Technical Report

IRI-TR-16-06

# Protein Rigidity Decomposition using the Pebble Game Algorithm

Helen Abadiotakis

July 2016

Institut de Robòtica i Informàtica Industrial

## Abstract

This technical report summarizes the work done by Helen Abadiotakis during her research stay at IRI in the summer of 2016. The objective of the work was to implement the 2D and 3D Pebble Game rigidity detection algorithms with the aim of applying them to the detection of rigid clusters of atoms in proteins.

**Institut de Robòtica i Informàtica Industrial (IRI)**
Consejo Superior de Investigaciones Científicas (CSIC)
Universitat Politècnica de Catalunya (UPC)
Llorens i Artigas 4-6, 08028, Barcelona, Spain

Tel (fax): +34 93 401 5750 (5751)
`http://www.iri.upc.edu`

**Corresponding author:**

J. M. Porta
tel: +34 93 401 0775
`porta@iri.upc.edu`
`http://www.iri.upc.edu/staff/porta`

## 1 Introduction

Proteins by nature are not static, but instead dynamic assemblies of rigid and flexible subparts working together as one intricate machine. The Protein Data Bank has comprehensively annotated the sequence of amino acids of many proteins, cataloging and documenting proteins on the atomic level as well as its three-dimensional structure, showing clear connections between its individual components. However, it remains quite difficult to report holistic protein movement and motion.

The complex nature of proteins does not lend itself to be easily simulated. Protein motions are typically analyzed with molecular dynamics, which rely on complex all-atom force fields. However, this method can only simulate small amplitude motions. Approaches to simulate large amplitude protein motions use coarse-grained models, where atoms are not considered individually as nodes but instead are grouped. An alternative option is to use the "rigid geometry assumption," where atom bonds and angles are considered fixed. Under this assumption, proteins become long chains of atoms articulated with revolute joints. The conformations that these chains fold up into are stabilized by bonds between distant atoms such as hydrogen bonds and disulfide bridges. In this way, proteins can be seen as rigid groups of atoms connected by flexible chains.We decompose the proteins into rigid bodies, finding areas of structural rigidity and those that are flexible. This gives rise to a chain of interconnected links giving the protein several degrees of freedom. In the figures below [5], the protein Crambin is analyzed and discovered to find one large rigid body within the core of the protein and a smaller, separate rigid part along one alpha helix.

These pattern of connections can be quite complex, hindering the simulation of motion, even after careful rigidity decomposition. To model proteins effectively as robots, we have implemented the pebble game with respect to protein structure to analyze motion. The pebble game algorithm, which has already been outlined by Jacobs and Hendrickson [2], determines the independence of a set of edges in two dimensions. This theory is then applied to a set of residue connections within a protein, which can be found in the protein data bank. This analysis was made to further extend the CuikSuite package developed in the Kinematics and Robot Design group at IRI. The recent advances in rigidity theory with the motion analysis tools has made it possible decompose the proteins in rigid parts, and generate the possible relative motions between such parts. The CuikSuite offers tools to analyze the motions of mechanisms with arbitrary architecture. Thus, this project aims to integrate CuikSuite with the rigidity theory tools necessary to detect the rigid clusters of atoms and their flexible connections.

In Section 2, we review the background of rigidity theory and its application to the pebble game. We discuss our implementation in Section 3 and the 2-D and 3-D working modes we have developed. In Section 4, we discuss our results of our testing. Lastly, Section 5 presents a brief summary and future goals of completing our objective.

## 2 Background

Rigidity is the property of a structure such that its shape is preserved and does not bend or flex at its hinges. The opposite of rigidity is flexibility, where there is not a preserved distance between adjacent vertices. Rigidity theory predicts the flexibility of structures formed by rigid vertices that are linked or hinged. With respect to proteins, the rigid bodies are residues and the linkages are bonds. To determine whether a graph is rigid or flexible, we can take the following theorem into consideration [1]:

THEOREM 2.1. *If a graph has a single infinitesimally rigid realization, then all its generic realizations are rigid.*

We can recognize a rigid graph by examining its edges and classifying them as independent

