ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΜΕ ΕΦΑΡΜΟΓΕΣ ΣΤΗΝ ΒΙΟΪΑΤΡΙΚΗ

ΑΝΑΔΙΠΛΩΣΗ ΠΡΩΤΕΪΝΗΣ ΣΕ ΚΥΒΙΚΟ ΠΛΕΓΜΑ ΜΕ ΕΛΑΧΙΣΤΟΠΟΙΗΣΗ ΤΗΣ ΑΠΟΚΛΙΣΗΣ ΑΠΟ ΤΗΝ ΑΝΑΔΙΠΛΩΣΗ ΤΗΣ ΣΤΟ ΧΩΡΟ

Παπαδοπούλου Μαρία

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Επιβλέπων

Μάρκου Ευριπίδης

Λαμία, 2015

Τριμελής Επιτροπή:

Μάρκου Ευριπίδης, Επίκουρος Καθηγητής

Αδάμ Μαρία, Επίκουρος Καθηγήτρια

Μπάγκος Παντελής, Αναπληρωτής Καθηγητής

ΠΕΡΙΛΗΨΗ

Ο υπολογισμός (πρόβλεψη) της τρισδιάστατης δομής μιας πρωτεΐνης, όταν είναι γνωστή μόνο η ακολουθία των αμινοξέων που την αποτελούν, είναι ένα διάσημο πρόβλημα στην περιοχή της Υπολογιστικής Βιολογίας. Το πρόβλημα αυτό είναι γνωστό ως πρόβλημα της Αναδίπλωσης μιας Πρωτεΐνης (Protein Folding problem), και η επίλυσή του είναι δύσκολη (NP-hard) ακόμη και σε απλοποιημένα μοντέλα, με την έννοια ότι οι μόνοι γνωστοί αλγόριθμοι που το λύνουν απαιτούν εκθετικό χρόνο εκτέλεσης ως προς το πλήθος των αμινοξέων της πρωτεΐνης. Το πρόβλημα παραμένει δύσκολο (NP-hard) ακόμη και όταν αναζητούμε τη βέλτιστη αναδίπλωση μιας πρωτεΐνης σε δισδιάστατα πλέγματα (lattices).

Σε αυτή την εργασία μελετήσαμε το παρακάτω σχετικό πρόβλημα: Δίνεται η τρισδιάστατη αναδίπλωση μιας πρωτεΐνης (δηλαδή οι συντεταγμένες στο χώρο, των αμινοξέων που την αποτελούν), και θέλουμε να βρούμε μια βέλτιστη αναδίπλωση της πρωτεΐνης πάνω σε ένα τρισδιάστατο πλέγμα, τέτοια ώστε να είναι όσο το δυνατόν πιο ‘κοντά’ με την αναδίπλωσή της στο χώρο. Αυτό το πρόβλημα είναι γνωστό ως το πρόβλημα της Εμφύτευσης της Πρωτεΐνης σε Πλέγμα (Protein-Chain Fitting on Lattice (PCLF)). Η λύση του προβλήματος αυτού θα ήταν πολύ χρήσιμη για την επίλυση του προβλήματος Protein Folding, καθώς θα περιόριζε τα πλέγματα στα οποία αναζητούμε τη βέλτιστη αναδίπλωση μιας πρωτεΐνης. Το πρόβλημα PCLF έχει αποδειχθεί ότι είναι NP-hard για συγκεκριμένες οικογένειες τρισδιάστατων πλεγμάτων.

Εδώ ασχολούμαστε με την επίλυση του PCLF προβλήματος χρησιμοποιώντας δύο κριτήρια αξιολόγησης της βέλτιστης αναδίπλωσης: την ελαχιστοποίηση της απόκλισης των συντεταγμένων (Coordinate Root Mean Squared (CRMS) deviation), και την ελαχιστοποίηση της απόκλισης των αποστάσεων (Distance Root Mean Squared (DRMS) deviation) μεταξύ της αναδίπλωσης στο χώρο και στο πλέγμα. Σχεδιάζουμε και υλοποιούμε αλγόριθμους που παίρνουν ως είσοδο την τρισδιάστατη δομή μιας πρωτεΐνης και ένα συγκεκριμένο πλέγμα και επιστρέφουν τον ορισμό προβλημάτων Ακέραιου Προγραμματισμού. Στη συνέχεια λύνουμε τα προβλήματα ακέραιου προγραμματισμού (χρησιμοποιώντας ένα εμπορικό πρόγραμμα) και υπολογίζουμε βέλτιστες αναδιπλώσεις της πρωτεΐνης στο δεδομένο πλέγμα και για τα δύο κριτήρια. Εφαρμόζουμε τις παραπάνω τεχνικές σε συγκεκριμένες ομάδες πρωτεϊνών με κοινά μακροσκοπικά χαρακτηριστικά και μια σειρά από πλέγματα. Τα πρώτα αποτελέσματα μας δίνουν ενδείξεις για την εξάρτηση της εύρεσης ενός ‘καλού’ πλέγματος για κάποια πρωτεΐνη από τη χρήση των κριτηρίων CRMS ή DRMS.

Abstract

The computation (prediction) of the 3D structure of a protein, given its amino-acids sequence, is a well known and important problem in Computational Biology. This problem is known as the Protein Folding problem and it has been proved to be NP-hard even in simplified models, meaning that the existence of an efficient algorithm to solve it, is very unlikely. The problem remains NP-hard even considered on a 2D lattice.

We consider here the following closely related problem: given the 3D folding of a protein (i.e., coordinates of its atoms), find the 'closest' 3D lattice representation of this folding. This problem is known as the Protein Chain Lattice Fitting (PCLF) problem. A solution to this problem would indicate which lattice to test in the original Protein Folding problem. It has been shown that the PCLF problem is NP-hard for specific cubic lattices.

