Protein Legos: Algorithm Development and Optimisation

Joy Yeh (cy13308), supervised by Fabio Parmeggiani and Simon McIntosh-Smith

University of Bristol

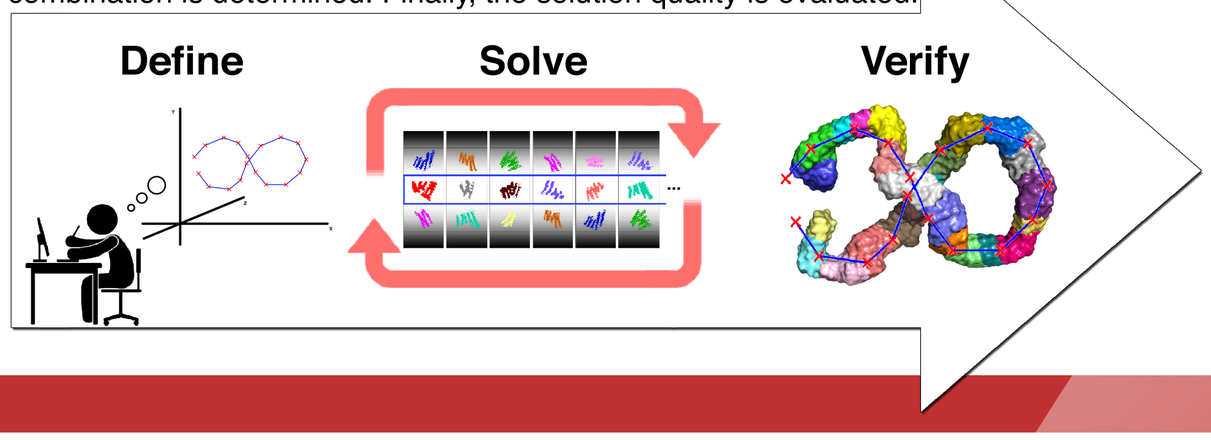
Introduction

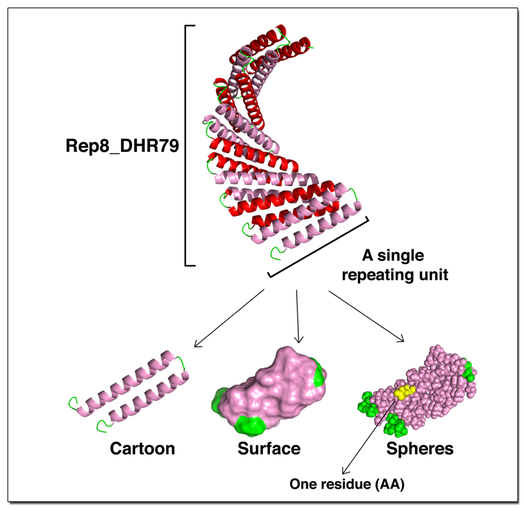
We present Elfin - the first *de novo* protein design application developed to take advantage of what we call "Repeat Unit Construction" (RUC). Compared to existing approaches, RUC offers the following:

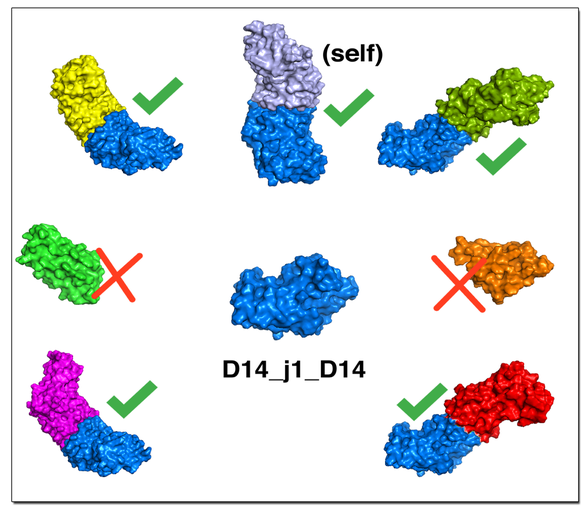
- Greatly reduced computational complexity
- Exploration of large to very large designs
- Increased likelihood of in-lab experiment success

Through mathematical abstractions and a novel algorithm, we were able to implement RUC as a Genetic Algorithm (GA), which we call Elfin. Despite reduced complexity, the search space (and thus design freedom) is still enormous. A great part of this project therefore undertook performance optimisations, with an emphasis on portability.

Protein design is the reverse of protein folding. Its aim is to find an amino-acid (AA) sequence that folds into some desired 3D shape. Below is an input-process-output illustration: akin to building a structure using Lego blocks, the designer first conceives a 3D shape. Then, through some trial and error, a building block combination is determined. Finally, the solution quality is evaluated.







A protein is made of ~O(10^2) residues, and each residue can be one of 20 natural AAs in several possible conformations. Hence, designing a N-residue protein requires evaluating more than ~O(20^N) combinations - clearly an intractable search problem. Consequently, past research has focussed on reducing this complexity, with a number of designs successfully validated in experiment. Regan et al. offer a more complete review in [1].

