteins.

Judging by the result of the statistical independence test the similarity consensus seems to be exactly what it was expected to be, a combination of knowledge contained in individual similarity measures. However, the aggregations used to construct the consensus were very simple. Maybe a different approach based on rank aggregation or majority voting would be more interesting way of combining this knowledge.

It shouldn't be also difficult to introduce a model of user preferences. Either as a simple weighted sum of similarity measures or as a reference ranking given for selected decoys by an expert.

4.5 Conclusions

The amount of information shared between the **mean similarity consensus** and the individual similarity measures was found to be higher than the one possible by chance

(at significance level of 5%). This confirms the expected inclusive character of the consensus measure and is strong indication that it contains the sum of knowledge of all the individual comparison methods (what supports the **H1** hypothesis that the consensus might positively influence the evolvability of energy functions).

In addition, it was shown that for the **mean RMSD consensus** exclusion of the MaxCMO and USM measures resulted in no shared information between them and the consensus for almost 30% of the proteins. This means that the notion of similarity used by this measures adds a new information, not contained in the other measures in the 30% cases. Therefore, it was a reasonable idea to include these method in the **mean similarity consensus**. It would be interesting to test if the same could be said about the other methods. For that further experiments would be required with n-1 consensuses (single method excluded) compared against the individual methods.

Another interesting aspect of the performed analysis is a discovery of the negative correlation between the consensus disagreement and the consensus value. This seems to suggest that decoys more similar to the native structure are more difficult to asses by the protein comparison methods, as they were not designed for the decoys comparison. An interesting future work would be a test of disagreement between the decoy-specific measures and all measures, decoy and protein-specific, together.

Finally, a fully functional prototype of a cloud computing application for protein decoys comparison has been proposed and tested for performance. In the future this application could be extended with a consensus construction methods and become an alternative service to the current ProCSKI meta-server.

Chapter 5

Genetic Programming Experiment

5.1 Introduction

The application of evolutionary algorithms (EA) to protein structure prediction is not new. Many methods have been studied in the past [Ung04] using a variety of protein structure and energy models, ranging from a very simple H–P lattice model [Dil90] in works of Krasnogor et al. [KHSP99, KBHB02] and Santana et al. [SLL08] to all-atom force fields such as CHARMM used by Day et al. [DLP03] and Cutello et al. [CNN06] or AMBER used by Djurdjevic et al. [DB06]. All these methods, however, use the EA framework to optimise the model of protein structure with respect to a fixed, i.e. given, energy function.

In this work, I focus instead on the improvement of the energy function itself. I do this, because even the optimal model would be as good as the energy functions (i.e. objective functions) used to optimise it. The analysis of the most successful prediction methods, namely I-TASSER and Robetta, suggests, that the energy functions are far from perfect (see Section 2.3), as it is not highly correlated with similarity to the native, and thus it could be improved. I hypothesise that a more general functional combination of energy terms will result in higher correlation between the energy function and the similarity of the candidate protein structures to the true native structure (hypothesis **H2**).

To test my hypothesis I conducted a large number of experiments applying genetic programming (GP) to evolve the energy function used to evaluate protein structures. As a test set I used real decoys generated by I-TASSER predictor during the structure optimisation process [WSZ07]. I believe that this is more accurate than using decoys generated by randomisation of the native structure (as in the original work of Zhang et al. [ZKS03]), because in practise, predictors have no a priori knowledge of the native structure. I have selected a subset of eight energy terms used by I-TASSER and pre-calculated their values for all decoys. Experiments were then carried out to evolve non-linear energy functions featuring a range of basic algebraic operators and transcendental functions. Using several different fitness measures I tried to determine how difficult it is to evolve an energy function that is highly correlated with structural similarity to the native state. As

a baseline control experiment, I compared the best evolved energy functions with results of a random walk.

I decided to use GP as it has been proved successful in solving symbolic regression problems and being able to produce human-competitive results [PLM08]. The standard tree-based GP was chosen to provide a solid comparison baseline for further research, as this is a really pioneering work and very little is known about the structure of the problem, but there are many possible alternative approaches including graph-based GP, linear GP, grammatical evolution or estimated distribution algorithms (see Chapter 7 for discussion of possible directions for future research).

The results reported in this chapter are further analysed in Chapter 6, providing control experiment with linear combination of terms and discussion of the method limitations including the influence of the consensus similarity.

5.2 Methods

5.2.1 Energy Terms

I have implemented eight I-TASSER energy terms. These include: three short-range potentials between C_α atoms E_{13}, E_{14} and E_{15}, long-range pairwise potential between side chain centres of mass E_{pair}, environment profile potential E_{env}, local stiffness potential E_{stiff} and electrostatic interactions potential $E_{electro}$ as described in [ZKS03, ZS04c] and the hydrogen bonds potential E_{HB} as defined in supplementary materials to [ZHA+06b]. See Section 2.3.3 for detailed description of these terms.

I left out potentials using data from the threading process (e.g. distance map or contact order) and the hydrophobic potential introduced in [WSZ07] using neural network [CZ05] as they depend on external feature predictors which were not available for local use at the time of writing this chapter and would have made pre-calculation much slower (see Section 6.3).

5.2.2 Construction of the Ranking

In my optimisation experiments I have used 54 small non-homologues protein chains used by Zhang et. al [WSZ07]. From the set of decoys generated by I-TASSER during the Monte Carlo based structure optimisation process [ZKS02] (available online [WSZ07]) I have taken a 10% sample (one decoy every 10 I-TASSER iterations) to eliminate highly similar decoys. This resulted in a training set of 1250–2000 decoys per protein. For each decoy I have pre-calculated the values of energy terms listed in the previous section.

For each protein I have measured the similarity between the generated decoys and the known native structure. As a measure I used the root mean square deviation (RMSD) between 3D coordinates of C_α atoms of two structures minimised with respect to the rotation using Kabsch algorithm [Kab78, CSD04]. Despite the limitation of the RMSD discussed in Section 2.4.4 I use it here to make a fair comparison to the previous work [ZKS03, WSZ07].

To compensate for the inaccuracies of the RMSD as a similarity measure, we decided not to rely on the absolute value of the similarity directly, but rather on the relative rank order.

For given decoys A and B I decide only if $RMSD(A, native) < RMSD(B, native)$ and ignore the scale of absolute difference in the distance to a native $\delta = RMSD(A, native) - RMSD(B, native)$. By doing this, I also simplify the optimisation objective, as linear hierarchy is easier to reflect in energy function than exact distances between all pairs of decoys.

For each protein I sorted all decoys in increasing order of the original I-TASSER energy to obtain the initial ranking R_0 (see Figure 5.1). That is, decoys with low ranks (near the beginning of the ranking) have lower energy, while those with high ranks have higher energy. Based on R_0 I created two variants of the final ranking R_1 and R_2. Ranking R_1 was constructed by sorting R_0 by RMSD (in ascending order). In case of a tie, the rank from R_0 was used as a second sorting criterion. Effectively R_1 is a permutation of R_0. R_2 in contrast, is a list of ranks assigned to each element of R_0 according to its position in R_1. The ranks in R_2 were averaged in case of a RMSD tie. A tie between decoys was called when RMSD values were the same up to the first two decimal places. This gave us a precision of a 1 picometer (for reference, the radius of hydrogen atom is 25 pm).

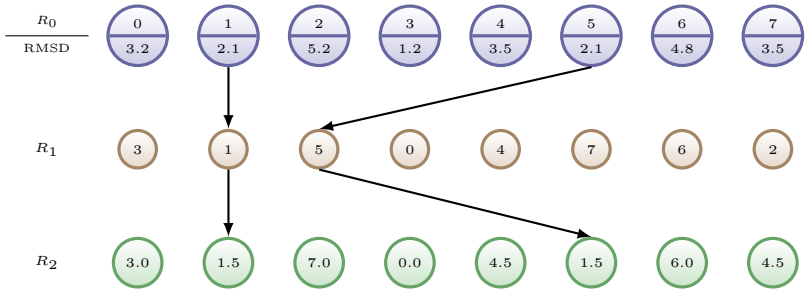

Figure 5.1: The figure shows the two approaches used to construct the ranking. R_0 is the initial ranking where decoys are sorted by the original I-TASSER energy (leftmost decoy has the lowest energy). R_1 is a permutation of R_0 where decoys are sorted by RMSD (leftmost decoy is most similar to a native structure) and a decoy index from R_0 is used to break ties (e.g. decoys 1 and 5 share the same RMSD value but as $1 < 5$, decoy 1 precedes 5 in R_1). R_2 is a list of ranks based on the order of decoys in R_1 (e.g. decoy 0 is in position 3 in R_1, so $R_2[0] = 3$). In case of RMSD ties the ranks in R_2 are averaged (e.g. decoys 1 and 5 have the same RMSD values, therefore they share the average rank of $\frac{1+2}{2} = 1.5$).

For the same set of protein chains I ran the Rosetta ab initio prediction [RSMB04] and obtained the same number of decoys as in I-TASSER case. These decoys were only used for visual assessment of correlation between Rosetta energy and RMSD (see Section 5.4).

5.2.3 Experiment Plan

I used a set of 16 terminals and 8 operators. Half of the terminals were the energy terms T_1–T_8 described in Section 5.3.2 (see Table 5.2 for the mapping to I-TASSER terms), half were ephemeral random constants in range [-1,1]. Half of the operators were binary (addition, subtraction, multiplication, division), half were unary (sine, cosine, exponential function, natural logarithm). I did not impose any selection bias towards any of the primitives.

The key element of my evolutionary mechanism is the objective function used to calculate the fitness of the candidate energy functions in the population. For each protein the

objective function was used to rank the decoys using the evolved energy function. Then this ranking was compared to the reference ranking (obtained in the preprocessing stage) and the normalised distance between the two was averaged for all proteins in the training set, producing the total fitness.

I used several different methods to calculate the distance between rankings:

- *Levenshtein edit distance*[Lev66], 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*[Kni66], 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 footrule distance*[DKNS01], the sum of differences between the ranks of elements.

$$S(a,b) = \sum_{i}^{n} |a(i) - b(i)|$$

Notice that the measures differ in the computational cost. For the Levenshtein distance a dynamic programming algorithm has to be used with a complexity of $O(n^2)$. Kendall distance can be computed in $O(nlogn)$ time by counting inversions during the merge sort procedure. Spearman distance is the simplest measure of these three and can be calculated in a linear time.

Both Kendall and Spearman distances are bounded by $O(n^2)$, having the maximum possible distance equal to respectively $\frac{n(n-1)}{2}$ and $\frac{1}{2}n^2$ for the reversed ranking. Levenshtein distance, similar to many other editing distance metrics on permutations such as Hamming metric, Cayley distance or Ulam metric, is bounded by $O(n)$.

For the Spearman distance I have applied an additional weighting mechanism to promote correct order at the beginning of the ranking (more native-like) and to be more forgiving for differences in the order at the end (less native-like). I used two weighting functions:

- linear function decreasing from 1 to 0 along the position in the ranking,

$$w_{LIN}(i) = 1 - i/N, \text{ for } 0 \leq i < N \tag{5.1}$$

- sigmoid function with inflection point (weight 0.5) at 25% of the ranking length.

$$w_{SIG}(i) = \frac{1}{1 + \exp(\frac{i - 0.25N}{scale * N})}, \text{ where } scale = \begin{cases} \frac{0.25N}{width} & \text{if } i < 0.25N \\ \frac{0.75N}{width} & \text{if } i \geq 0.25N \end{cases} \tag{5.2}$$

Additional experiments were performed with the reduced data sets. Instead of using all decoys for each protein, I used a small sample of decoys. To generate them I have applied two sampling methods: simple selection of k decoys and noise filtering. The first method was used with $k = 100$ in three variants: random selection, uniform selection (every n/kth decoy), decoys with the lowest original I-TASSER energy. The goal of the noise filtering method was to obtain a uniformly sampled set of decoys for which the original I-TASSER energy is correlated with similarity. Starting from the decoys most similar to the native, the set of all decoys was divided into bins. Two variants of bins were used: equal size bins (same number of decoys) and equal distance bins (same RMSD range). From each bin a single most similar decoy was selected such that its original I-TASSER energy value was greater than for the decoy selected from the previous bin. If none such decoy existed no sample was selected from the bin. Along arbitrary selected number of bins $b = 100$ I also used b for which the average number of samples obtained for all proteins was the largest. I found it to be 42 for equal size bins and 58 for equal distance bins.

I have implemented the genetic programming algorithm using the Open BEAGLE framework [GP06]. Base GP configuration used in all my experiments included two replacement strategies: generational and steady-state [Sys90], the tournament selection [GD91] and the population size set to 100. Mutation was done using three different operators: sub-tree replacement with a random tree, sub-tree swap or tree shrink where a tree node is replaced by one of its child nodes. Table 5.1 shows the probabilities of all evolutionary operators used in my experiments.

operation	**operator**	**probability**
crossover	non-leaf crossover point	0.70
	leaf crossover point	0.10
mutation	sub-tree replacement	0.05
	tree shrink	0.05
	sub-tree swap	0.05
reproduction	copy without modification	0.05

Table 5.1: List of the evolutionary probabilities used in all experiments.

This configuration is derived from an initial exploratory trial performed on a set of first two proteins (*1abv* and *1af7*). As for the more complex configurations the evolutionary progress was similar or even smaller than in case of the basic ones (see Figures A.5–A.7 in the Appendix), I decided to keep the experiments simple and not to use the more advanced features of the Open BEAGLE framework such as constrained operators [Mon95], multi-deme populations, evolutionary module acquisition (EMA) [AP93] or hierarchical fair competition (HFC) replacement [HGS+05]. I used the traditional low rate for mutation to have a majority of the offspring created by the crossover [PLM08]. I kept the population size low due to complexity of the fitness evaluation both in terms of CPU time and the memory requirements, however, it would be an interesting future work to investigate the behaviour of the algorithm with much larger populations.

I have run my experiments in three rounds changing the factors of the next round based on the results of the previous one. As a result, the first two rounds are of an exploratory nature, while the third round is more aggressive towards increasing the correlation. Configuration of all rounds is shown in Figure 5.2.

In the first round I used Levenshtein, Kendall and non-weighted Spearman distances. In the second round the ineffective Levenshtein distance was rejected and the linear and sigmoid weighting was added to the Spearman distance. To have more selection pressure I changed the tournament size from 2 in previous round to 4, 6 and 8. In the third round I used only sigmoid weighted Spearman's distance together with the list of ranks R_2 instead of permutational ranking R_1 used in both previous rounds. The tournament size has been set to 8 for generational and 6 for steady-state replacement. Both these strategies were used alone, with strong 5-elitism or with automatically defined functions (ADF) operators [Koz94]. The set of all decoys was extended with 5 types of reduced sets and the number of generations was increased to 2000 from 1000 used in previous rounds.

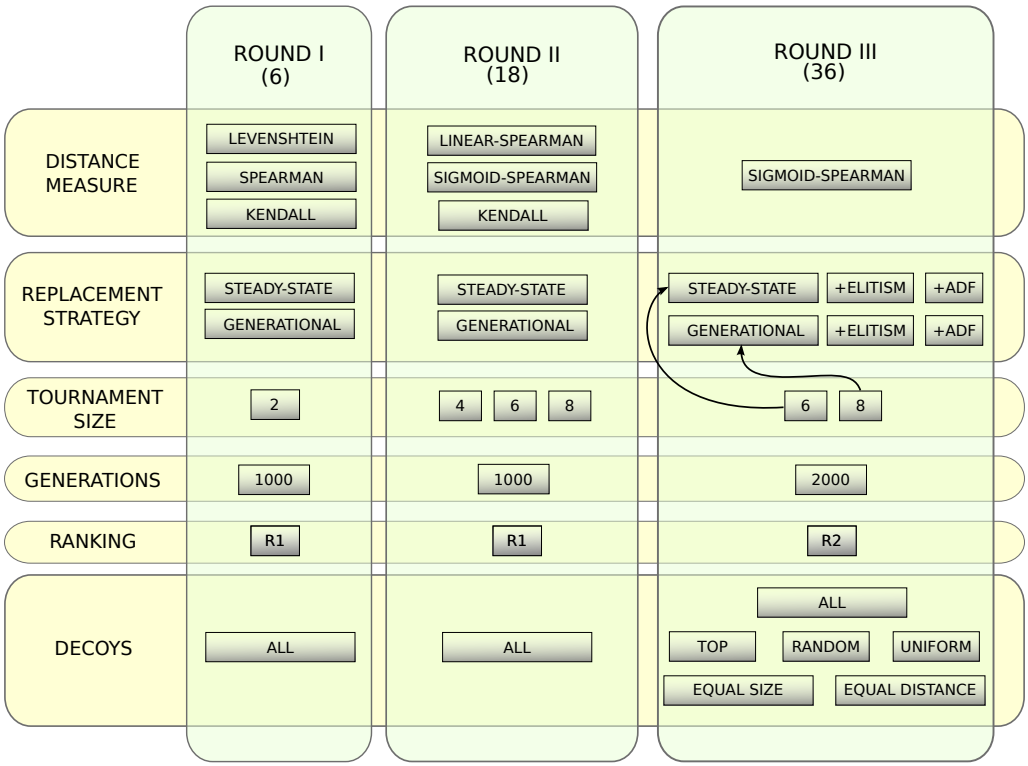

Figure 5.2: Configuration of the three rounds of experiments. The number in brackets is the number of experiments run in each round.

In all experimental rounds, a random walk was performed as a baseline for comparison. At each generation the population was created using the half-and-half initialization operator [Koz92, LP01].

Each experiment was repeated 5 times with different random seeds. In total I have conducted 60 different experiments using over 100 CPU days to perform 300 GP runs. In the next section I report only the results of a best run, as I am not interested in the average behaviour of the algorithm but rather in the fittest GP-designed energy function.

Experiments from the last round were repeated using the similarity consensus. Instead of using the RMSD as a similarity measure, the mean similarity consensus (see Chapter 4) was used to construct the reference list of ranks R_2. The ties between decoys were decided using 4 different precision levels: 2, 3, 4 and all 12 decimal places.

5.3 Initial Analysis

5.3.1 Energy Correlation

To compare the ability of the original energy functions used by I-TASSER and Rosetta to distinguish between native-like and non-native structures, I plotted the relation between the energy and similarity to the native for all decoys.

The average correlation between the original I-TASSER energy and the similarity to the native structure measured by RMSD is shown in Figure 5.4. Each panel represents a set of decoys for a single protein and the Pearson correlation coefficient $\rho(x, y) = \frac{cov(x,y)}{\sigma_x \sigma_y}$ is given in brackets. The average correlation coefficient for all proteins was 0.44 ± 0.23 (second value is a standard deviation). Interestingly, even the highest correlation coefficient e.g. for *1f05A* or *2f3nA* (see the close-up in Figure 5.3), is not enough to point to the most native-like structure as we observe a flat cloud in the lowest energy region stretched over RMSD range of 0.1–0.2nm. This cloud becomes wider with a decrease of the correlation coefficient and its center tends to shift towards greater values of RMSD.

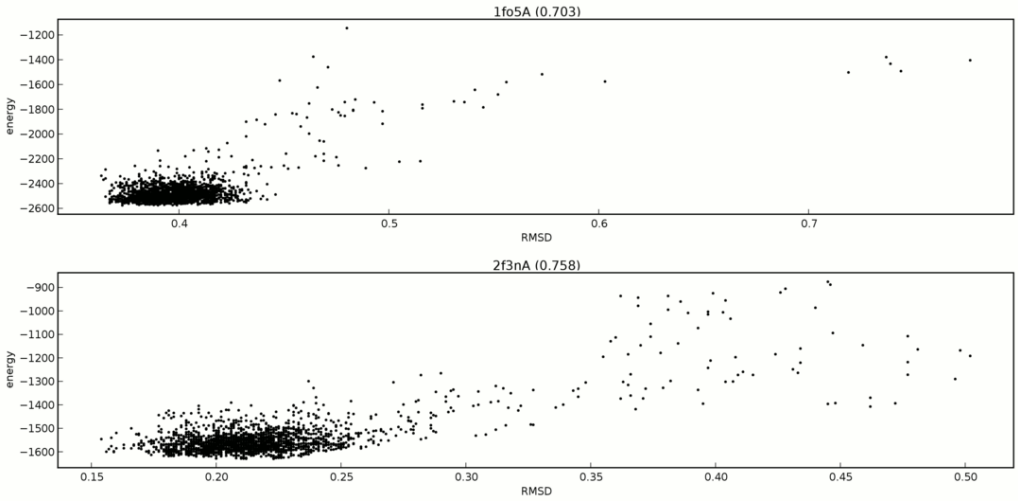

Figure 5.3: Scatter plots of I-TASSER energy vs. RMSD. Close-up for two proteins with high correlation coefficient (given in brackets). Distance is given in nanometers.

This difficulty in selecting the most native-like decoys is even more visible when the energy is plotted against the rank (see Figure 5.5). Instead of a clear trend with the energy decreasing along the decreasing rank, the trend line is very flat and thick. Several slightly slanted vertical stripes are visible in regions were a number of decoys equally distant from the native differs in energy (*2f3nA*, *10f9A*). Overall, the correlation to the rank was almost half as low as in the case of RMSD with the total average of 0.25 ± 0.16.

Similar plots for decoys generated by Rosetta are shown in Figures 5.6, 5.7. As the decoys cover a wider RMSD range and are not concentrated in a single region, the total average correlation coefficient to the rank, equal to 0.28, is only 0.02 lower than the coefficient of correlation to RMSD. However, as in case of the I-TASSER energy, pointing out the native-like decoys using the value of Rosetta energy is in most cases very difficult.

The comparative plots with both I-TASSER and Rosetta decoys marked on a shared RMSD/rank scale are presented in the Appendix (Figures A.1–A.4).

5.3.2 I-TASSER Energy Terms

Coefficients for individual energy terms are shown in Table 5.2. The values for the decoys I used are significantly lower than the one reported by Zhang et al. [ZKS03]. Notice the negative correlation of selected terms which decreases the average correlation nearly to zero. The low values of the ρ_2 coefficient could, however, be somewhat misleading as they are hiding the spread amongst different proteins. The relative standard deviation for ρ_2 ranged from 82% for T_2 to 942% for T_6.

