Repairing gaps in Kinari-2 for large scale protein and flexibility analysis applications

Magdalena Metlicka
College of Inf. and Comp. Sciences
University of Massachusetts
Amherst, Massachusetts, USA
mmetlicka@umass.edu

Mojtaba Nouri Bygi
Department of Computer Science
Smith College
Northampton, Massachusetts, USA
mnouribygi@smith.edu

Ileana Streinu
Department of Computer Science
Smith College
Northampton, Massachusetts, USA
and
College of Inf. and Comp. Sciences
University of Massachusetts
Amherst, Massachusetts, USA
istreinu@smith.edu

***Abstract*—Pebble game rigidity analysis is a combinatorial method, implemented in our free web server KinariWeb, for extracting protein rigidity and flexibility information without performing costly molecular dynamics simulations. Due to the idiosynchrasies of the data in the Protein Data Bank (PDB), Kinari succeeds only on a fraction of the available files. Motivated by large scale applications, aiming at processing almost all the PDB files, we have recently developed a faster and more robust version, the phased pebble game algorithm. It is specifically designed to take advantage of the sequential structure of biopolymers. However, the structural data available in the Protein Data Bank for proteins solved with X-ray crystallography is very rarely complete: missing residues induce gaps in the atom sequence, leading to incomplete rigidity analysis results. In this paper, we describe the pre-processing component of the new version Kinari-2, which fixes such gaps for the purpose of producing a valid input for the phased pebble game algorithm described in [19].**

***Index Terms*—protein rigidity, flexibility, simulated unfolding, dilution, rigid clusters,gaps, sequence, curation, pdb file.**

I. Introduction

Proteins are vital components of living organisms, involved in all cellular processes. The primary structure of a protein consists of chains of amino-acids, connected by strong covalent and peptide bonds. The weaker, non-covalent interactions between these amino-acids are the reason for protein folding and the formation of secondary, tertiary and quaternary structures [3]. Protein folding remains an extremely challenging problem to study, both experimentally and in computer simulations. Recent studies have focused instead on general protein flexibility and slow motions. Various coarse-grained approaches have been proposed for gaining *insights* into what might cause *large conformational changes* near a protein's native state, as these are evidence of important functions. *Rigidity analysis* is one such method, which has been successfully applied to extract useful flexibility and *simulated unfolding* information from protein crystallographic data.

Funded by the NIH 1R01GM109456 and NSF CCF-1319366 of IS.

KINARI (KINematics And RIgidity Analysis of biomolecules) is an on-going project in the group of the senior author, aiming at overcoming many of the limitations of previous rigidity-based implementations. The first version was released in 2011 as a free open-access web server at http://kinari.cs.umass.edu. We are currently developing the second major revision of the software. For comparison purposes, we refer to the 2011 version [10] as Kinari-1 and use Kinari-2 for the new, on-going project.

Kinari-1 was tested and profiled on approx. 28000 entries from the Protein Data Bank (PDB) [1]. It fails to run to completion on many other entries, primarily because of the idiosyncrasies of the PDB file formats, but also because of the software implementation limitations in processing very large size molecules. The goal of Kinari-2 is to make possible the efficient *large scale study of protein flexibility.* The new design has been described in [20], and various aspects of the new system have been previously reported in [2], [8], [9]. Very recently, we presented in [19] a key algorithmic improvement engineered for speeding up the software kernel of Kinari-2.

The *current paper* focuses on the critical pre-rigidity analysis processing, or *curation of PDB data,* in order to make it possible for the future users of Kinari-2 to *automatically* and manually overcome the severe limitations due to gaps and ommisions in the sequence, missing atoms, errors and ambiguities in the experimentally-reported data available in the PDB.

Overview. In this paper, we describe the new pre-processing phase of Kinari-2, whose output is fed into the phased pebble game Following an enhanced curation stage, this pre-processing aims first at building an atom-bond network which is as complete as possibly can be inferred from *both* the sequence (SEQRES lines) and the structural data (atom coordinates in ATOM/HETATM lines) available in the input PDB file.

This paper focuses on 'methods' by describing the principles underlying the pre-processing outlined above, which is necessary for completing the implementation and integration

in Kinari-2 of our new phased pebble game for protein rigidity analysis. The methods have been tested for correctness and accuracy on a variety of PDB-native and PDB-engineered data, but the idiosynchrasies of the data in PDB files offer new surprises that can only be overcome through a large scale, whole PDB survey of all the components described in this paper. This process is under way, and the results are planned to be reported in the long, journal version of this paper.