RUC is a new method that further reduces the search complexity. We propose to use repeat protein units as the basic building block to construct an overall structure. A repeat protein contains two or more identical repeating units. Units from different repeat proteins might or might not bind with one another. The crux of this model is in the rigidity assumption we can make and experimentally verified stability of these proteins.

Methodology

On the right-hand-side is a diagram of the protein design cycle including RUC. Starting from laboratory experiments, repeat unit pairs that are validated get collected into a database, which grows over time. Then, each repeat unit pair gets abstracted down from their atomic representations to a very compact 6-float transformation.

Using the database of pair relationships abstractions, we devise two essential subroutines that are independent of metaheuristic:

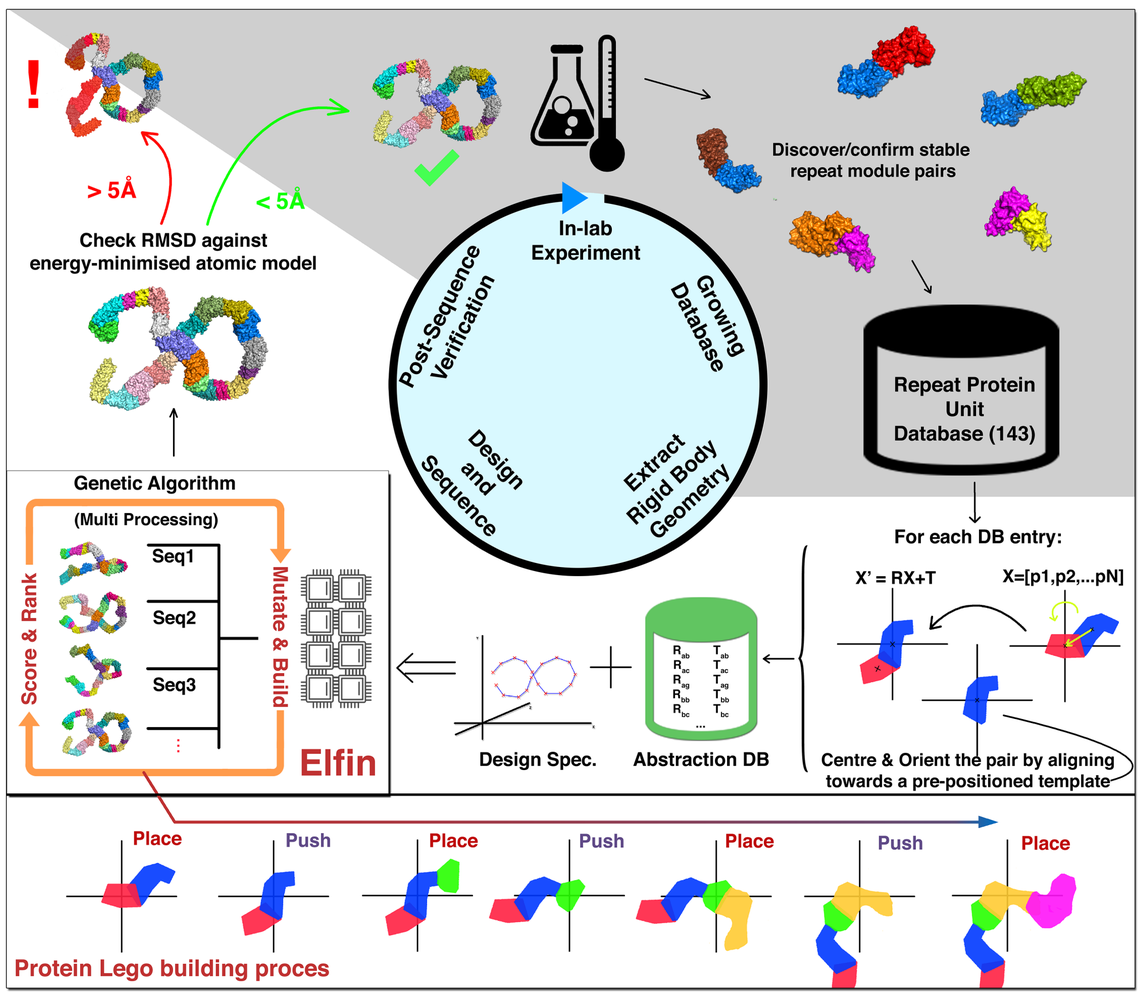
- A protein construction procedure
- A score function to gauge how well a constructed protein fits the input.

For the first problem, we developed a Ribosome-like, "place-and-push" process described near the bottom of the figure. For scoring, we identify it as a partial Procrustes superimposition problem, which is solved using Kabsch's absolute orientation [2] on top of proportional data re-sampling.