The average correlation coefficient for the naive sum of energy terms $E_N = \sum_{i=1}^{8} T_i$ was 0.12 ± 0.16. Correlation between the naive sum of energy terms and the rank was lower as in the case of original I-TASSER energy, and the coefficient value was 0.07 ± 0.16.

energy term	ρ_1	ρ_2	σ_2	ρ_E	σ_E
T_1 (E_{13})	0.27	0.03	0.11	0.08	0.15
T_2 (E_{14})	0.56	0.20	0.17	0.38	0.16
T_3 (E_{15})	0.33	0.15	0.15	0.34	0.19
T_4 (E_{stiff})	0.25	0.24	0.22	0.44	0.24
T_5 (E_{HB})	0.51	-0.16	0.20	-0.36	0.23
T_6 (E_{pair})	0.38	0.01	0.14	0.12	0.13
T_7 ($E_{electro}$)	0.27	-0.20	0.23	-0.34	0.26
T_8 (E_{env})	0.34	0.04	0.16	0.03	0.15
average	0.36	0.04	0.17	0.09	0.19

Table 5.2: Both ρ_1 and ρ_2 represent the average correlation between a single energy term and the similarity to native structure measured by RMSD. ρ_1 is the coefficient originally reported by Zhang et al. [ZKS03] and ρ_2 is the coefficient calculated for my implementation of I-TASSER energy terms on 54 proteins used in my experiments. ρ_E is the correlation coefficient between a single energy term and the original I-TASSER energy. σ_2 and σ_E represent the standard deviation of ρ_2 and ρ_E coefficients. In case of the hydrogen bonds potential E_{HB}, ρ_1 and ρ_2 cannot be directly compared, as the latter apply to the new implementation of this term [ZHA+06b].

5.3.3 Fitness Distance Correlation

The optimisation objective was to minimise the distance $d(R_r, R_e)$ between the reference ranking R_r and the ranking R_e produced by the evolved function. The range of distances was normalised to the [0,1] fitness range, where maximum distance (comparison with reversed R_r) gives fitness equal to 0 and zero distance corresponds to fitness equal to 1.

To compare the landscape difficulty of different fitness functions I have measured the fitness distance correlation on the phenotype level. Starting from a random reference ranking R_r of length 100, for each of $t \in \{1, \ldots, 1000\}$ steps 20 new permutations were generated by applying a random transposition to the permutations from previous step $t-1$. The correlation of the fitness functions to the distance to R_r measured in number of applied transpositions, as well as the direct correlation between the fitness functions, is shown in Figure 5.8.

Notice that the minimum number of transpositions needed to transform n-element permutation a to b, known as Cayley distance, is bounded by $O(n-1)$ and equal to $n-c$, where c is the number of cycles in the disjoint cycle decomposition of ab^{-1}. Because of that, for $n > 100$ a fluctuation of the fitness value is observed in plots A–C of Figure 5.8.

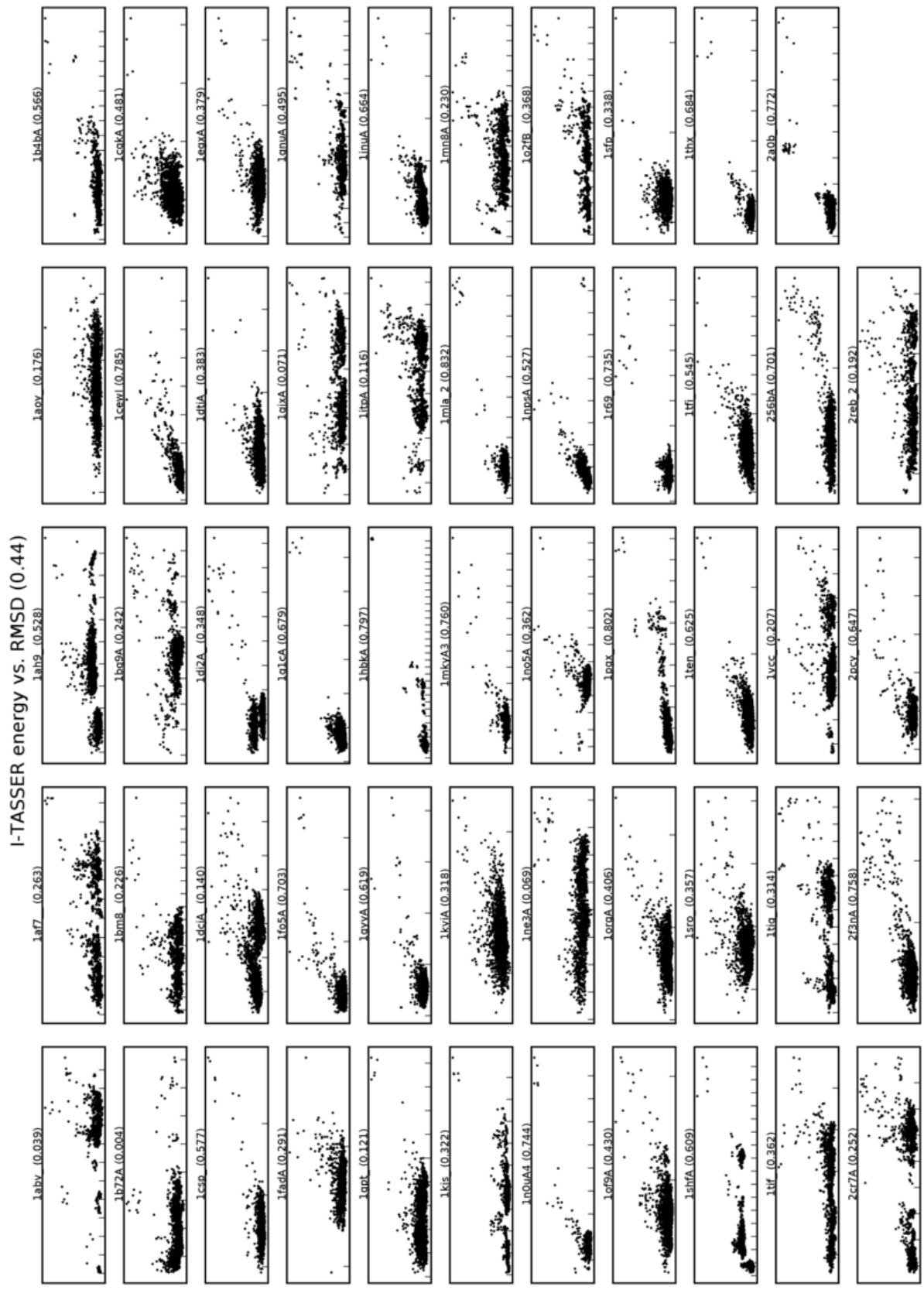

Figure 5.4: Scatter plots of I-TASSER energy (vertical axis) vs. RMSD (horizontal axis). Each plot contains all decoys for a single protein. Correlation for each plot is given in brackets. Distance between ticks on horizontal axis is 0.1nm.

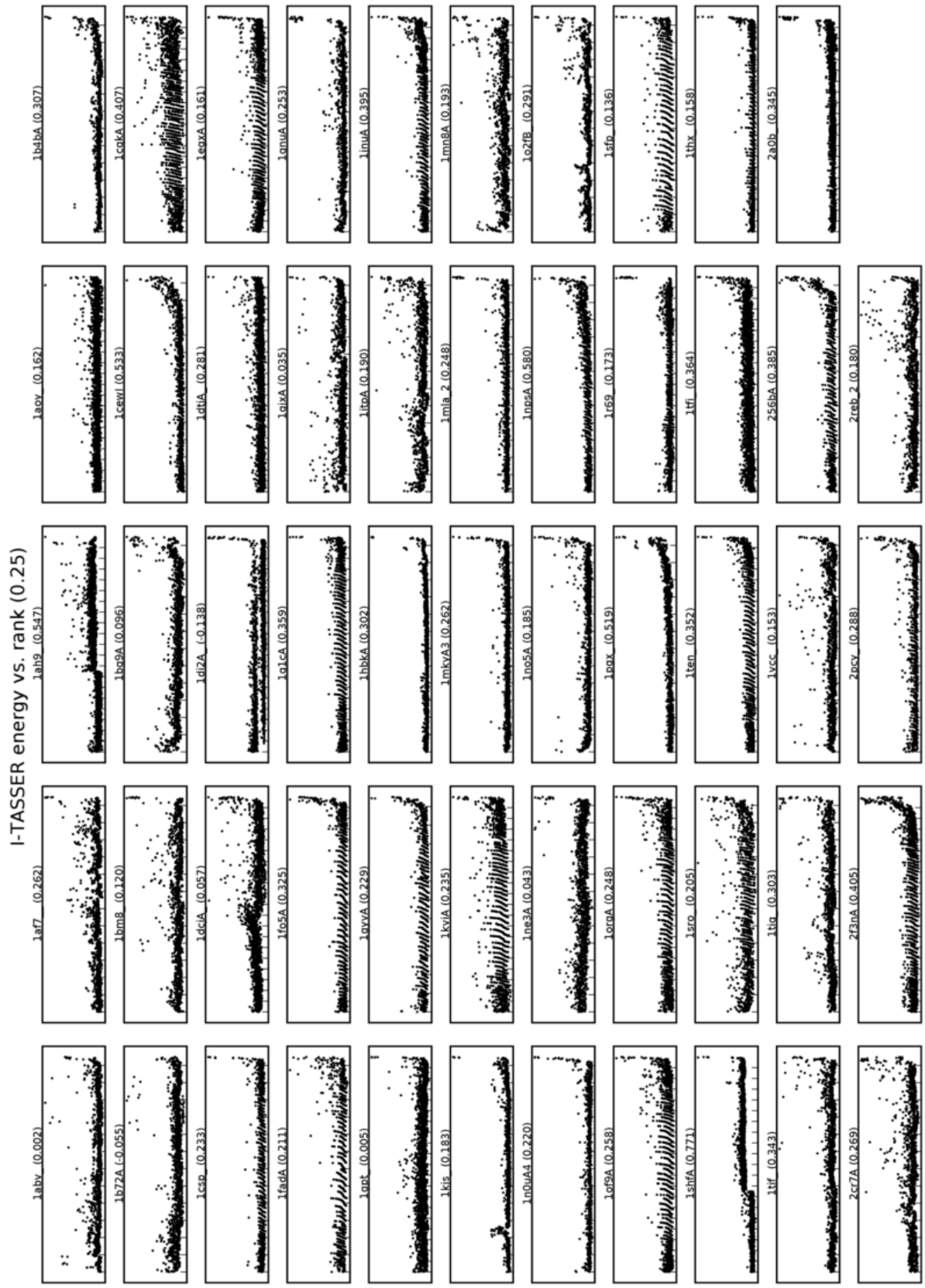

Figure 5.5: Scatter plots of I-TASSER energy (vertical axis) vs. rank (horizontal axis). Each plot contains all decoys for a single protein. Correlation for each plot is given in brackets. Distance between ticks on horizontal axis is 100 ranks.

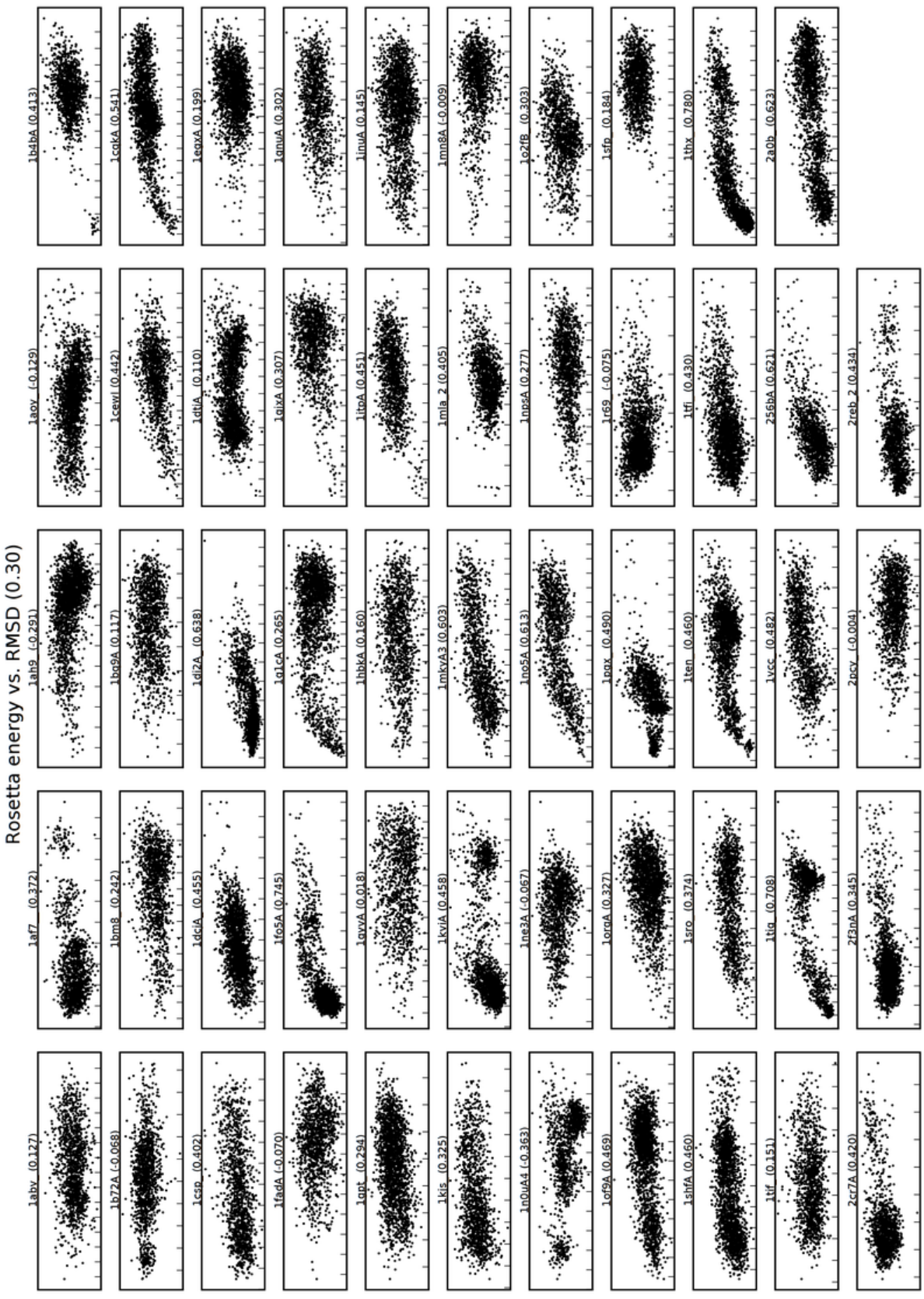

Figure 5.6: Scatter plots of Rosetta energy (vertical axis) vs. RMSD (horizontal axis). Each plot contains all decoys for a single protein. Correlation for each plot is given in brackets. Distance between ticks on horizontal axis is 0.1nm.

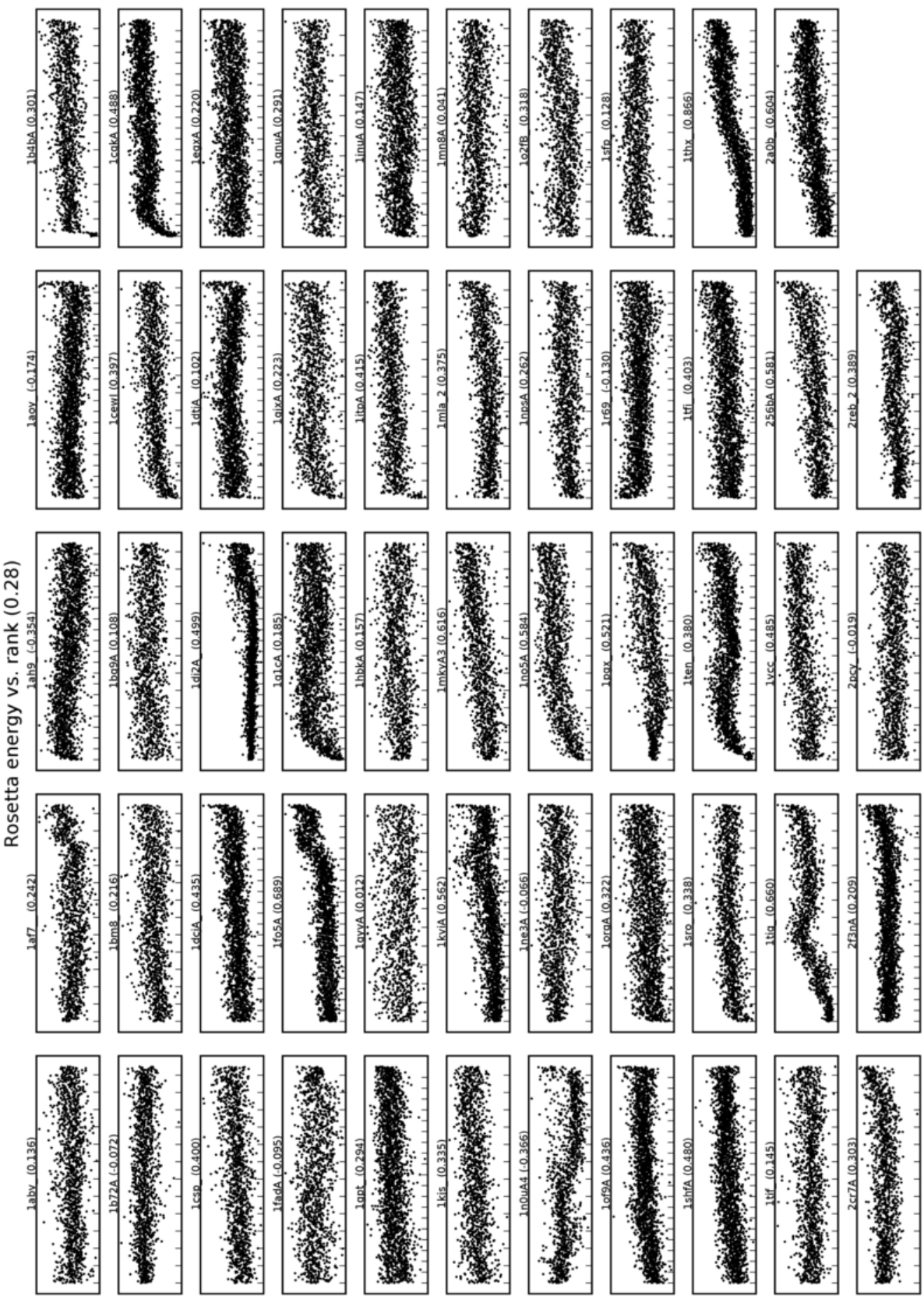

Figure 5.7: Scatter plots of Rosetta energy (vertical axis) vs. rank (horizontal axis). Each plot contains all decoys for a single protein. Correlation for each plot is given in brackets. Distance between ticks on horizontal axis is 100 ranks.

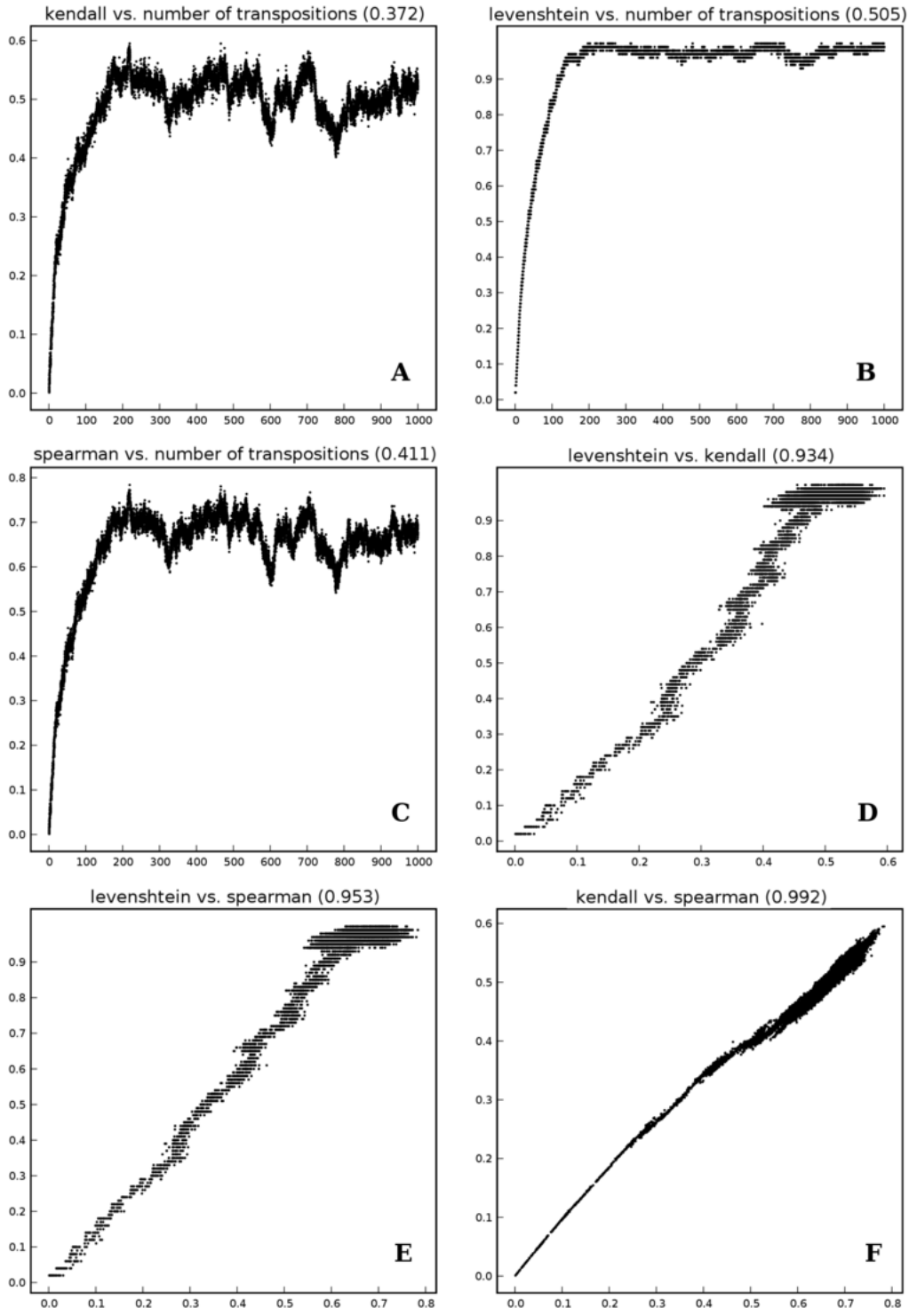

Figure 5.8: Correlation between fitness and number of transpositions applied to the reference ranking (plots A–C) and direct correlation between fitness functions (plots D–F). The correlation coefficients are given in brackets.