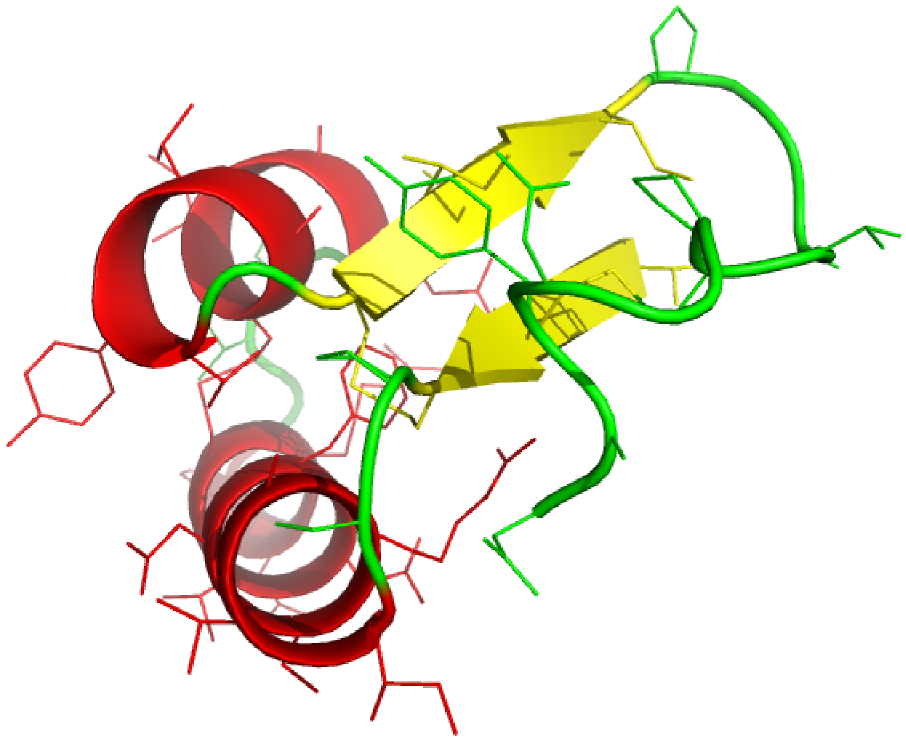

**Figure 1:** Crambin (PDB file 1CRN), a 46 amino acid protein with two alpha helices and antiparallel beta sheet

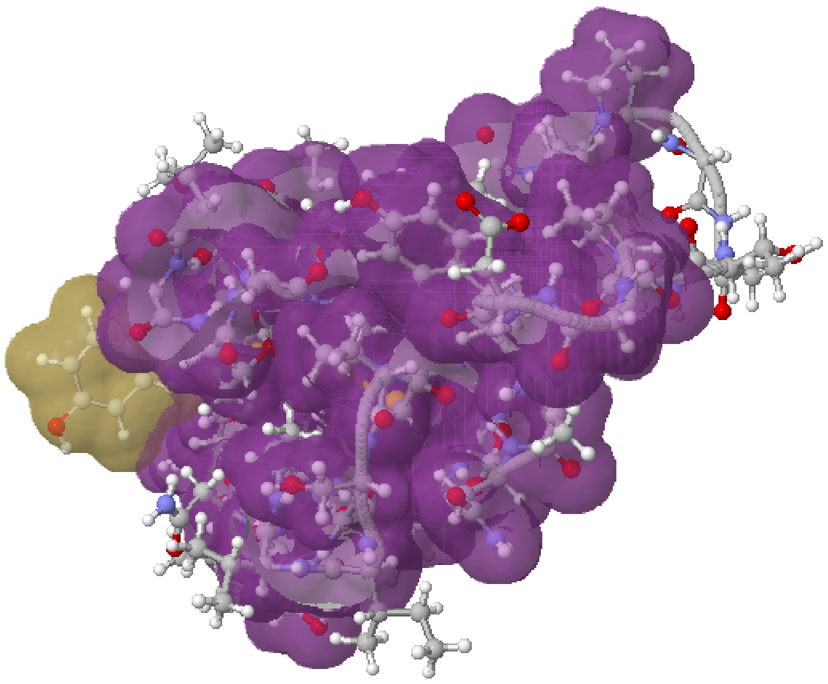

**Figure 2:** Crambin decomposed into rigid parts, contains one large rigid cluster in core of the protein

or redundant. In a two-dimensional plane, we classify a framework as rigid if it has $2n-3$ independent edges. These edges are enough to eliminate any degrees of freedom that the graph may contain. We can use the Laman theorem to classify edges under inspection [3].