With the above data and subroutines, we first experimented with a greedy approach. As expected, while it solved all benchmark problems (known solutions that were randomly generated), it stalled on nearly all real inputs (hand-drawn 3D shapes). These were due to the algorithm's inability to backtrack when it greedily enters a collision dead-end.

Inspired by natural evolution, we implemented Elfin as a GA with the following operators: crossover, point mutation, and "limb" mutation. Some key characteristics of Elfin are:

- Population diversity enforcement – sacrifice trial per second for faster convergence
- Effectiveness verified through benchmark tests
- Geometry and other mathematics verified through unit tests
- Written in C++, ported to OpenMP, and uses only STL and a JSON parsing library

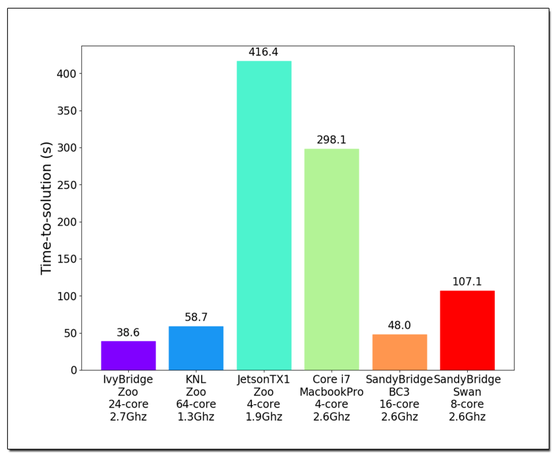


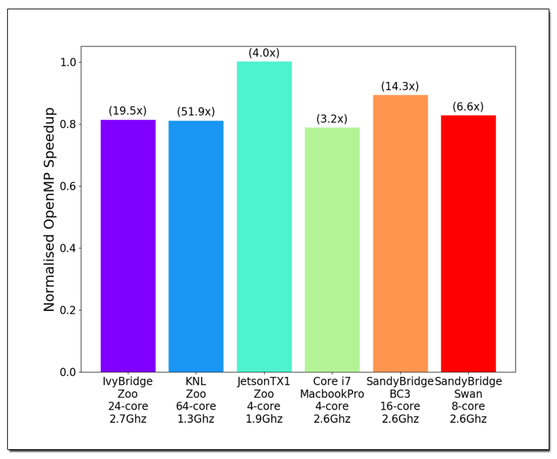
Results

To verify that RUC produces designs that actually fold into the intended 3D shape, we compared Elfin's outputs against Rosetta's [3] energy-minimised version of the same proteins. The result was that all benchmarks passed the 5Å international standard. Past the realm of known solutions, we tested Elfin with custom input. Its effectiveness is shown in both the infinity-sign example in above figures, and the Bristol logo on the top-right corner of this poster.

Second to effectiveness is performance. On the right-hand-side, the top chart displays time taken to solve the infinity sign design, while the bottom chart shows strong scaling effects. These graphs suggest:

- Multiple algorithm and memory access optimisations made Elfin reasonably fast for solving a single design
- Elfin is not yet at interactive speeds
- Elfin strongly scales
- Distributed computing and accelerators should further boost speeds





Future Work

We have identified potentials for further serial optimisation, such as tighter control over dynamic memory allocation and ensuring the presence of optimising cues throughout. Currently, Elfin is being ported to MPI to enable distributed computing, and OpenMP 4.5 to exploit GPGPU to further reduce time-to-solution per design.

In fact, Elfin can solve general rigid-body building block design problems. It is also easily extensible to accommodate new protein units, such has those with more than two terminals (using tree representations), and those with functional attachments (using additional scoring metric).

Overall, we believe Elfin is an effective implementation of the RUC design method, and our optimisation efforts have made Elfin suitable for real scientific use. It is the first application of its kind and will pave way for more research on RUC. A journal paper on computational Structural Biology is under preparation. It is intended that Elfin ultimately becomes part of the Rosetta software suite when it is more thoroughly optimised and documented.

References

[1] Regan, Lynne, et al. "Protein design: Past, present, and future." Peptide Science 104.4 (2015): 334-350.
[2] Kabsch, Wolfgang. "A solution for the best rotation to relate two sets of vectors." Acta Crystallographica Section A: Crystal Physics, Diffraction, Theoretical and General Crystallography 32.5 (1976): 922-923.
[3] RosettaCommons, Rosetta Software Suite 3.8, Computer Software, URL: https://www.rosettacommons.org, accessed 27 Mar 2017

Repeat protein data (unpublished) used with permission from:
TJ Brunette and David Baker, The Baker Lab, Institutde of Protein Design, University of Washington

Graphics attributions:
Designer icon: clipartfest.com; Chemistry icon: freeiconspng.com; All other icons: iconfinder.com

Special Thanks to:
UoB HPC Group for access to BlueCrystal and the Zoo
BrisSynBio for access to BlueGem
Cray Inc. for access to Swan and Falcon

BrisSynBio biomolecules to biosystems from understanding to design

OpenMP

UOB HPC