As the range of values that the fitness function based on the Levenshtein distance can obtain is a square of the range obtainable for Spearman and Kendall distances, the fluctuation range is also lower.

This limitation is visible even more clearly in plots D and E of Figure 5.8. The distinct horizontal stripes appear for groups of permutations equally distant from R_r in the Levenshtein metric space but easily distinguishable by the other two distances. The gaps between the stripes are another indicator of sparse space of values of the Levenshtein distance. The horizontal stripes become longer near the maximum of Levenshtein distance, showing inablity of this measure to distinguish between many samples slightly above the middle of other two distances range.

5.4 Results

5.4.1 Round I: initial fitness experiments

As could be seen in Figure 5.9 the average fitness for the Levenshtein distance varies in a tiny range just above zero (the maximum distance). For the Spearman distance the average fitness improved quickly in the first 50–100 generations and saturated later around 40% of the maximum fitness. The initial improvement was more rapid in case of the Kendall distance but the spread of the fitness was very small, covering only 3% range of the maximum fitness. The best evolved functions were only slightly (1.3% and 5.5% for the Kendall and Spearman distances respectively) better than the best function found by the random walk (see Table 5.4.1).

To check the statistical significance of the results I have performed the Kruskal-Wallis one-way analysis of variance to test the null hypothesis that medians of two average fitness distributions shown in Figure 5.9 are equal. The hypothesis was rejected for both 9 vertical (across measures of distance) and 9 horizontal (across configuration) pairwise comparisons at the 99.9% confidence level ($p < 0.001$).

	steady-state		**generational**	
measure	**max**	**improvement**	**max**	**improvement**
Levenshtein	0.004	25.91%	0.003	13.01%
Spearman	0.417	4.72%	0.420	5.49%
Kendall	0.520	0.97%	0.522	1.34%

Table 5.3: Maximum fitness and improvement over the random walk for two run configurations and three fitness functions used in the 1st round of experiments.

5.4.2 Round II: increased selection pressure

The linear weighting mechanism did not change the fitness landscape but the sigmoid weighting did. As shown in Figure 5.11 the Spearman distance with sigmoid weights reached over 20% higher average and maximum fitness value than the non-weighted Spearman distance. The evolutionary progress for the best runs of both linear and sigmoid weighted Spearman distance continues steadily without the early saturation

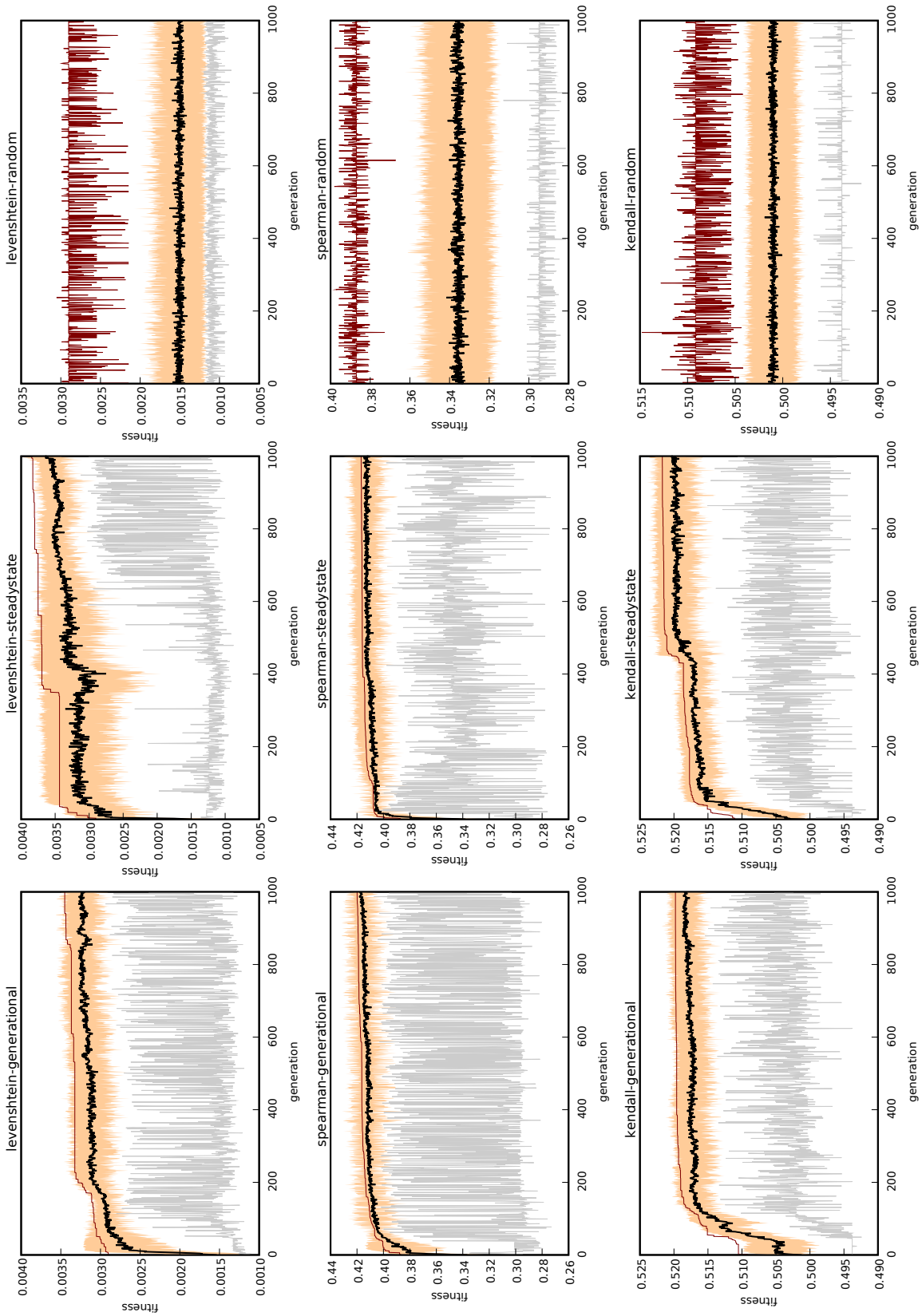

Figure 5.9: Fitness throughout the generations in the first round of experiments. Lines show the average (thick black), minimum (thin grey) and maximum (thin red) fitness in the population. Filled area around the average represents the standard deviation. Each row corresponds to a single fitness function and each column corresponds to a single GP configuration.

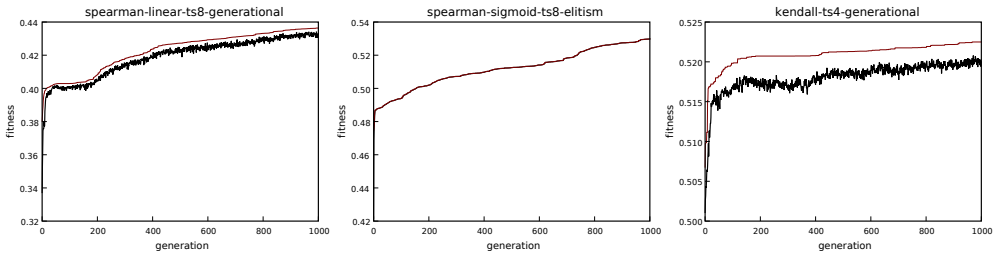

Figure 5.10: Fitness throughout the generations in the second round of experiments for the best run for each fitness function. The lines show average (thick black line) and the maximum (thin red line) fitness in the population.

observed in the first round of experiments (see Figure 5.10). However the spread of fitness values is again low, covering only about 10% of the fitness range (see Figure 5.11).

I did not observe a significant change in the evolutionary improvement over the random walk for the Kendall distance. In case of the Spearman distance with linear weights the improvement seems to be even twice as big (up to 11%) as in the previous round. However, the maximum fitness values are still in the 0.4–0.5 range, so similarly far away from the theoretical maximum as in the first round of experiments.

As the increase of the tournament size did not result in increased fitness of the best solutions, for the next round of experiments I have arbitrarily picked the tournament size 6 for the steady-state replacement and 8 for the generational one, just by the visual assessment of the fitness progression throughout generations.

		steady-state		**generational**		**elitism**	
measure	**ts**	**max**	**impr**	**max**	**impr**	**max**	**impr**
Kendall	4	0.516	0.19%	0.522	1.49%	0.514	-0.08%
	6	0.515	0.01%	0.517	0.42%	0.513	-0.25%
	8	0.520	1.03%	0.514	-0.12%	0.517	0.45%
Spearman linear	4	0.416	6.24%	0.429	9.47%	0.418	6.67%
	6	0.430	9.79%	0.404	3.10%	0.408	4.18%
	8	0.419	6.78%	0.436	11.34%	0.423	7.80%
Spearman sigmoid	4	0.518	7.49%	0.503	4.38%	0.527	9.47%
	6	0.516	7.11%	0.514	6.71%	0.511	6.04%
	8	0.515	6.98%	0.508	5.38%	0.530	9.96%

Table 5.4: Maximum fitness and improvement over the random walk for three run configurations with different tournament size and three fitness functions used in the 2nd round of experiments.

5.4.3 Round III: shared ranks and reduced data sets

Since in this round I used averaged ranks and in consequence the ranking was not a permutation any more, I did not use the Kendall distance. The fitness function based on the sigmoid weighted Spearman distance was used in all experiments. The addition of ADFs or strong elitism improved the best fitness in several cases but I did not find any tendency that would be common for all sets of decoys.

For the set of all decoys, the value of the average and maximum fitness was around 40% higher compared to the permutational ranking. However, as shown in Table 5.5 the

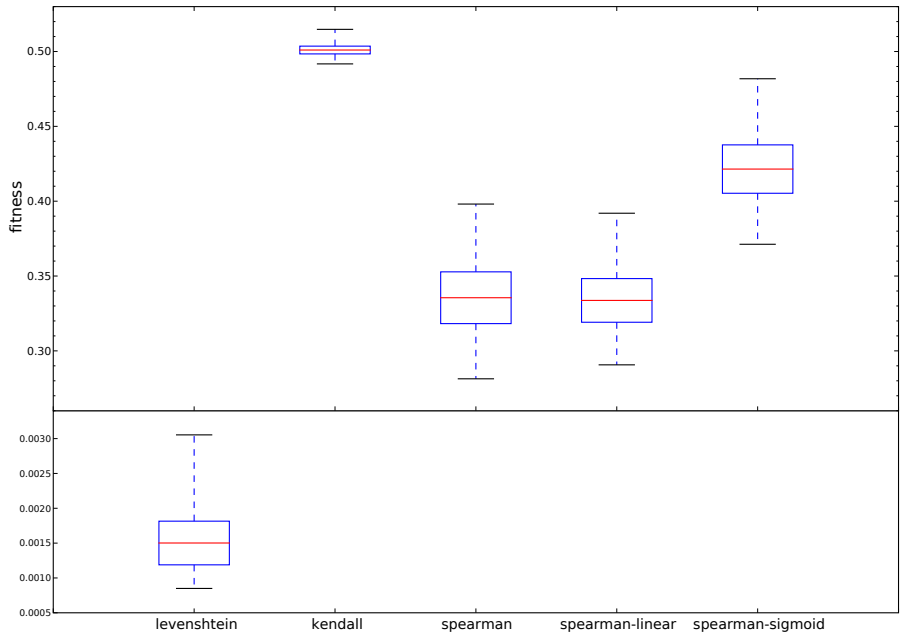

Figure 5.11: Box plot of the fitness distribution achieved by a random walk for ranking distances used in first two rounds of experiments. Middle line is the median of the average fitness in population across all generations. Box size represents the median of the population fitness standard deviation. Top and bottom whiskers marks maximum and minimum fitness across all individuals.

evolution process could not improve significantly over the random walk, despite the fact that the best randomly found energy function was as simple as $f = 0.412/T_6$. Similar results were obtained for the three sets of 100 selected decoys: random, uniform and top. For the noise filtered set of decoys, where the GP trees of the best functions were over two times larger, the improvement over random walk was greater than 7%.

		best fitness		**best tree size**		**best tree depth**	
decoys set	**impr**	**max**	**avg**	**max**	**avg**	**max**	**avg**
all	0.78%	0.714	0.710	380	186.2	18	16.8
uniform-100	0.96%	0.716	0.710	289	151.5	18	17.3
random-100	1.28%	0.720	0.713	289	151.5	18	17.3
top-100	1.93%	0.709	0.701	266	118.7	18	15.5
f-42	7.76%	0.729	0.713	1110	485.0	18	17.7
f-100	7.64%	0.788	0.772	629	414.7	18	17.5
d-58	8.21%	0.809	0.780	752	388.7	18	17.3
d-100	10.88%	0.835	0.804	793	329.2	18	17.5

Table 5.5: Comparison of the best evolved functions for different sets of decoys. Second column shows percentage fitness improvement over the random walk. The next columns show the maximum and the average value of fitness, tree size and tree depth for the best functions of all six run configurations.

The change in fitness landscape for the noise filtered sets of decoys is shown in Figure 5.12. The range of fitness values for the random walk increased up to 40% of the maximum fitness from 15% observed previously for all decoys. Although the average fitness was lower, the fitness of the best individuals has improved reaching 0.75 for the set of decoys created using 100 equal distance bins (d-100).

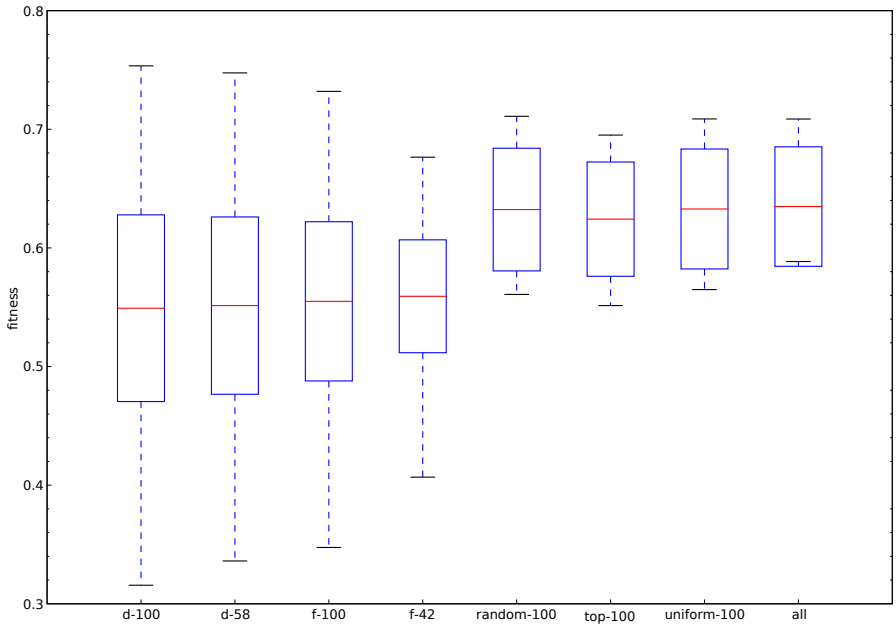

Figure 5.12: Box plot of the fitness distribution achieved by a random walk with sets of decoys used in the third round of experiments. Middle line is the median of the average fitness in population across all generations. Box size represents the median of the population fitness standard deviation. Top and bottom whiskers marks maximum and minimum fitness across all individuals.

The overall fittest evolved individual and the greatest evolutionary improvement over the random walk was observed again for the d-100 set. The best evolved energy function had almost 11% greater maximum fitness (0.835) than $f = 0.3433 * T1 + T3 - T6$ found by chance. However, the GP tree was difficult to analyse because of a bloat. As I did not introduce any size penalty in the fitness function, the average size of a GP tree was increasing through generations as shown in Figure 5.13.

The distribution of the terminals and operators used in the best functions evolved for each set of decoys is summarised in Table 5.6. The most frequently used energy terms were T_4 and T_5. Interestingly, T_4 had the highest correlation to RMSD and T_5 the second lowest one (see Table 5.2). Similarly, the least frequently used energy terms, T_1 and T_6, were the ones with the correlation to RMSD closest to zero. Therefore, the GP optimisation based on the distance between ranks was able to discover an analogous hierarchy of the energy terms. Across the operators, the most frequent were addition and division with transcendental functions (sine, cosine and natural logarithm) being the least frequent.

decoys set	T_1	T_2	T_3	T_4	T_5	T_6	T_7	T_8	**add**	**sub**	**mul**	**div**	**log**	**exp**	**sin**	**cos**	**total**
all	0	0	10	18	19	3	10	1	99	2	0	0	8	104	4	4	323
uniform-100	4	0	0	1	2	1	3	4	0	15	1	0	4	21	0	1	59
random-100	1	4	5	3	1	5	14	6	6	1	27	46	12	24	2	4	203
top-100	0	0	0	0	0	3	9	9	0	0	9	61	4	48	64	3	260
f-42	11	49	76	84	26	3	105	19	310	38	25	174	5	8	1	1	1110
f-100	1	42	45	50	48	14	3	33	158	39	65	22	14	32	13	1	629
d-58	26	8	1	40	69	32	21	3	193	34	26	90	4	28	6	27	752
d-100	6	6	12	60	41	17	19	22	52	102	37	78	0	34	0	17	590

Table 5.6: The distribution of terminals and operators in the best evolved functions for different sets of decoys. Ephemerals are not shown.

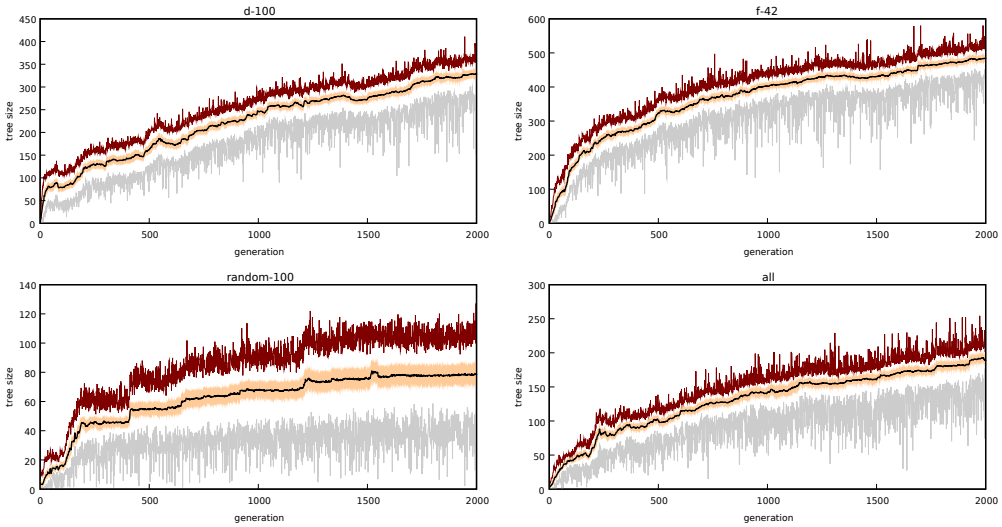

Figure 5.13: Tree size throughout the generations. Sizes are averaged over all six GP configurations for a selected set of decoys from the third round of experiments. Lines show the average (thick black), minimum (thin grey) and maximum (thin red) tree size in the population. Filled area around the average represents standard deviation.

The best energy function evolved for the d-100 set was correlated to RMSD with the coefficient of 0.76 ± 0.19 and to the rank with coefficient of 0.74 ± 0.18. When the best evolved function was applied to the set of all decoys, the corresponding correlation coefficients dropped to 0.30 ± 0.18 (shown in Figure 5.14) and 0.20 ± 0.17. Still, compared to naive combination of terms the evolutionary optimisation improved the correlation coefficients 2.5 and 2.85 times respectively.