II. Methods and Implementation

We split the Kinari-2 pre-processing into three parts:

1) **Curation: repair, build and extract.** A pdb file is enhanced with additional hydrogen atoms, if they are missing **(repair)**, then larger structures such as bio-assemblies are **built**, if desired for further processing and, finally, one or more chains of single models or molecular conformations are **extracted** for further processing.
   The next two parts are described for a single chain, but executed (taking symmetries into account for avoiding repetitive calculations) for all existing chains in this resulting pdb file.
2) Build a **tree-structured atom-bond network** of proper and virtual atoms and bonds. The result is an xml-formatted file of type .bnd (from *bond*).
3) Build a **tree-structured body-bar-hinge-pin** mechanical model from a single chain .bnd file. The result is an XML-formatted file of type .bbh (from *body-bar-hinge*).

The .bbh file is sent to the new phased pebble game described in [19] and the properly grouped tree and not-tree information is processed in one of the two phases of this new algorithm.

The next sections give an overview of these three parts of Kinari-2 pre-processing, along with some minimal theoretical explanation justifying the use of the mechanical model. A more detailed description is outside the scope of this extended abstract, and can be found in the references describing the previous version of our software, Kinari-1 [5], [7], [10]–[16], [21], in the case studies and educational resources available from Kinari-1 web site [18] and in the overview of the new Kinari-2 design from [2], [8], [9], [20].

III. Curation of a PDB file

Curation of data for using in Kinari-2 While developing the next version of Kinari software, the first, most basic and possibly the most tedious challenge we faced was the data quality bottleneck: the large and often poorly documented variations in format and available information recorded (or not) in molecular structure files, and the difficulty of automatically extracting some of this information. To respond to the need for reproducible curation results, the new software tools (interactive or automated) should record the curator's decisions and generate quality measurements suitable for integration in further method-comparison or validation steps. In this category, we identified a need for going beyond existing tools for repairing molecular structures by developing or integrating third-party tools for handling missing parts (loops, residues, atoms), "fixing" low resolution protein structures or C_α-only backbones, modeling and quantifying the impact of alternative conformations, detecting "defects" such as gaps in the protein structure (compared to the known protein sequence), or performing structural "repairs" in proteins.

The curation process. Unless ran with predefined default options, curation is meant to be an interactive phase consisting of a series of steps, in which the user may perform *in silico surgery* on the molecular complex present in the PDB file, by deleting solvent or ligands, selecting specific chains, or building an entire molecular bio-assembly or crystal fragment, if the user opts to work with such larger structures. Some necessary repairs are also performed, such as adding missing Hydrogen atoms (using the third-party software Reduce [22]) to structures solved by X-ray crystallography: the hydrogens are needed for computing the hydrogen bonds. Finally, a network of various types of bonds and interactions between the atoms is computed. An energy value is calculated for each particular bond or interaction, according to its type, and the user specifies a cutoff value for hydrogen bonds and hydrophobic interactions.

In Kinari-2 we are placing maximum emphasis on the management and reproducibility of curation experiments [2]. Fig. 1 illustrates the difference between the original pdb file and one that has been pre-processed with Reduce for placing its missing hydrogen atoms. Significantly different rigidity results will be reported on these two files, although they represent the same protein.

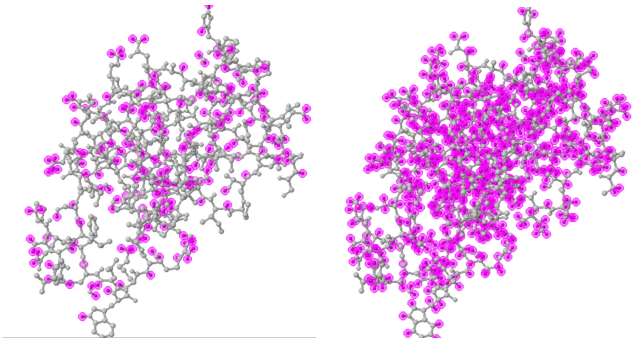

Fig. 1. Highlighted in magenta, the hydrogen atoms in the original PDB file 1AZ5 (left) and after repairing them with Reduce (right).