THEOREM 2.2. *The edges of a graph G are independent in two dimensions if and only if no subgraph G' has more than* $2n-3$ *edges.*

A subgraph $G'$ with $n'$ vertices and $2n'-3$ independent edges is characterized as a Laman subgraph. The following Laman theorem helps us to determine whether edges are independent or not.

THEOREM 2.3. *For a graph G have* $m$ *edges and* $n$ *vertices, the following are equivalent:*

1. *The edges of G are independent in two dimensions.*

2. *For each edge in g, the graph formed by adding three additional edges has no induced subgraph* $G'$ *in which* $m' < 2n'$.

Thus a new edge is only added if it satisfies this above theorem and is discovered to be independent of the current set.

## 2.1 2-D Pebble Game

Using the bar-and-joint model, the bodies can be described as a graph $G$ with set of vertices $V$ from 1 to n, and a set of edges $E$ from 1 to m. Each connects a pair of vertices *(u,v)*, with no self loops, such that there exists a bond between $u$ and $v$. The game begins with two pebbles on each node. These pebbles are used to cover up to two independent edges incident to the vertex. The algorithm grows a maximal set of independent edges by adding edges only if four pebbles can be gathered at its two nodes. If a pebble covering can be found, the edge being inspected is independent with respect to the graph's current state. In the pebble game, no more than two pebbles may be present on a vertex at any time. The algorithm by Jacobs and Hendrickson that was implemented uses breadth-first search to find a free pebble to cover an edge $< u, v >$. The algorithm first looks for pebbles for a node $u$. Once it finds 4 pebbles (2 in $u$ and 2 in $v$), we can cover the edge. If a pebble is found, the edges traversed to find the pebble are reversed and the pebble is moved along the path until it reaches $u$. If the edge is accepted as independent, then edge $< u, v >$ is inserted as a directed edge from $u$ to $v$ and a pebble is removed from $u$.

The edges are considered in the order presented. Any edges for which a valid pebble covering cannot be found is marked as redundant. Once all edges have been processed, the algorithm ends and determines if three pebbles remain. If this is true, the input graph is rigid, if not, the graph is flexible. Each free pebble left in the graph corresponds to a degree of freedom. The three degrees of freedom at the end of a game is required to specify the position of a rigid body in two dimensions, accounting for the trivial motions of two translations and a rotation [2].

## 2.2 3-D Pebble Game

The results can be extended to a 3-dimensional pebble game with respect to the body-and-bar model. However, in this case, $6n-6$ independent edges are needed to removed all nonrigid motions of the graph. The 3-dimensional pebble game runs very similarly to the 2-dimensional version. However, in this case, each vertex starts with six pebbles to account for the increased degrees of freedom of each vertex. Each vertex is allowed up to six adjacent edges to cover with its pebbles. The game ends once all edges are processed. The edges of a graph $G$ are independent in two dimensions if and only if no subgraph *G'* has more than $6n-6$ edges.

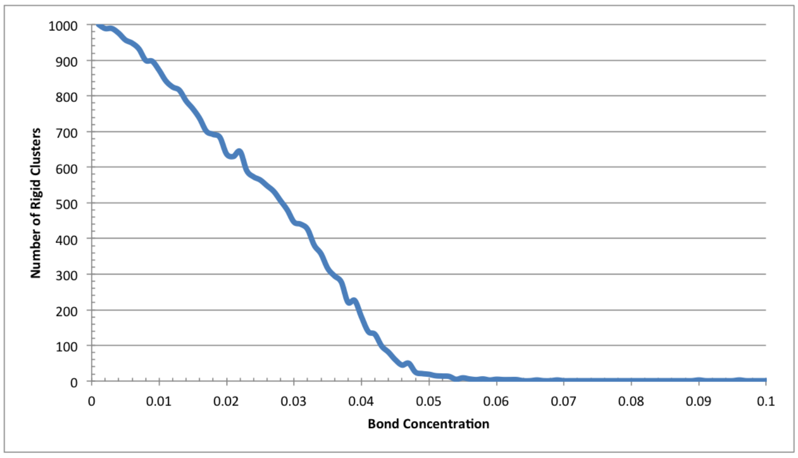

**Figure 3:** The number of rigid clusters as determined by the 2D pebble game. Decreases linearly as bond concentration increases until the graph becomes rigid at around .055.

## 3 Implementation

Enlarge cover attempts to find a pebble covering so that all the edges processed are covered. In its first iteration, a free pebble is attempted to be found. If possible, the algorithm 'quadruples' the edge in an attempt to find four pebbles between the two nodes. Helper functions *freePebble* and *movePebbles* attempt to find and move pebbles to their respective places.