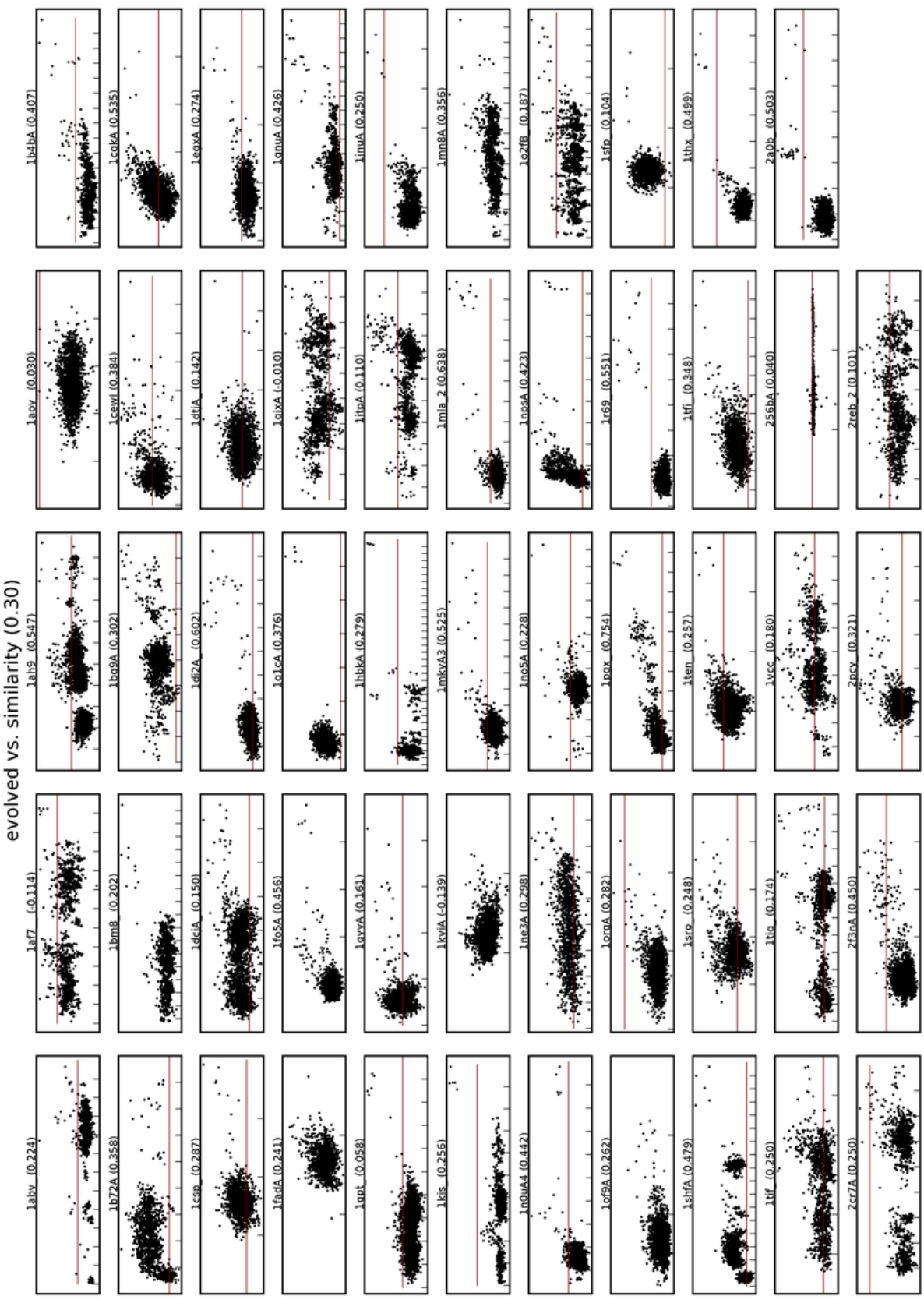

Figure 5.14: Scatter plots of the best energy function evolved for d-100 set (vertical axis) vs. RMSD (horizontal axis). Red horizontal line marks the energy of the native structure. Each plot contains all decoys for a single protein. Correlation for each plot is given in brackets. Distance between ticks on horizontal axis is 0.1nm.

Chapter 6

Evolvability of Energy Functions

In this chapter I further analyse the result of the genetic programming experiment reported in Chapter 5. The best evolved energy functions are compared against a baseline experiment with a linear combination of energy terms, where weights were optimised using the Nelder-Mead downhill simplex method. Successful and unsuccessful GP runs are analysed using a number of population diversity statistics. The evolvability of the energy functions with rankings based on the similarity consensus are presented and the limitations of the GP approach are discussed. The chapter is concluded with a summary of a work done and suggestions of possible directions for future work.

6.1 Final Analysis

6.1.1 Comparison to Linear Combination of Terms

As I demonstrated before, the best energy function found by my GP algorithm provide significantly better prediction guidance than the naive combination of terms or best functions found by the random walk. Moreover, the GP algorithm was able to automatically discover the most and the least useful energy terms without having any knowledge how these terms alone are correlated to RMSD.

To put these results in context, I used the Nelder-Mead downhill simplex method [NM64, McK99] to find the best weights of the energy function given as a linear combination of terms $E_L = \sum_{i=1}^{8} w_i T_i$, similar to what have been done in the original work by Zhang et al. [ZKS03]. I ran the SciPy [JOP$^+$] implementation of the algorithm using a vector of weights $\vec{w} = [w_1, \ldots, w_8]$ as a variable and minimising either the sigmoid weighted Spearman distance or the correlation coefficient between energy and rank directly. The method converged in a fraction of time allowed for GP optimisation (minutes vs. hours) performing on average only about 500 evaluations of the objective function. Table 6.1 shows the maximum objective function values obtained for d-100 and all decoy sets compared with the results of the best evolved functions. The fitness of GP-evolved functions was in all cases over 10% higher.

	spearman-sigmoid		correlation	
method	**d-100**	**all**	**d-100**	**all**
simplex	0.734	0.638	0.650	0.166
GP	0.835	0.714	*0.740	*0.200

Table 6.1: The results of the simplex method optimisation of the weighted sum of terms compared to the best GP-evolved functions. Notice that the correlation coefficient for GP marked with star was calculated after the evolutionary optimisation, while in case of the downhill simplex method it was directly used as an objective function.

6.1.2 Population Diversity Analysis

The mapping between the tree representation of a function and its fitness is very complex as it involves an evaluation of the energy of thousands of decoys and a comparison of the evolved ordering with a reference ranking for several proteins. It would not be surprising if this mapping will result in the loss of diversity between the levels as shown on Figure 6.1.

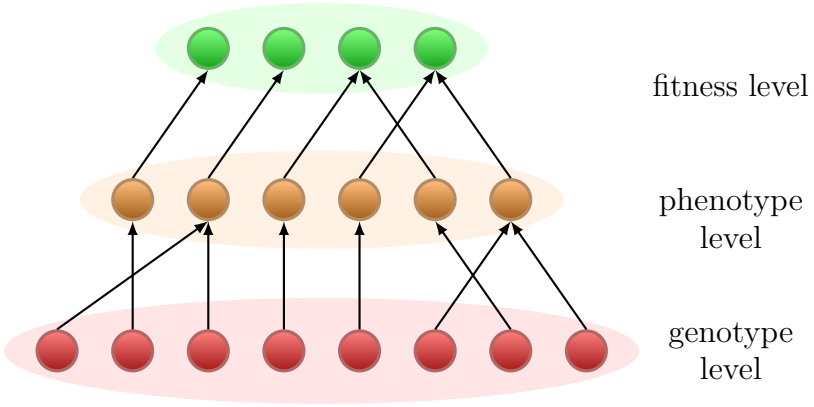

Figure 6.1: Illustration of possible mapping between GP trees, decoys ordering created by the evolved function and the total fitness.

To gain more insight into the evolutionary process and usefulness of the proposed fitness functions, I have collected a wide range of population diversity statistics for the best configurations from the last round of experiments. Burke et al. [BGKK02, BGK04] have shown that to understand the evolutionary dynamics the diversity should be measured on several levels. As suggested there, I measured the population diversity on three levels:

- *genotype*, I measured the number of unique trees using a pseudo-isomorphism measure [BGK04], where each tree is described by a triple <# terminals, # non-terminals, tree depth>
- *phenotype*, for each of n individuals in the population I ranked the decoys and measured the average root mean square distance between the ranks (using the Spearman footrule distance), which I then have averaged for all m proteins obtaining phenotype rmsd:

$$\frac{1}{m}\sum_{p=0}^{m}\frac{2}{n(n-1)}\sqrt{\sum_{i=0}^{n}\sum_{j>i}^{n}d(r_{pi},r_{pj})^2} \tag{6.1}$$

- *fitness*, I measured the entropy in the population based on the frequency of occurrence of fitness values (using a precision of three decimal places):

$$\sum_{i=0}^{n} p_i * log(p_i), \text{ where } p_i = \frac{1}{n} duplicates_i \tag{6.2}$$

For individual runs I have observed a rapid loss of the diversity on both genotype and phenotype levels after a few initial generations. However, it was not the case for the diversity on the fitness level, which usually did not decrease even for late generations. Moreover, for the best individual runs the maximum fitness is not stagnating but slowly improving throughout 2000 generations (see Figure 6.2). Hence, the early saturation of the average fitness does not seem to be related to a converged population.

I have analysed the common diversity characteristics of a group of the most successful runs and I found the best evolutionary progress to be related to a gradual decrease of the phenotype diversity and high or increasing diversity on the fitness level (see runs A and C in Figure 6.2). Interestingly, a high diversity on the fitness or phenotype level alone did not result in good evolutionary progress (see runs D and E in Figure 6.3). A high diversity on tree level for late generations seems to indicate cases when evolution was trapped around a low quality local optima.

6.1.3 Similarity Consensus

The I-TASSER energy terms were designed by experts with the RMSD correlation in mind. Some of the parameters of these terms were tweaked to maximise that correlation introducing some bias towards RMSD. Because of that, the correlation between the energy terms and the similarity consensus might be lower than in case of RMSD. This was indeed observed for the RMSD consensus. However, correlation coefficients for the similarity consensus were very similar to the coefficients of the correlation to RMSD (see Table 6.2 and compare with Table 5.2).

	RMSD consensus		**similarity consensus**	
	ρ_{sim}	ρ_{rank}	ρ_{sim}	ρ_{rank}
T_1	0.01 ± 0.10	0.01 ± 0.11	0.04 ± 0.13	0.03 ± 0.13
T_2	0.12 ± 0.16	0.10 ± 0.16	0.23 ± 0.15	0.19 ± 0.14
T_3	0.06 ± 0.13	0.04 ± 0.12	0.17 ± 0.13	0.13 ± 0.12
T_4	0.08 ± 0.18	0.05 ± 0.18	0.21 ± 0.22	0.16 ± 0.20
T_5	-0.07 ± 0.17	-0.04 ± 0.17	-0.16 ± 0.21	-0.11 ± 0.19
T_6	-0.02 ± 0.11	-0.03 ± 0.12	0.03 ± 0.14	0.01 ± 0.13
T_7	-0.07 ± 0.19	-0.06 ± 0.18	-0.05 ± 0.23	-0.02 ± 0.22
T_8	0.03 ± 0.14	0.04 ± 0.15	0.02 ± 0.17	0.02 ± 0.17

Table 6.2: Average correlation between I-TASSER energy terms and the consensus similarity. For each type of consensus two coefficients are given ρ_{sim} and ρ_{rank} representing direct correlation to the consensus similarity or correlation to rank.

The average correlation coefficient of the original I-TASSER energy and the mean similarity consensus was over 20% lower than in the case of RMSD (compare Figure 5.4 and Figure 4.9). Correlation to rank was also lower but only by 8%.

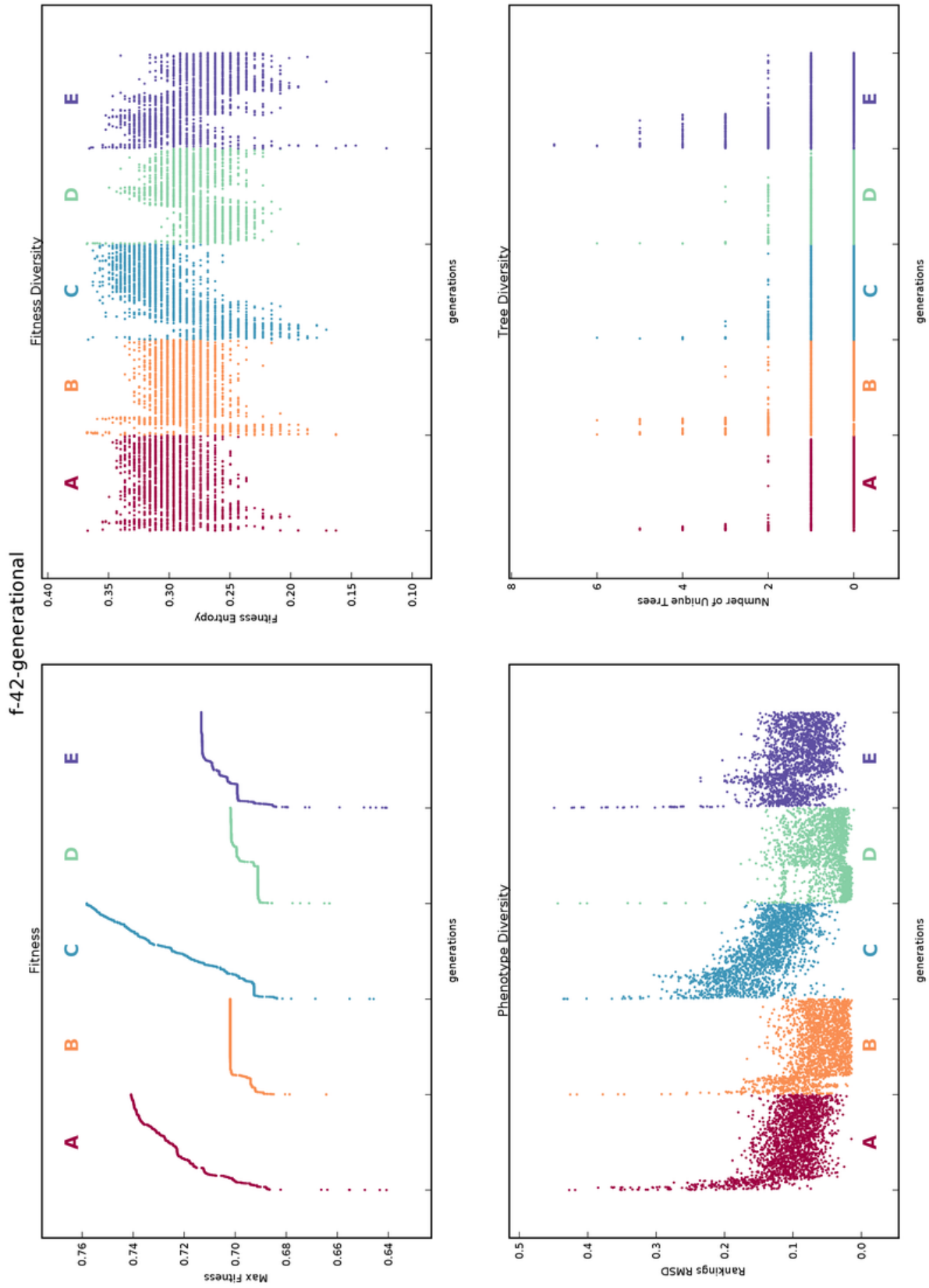

Figure 6.2: Maximum fitness throughout generations compared to population diversity on fitness, phenotype and GP tree levels (positive example). Each plot shows side by side five different runs (A-E) for a selected GP configuration.

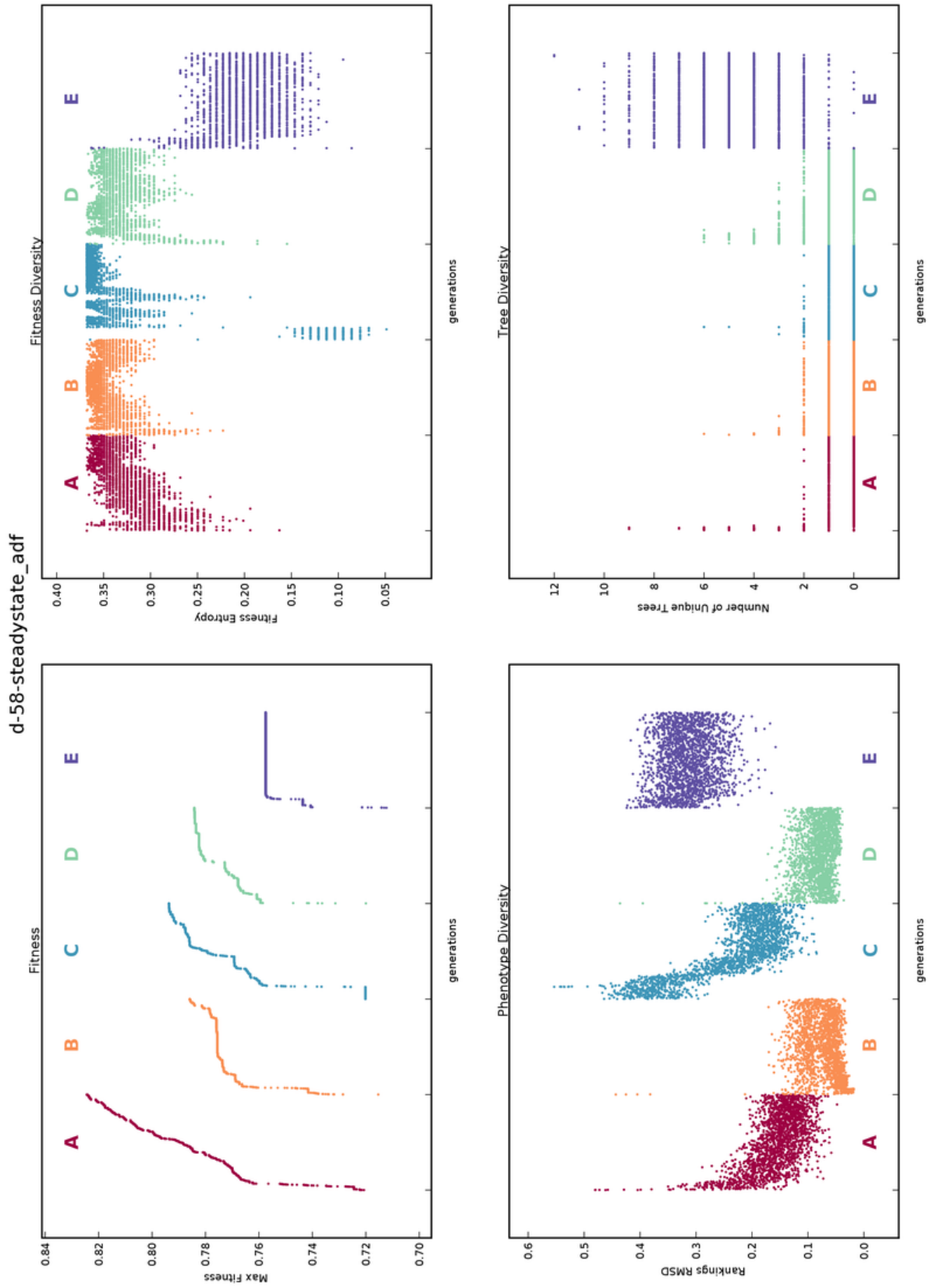

Figure 6.3: Maximum fitness throughout generations compared to population diversity on fitness, phenotype and GP tree levels (negative example). Each plot shows side by side five different runs (A-E) for a selected GP configuration.

The GP configuration from the round III of experiments (see Section 5.2.3) was used to test how different similarity measure will change the problem landscape. The best evolved functions (with the highest fitness) were obtained for ranks based on consensus using 4 decimal places precision (see the Appendix for all results). The fitness of the best functions evolved for the reduced sets was almost 4% higher than in case of RMSD (compare Table 6.3 and Table 5.5). On the other hand the best functions for the set of all proteins was almost twice as small as for the RMSD.

		best fitness		**best tree size**		**best tree depth**	
decoys set	**impr**	**max**	**avg**	**max**	**avg**	**max**	**avg**
all	0.47%	0.359	0.359	879	424.5	18	17.0
d-100	8.84%	0.868	0.847	783	454.3	18	17.2
d-58	10.32%	0.853	0.834	720	418.3	18	17.5
f-100	9.95%	0.811	0.795	886	540.5	18	17.5
f-42	9.42%	0.729	0.712	944	604.2	18	17.5

Table 6.3: Comparison of the best evolved functions for different sets of decoys for reference ranking based on the similarity mean consensus with rounding precision of 0.0001. Second column shows percentage fitness improvement over the random walk. The next columns show the maximum and the average value of fitness, tree size and tree depth for the best functions of all six run configurations.

The best energy function evolved for the d-100 set was correlated to mean similarity consensus with the coefficient of 0.76 ± 0.22 and to the rank with the coefficient of 0.74 ± 0.21. This is almost identical as in the RMSD case, only standard deviation is about 15% higher. When the best evolved function was applied to the set of all decoys, the corresponding correlation coefficients dropped to 0.07 ± 0.14 and 0.04 ± 0.10 respectively. This not only about 5 times lower than in case of RMSD but also 2 times lower than the naive sum of terms (0.14 ± 0.16 and 0.11 ± 0.15).

The fitness distribution plotted for the random walk (see Figure 6.4) clearly shows that for the set of all proteins the GP algorithm was trapped in a very narrow fitness range. This observation is also confirmed by the diversity analysis. For the set of all decoys the fitness diversity was almost twice as low as in case of reduced set of decoys or ranks based on RMSD (compare Figure 6.5 and Figure 6.2).

6.2 Discussion

In protein structure prediction a useful energy function is the one which guides the structural optimisation process towards the region of native-like structures. Therefore, it seems natural to measure this usefulness with a correlation coefficient between the energy and similarity to native. However, as I have shown, even a high correlation coefficient (> 0.7) does not guarantee that distinguishing the native-like structure from the others would be easy. This is reflected in the lower correlation to rank, since ranking ignores the scale. The lack of power to differentiate between the decoys is best observed on the energy vs. rank plots, where for several consecutive rank bins the assigned decoys are within the same energy range.

The difference between the correlation of single energy terms in my experiments and in the original work by Zhang et al. shows the difference in difficulty of the decoy sets used. It

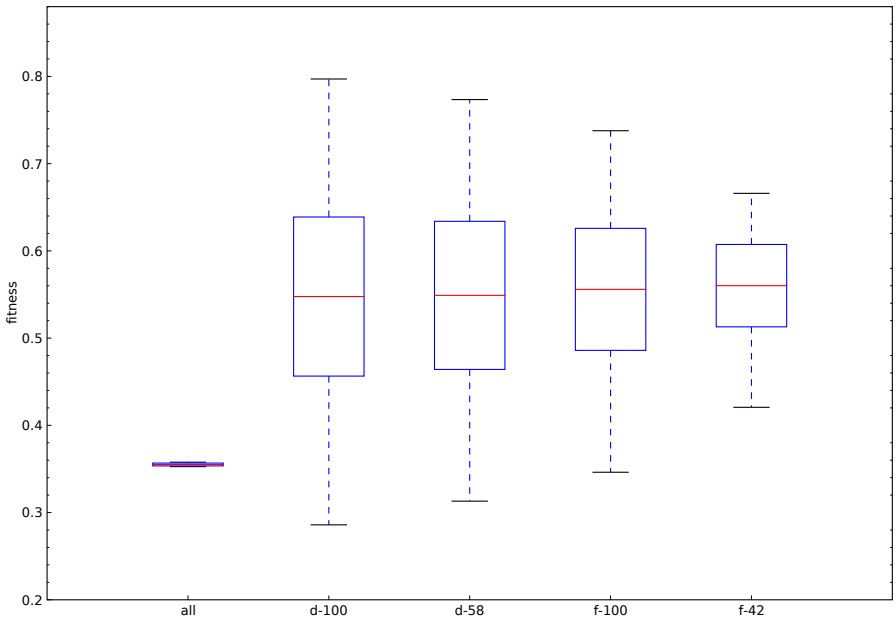

Figure 6.4: Box plot of the fitness distribution achieved by a random walk with sets of decoys used in the experiments with mean similarity consensus. Middle line is the median of the average fitness in population across all generations. Box size represents the median of the population fitness standard deviation. Top and bottom whiskers marks maximum and minimum fitness across all individuals.

seems to be more difficult to choose a native-like structure from the set of decoys sampled from the structural optimisation process, than from a set generated by randomisation of the native used by Zhang. The former starts from fragments of other proteins similar to the target on a sequence level and has no knowledge of its native structure. The latter is using the native structure directly resulting in a biased set. Moreover, the decoys used in my experiments are often very similar to each other, whereas Zhang kept them separated by large 0.35nm RMSD distance. As my results show, decoys generated by the predictor are more difficult to assess and thus optimising the energy based on the randomised and highly separated set of decoys might be inadequate as this is not what predictors have to deal with in practise.