Reproducibility of protein data curation. It is well known that the files deposited in the Protein Data Bank are not of uniform quality: to help with judging the quality of the experimental data deposited in the PDB, resolution and B-factors are parameters recorded with X-ray solved protein data. A number of entries in the PDB have been declared obsolete and replaced by others. Tools for checking the quality of crystallographic data are also available, such as MolProbity [4], [6]. The accuracy of the molecular model is relevant during the preprocessing of PDB files (prior to rigidity analysis)

because of its implications on bond calculations. Some types of bonds are computed with third party software, and different software performing the same task may produce different results, which may in turn lead to different rigidity analysis results. The critical steps include: adding the hydrogen atoms if the data come from an X-ray crystallography experiment; pruning the hydrogen bonds according to a user-selected cut-off value; selecting the model from among several available in a file containing data from an NMR experiment; computing the hydrogen bonds and hydrophobic interactions; building a biological assembly or, possibly, a small crystal, etc.

A curated molecule is next processed to compute the set of all relevant bonds and interactions. They are recorded in separate categories, including covalent (single and double) bonds, peptide bonds, disulphide bonds, hydrogen bonds, and hydrophobic interactions. For comparison purposes, alternate methods have been explored or some will be integrated in Kinari-2, including bond calculations with third-party software such as JMol, HBPLUS, and bndlist. Some scripts used in Kinari-1 are adapted to be used in Kinari-2, but there are some important differences, described next.

Detecting gaps and other missing atoms. A new and important goal in Kinari-2 is to repair residue and atom-level errors in PDB files. The structure data (atom coordinates) is in general noisy, often incomplete and sometimes presented in a non-standard manner. This noise has detrimental effects on the quality of rigidity analysis results, and in some cases it is impossible to run the Kinari pipeline to completion without some prior repair work. For this purpose, Kinari-2 uses both the SEQRES and ATOM/HETATM lines. The SEQRES entries provide information about the *sequence of residues* in each chain, corresponding to the primary protein structure. The ATOM/HETATM entries give the 3D coordinates of experimentally observed atoms, but those that have not been experimentally 'solved' are omitted. Ambiguities and experimental noise thus lead to omissions or multiple reported coordinates for atoms in ATOM/HETATM entries. But implicit in SEQRES are the identities and strong (covalent) connectivities of *all* the atoms in a given chain. Kinari-2 makes use of this information in ways that are relevant for the graph-theoretic model used by our software.

IV. Repairing Gaps and Missing Atoms

For the sake of a simpler presentation, we discuss how the repairing process works on a *single* protein chain. The goal is to reconstruct the entire strong (covalent) atom-bond network.

SEQRES reconstruction of an atom-bond network. This is a three-step process: 1) Using the SEQRES sequence, we build a *virtual* atom-bond network of the protein chain consisting in all atoms (without explicit coordinates) and strong bonds that should be present along the protein's backbone and residues, annotated by type (single, double, peptide bonds, etc.). 2) Using the ATOM/HETATM lines, fill in the previously constructed model with the coordinates for all the atoms that are present. These are referred to as *proper* atoms, and are distinguished from the remaining *virtual* ones, for which no coordinates are known. 3) Using third-party software and code from Kinari-1, we calculate the weak, distance dependent interactions, such as hydrogen bonds or hydrophobics. We add them to the atom-bond network, to obtain an XML-formatted file of type .bnd (from *bond*).

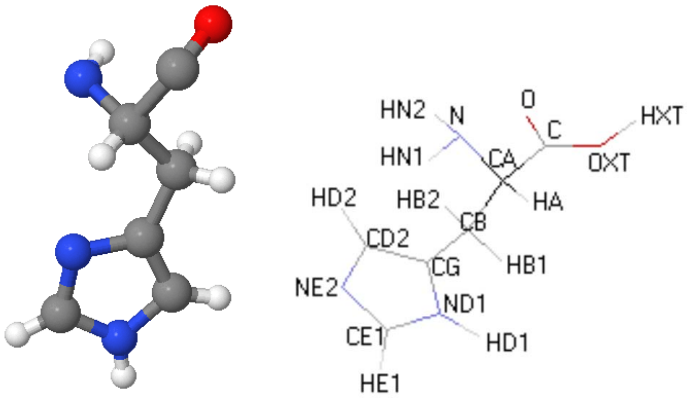

Fig. 2. Histidine (His, H)

V. Atom-Bond Network and Underlying Tree

We now make use of the linearity of the protein backbone and the *almost* tree-structure obtained by attaching the residues to it. With the exception of five residues which have cycles in their atom-bond network (HIS, PHE, PRO, TRP, TYR), the standard amino-acids lead immediately to such a tree structure. The exceptions are treated separately by breaking the cycles arbitrarily and remove some bonds. These loop bonds will be processed along with the weak bonds. This resulted tree is referred to as a *covalent atom-bond tree.* Fig. 3 illustrates the procedure. Combining all the bonds, we will have the full atom-bond network resulted from a (h-repaired) PDB file. This includes both proper and virtual atoms.