Once all edges have been processed, the algorithm finds the rigid clusters of the graph. To do so, three pebbles need to be gathered at two adjacent nodes. Pin these pebbles down and create a new cluster. Examine all adjacent nodes and attempt to free a pebble. If a free pebble cannot be found, the site is mutually rigid with respect to the initial bond and should be included in the current rigid cluster. If a free pebble is found, this site should be marked floppy as it is not mutually rigid with respect to the initial bond. Continue to repeat this until all bond are given a cluster label.

In our implementation, we reuse the functions enlargeCover, findPebble, and movePebbles depending on the objective of the iteration. When quadrupling an edge, a 'testQuadrupledEdge' mode is turned on to ensure an edge is not redundant. When finding rigid clusters, a 'testRigid' mode is on so that pebbles can be pinned and edges can be marked as rigid or floppy.

## 4 Results

The algorithm runs in worst case time $O(nm)$. Each edge must be tested for independence, with worst case pebble covering search time of $O(n)$ for searching all vertices for a free pebble. In a sample file listing 1000 vertices and 101006 edges, the algorithm determines the graph to be rigid in 2.26 seconds. In more typical files with about 14000 edges, the algorithm runs in under half a second.

The objective of the experiments was to implement a powerful algorithm that could quickly identify rigid clusters of a protein. The data from the experiments were generated randomly, using 1000 vertices and bonds placed between vertices located within a certain threshold, $\lambda$. Results of experiments using the 2D and 3D version are pictured below.

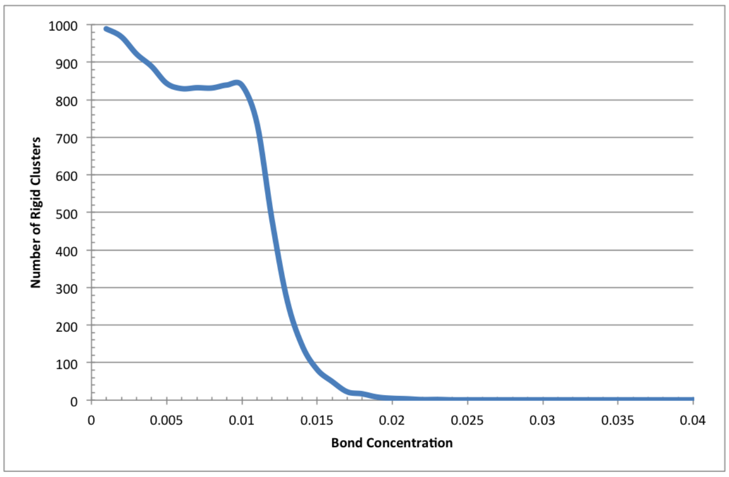

**Figure 4:** The number of rigid clusters as determined by the 3D pebble game. Sharp crystallizing point at around bond concentration of .013 when graph become rigid.

## 5 Conclusions

This algorithm can be used to accurately predict the rigidity of a large protein. It will filter data to find the independent connections in the network and also give the various edges and vertices contained in each rigid cluster. This code was implemented in C++ and has been implemented as an independent library which can easily be interfaced with other software packages.

The next step after this project is to integrate the implemented library with the CuikSuite [4].

## References

[1] H. Gluck. “Almost all simply connected closed surgaces are rigid,” in *Geometric Topology*, Lecture Notes in Mathematics, No. 438, pp. 225-239, 1975

[2] D. Jacobs and B. Hendrickson.“An Algorithm for Two-Dimensional Rigidity Percolation: The Pebble Game,” Journal of Computational Physics, 1997.

[3] G. Laman.“On graphs and rigidity of plane skeletal structures,” Journal of Engineering Mathematics, vol. 4, pp. 331, 1970.

[4] J. M. Porta, L. Ros, O. Bohigas, M. Manubens, C. Rosales, L. Jaillet. “The CUIK Suite: Analyzing the Motion Closed-Chain Multibody Systems,” IEEE Robotics & Automation Magazine, 21(3):105-114, 2014

[5] I. Streinu. “Kinary Project, KINARI Mutagen for kinematic and rigidity analysis of mutated proteins,” 2011.

## Acknowledgements

This work was supervised by Josep M. Porta.

## IRI reports