Although the naive combination of energy terms used in my experiments compared to the original I-TASSER energy was much less correlated to RMSD, the genetic programming optimisation was able to evolve energy functions significantly decreasing this gap. Considering the fact that this is the first time this work has been done, notwithstanding the maturity of the protein structure prediction field (several decades), these results are extremely encouraging.

One of the two biggest difficulties in this research was deciding how to build the ranking of decoys in a way that would lead to learning of the energy function. The second difficulty was a design of the fitness function that would result in an easy to search fitness landscape. These are discussed next.

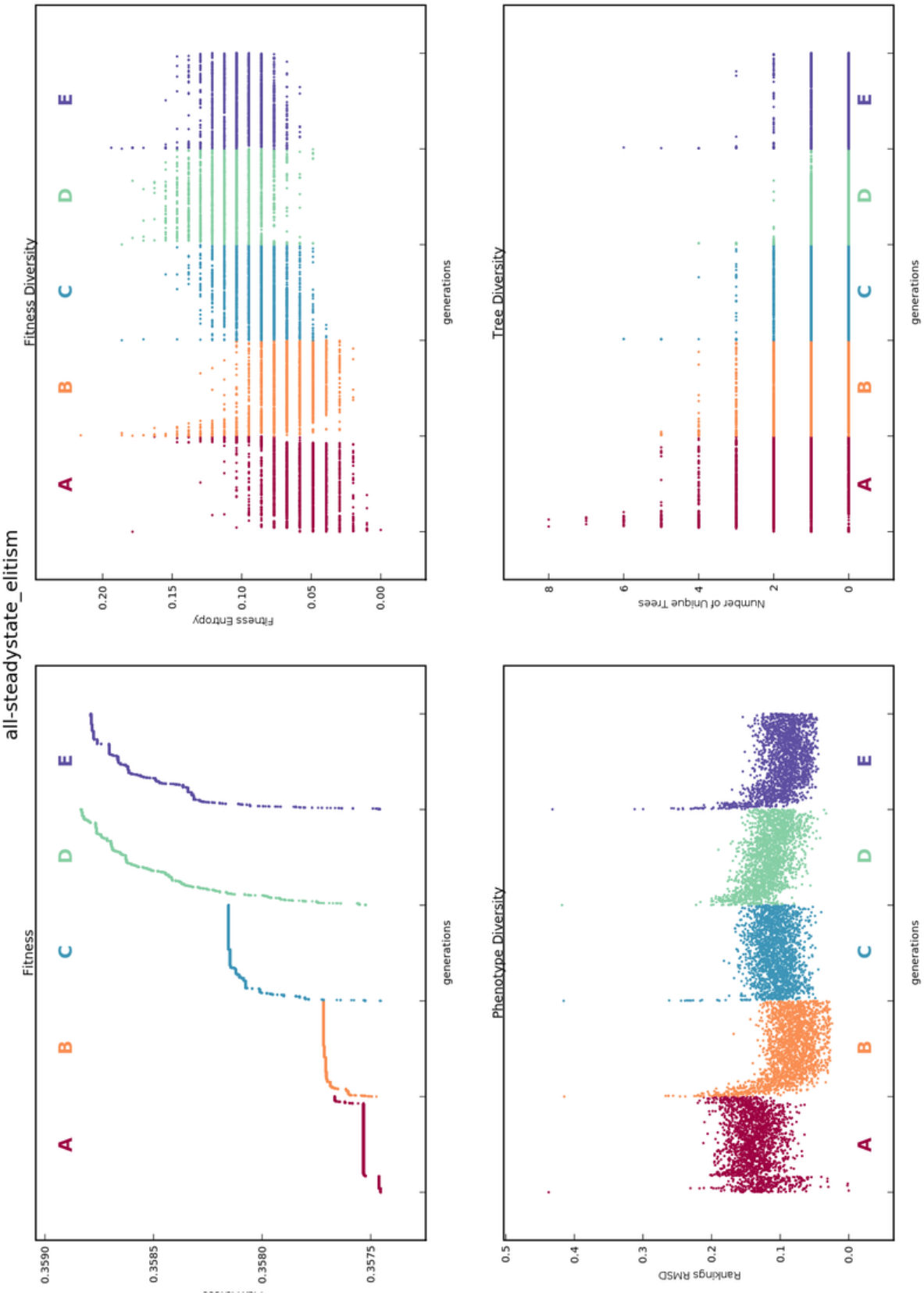

Figure 6.5: Maximum fitness throughout generations compared to population diversity on fitness, phenotype and GP tree levels in consensus experiments. Each plot shows side by side five different runs (A-E) for a selected GP configuration.

6.2.1 Decoy Ranking

Since in the structural optimisation process it is important to be able to measure the energy difference even for small structural changes, I decided to build the ranking with a picometer RMSD precision. Despite the high precision, I was not able to avoid ties in the ranking. The initial permutational approach, in which the tie was decided by the original I-TASSER energy has been shown less efficient in terms of the fitness distribution than averaging the ranks. It is not really surprising, as the I-TASSER energy itself was not highly correlated with RMSD.

The use of the ranking has been proven to be a good method to avoid the constraints of the direct comparison. This linear normalisation frees the evolution process from the need to reflect the scale of similarity or the direct differences between pairs of decoys.

In experiments with reduced sets of decoys I have shown that a wise selection of the sample representing the whole set improved the learning. But evolved functions were not proven useful when applied to the set of all decoys. Another way of pre-selecting the decoys, that would not depend on the original I-TASSER energy but rather purely on the decoys similarity might give better results. The similarity itself might be also measured differently. The RMSD as a non-weighted average of all C_α–C_α distances is sensitive to local errors and might return high values of distance even if global topology is correct.

The overall conclusion is that the influence of the parametrisation of the evolutionary process on the final result was not as important as the choice of the method to build the ranking. This leads to further work on how the decoys are ranked by the evaluation operator. Introduction of equal rank bins based on the distribution of the evolved energy values should possibly make the rankings comparison more accurate.

6.2.2 Fitness Function

For all fitness functions used in my experiments the average fitness saturated around the maximum after initial 50–200 generations. Although experiments with increased number of generations have shown a continuous improvement even after 2000 generations, the scale of this improvement was very small.

There are several factors that may cause this early saturation. The major one is a polynomial bound on the possible values of fitness functions. The maximum distance between two rankings of length n for Levenshtein distance is n (substitution of all characters). For Spearman and Kendall distance it is bounded by $O(n^2)$ being respectively $\frac{1}{2}n^2$ and $\frac{1}{2}n(n-1)$ for the reverse ranking. As a result, many different energy functions have the same value of the fitness function. This is why for the Levenshtein distance all individuals had the fitness very close to minimum and why for the Kendall distance the fitness variety amongst the population was so limited.

The analysis of population diversity for the best configurations has revealed that higher diversity on the fitness and phenotype level leads to a better evolutionary progress. It might be then useful to design a mechanism similar to the fitness sharing to promote the population diversity on both these levels.

Another important factor is the use of non-weighted average to calculate the total fitness. A very low fitness value for a single protein also significantly lowers the total fitness.

To overcome this a k outliers could be exclude from the total score. It might also be a good idea to use more complex averaging scheme, for example weighted by the native structures similarity to each other, so that more frequent but similar structures in the training set will have lower impact on the total fitness.

6.2.3 Similarity Consensus

It was reasonable to expect that more reliable measure of structural comparison could have make the problem landscape smoother, by eliminating some noise. The similarity consensus that is derived from a set of different similarity measures and combines strengths of the individual methods seemed to be a robust alternative. The results however, indicates that the use of similarity consensus made the landscape more difficult as the evolution did not progress at all for the set of all decoys due to low fitness diversity and the best function evolved for the d-100 set although having higher fitness than in case of the RMSD, turned out to be less correlated with the similarity than a naive combination of terms. Still, as the GP parameters were simply copied from the best ones for RMSD there might be room for improvement with more carefull parametrisation.

6.2.4 GP Challenge

The problem of the design of energy functions that would be useful in protein structure prediction is a new and truely difficult challenge for the GP. To encourage the reader to take on this challenge I made the input data publicly available. The following web page: `http://www.infobiotics.org/gpchallenge/` contains clearly annotated files with the precomputed energy potentials as well as the distances and ranks for all the decoys used in my experiments.

6.3 Conclusions and Future Work

In the last two chapters I have proposed the use of genetic programming to evolve novel forms of energy function for protein structure prediction. I have shown that this problem is very challenging, mainly due to the need of complex mapping between a GP tree and the total fitness, large amount of data to process and the requirement to generalise over different proteins, and that evolving a high quality energy functions is not an easy task. I have demonstrated a GP design generating functions that outperform the optimised weighted sum of terms used in previous works. I believe that this new GP-based approach might lead to significant improvements in the quality of protein structure prediction.

However, there is still plenty of scope for improvement and several questions remain unanswered. The definition of the fitness function needs improvement to better handle the problem of equal ranks and to relax the polynomial bound on the measure of distance between rankings (by enumerating the permutations for example). Moreover, additional objectives could be added to the fitness function (rank of the native structure for example) to evolve energy functions that are compact and easy to compute.

In the future work, the currently limited set of energy terms could be extended with data from protein feature predictors such as distance maps, contact order, contact restraints

or solvent accessibility [BSK+06, SBH+09, SBHK08]. Especially the last one might be meaningful, since the hydrophobic effect is considered to be one of the main forces in protein folding.

Finally, it is not yet understood how general the evolved functions are or whether the use of different decoys (either generated by different prediction methods or for different set of proteins) will increase their ability to select the near-native structures.

Chapter 7

Conclusions

The process of *de novo* structure prediction of proteins is based on the Anfinsen's thermodynamic hypothesis and is driven by a minimisation of the structure energy. The energy functions used in the state-of-the-art predictors to evaluate protein structures are based on several terms derived from the statistical analysis of known protein structures. These terms do not represent the real potential energy of a protein molecule, but rather measure how common are the structural features of the evaluated structure across the known protein structures.

Even though those terms do not represent explicitly the physical energy or the probability of a structure being native-like, the best CASP predictors simply adds them together in their energy functions. The only optimisation applied to these energy functions is a selection of weights for each of the terms. This optimisation is done using a sparse set of structures generated by randomisation of a known native structure. The weights are optimised to maximise a correlation between the energy and the distance to a native structure, measured using RMSD. It is known however, that the RMSD has several disadvantages: its sensitive to outliers and its absolute value depends on the structure length.

In this dissertation I addressed two main issues: the way the energy functions are designed and the imperfectness of the similarity measure this design is based on. The protein structure comparison server developed in my research group - ProCKSI, was extended with a new robust comparison method - Max-CMO (see Chapter 3). Then ProCKSI was modified to be able to apply the concept of similarity consensus (proven to be robust in case of protein classification) to measure distance between the native structure and a set of decoys. Six different comparison methods (including Max-CMO) were aggregated to obtain a consensus distance for each decoy and an interesting increase in methods disagreement for more native-like decoys was discovered (Chapter 4). In addition, a fully functional prototype of a cloud computing application for protein decoys comparison was presented.

Finally in Chapters 5 and 6 several variants of a genetic programming algorithm were proposed to design the energy functions as a general functional combination of energy terms and tested with both RMSD and similarity consensus used as a reference distance measure. For the RMSD the GP approach was found to be more accurate than a baseline experiments with random walk and Nelder-Mead simplex method applied to the linear

combination of terms showing that GP does indeed improve the energy functions what is in agreement with the hypothesis **H2**. This effect however, was not fully achieved with the similarity consensus, even though it had a desirable property of containing the sum of information from different similarity methods (as was shown in Chapter 4). Therefore the hypothesis **H1** that the use a similarity consensus will improve the prediction quality by eliminating the noise and smoothing out the landscape was rejected (possible reasons for this disagreement with experimental results are discussed below). Last but not least, by using decoys generated by the real predictors I have avoided the bias towards the native structure and made the reported research practically relevant.

The presented work has three main limitations: (1) evolved functions are complex and difficult to analyse, (2) operators used in GP algorithm do not contain more sophisticated programming language concepts such as conditions or loops and (3) the energy terms used in GP algorithm have been already designed with correlation to RMSD in mind. Incorporation of one of the bloat control techniques in the GP algorithm, even as simple as inclusion of the tree size component in the fitness function might lead to smaller trees and interpretable functions. An alternative approach could be a use of the Cartesian Genetic Programming (CGP) [Mil99], which uses indexed graphs to represent programs (instead of trees) and eliminates the problem of a bloat by allowing a selective genotype $\leftrightarrow$ phenotype mapping. This means that even though the genotype remains large throughout the evolution only a part of it may be used to compute the fitness. Moreover, in a self-modifying variant of the CGP (SM-CGP) the phenotype can evolve together with the genotype, so that not only a best solution but also a best representation for it would be found [HMB09]. By applying the SM-CGP in an iterative manner, each time for different protein, the algorithm could first evolve the graph structure capable of generalising the solution for all the cases and then evolve with it a best energy function for all the proteins.

One can also imagine that more effective energy functions might be evolvable only if more sophisticated operators are used, such as conditions and loops. Especially the conditions as they would allow to variant the energy computations with respect to different types of proteins. As the energy terms are often representing an aggregation of features measured across a range of residues, it might be also possible to break them down to these subcomponents and let the GP algorithm to use the loop operator to construct its own aggregations. This however, would require a careful selection of such terms as to many low-level components might introduce too much noise to the system and have a negative effect on evolvability.

The failure of the GP algorithm to evolve good energy functions that would reflect the order of decoys ranked by the similarity consensus might have two explanations. The sensitivity of protein comparison methods used to compute the consensus could be too low in case of decoys. It have to be remembered that these method were created to compare proteins and two different native protein structures are usually much more different from each other than the decoy and the native structure which share the same amino acid sequence. Stronger disagreement for decoys closer to the native seems to support this hypothesis. To verify it, the GP algorithm could be applied to a consensus based on the decoy-specific comparison measures, e.g. the ones used in the Protein Decoy Comparator cloud application presented in Chapter 4.

However, even an application of more specific measures might not change the evolvability of the energy function when the consensus is used. As the individual energy terms themselves

were in most cases already individually parametrised to increase the correlation to RMSD, they might just not be that useful with any other similarity measure. To overcome this problem the parameters of these energy terms should be designed from the start with the similarity consensus as a target distance measure. Or even better, the similarity consensus could be combined with GP running on low-level energy terms components and the right combination of them would be evolved automatically without the need to involve human experts in the process of their design.

Bibliography

[AHC+08] Antonina Andreeva, Dave Howorth, John-Marc Chandonia, Steven E. Brenner, Tim J. P. Hubbard, Cyrus Chothia, and Alexey G. Murzin. Data growth and its impact on the SCOP database: new developments. *Nucl. Acids Res.*, 36(suppl1):D419–425, January 2008. doi:10.1093/nar/gkm993.

[Anf73] Christian Anfinsen. Principles that Govern the Folding of Protein Chains. *Science*, 181(4096):223–30, July 1973. doi:10.1126/science.181.4096.223.

[AP93] Peter Angeline and Jordan Pollack. Evolutionary Module Acquisition. In *Proceedings of the Second Annual Conference on Evolutionary Programming*, pages 154–163, 1993.

[BB05] Roberto Battiti and Mauro Brunato. Reactive Search: Machine Learning For Memory-Based Heuristics. Technical Report DIT-05-058, Informatica e Telecomunicazioni, University of Trento, Italy, sep 2005.

[BBO+83] Bernard Brooks, Robert Bruccoleri, Barry Olafson, David States, S Swaminathan, and Martin Karplus. CHARMM: A program for macromolecular energy, minimization, and dynamics calculations. *Journal of Computational Chemistry*, 4(2):187–217, 1983. doi:10.1002/jcc.540040211.

[BCD97] Michael J. Bower, Fred E. Cohen, and Roland L. Dunbrack. Prediction of protein side-chain rotamers from a backbone-dependent rotamer library: a new homology modeling tool. *Journal of Molecular Biology*, 267(5):1268–1282, April 1997. doi:10.1006/jmbi.1997.0926.

[BCM+03] Philip Bradley, Dylan Chivian, Jens Meiler, Kira Misura, Carol Rohl, William Schief, William Wedemeyer, Ora Schueler-Furman, Paul Murphy, Jack Schonbrun, Charles Strauss, and David Baker. Rosetta predictions in CASP5: Successes, failures, and prospects for complete automation. *Proteins: Structure, Function, and Genetics*, 53(S6):457–68, 2003. doi:10.1002/prot.10552.

[BDM+08] Daniel Branton, David W Deamer, Andre Marziali, Hagan Bayley, Steven A Benner, Thomas Butler, Massimiliano Di Ventra, Slaven Garaj, Andrew Hibbs, Xiaohua Huang, Stevan B Jovanovich, Predrag S Krstic, Stuart Lindsay, Xinsheng Sean Ling, Carlos H Mastrangelo, Amit Meller, John S Oliver, Yuriy V Pershin, J Michael Ramsey, Robert Riehn, Gautam V Soni, Vincent Tabard-Cossa, Meni Wanunu, Matthew Wiggin, and Jeffery A Schloss. The potential and challenges of nanopore sequencing. *Nature Biotechnology*, 26(10):1146–1153, October 2008. doi:10.1038/nbt.1495.

[BDNBP+09] Moshe Ben-David, Orly Noivirt-Brik, Aviv Paz, Jaime Prilusky, Joel L. Sussman, and Yaakov Levy. Assessment of CASP8 structure predictions for template free targets. *Proteins: Structure, Function, and Bioinformatics*, 77(S9):50–65, 2009. doi:10.1002/prot.22591.

[Ber08] Helen M. Berman. The Protein Data Bank: a historical perspective. *Acta Crystallographica Section A*, 64(1):88–95, 2008. doi:10.1107/S0108767307035623.

[BGCZ07] Fabian Birzele, Jan E. Gewehr, Gergely Csaba, and Ralf Zimmer. Vorolign–fast structural alignment using Voronoi contacts. *Bioinformatics*, 23(2):e205–211, 2007. doi:10.1093/bioinformatics/btl294.

[BGK04] E.K. Burke, S. Gustafson, and G. Kendall. Diversity in genetic programming: an analysis of measures and correlation with fitness. *Evolutionary Computation, IEEE Transactions on*, 8(1):47–62, 2004. doi:10.1109/TEVC.2003.819263.

[BGKK02] E.K. Burke, S. Gustafson, G. Kendall, and N. Krasnogor. Advanced Population Diversity Measures in Genetic Programming. In J.J. Merelo Guervós, P. Adamidis, H.-G. Beyer, J.-L. Fernández-Villacañas, and H.-P. Schwefel, editors, *7th International Conference Parallel Problem Solving from Nature*, volume 2439 of *Springer Lecture Notes in Computer Science*, pages 341–350, Granada, Spain, September 2002. PPSN, Springer Berlin / Heidelberg. doi:10.1007/3-540-45712-7_33.

[BHBK07] Daniel Barthel, Jonathan D. Hirst, Jacek Blazewicz, and Natalio Krasnogor. ProCKSI: A Decision Support System for Protein (Structure) Comparison, Knowledge, Similarity and Information. *BMC Bioinformatics*, 8(1):416, 2007. doi:10.1186/1471-2105-8-416.

[BK05] E.K. Burke and G. Kendall, editors. *Search Methodologies: Introductory Tutorials in Optimization and Decision Support Techniques*. Springer, 2005. doi:10.1007/0-387-28356-0.

[BKB+07] James N. D. Battey, Jürgen Kopp, Lorenza Bordoli, Randy J. Read, Neil D. Clarke, and Torsten Schwede. Automated server predictions in CASP7. *Proteins: Structure, Function, and Bioinformatics*, 69(S8):68–82, 2007. doi:10.1002/prot.21761.

[BKR06] Michal Brylinski, Leszek Konieczny, and Irena Roterman. Hydrophobic collapse in (in silico) protein folding. *Computational Biology and Chemistry*, 30(4):255–267, August 2006. doi:10.1016/j.compbiolchem.2006.04.007.

[BMQ+05] Philip Bradley, Lars Malmström, Bin Qian, Jack Schonbrun, Dylan Chivian, David E. Kim, Jens Meiler, Kira M.S. Misura, and David Baker. Free modeling with Rosetta in CASP6. *Proteins: Structure, Function, and Bioinformatics*, 61(S7):128–34, 2005. doi:10.1002/prot.20729.

[Bou03] Philip E. Bourne. *Structural Bioinformatics*, chapter CASP and CAFASP experiments and their findings, pages 499–505. Wiley-Liss, 2003. doi:10.1002/0471721204.ch24.

[BR03] Christian Blum and Andrea Roli. Metaheuristics in combinatorial optimization: Overview and conceptual comparison. *ACM Computing Surveys*, 35(3):268–308, Sep 2003. doi:10.1145/937503.937505.

[BSH+07] Jaume Bacardit, Michael Stout, Jonathan D. Hirst, Kumara Sastry, Xavier Llorà, and Natalio Krasnogor. Automated alphabet reduction method with evolutionary algorithms for protein structure prediction. In *Proceedings of the 9th annual conference on Genetic and evolutionary computation*, pages 346–353, London, England, 2007. doi:10.1145/1276958.1277033.