Putting everything together, there are two categories of bonds. The first category is the bonds from the covalent tree, or tree bonds. Every other bonds in the chain belongs to the

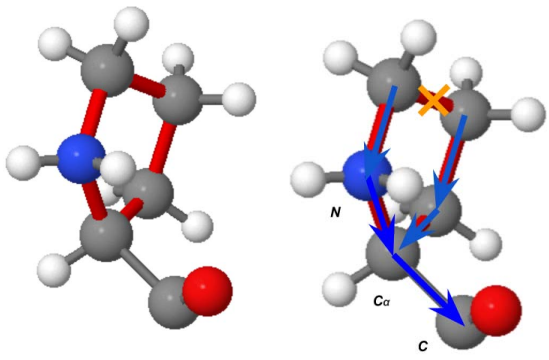

Fig. 3. The Proline residue (left) with colored bonds (left). Removing one loop edge, we obtain a covalent tree, and orient its edges towards the backbone (right).

second category. It consists of every other bonds, such as covalent bonds from SEQRES network to break their loops, and weak bonds (for example h bonds).

Direction of bonds. If the tree bonds are in the backbone, they are oriented from N terminus to C terminus, and if they are hanging residues in the tree, they are oriented towards the backbone. For the other bonds in the molecule, the direction is towards the covalent tree.

Phased pebble game. A direct usage for atom-bond network is in phased pebble game, introduced in [19]. The directed atom-bond tree is processed in the first phased of the game in linear time, which is one order of magnitude faster that the original algorithm [17]. The rest of the atom-bond network will be passed to the second phase

VI. Results

The methods described in this paper have been implemented and tested for correctness and accuracy on a variety of PDB-native and PDB-engineered data. Along the way, we have discovered many poorly documented or simply undocumented aspects of the PDB format, and the idiosynchrasies of the data in PDB files offered sufficient surprises to make us understand the challenges implicated in running large scale, whole PDB surveys for all the components described in this paper. In particular, the examples chosen to illustrate this paper have been created with the current, not-yet-released implementation of Kinari-2. We are confident that after integrating the components described here into the full Kinari-2 pipeline, we will cover a much larger fraction of the PDB than previous implementations have done (in particular, significantly more than Kinari-1). This process is under way, and the results are planned to be reported in the long, journal version of this paper.

VII. Conclusion

Rigidity analysis is a promising method for protein flexibility analysis, but it requires a large scale validation to demonstrate its full potential.

Large scale surveys, e.g. of the entire Protein Data Bank, require not only more efficient algorithms and software implementations, but also significantly improved methods for curating and pre-processing the data. In this paper, we presented a methodology for automating the curation and pre-processing steps of rigidity analysis, with the goal of taking advantage of a carefully engineered algorithms designed to integrate the specificities of protein structures in order to improve the performance of the new version of Kinari-2. A comprehensive testing and profiling of the new implementation is currently under way and the results will be reported in the full version of this paper.

References

[1] H. M. Berman, J. Westbrook, Z. Feng, G. Gilliland, T. N. Bhat, H. Weissig, I. N. Shindyalov, and P. E. Bourne. The protein data bank. *Nucleic Acids Research*, 28(1):235–242, 2000.

[2] J. C. Bowers, R. T. John, and I. Streinu. Managing reproducible computational experiments with curated proteins in kinari-2. *Proc. Bioinformatics Research and Applications (ISBRA'15), Lecture Notes in Computer Science*, 9096:72–83, 2015. ISBN: 978-3-319-19047-1 (Print) 978-3-319-19048-8 (Online).

[3] C. Branden and J. Tooze. *Introduction to Protein Structure*. Garland Publishing, Inc., New York, 1998.

[4] V. B. Chen, W. B. Arendall, III, J. J. Headd, D. A. Keedy, R. M. Immormino, G. J. Kapral, L. W. Murray, J. S. Richardson, and D. C. Richardson. *MolProbity*: all-atom structure validation for macromolecular crystallography. *Acta Crystallographica Section D*, 66(1):12–21, Jan 2010.