[BSH+09] Jaume Bacardit, Michael Stout, Jonathan Hirst, Alfonso Valencia, Robert Smith, and Natalio Krasnogor. Automated Alphabet Reduction for Protein Datasets. *BMC Bioinformatics*, 10(1):6, 2009. doi:10.1186/1471-2105-10-6.

[BSK+06] J. Bacardit, M. Stout, N. Krasnogor, J.D. Hirst, and J. Blazewicz. Coordination Number Prediction using Learning Classifier Systems: Performance and Interpretability. In *Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation (GECCO '06)*, pages 247–254. ACM Press, July 2006. doi:10.1145/1143997.1144041.

[BSR+02] Richard Bonneau, Charlie E. M. Strauss, Carol A. Rohl, Dylan Chivian, Phillip Bradley, Lars Malmstrom, Tim Robertson, and David Baker. De Novo Prediction of Three-dimensional Structures for Major Protein Families. *Journal of Molecular Biology*, 322(1):65–78, September 2002. doi:10.1016/S0022-2836(02)00698-8.

[BTR+01] Richard Bonneau, Jerry Tsai, Ingo Ruczinski, Dylan Chivian, Carol Rohl, Charlie Strauss, and David Baker. Rosetta in CASP4: Progress in ab initio protein structure prediction. *Proteins: Structure, Function, and Genetics*, 45(S5):119–26, 2001. doi:10.1002/prot.1170.

[CCD+05] David Case, III Thomas Cheatham, Tom Darden, Holger Gohlke, Ray Luo, Jr Kenneth Merz, Alexey Onufriev, Carlos Simmerling, Bing Wang, and Robert Woods. The Amber biomolecular simulation programs. *Journal of Computational Chemistry*, 26(16):1668–88, 2005. doi:10.1002/jcc.20290.

[CCS06] Orhan Camoglu, Tolga Can, and Ambuj K. Singh. Integrating multi-attribute similarity networks for robust representation of the protein space. *Bioinformatics*, 22(13):1585–1592, 2006. doi:10.1093/bioinformatics/btl130.

[CGP+98] Pierluigi Crescenzi, Deborah Goldman, Christos H. Papadimitriou, Antonio Piccolboni, and Mihalis Yannakakis. On the Complexity of Protein Folding. *Journal of Computational Biology*, 5(3):423–466, 1998. Available from: http://citeseer.ist.psu.edu/31865.html [cited 2007-03-13].

[Chi06] David Chivian. CASP7 server ranking for FM category (GDT MM) [online]. 2006 [cited 2007-08-06]. Available from: http://robetta.bakerlab.org/CASP7_eval/CASP7.FR_A-NF.Best-GDT_MM.html.

[CHK+02] Robert Carr, William Hart, Natalio Krasnogor, Jonathan Hirst, Edmund K. Burke, and James Smith. Alignment Of Protein Structures With A Memetic Evolutionary Algorithm. In W. B. Langdon, E. Cantu-Paz, K. Mathias, R. Roy, D. Davis, R. Poli, K. Balakrishnan, V. Honavar, G. Rudolph, J. Wegener, L. Bull, M. A. Potter, A. C. Schultz, J. F. Miller, E. Burke, and N. Jonoska, editors, *GECCO '02: Proceedings of the Genetic and Evolutionary Computation Conference*, pages 1027–1034, New York, USA, July 2002. GECCO, Morgan Kaufmann Publishers.

[CK02] L.P. Chew and K. Kedem. Finding Consensus Shape for a Protein Family. In *18th ACM Symp. on Computational Geometry*, Barcelona, Spain, 2002. doi:10.1145/513400.513408.

[CKM+05] Dylan Chivian, David Kim, Lars Malmström, Jack Schonbrun, Carol Rohl, and David Baker. Prediction of CASP6 structures using automated robetta protocols. *Proteins: Structure, Function, and Bioinformatics*, 61(S7):157–66, 2005. doi:10.1002/prot.20733.

[CL02] Alberto Caprara and Giuseppe Lancia. Structural alignment of large–size proteins via lagrangian relaxation. In *RECOMB '02: Proceedings of the sixth annual international conference on Computational biology*, pages 100–108. ACM Press, January 2002. doi:10.1145/565196.565209.

[CNN06] Vincenzo Cutello, Giuseppe Narzisi, and Giuseppe Nicosia. A multi-objective evolutionary approach to the protein structure prediction problem.

Journal of The Royal Society Interface, 3(6):139–151, 2006. Applies MOO for CHARMM27 energy (computed with TINKER). doi:10.1098/rsif.2005.0083.

[CSD03] Adrian A. Canutescu, Andrew A. Shelenkov, and Jr. Dunbrack, Roland L. A graph-theory algorithm for rapid protein side-chain prediction. *Protein Sci*, 12(9):2001–2014, September 2003. Available from: http://dunbrack.fccc.edu/SCWRL3.php [cited 2007-08-09], doi:10.1110/ps.03154503.

[CSD04] Evangelos A. Coutsias, Chaok Seok, and Ken A. Dill. Using quaternions to calculate RMSD. *Journal of Computational Chemistry*, 25(15):1849–1857, 2004. doi:10.1002/jcc.20110.

[CZ05] Huiling Chen and Huan-Xiang Zhou. Prediction of solvent accessibility and sites of deleterious mutations from protein sequence. *Nucleic Acids Research*, 33(10):3193–3199, June 2005. doi:10.1093/nar/gki633.

[CZF+01] Susana Cristobal, Adam Zemla, Daniel Fischer, Leszek Rychlewski, and Arne Elofsson. A study of quality measures for protein threading models. *BMC Bioinformatics*, 2(1):5, 2001. doi:10.1186/1471-2105-2-5.

[DB06] Dusan P. Djurdjevic and Mark J. Biggs. Ab initio protein fold prediction using evolutionary algorithms: Influence of design and control parameters on performance. *Journal of Computational Chemistry*, 27(11):1177–1195, 2006. doi:10.1002/jcc.20440.

[DC97] Ken A. Dill and Hue Sun Chan. From Levinthal to pathways to funnels. *Nat Struct Mol Biol*, 4(1):10–19, January 1997. doi:10.1038/nsb0197-10.

[Dil90] Ken A. Dill. Dominant forces in protein folding. *Biochemistry*, 29(31):7133–7155, August 1990. doi:10.1021/bi00483a001.

[DKNS01] Cynthia Dwork, Ravi Kumar, Moni 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. doi:10.1145/371920.372165.

[DLP03] Richard O. Day, Gary B. Lamont, and Ruth Pachter. Protein Structure Prediction by Applying an Evolutionary Algorithm. In *Proceedings of the 17th International Symposium on Parallel and Distributed Processing*, page 155.1. IEEE Computer Society, 2003. doi:10.1109/IPDPS.2003.1213291.

[DQR+07] Rhiju Das, Bin Qian, Srivatsan Raman, Robert Vernon, James Thompson, Philip Bradley, Sagar Khare, Michael D. Tyka, Divya Bhat, Dylan Chivian, David E. Kim, William H. Sheffler, Lars Malmström, Andrew M. Wollacott, Chu Wang, Ingemar Andre, and David Baker. Structure prediction for CASP7 targets using extensive all-atom refinement with Rosetta@home. *Proteins: Structure, Function, and Bioinformatics*, 69(S8):118–128, 2007. doi:10.1002/prot.21636.

[DSS+00] Aaron R. Dinner, Andrej Sali, Lorna J. Smith, Christopher M. Dobson, and Martin Karplus. Understanding protein folding via free-energy surfaces from theory and experiment. *Trends in Biochemical Sciences*, 25(7):331–339, July 2000. doi:10.1016/S0968-0004(00)01610-8.

[ED05] David J. Earl and Michael W. Deem. Parallel tempering: Theory, applications, and new perspectives. *Physical Chemistry Chemical Physics*, 7(23):3910–3916, 2005. doi:10.1039/b509983h.

[FB99] Daniel Fischer and Christian Barret. CAFASP-1: Critical assessment of fully automated structure prediction methods. *Proteins: Structure, Function, and Genetics*, 37(S3):209–217, January 1999.

[GD91] David E. Goldberg and Kalyanmoy Deb. A Comparative Analysis of Selection Schemes Used in Genetic Algorithms. In Gregory J. E. Rawlins, editor, *Foundations of Genetic Algorithms*, pages 69–93. Morgan Kaufmann, 1991.

[GEFR03] Krzysztof Ginalski, Arne Elofsson, Daniel Fischer, and Leszek Rychlewski. 3D-Jury: a simple approach to improve protein structure predictions. *Bioinformatics*, 19(8):1015–1018, May 2003. doi:10.1093/bioinformatics/btg124.

[GGGR05] Krzysztof Ginalski, Nick V. Grishin, Adam Godzik, and Leszek Rychlewski. Practical lessons from protein structure prediction. *Nucl. Acids Res.*, 33(6):1874–1891, April 2005. doi:10.1093/nar/gki327.

[GJ79] Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., January 1979.

[GK03] Fred W. Glover and Gary A. Kochenberger, editors. *Handbook of Metaheuristics*. Springer, 2003. doi:10.1007/b101874.

[GK08] Dominik Gront and Andrzej Kolinski. Utility library for structural bioinformatics. *Bioinformatics*, 24(4):584–585, February 2008. doi:10.1093/bioinformatics/btm627.

[GLA+07] Lesley H. Greene, Tony E. Lewis, Sarah Addou, Alison Cuff, Tim Dallman, Mark Dibley, Oliver Redfern, Frances Pearl, Rekha Nambudiry, Adam Reid, Ian Sillitoe, Corin Yeats, Janet M. Thornton, and Christine A. Orengo. The CATH domain structure database: new protocols and classification levels give a more comprehensive resource for exploring evolution. *Nucl. Acids Res.*, 35(suppl1):D291–297, 2007. doi:10.1093/nar/gkl959.

[Glo89] Fred Glover. Tabu Serach – Part I. *ORSA Journal on Computing*, 1(3):190–206, 1989.

[Glo90] Fred Glover. Tabu Serach – Part II. *ORSA Journal on Computing*, 2(1):4–32, 1990.

[GP05] Michel Gendreau and Jean-Yves Potvin. Metaheuristics in Combinatorial Optimization. *Annals of Operations Research*, 140(1):189–213, November 2005. doi:10.1007/s10479-005-3971-7.

[GP06] Christian Gagné and Marc Parizeau. Genericity in Evolutionary Computation Software Tools: Principles and Case-study. *International Journal on Artificial Intelligence Tools*, 15(2):173–194, 2006. doi:10.1142/S021821300600262X.

[GPI99] Deborah Goldman, Christos H. Papadimitriou, and Sorin Istrail. Algorithmic Aspects of Protein Structure Similarity. In *FOCS '99: Proceedings of the 40th Annual Symposium on Foundations of Computer Science*, pages 512–512. IEEE Computer Society, January 1999. doi:10.1109/SFFCS.1999.814624.

[Gra04] J. Gramm. A polynomial-time algorithm for the matching of crossing contact-map patterns. *Computational Biology and Bioinformatics, IEEE/ACM Transactions on*, 1(4):171–180, January 2004. doi:10.1109/TCBB.2004.35.

[GSK93] Adam Godzik, Jeffrey Skolnick, and Andrzej Kolinski. Regularities in interaction patterns of globular proteins. *Protein Engineering Design and Selection*, 6(8):801–810, November 1993. Max-CMO measure definition. doi:10.1093/protein/6.8.801.

[HAB+99] TJ Hubbard, B Ailey, SE Brenner, AG Murzin, and C Chothia. SCOP: a Structural Classification of Proteins database. *Nucl. Acids Res.*, 27(1):254–256, January 1999. doi:10.1093/nar/27.1.254.

[Hay98] Brian Hayes. Prototeins. *American Scientist*, 86(3):216–, May–June 1998. doi:10.1511/1998.3.216.

[HGS+05] Jianjun Hu, Erik Goodman, Kisung Seo, Zhun Fan, and Rondal Rosenberg. The Hierarchical Fair Competition (HFC) Framework for Sustainable Evolutionary Algorithms. *Evolutionary Computation*, 13(2):241–277, 2005. doi:10.1162/1063656054088530.

[HH09] Hitomi Hasegawa and Liisa Holm. Advances and pitfalls of protein structural alignment. *Current Opinion in Structural Biology*, 19(3):341–348, June 2009. doi:10.1016/j.sbi.2009.04.003.

[HMB09] Simon Harding, Julian F. Miller, and Wolfgang Banzhaf. Evolution, development and learning using self-modifying cartesian genetic programming. In *Proceedings of the 11th Annual conference on Genetic and evolutionary computation*, pages 699–706, Montreal, Canada, jul 2009. doi:10.1145/1569901.1569998.

[HNLS05] Ling-Hong Hung, Shing-Chung Ngan, Tianyun Liu, and Ram Samudrala. PROTINFO: new algorithms for enhanced protein structure predictions. *Nucleic Acids Res*, 33(suppl2):W77–W80, January 2005. doi:10.1093/nar/gki403.

[HP00] Liisa Holm and Jong Park. DaliLite workbench for protein structure comparison. *Bioinformatics*, 16(6):566–567, June 2000. doi:10.1093/bioinformatics/16.6.566.

[HTdW95] A. Hertz, E. Taillard, and D. de Werra. A Tutorial on Tabu Search. In *Proc. of Giornate di Lavoro AIRO'95 (Enterprise Systems: Management of Technologicaland Organizational Changes)*, pages 13–24, Italy, 1995. Available from: http://citeseer.ist.psu.edu/hertz92tutorial.html.

[Jac98] Sophie E. Jackson. How do small single-domain proteins fold? *Folding and Design*, 3(4):R81–R91, August 1998. doi:doi:10.1016/S1359-0278(98)00033-9.

[JBC+05] D. T. Jones, K. Bryson, A. Coleman, L. J. McGuffin, M. I. Sadowski, J. S. Sodhi, and J. J. Ward. Prediction of novel and analogous folds using fragment assembly and fold recognition. *Proteins: Structure, Function, and Bioinformatics*, 61(S7):143–151, January 2005. doi:10.1002/prot.20731.

[JOP+] Eric Jones, Travis Oliphant, Pearu Peterson, et al. SciPy: Open source scientific tools for Python, 2001–. Available from: http://www.scipy.org/ [cited 2008-07-25].

[Kab78] 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. doi:10.1107/S0567739478001680.

[KB03] Michal A. Kurowski and Janusz M. Bujnicki. GeneSilico protein structure prediction meta-server. *Nucl. Acids Res.*, 31(13):3305–3307, July 2003. doi:10.1093/nar/gkg557.

[KBHB02] N. Krasnogor, B. Blackburn, J.D. Hirst, and E.K. Burke. Multimeme Algorithms for Protein Structure Prediction. In Juan J. Merelo, Panagiotis Adamidis, and Hans-Georg Beyer, editors, *Parallel Problem Solving from Nature - PPSN VII*, volume 2439 of *Springer Lecture Notes in Computer Science*, pages 769–778. Springer, 2002. doi:10.1007/3-540-45712-7_74.

[KHSP99] N. Krasnogor, W.E. Hart, J. Smith, and D. Pelta. Protein Structure Prediction With Evolutionary Algorithms. In Banzhaf, Daida, Eiben, Garzon, Honovar, Jakiela, and Smith, editors, *International Genetic and Evolutionary Computation Conference (GECCO99)*, pages 1569–1601. Morgan Kaufmann, 1999.

[KK03] Kevin Karplus and Rachel Karchin. Combining local-structure, fold-recognition, and new fold methods for protein structure prediction. *Proteins: Structure, Function, and Genetics*, 53(S6):491–496, January 2003. doi:10.1002/prot.10540.

[KK05] Kevin Karplus and Sol Katzman. SAM-T04: What is new in protein-structure prediction for CASP6. *Proteins: Structure, Function, and Bioinformatics*, 61(S7):135–142, January 2005. doi:10.1002/prot.20730.

[KKL05] Rachel Kolodny, Patrice Koehl, and Michael Levitt. Comprehensive Evaluation of Protein Structure Alignment Methods: Scoring by Geometric Measures. *Journal of Molecular Biology*, 346(4):1173–1188, March 2005. doi:10.1016/j.jmb.2004.12.032.

[KLZ+03] N. Krasnogor, G. Lancia, A. Zemla, W.E. Hart, R.D. Carr, J.D. Hirst, and E.K. Burke. A Comparison of Computational Methods for the Maximum Contact Map Overlap of Protein Pairs. submitted, 2003.

[Kni66] William 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.

[Kol04] Andrzej Kolinski. Protein modeling and structure prediction with a reduced representation. *Acta Biochimica Polonica*, 51(2):349–371, 2004. Available from: http://www.actabp.pl/html/2_2004/349.html [cited 2007-08-06].

[Koz] John R. Koza. 36 Human-Competitive Results Produced by Genetic Programming [online, cited 2010-02-22]. Available from: http://www.genetic-programming.com/humancompetitive.html.

[Koz92] John R. Koza. *Genetic programming: on the programming of computers by means of natural selection and genetics*. MIT Press, 1992.

[Koz94] John R. Koza. Scalable learning in genetic programming using automatic function definition. In Kenneth E. Jr. Kinnear, editor, *Advances in Genetic Programming*, chapter 5, pages 99–117. MIT Press, 1994.

[KP04] N. Krasnogor and D. A. Pelta. Measuring the similarity of protein structures by means of the universal similarity metric. *Bioinformatics*, 20(7):1015–1021, May 2004. doi:10.1093/bioinformatics/bth031.

[Kra04] N. Krasnogor. Self-Generating metaheuristics in bioinformatics: the protein structure comparison case. *Genetic Programming and Evolvable Machines*, 5(2):181–201, 2004.

[KS98] Andrzej Kolinski and Jeffrey Skolnick. Assembly of protein structure from sparse experimental data: An efficient Monte Carlo model. *Proteins: Structure, Function, and Genetics*, 32(4):475–494, January 1998. http://dx.doi.org/10.1002/(SICI)1097-0134(19980901)32:4¡475::AID-PROT6¿3.0.CO;2-F doi:10.1002/(SICI)1097-0134(19980901)32:4<475::AID-PROT6>3.0.CO;2-F.

[KSAG03] Alexander Kraskov, Harald Stogbauer, Ralph G. Andrzejak, and Peter Grassberger. Hierarchical Clustering Based on Mutual Information, 2003. Available from: http://arxiv.org/abs/q-bio/0311039 [cited 2010-02-01].

[KVFM05] Andriy Kryshtafovych, Ceslovas Venclovas, Krzysztof Fidelis, and John Moult. Progress over the first decade of CASP experiments. *Proteins: Structure, Function, and Bioinformatics*, 61(S7):225–36, 2005. doi:10.1002/prot.20740.

[LCWI01] Giuseppe Lancia, Robert Carr, Brian Walenz, and Sorin Istrail. 101 optimal PDB structure alignments: a branch-and-cut algorithm for the maximum contact map overlap problem. In *RECOMB '01: Proceedings of the fifth annual international conference on Computational biology*, pages 193–202. ACM Press, January 2001. doi:10.1145/369133.369199.

[Lev66] V. I. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Dokl.*, 10(8):707–710, February 1966.

[Lev69] Cyrus Levinthal. How to Fold Graciously. In J. T. P. DeBrunner and E. Munck, editors, *Mossbauer Spectroscopy in Biological Systems: Proceedings of a meeting held at Allerton House*, pages 22–24, Monticello, Illinois, 1969. University of Illinois Press. Available from: http://www-wales.ch.cam.ac.uk/~mark/levinthal/levinthal.html [cited 2007-03-13].

[LK99] Themis Lazaridis and Martin Karplus. Effective energy function for proteins in solution. *Proteins: Structure, Function, and Genetics*, 35(2):133–152, 1999. http://dx.doi.org/10.1002/(SICI)1097-0134(19990501)35:2¡133::AID-PROT1¿3.0.CO;2-N doi:10.1002/(SICI)1097-0134(19990501)35:2<133::AID-PROT1>3.0.CO;2-N.

[LKR03] Jacek Leluk, Leszek Konieczny, and Irena Roterman. Search for structural similarity in proteins. *Bioinformatics*, 19(1):117–124, Jan 2003. doi:10.1093/bioinformatics/19.1.117.

[LL06] Shuai Cheng Li and Ming Li. On the Complexity of the Crossing Contact Map Pattern Matching Problem. In *Algorithms in Bioinformatics*, volume 4175 of *Lecture Notes in Computer Science*, pages 231–241. Springer, January 2006. doi:10.1007/11851561_22.

[LMC99] Manuel Laguna, Rafael Marti, and Vicente Campos. Intensification and diversification with elite tabu search solutions for the linear ordering problem. *Computers & Operations Research*, 26(12):1217–1230, October 1999. doi:10.1016/S0305-0548(98)00104-X.

[LOC+04] A. Liwo, S. Oldziej, C. Czaplewski, U. Kozlowska, and H.A. Scheraga. Parametrization of Backbone-Electrostatic and Multibody Contributions to the UNRES Force Field for Protein-Structure Prediction from Ab Initio Energy Surfaces of Model Systems. *J. Phys. Chem. B*, 108(27):9421–9438, 2004. doi:10.1021/jp030844f.

[LP00] R. B. Lyngsø and C. N. S. Pedersen. Protein folding in the 2D HP model. In *Proceedings of the 1st journees ouvertes: biologie, informatique et mathematiques (JOBIM 00)*, 2000.

[LP01] Sean Luke and Liviu Panait. A survey and comparison of tree generation algorithms. In Lee Spector, Erik D. Goodman, Annie Wu, W. B. Langdon, Hans-Michael Voigt, Mitsuo Gen, Sandip Sen, Marco Dorigo, Shahram Pezeshk, Max H. Garzon, and Edmund Burke, editors, *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001)*, pages 81–88, San Francisco, California, USA, July 2001. Morgan Kaufman. Available from: http://en.scientificcommons.org/453130 [cited 2008-04-08].

[Luk09] Sean Luke. *Essentials of Metaheuristics*. self-published, 2009. Available from: http://cs.gmu.edu/~sean/book/metaheuristics/ [cited 2010-06-03].

[Mac04] Jr Alexander MacKerell. Empirical force fields for biological macromolecules: Overview and issues. *Journal of Computational Chemistry*, 25(13):1584–1604, 2004. doi:10.1002/jcc.20082.

[McK99] K. I. M. McKinnon. Convergence of the Nelder-Mead simplex method to a nonstationary point. *SIAM Journal on Optimization*, 9:148–158, 1999.

[MF05] John Moult and Krzysztof Fidelis. Critical assessment of methods of protein structure prediction (CASP) - Round 6. *Proteins: Structure, Function, and Bioinformatics*, 61(S7):3–7, January 2005. doi:10.1002/prot.20716.

[Mil99] Julian F. Miller. An empirical study of the efficiency of learning boolean functions using a Cartesian Genetic Programming approach. In Wolfgang Banzhaf, Jason Daida, Agoston E. Eiben, Max H. Garzon, Vasant Honavar, Mark Jakiela, and Robert E. Smith, editors, *Proceedings of the Genetic and Evolutionary Computation Conference*, volume 2, pages 1135–1142, Orlando, Florida, USA, 13-17 July 1999. Available from: http://citeseer.ist.psu.edu/153431.html [cited 2010-02-22].

[Mon95] David J. Montana. Strongly Typed Genetic Programming. *Evolutionary Computation*, 3(2):199–230, 1995. doi:10.1162/evco.1995.3.2.199.

[New02] Alantha Newman. A new algorithm for protein folding in the HP model. In *SODA 2002: Proceedings of the thirteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 876–884, Philadelphia, USA, 2002. Society for Industrial and Applied Mathematics.

[NM64] J.A. Nelder and R. Mead. A simplex method for function minimization. *The Computer Journal*, 7:308–313, 1964.

[OSO02] Angel R. Ortiz, Charlie E.M. Strauss, and Osvaldo Olmea. MAMMOTH (Matching molecular models obtained from theory): An automated method for model comparison. *Protein Science*, 11(11):2606–2621, 2002. MAMMOTH is using URMS to build a similarity matrix. Then uses Needleman&Wunsch to align the backbones (with gaps penalties) and MaxSub heuristic to find a subset of residues in distance ¡= 4A. doi:10.1110/ps.0215902.

[PBB+03] F. M. G. Pearl, C. F. Bennett, J. E. Bray, A. P. Harrison, N. Martin, A. Shepherd, I. Sillitoe, J. Thornton, and C. A. Orengo. The CATH database: an extended protein family resource for structural and functional genomics. *Nucl. Acids Res.*, 31(1):452–455, 2003. doi:10.1093/nar/gkg062.

[PBC+03] Vijay S. Pande, Ian Baker, Jarrod Chapman, Sidney P. Elmer, Siraj Khaliq, Stefan M. Larson, Young Min Rhee, Michael R. Shirts, Christopher D. Snow, Eric J. Sorin, and Bojan Zagrovic. Atomistic protein folding simulations on the submillisecond time scale using worldwide distributed computing. *Biopolymers*, 68(1):91–109, 2003. doi:10.1002/bip.10219.

[PBW+05] James C. Phillips, Rosemary Braun, Wei Wang, James Gumbart, Emad Tajkhorshid, Elizabeth Villa, Christophe Chipot, Robert D. Skeel, Laxmikant Kale, and Klaus Schulten. Scalable molecular dynamics with NAMD. *Journal of Computational Chemistry*, 26:1781–1802, 2005. Available from: http://www.ks.uiuc.edu/Research/namd/ [cited 2007-03-14].

[PC03] Jay W Ponder and David A Case. Force fields for protein simulations. *Adv Protein Chem*, 66:27–85, 2003.

[PKBC+05] D.A. Pelta, N. Krasnogor, C. Bousono-Calzon, J.L. Verdegay, J. Hirst, and E.K. Burke. A fuzzy sets based generalization of contact maps for the overlap of protein structures. *Journal of Fuzzy Sets and Systems*, 152(2):103–123, 2005. doi:10.1016/j.fss.2004.10.017.

[PLM08] Riccardo Poli, William B. Langdon, and Nicholas Freitag McPhee. *A field guide to genetic programming*. Published via http://lulu.com., 2008. with contributions by J.R. Koza. Available from: http://www.gp-field-guide.org.uk [cited 2010-06-09].

[RSMB04] Carol A. Rohl, Charlie E. M. Strauss, Kira M. S. Misura, and David Baker. Protein Structure Prediction Using Rosetta. In Ludwig Brand and Michael L. Johnson, editors, *Numerical Computer Methods, Part D*, volume Volume 383 of *Methods in Enzymology*, pages 66–93. Academic Press, January 2004. doi:10.1016/S0076-6879(04)83004-0.

[SB98] IN Shindyalov and PE Bourne. Protein structure alignment by incremental combinatorial extension (CE) of the optimal path. *Protein Eng.*, 11(9):739–747, 1998. doi:10.1093/protein/11.9.739.

[SBH+09] Michael Stout, Jaume Bacardit, Jonathan Hirst, Robert Smith, and Natalio Krasnogor. Prediction of topological contacts in proteins using learning classifier systems. *Soft Computing - A Fusion of Foundations, Methodologies and Applications*, 13(3):245–258, Feb 2009. doi:10.1007/s00500-008-0318-8.

[SBHK08] Michael Stout, Jaume Bacardit, Jonathan D. Hirst, and Natalio Krasnogor. Prediction of recursive convex hull class assignments for protein residues. *Bioinformatics*, 24(7):916–923, 2008. doi:10.1093/bioinformatics/btn050.

[SBS05] Dawn M. Strickland, Earl Barnes, and Joel S. Sokol. Optimal Protein Structure Alignment Using Maximum Cliques. *OPERATIONS RESEARCH*, 53(3):389–402, May 2005. doi:10.1287/opre.1040.0189.

[SF01] N. Siew and D. Fischer. Convergent evolution of protein structure prediction and computer chess tournaments: CASP, Kasparov, and CAFASP. *IBM Syst. J.*, 40(2):410–425, January 2001. doi:10.1147/sj.402.0410.

[SKD+02] R. Steuer, J. Kurths, C. O. Daub, J. Weise, and J. Selbig. The mutual information: Detecting and evaluating dependencies between variables. *Bioinformatics*, 18(suppl2):S231–240, October 2002.

[SLL08] R. Santana, P. Larranaga, and J.A. Lozano. Protein Folding in Simplified Models With Estimation of Distribution Algorithms. *Evolutionary Computation, IEEE Transactions on*, 12(4):418–438, 2008. doi:10.1109/TEVC.2007.906095.

[SP00] Michael Shirts and Vijay Pande. COMPUTING: Screen Savers of the World Unite! *Science*, 290(5498):1903–4, December 2000. doi:10.1126/science.290.5498.1903.

[SRK+99] Kim T. Simons, Ingo Ruczinski, Charles Kooperberg, Brian A. Fox, Chris Bystroff, and David 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. http://dx.doi.org/10.1002/(SICI)1097-0134(19990101)34:1¡82::AID-PROT7¿3.0.CO;2-A doi:10.1002/(SICI)1097-0134(19990101)34:1<82::AID-PROT7>3.0.CO;2-A.

[SS05] Sébastien Sorlin and Christine Solnon. Reactive tabu search for measuring graph similarity. In Luc Brun and Mario Vento, editors, *5th IAPR-TC-15 workshop on Graph-based Representations in Pattern Recognition*, pages 172–182. Springer-Verlag, apr 2005. Available from: http://liris.cnrs.fr/publis/?id=1525 [cited 2007-03-13].

[Sys90] Gilbert Syswerda. A Study of Reproduction in Generational and Steady State Genetic Algorithms. In Gregory J. E. Rawlins, editor, *Foundations of Genetic Algorithms*, pages 94–101. Morgan Kaufmann, 1990.

[SZA+03] Jeffrey Skolnick, Yang Zhang, Adrian Arakaki, Andrzej Kolinski, Michal Boniecki, András Szilágyi, and Daisuke Kihara. TOUCHSTONE: A unified approach to protein structure prediction. *Proteins: Structure, Function, and Genetics*, 53(S6):469–79, 2003. doi:10.1002/prot.10551.

[Ung04] Ron Unger. *Applications of Evolutionary Computation in Chemistry*, volume 110 of *Structure & Bonding*, chapter The Genetic Algorithm Approach to Protein Structure Prediction, pages 153–175. Springer, Berlin, 2004. doi:10.1007/b13936.

[URLa] CMOS contact maps data sets [online, cited 2007-03-14]. Available from: http://eudoxus.scs.uiuc.edu/cgi-bin/cmosinstr.cgi.

[URLb] TINKER [online, cited 2007-03-14]. Available from: http://dasher.wustl.edu/tinker/.

[URL10a] Folding@home client statistics [online]. 2010 [cited 2008-10-10]. Available from: http://fah-web.stanford.edu/cgi-bin/main.py?qtype=osstats.

[URL10b] Protein Data Bank statistics [online]. 2010 [cited 2010-06-09]. Available from: http://www.pdb.org/pdb/statistics/holdings.do.

[VDSLH+05] David Van Der Spoel, Erik Lindahl, Berk Hess, Gerrit Groenhof, Alan E. Mark, and Herman J. C. Berendsen. GROMACS: Fast, flexible, and free. *Journal of Computational Chemistry*, 26(16):1701–18, 2005. doi:10.1002/jcc.20291.

[WA96] Gunter P. Wagner and Lee Altenberg. Complex Adaptations and the Evolution of Evolvability. *Evolution*, 50(3):967–97, Jun 1996. Available from: http://dynamics.org/Altenberg/PAPERS/CAEE/ [cited 2007-06-13].

[WFB03] Stefan Wallin, Jochen Farwer, and Ugo Bastolla. Testing similarity measures with continuous and discrete protein models. *Proteins: Structure, Function, and Genetics*, 50(1):144–157, 2003. doi:10.1002/prot.10271.

[WMBB00] S. J. Wheelan, A. Marchler-Bauer, and S. H. Bryant. Domain size distributions can predict domain boundaries. *Bioinformatics*, 16(7):613–618, 2000. doi:10.1093/bioinformatics/16.7.613.

[WSZ07] Sitao Wu, Jeffrey Skolnick, and Yang Zhang. Ab initio modeling of small proteins by iterative TASSER simulations. *BMC Biol*, 5(1):17, May 2007. doi:10.1186/1741-7007-5-17.

[XS06] Wei Xie and Nikolaos V. Sahinidis. A Branch-and-Reduce Algorithm for the Contact Map Overlap Problem. In Alberto Apostolico, Concettina Guerra, Sorin Istrail, Pavel A. Pevzner, and Michael S. Waterman, editors, *RECOMB*, volume 3909 of *Lecture Notes in Computer Science*, pages 516–529. Springer, 2006. doi:10.1007/11732990_43.

[XS07] Wei Xie and Nikolaos V. Sahinidis. A Reduction-Based Exact Algorithm for the Contact Map Overlap Problem. *Journal of Computational Biology*, 14(5):637–654, 2007. doi:10.1089/cmb.2007.R007.

[YK05] Golan Yona and Klara Kedem. The URMS-RMS Hybrid Algorithm for Fast and Sensitive Local Protein Structure Alignment. *Journal of Computational Biology*, 12(1):12–32, February 2005. doi:10.1089/cmb.2005.12.12.

[ZAS05] Yang Zhang, Adrian K. Arakaki, and Jeffrey Skolnick. TASSER: an automated method for the prediction of protein tertiary structures in CASP6. *Proteins*, 61 Suppl 7:91–8, January 2005. doi:10.1002/prot.20724.

[Zem03] Adam Zemla. LGA: a method for finding 3D similarities in protein structures. *Nucl. Acids Res.*, 31(13):3370–3374, 2003. doi:10.1093/nar/gkg571.

[Zha06a] Yang Zhang. CASP7 server ranking for FM category (TM-Score) [online]. 2006 [cited 2007-08-06]. Available from: http://zhang.bioinformatics.ku.edu/casp7/24.html.

[ZHA+06b] Yang Zhang, Isaac A. Hubner, Adrian K. Arakaki, Eugene Shakhnovich, and Jeffrey Skolnick. On the origin and highly likely completeness of single-domain protein structures. *PNAS*, 103(8):2605–2610, February 2006. doi:10.1073/pnas.0509379103.

[ZKS02] Yang Zhang, Daisuke Kihara, and Jeffrey Skolnick. Local energy landscape flattening: Parallel hyperbolic Monte Carlo sampling of protein folding. *Proteins: Structure, Function, and Genetics*, 48(2):192–201, 2002. doi:10.1002/prot.10141.

[ZKS03] Yang Zhang, Andrzej Kolinski, and Jeffrey Skolnick. TOUCHSTONE II: A New Approach to Ab Initio Protein Structure Prediction. *Biophys. J.*, 85(2):1145–1164, August 2003. Available from: http://www.biophysj.org/cgi/content/full/85/2/1145 [cited 2007-03-13].

[ZS04a] Yang Zhang and Jeffrey Skolnick. Automated structure prediction of weakly homologous proteins on a genomic scale. *PNAS*, 101(20):7594–7599, May 2004. doi:10.1073/pnas.0305695101.

[ZS04b] Yang Zhang and Jeffrey Skolnick. Scoring function for automated assessment of protein structure template quality. *Proteins: Structure, Function, and Bioinformatics*, 57(4):702–710, January 2004. doi:10.1002/prot.20264.

[ZS04c] Yang Zhang and Jeffrey Skolnick. Tertiary Structure Predictions on a Comprehensive Benchmark of Medium to Large Size Proteins. *Biophys. J.*, 87(4):2647–2655, October 2004. doi:10.1529/biophysj.104.045385.

[ZS05] Yang Zhang and Jeffrey Skolnick. TM-align: a protein structure alignment algorithm based on the TM-score. *Nucl. Acids Res.*, 33(7):2302–2309, April 2005. doi:10.1093/nar/gki524.

[ZVMF99] Adam Zemla, Ceslovas Venclovas, John Moult, and Krzysztof Fidelis. Processing and analysis of CASP3 protein structure predictions. *Proteins: Structure, Function, and Genetics*, 37(S3):22–29, January 1999. http://dx.doi.org/10.1002/(SICI)1097-0134(1999)37:3+¡22::AID-PROT5¿3.0.CO;2-W doi:10.1002/(SICI)1097-0134(1999)37:3+<22::AID-PROT5>3.0.CO;2-W.

[ZW05] Jianhua Zhu and Zhiping Weng. FAST: A novel protein structure alignment algorithm. *Proteins: Structure, Function, and Bioinformatics*, 58(3):618–627, 2005. doi:10.1002/prot.20331.

Appendix A

Supplementary materials

		best fitness		best tree size		best tree depth	
decoys set	**impr**	**max**	**avg**	**max**	**avg**	**max**	**avg**
all	0.62%	0.360	0.359	569	325.0	18	17.5
dnoise100	8.82%	0.863	0.851	1580	725.7	18	17.5
dnoise58	8.91%	0.852	0.833	1328	569.5	18	17.3
fnoise100	6.64%	0.785	0.777	610	468.2	18	17.3
fnoise42	9.71%	0.745	0.718	1191	658.3	18	17.7

Table A.1: Comparison of the best evolved functions for different sets of decoys for reference ranking based on the similarity mean consensus. Second column shows percentage fitness improvement over the random walk. The next columns show the maximum and the average value of fitness, tree size and tree depth for the best functions of all six run configurations.

		best fitness		best tree size		best tree depth	
decoys set	**impr**	**max**	**avg**	**max**	**avg**	**max**	**avg**
all	0.40%	0.359	0.359	418	265.8	18	17.7
dnoise100	7.36%	0.840	0.828	810	630.2	17	17.0
dnoise58	9.18%	0.849	0.827	872	524.5	18	17.3
fnoise100	7.66%	0.831	0.819	1020	657.5	18	17.3
fnoise42	8.10%	0.793	0.770	651	444.2	18	17.3

Table A.2: Comparison of the best evolved functions for different sets of decoys for reference ranking based on the similarity mean consensus with rounding precision of 0.01. Second column shows percentage fitness improvement over the random walk. The next columns show the maximum and the average value of fitness, tree size and tree depth for the best functions of all six run configurations.

		best fitness		best tree size		best tree depth	
decoys set	**impr**	**max**	**avg**	**max**	**avg**	**max**	**avg**
all	0.49%	0.359	0.359	769	381.2	18	17.3
dnoise100	8.61%	0.863	0.844	558	342.2	18	17.3
dnoise58	9.34%	0.854	0.838	750	457.8	18	17.7
fnoise100	9.90%	0.815	0.794	1026	513.7	18	17.2
fnoise42	11.27%	0.739	0.713	1159	673.3	18	17.3

Table A.3: Comparison of the best evolved functions for different sets of decoys for reference ranking based on the similarity mean consensus with rounding precision of 0.001. Second column shows percentage fitness improvement over the random walk. The next columns show the maximum and the average value of fitness, tree size and tree depth for the best functions of all six run configurations.

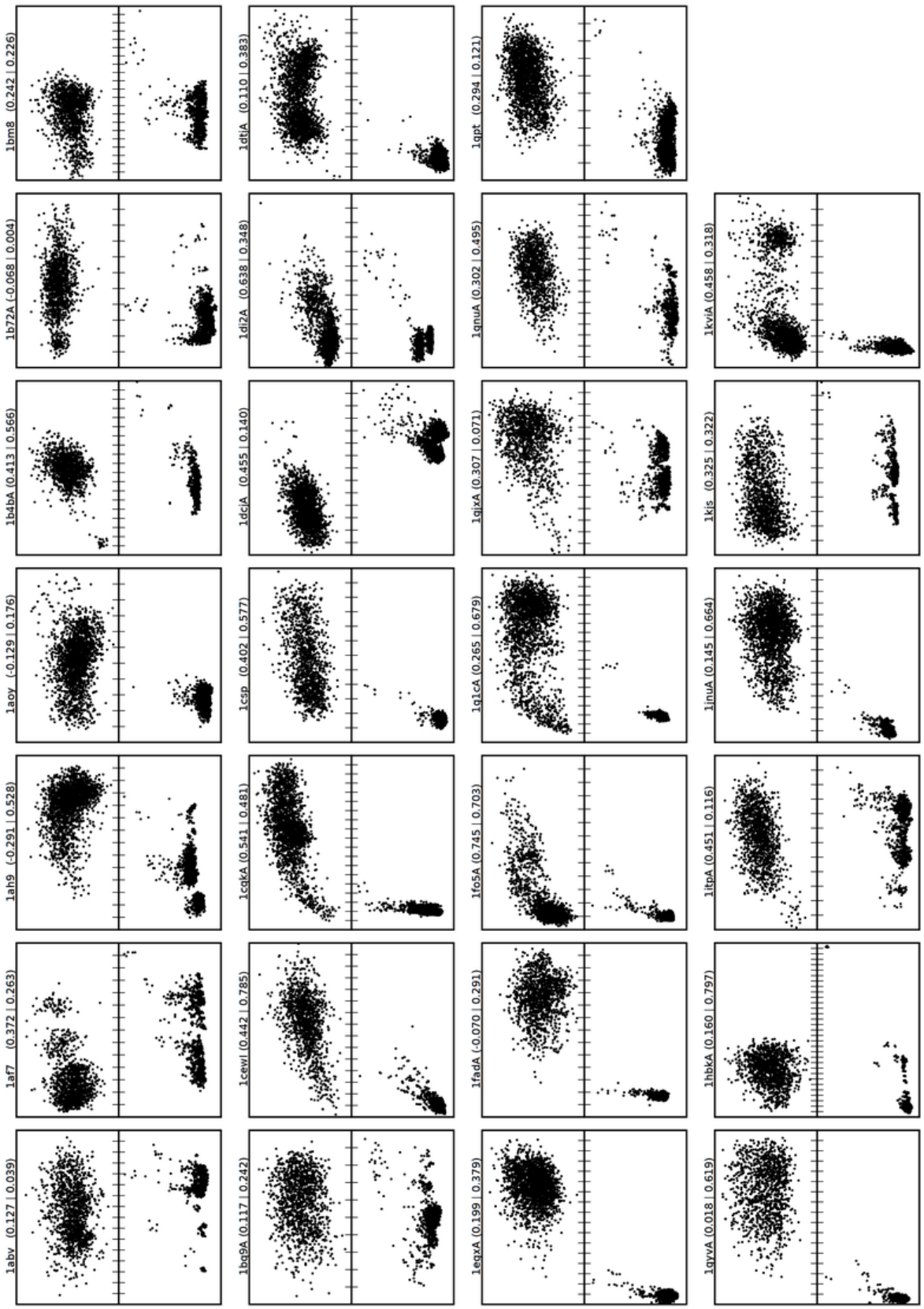

Figure A.1: Correlation of Rosetta (top) and I-TASSER (bottom) energy functions vs. similarity (same scale) — part 1