[5] P. Clark, J. Grant, S. Monastra, F. Jagodzinski, and I. Streinu. Periodic rigidity of protein crystal structures. In *2nd IEEE International Conference on Computational Advances in Bio and Medical Sciences (ICCABS'12). Feb. 23-25*, February 2012.

[6] I. W. Davis, A. Leaver-Fay, V. B. Chen, J. N. Block, G. J. Kapral, X. Wang, L. W. Murray, W. B. Arendall, III, J. Snoeyink, J. S. Richardson, and D. C. Richardson. Molprobity: all-atom contacts and structure validation for proteins and nucleic acids. *Nucleic Acids Research*, 35:W375–W383, 2007.

[7] E. Flynn, F. Jagodzinski, S. P. Santana, and I. Streinu. Rigidity and flexibility of protein-nucleic acid complexes. In *Proc. 3nd IEEE International Conference on Computational Advances in Bio and Medical Sciences (ICCABS'13)*, June 2013.

[8] E. Flynn and I. Streinu. Consistent visualization of multiple rigid domain decompositions of proteins. *Proc. Bioinformatics Research and Applications (ISBRA'15), Lecture Notes in Computer Science*, 9683:151–162, May 2016.

[9] E. Flynn and I. Streinu. Matching multiple rigid domain decompositions of proteins. *IEEE Transactions on Nanobioscience*, 16(2):1–10, 2017.

[10] N. Fox, F. Jagodzinski, Y. Li, and I. Streinu. Kinari-web: A server for protein rigidity analysis. *Nucleic Acids Research*, 39(Web Server Issue):W177–W183, 2011. doi:10.1093/nar/gkr482.

[11] N. Fox, F. Jagodzinski, and I. Streinu. Kinari-lib: a c++ library for pebble game rigidity analysis of mechanical models. In *Minisymposium on Publicly Available Geometric/Topological Software, Chapel Hill, NC, USA, Jun. 17-19, 2012*, 2012.

[12] N. Fox and I. Streinu. Towards accurate modeling for protein rigidity analysis. In *2012 IEEE 2nd International Conference on Computational Advances in Bio and medical Sciences (ICCABS)*, pages 1–6, Feb 2012.

[13] N. Fox and I. Streinu. Towards accurate modeling of noncovalent interactions for protein rigidity analysis. *BMC Bioinformatics*, 14(18):S3, Nov 2013.

[14] N. Fox and I. Streinu. *Redundant and Critical Noncovalent Interactions in Protein Rigid Cluster Analysis*, pages 167–196. Springer Berlin Heidelberg, Berlin, Heidelberg, 2014.

[15] F. Jagodzinski, P. Clark, T. Liu, J. Grant, S. Monastra, and I. Streinu. Rigidity analysis of periodic crystal structures and protein biological assemblies. *BMC Bioinformatics*, 14 (Suppl 18):S2, 2013. Selected articles from the Second IEEE International Conference on Computational Advances in Bio and Medical Sciences (ICCABS 2012): Bioinformatics.

[16] F. Jagodzinski, J. Hardy, and I. Streinu. Using rigidity analysis to probe mutation-induced structural changes in proteins. *Journal of Bioinformatics and Computational Biology*, 10(3):1242010, 2012.

[17] A. Lee and I. Streinu. Pebble game algorithms and sparse graphs. *Discrete Mathematics*, 308(8):1425 – 1437, 2008. Third European Conference on Combinatorics – Graph Theory and Applications.

[18] LinkageLab. *KINARI Web*. http://kinari.cs.umass.edu.

[19] M. Nouri Bygi and I. Streinu. Efficient pebble game algorithms engineered for protein rigidity applications. In *7th IEEE International Conference on Computational Advances in Bio and Medical Sciences (ICCABS'17)*, 2017.

[20] I. Streinu. Large scale rigidity-based flexibility analysis of biomolecules. *Structural Dynamics*, 3(012005), January 2016.

[21] I. Streinu, F. Jagodzinski, and N. Fox. Tutorial: Analyzing protein flexibility: an introduction to combinatorial rigidity methods and applications. In *IEEE International Conference Bioinformatics and Biomedicine (BIBM 2011), Atlanta, GA, Nov 12-15, 2011*, 2011.

[22] J. Word, S. C. Lovell, J. S. Richardson, and D. C. Richardson. Asparagine and glutamine: using hydrogen atom contacts in the choice of side-chain amide orientation1. *Journal of Molecular Biology*, 285(4):1735 – 1747, 1999.