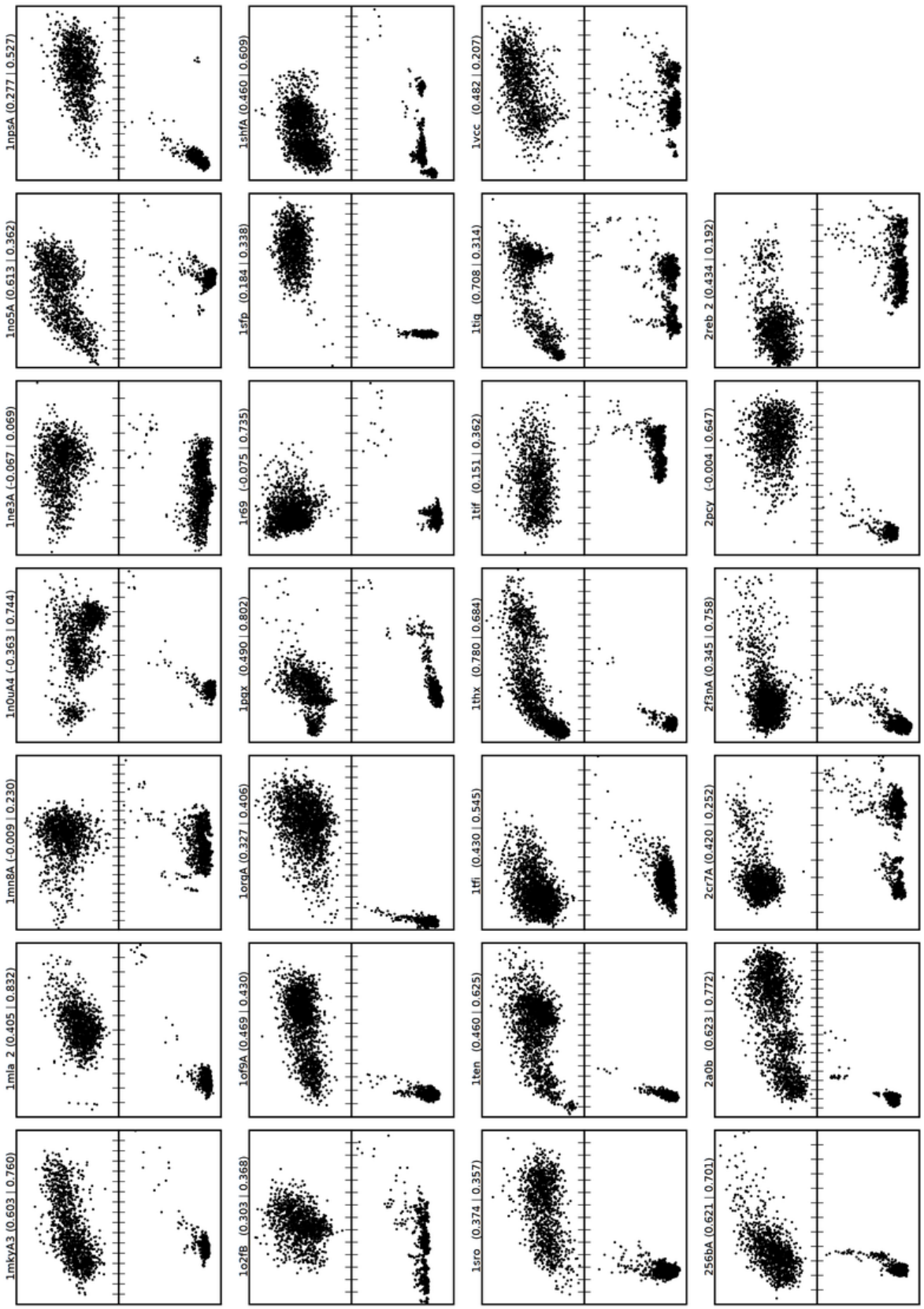

Figure A.2: Correlation of Rosetta (top) and I-TASSER (bottom) energy functions vs. similarity (same scale) — part 2

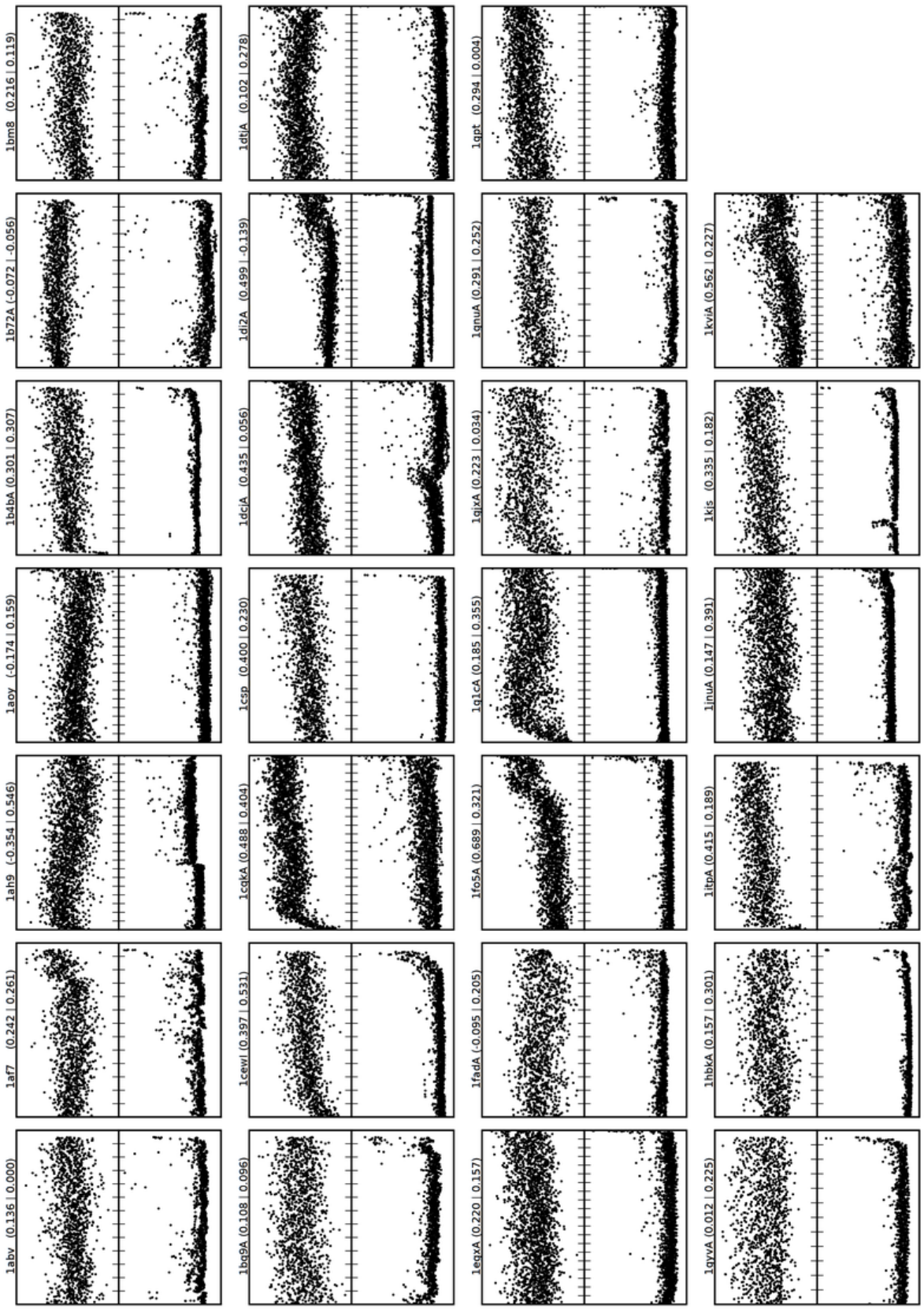

Figure A.3: Correlation of Rosetta (top) and I-TASSER (bottom) energy functions vs. rank (same scale) — part 1

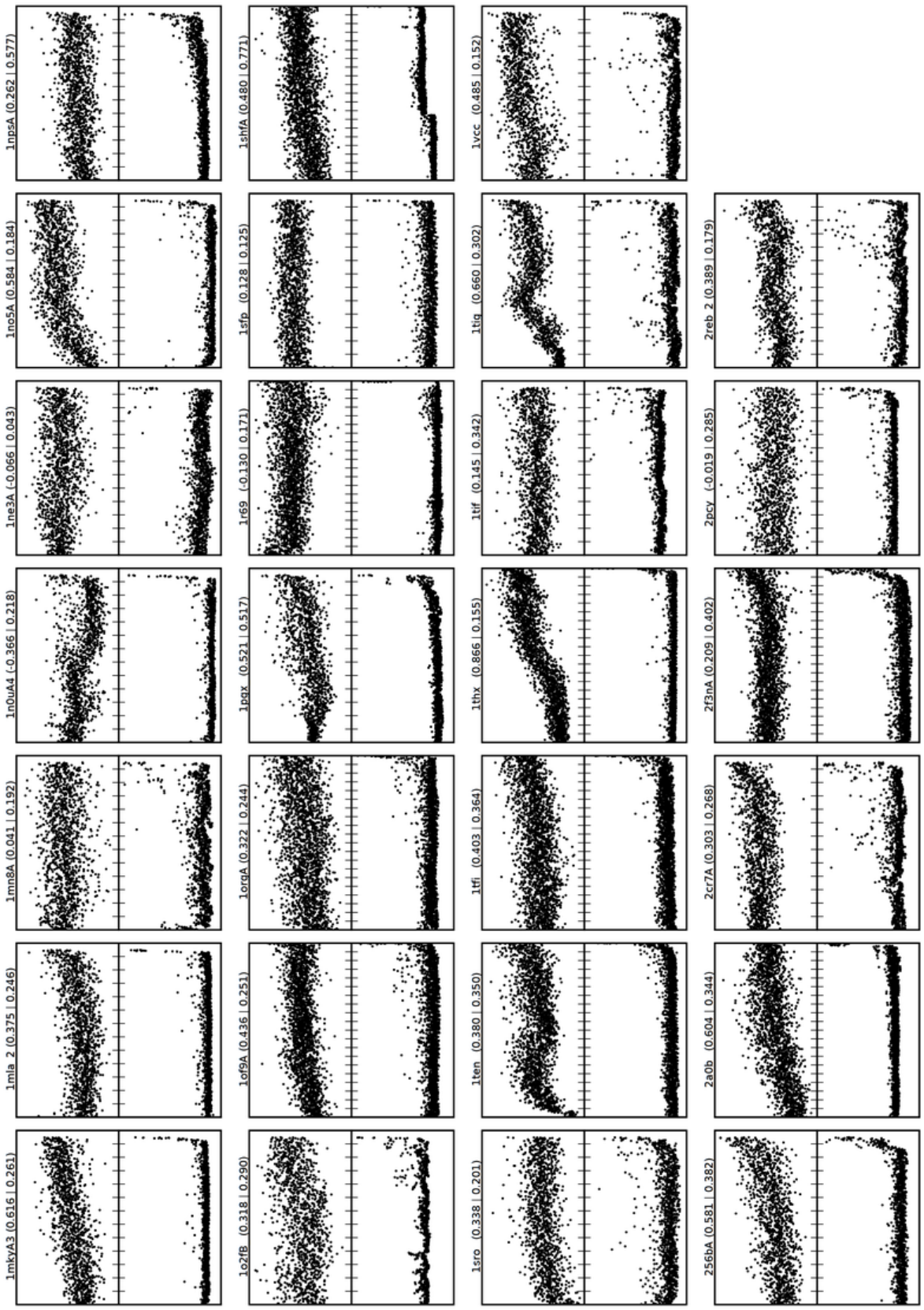

Figure A.4: Correlation of Rosetta (top) and I-TASSER (bottom) energy functions vs. rank (same scale) — part 2

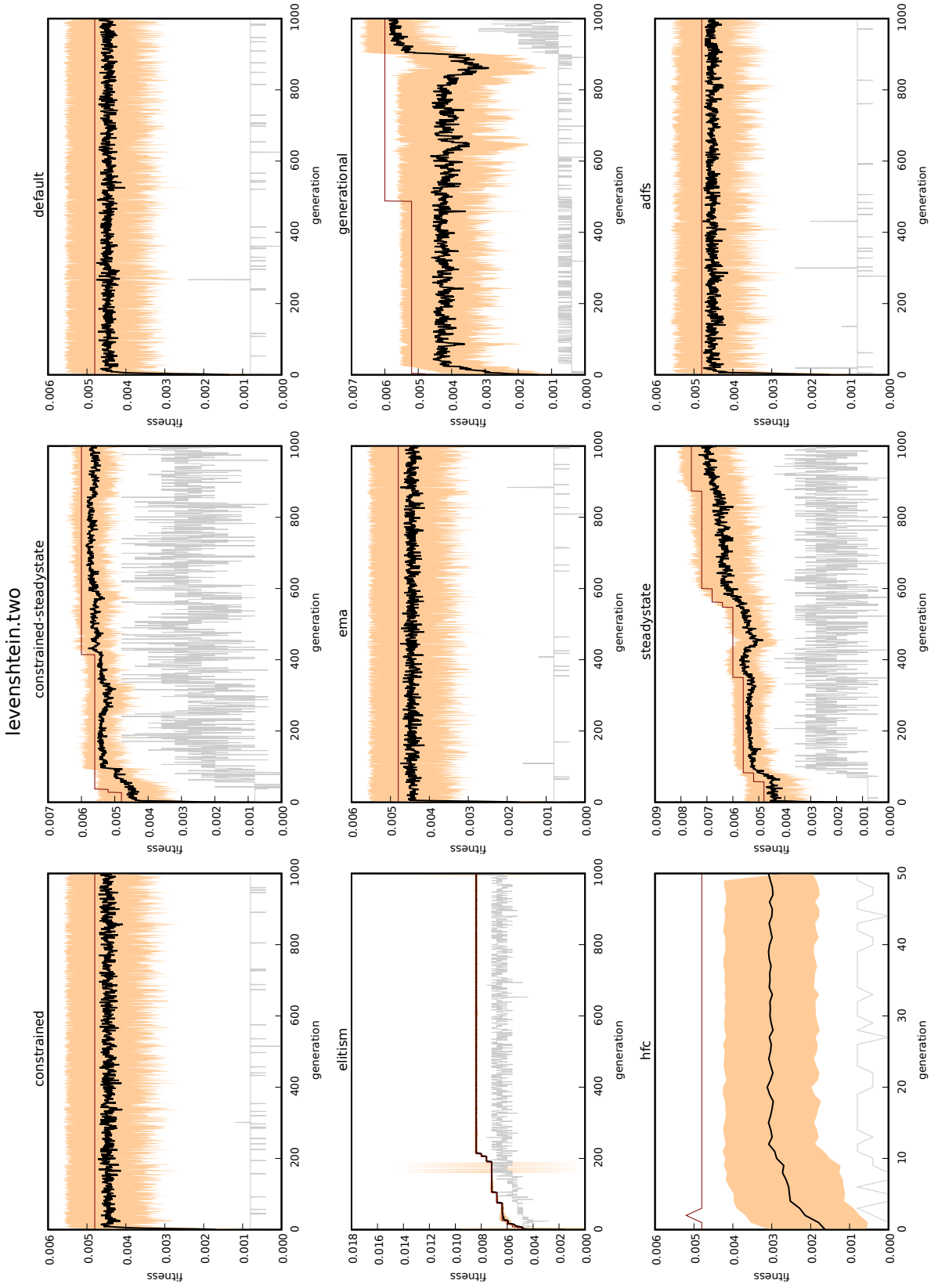

Figure A.5: Fitness throught generations in preliminary runs of GP with Levenshtein distance and first two proteins only.

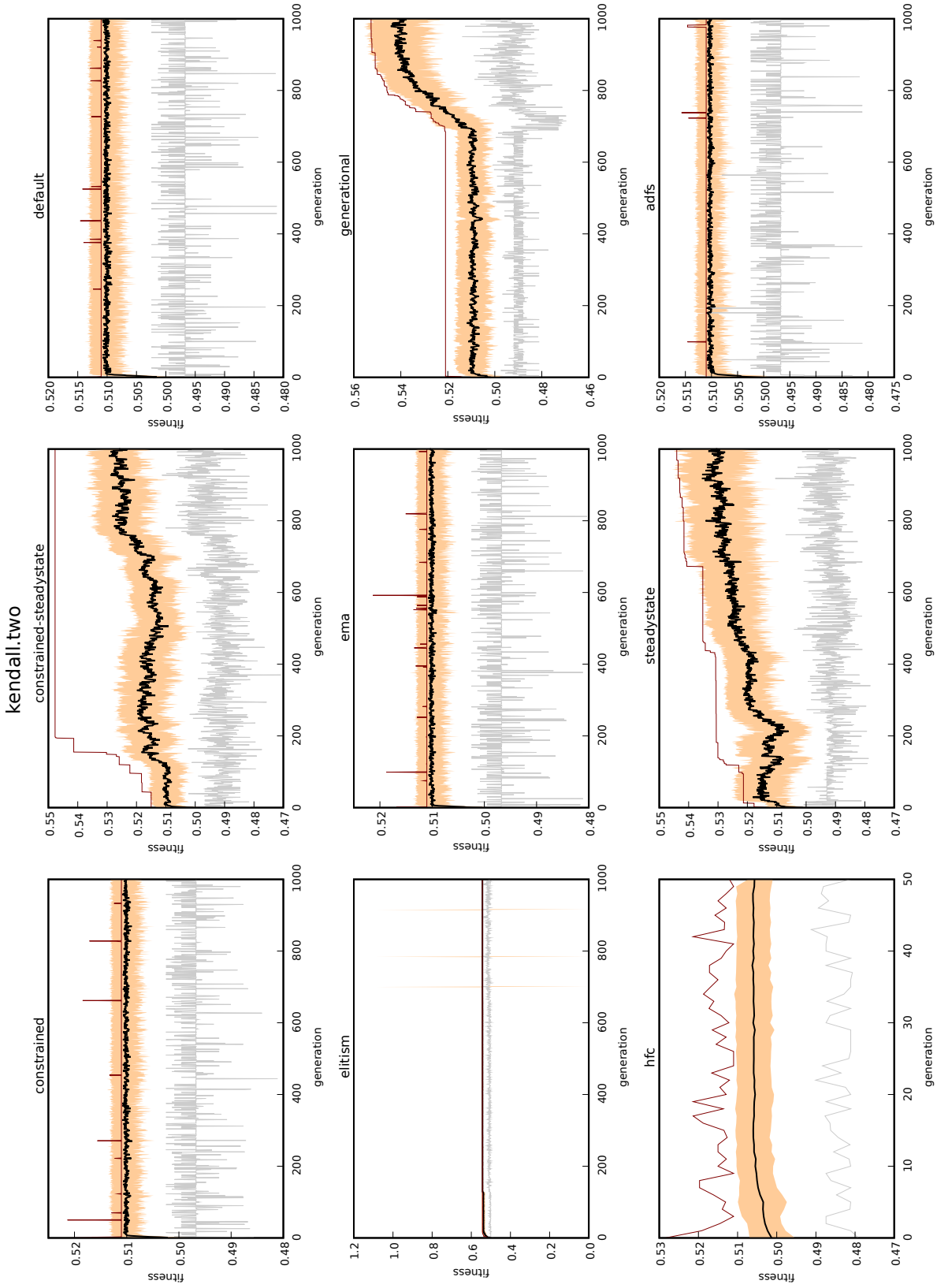

Figure A.6: Fitness throught generations in preliminary runs of GP with Kendall distance and first two proteins only.

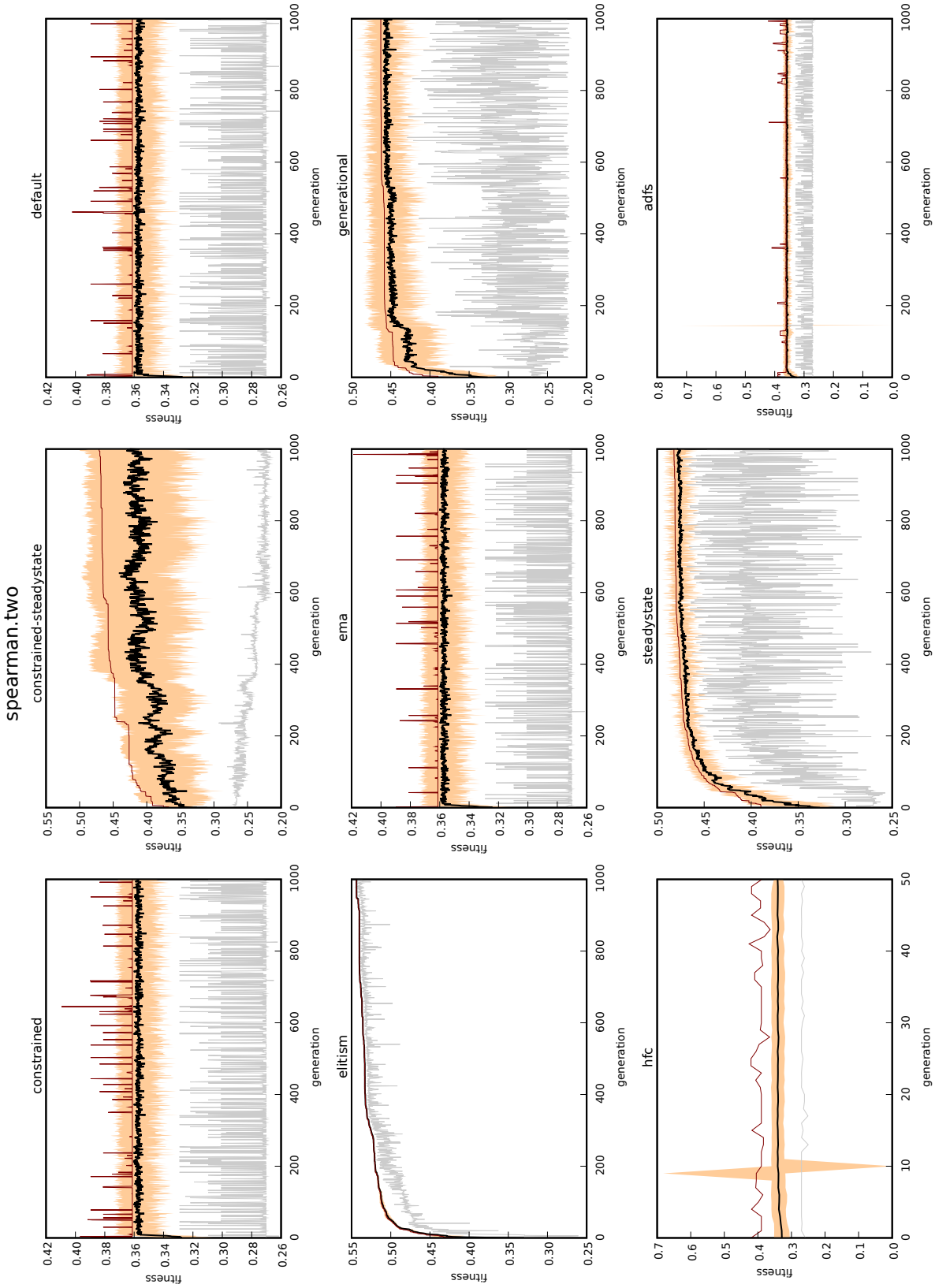

Figure A.7: Fitness throught generations in preliminary runs of GP with Spearman distance and first two proteins only.

Index