In this work we study the PCLF problem using two different criteria for the optimal (i.e., closest) representation: the minimization of the coordinates' deviation (Coordinate Root Mean Squared (CRMS) deviation), and the minimization of the distances' deviation (Distance Root Mean Squared (DRMS) deviation) between the two foldings. We design and implement algorithms which take as input the 3D structure of a protein along with a 3D lattice and return the definitions of Integer Programming problems. We next solve those problems (using a commercial software) computing the optimal foldings of the protein in the given lattice considering both criteria. We apply our techniques to specific protein data sets with similar macroscopic properties, considering a number of different lattices. Our experiments indicate the effect of CRMS and DRMS criteria on finding a 'good' lattice for a protein.

Περιεχόμενα

ΕΙΣΑΓΩΓΗ

Ένα πολύ σημαντικό πρόβλημα στη βιολογία με το οποίο ασχολούνται πολλοί ερευνητές είναι η πρόβλεψη της τρισδιάστατης απεικόνισης μιας πρωτεΐνης γνωρίζοντας την αλληλουχία των αμινοξέων που αποτελούν την πρωτεϊνική αλυσίδα.

Το παραπάνω πρόβλημα είναι γνωστό ως “Πρόβλημα Αναδίπλωσης Πρωτεΐνης” (Protein Folding Problem) και η λύση του είναι εξαιρετικά δύσκολη ακόμα και αν περιοριστούμε σε απλά μοντέλα όπου η απεικόνιση της πρωτεΐνης γίνεται πάνω σε δισδιάστατα ή τρισδιάστατα πλέγματα (lattices), και λαμβάνοντας υπόψιν μόνο κάποιες βασικές αλληλεπιδράσεις μεταξύ των αμινοξέων. Ένα τέτοιο απλό μοντέλο είναι το μοντέλο Hydrophobic-Polar (HP), στο οποίο λαμβάνονται υπόψιν μόνο δύο κατηγορίες αμινοξέων: τα υδρόφοβα και τα υδρόφιλα. Σε αυτήν την περίπτωση ο σκοπός είναι να βρεθεί το σχήμα της πρωτεΐνης πάνω σε ένα πλέγμα, έτσι ώστε ο αριθμός των υδρόφοβων αμινοξέων που δεν είναι γειτονικά στην πρωτεϊνική αλυσίδα και τοποθετούνται σε γειτονικούς κόμβους του πλέγματος να είναι ο μέγιστος δυνατός.

Δυστυχώς, ακόμη και για αυτό το απλοποιημένο μοντέλο, οι μόνοι γνωστοί βέλτιστοι αλγόριθμοι είναι μη-αποδοτικοί, δηλαδή η χρονική τους πολυπλοκότητα είναι μια εκθετική συνάρτηση του μήκους της πρωτεϊνικής αλυσίδας. Οι μόνοι γνωστοί αποδοτικοί αλγόριθμοι (δηλαδή με χρονική πολυπλοκότητα που είναι πολυωνυμική συνάρτηση του μήκους της πρωτεϊνικής αλυσίδας), καταφέρνουν να μας δώσουν ένα σχήμα της πρωτεΐνης στο πλέγμα που προσεγγίζει το μέγιστο πλήθος των γειτονικών υδρόφοβων αμινοξέων, ενώ έχει αποδειχθεί ότι δεν μπορούμε να πετύχουμε μεγάλη προσέγγιση της βέλτιστης λύσης σε πολυωνυμικό χρόνο. Ακόμη όμως και αν μπορούσαμε να βρούμε γρήγορα μια βέλτιστη αναδίπλωση της πρωτεΐνης πάνω σε ένα πλέγμα χρησιμοποιώντας είτε το μοντέλο HP είτε κάποιο άλλο μοντέλο, δεν θα ξέραμε αν το συγκεκριμένο πλέγμα αποδίδει μια απεικόνιση της πρωτεΐνης που είναι ‘κοντά’ στην απεικόνισή της στο χώρο.

Μια διαδικασία για την επιλογή ενός ‘καλού’ πλέγματος που έχει προταθεί και μελετάται τα τελευταία χρόνια είναι η εξής:

Διαλέγουμε ένα σύνολο πρωτεϊνών για τις οποίες ξέρουμε τις συντεταγμένες των αμινοξέων τους στο χώρο και ένα σύνολο από πλέγματα πάνω στα οποία θέλουμε να τοποθετήσουμε τις πρωτεΐνες. Στη συνέχεια τοποθετούμε κάθε πρωτεΐνη σε όλα τα πλέγματα έτσι ώστε η αναδίπλωση της πρωτεΐνης πάνω στο πλέγμα να είναι όσο το δυνατόν πιο 'κοντά' με το σχήμα της στο χώρο. Τέλος επιλέγουμε για κάθε πρωτεΐνη εκείνο το πλέγμα που μας δίνει κατά μέσο όρο την 'καλύτερη' εικόνα της.

Η αξιολόγηση της απεικόνισης μιας πρωτεΐνης πάνω σε ένα πλέγμα σε σχέση με το σχήμα της στο χώρο, συνήθως γίνεται χρησιμοποιώντας ως κριτήριο την ελαχιστοποίηση είτε της μέσης τετραγωνικής απόκλισης των συντεταγμένων (Coordinate Root Mean Squared Deviation -CRMS) είτε της μέσης τετραγωνικής απόκλισης των αποστάσεων (Distance Root Mean Squared Deviation -DRMS).

Το παραπάνω πρόβλημα της εύρεσης της καλύτερης εικόνας μιας πρωτεΐνης πάνω σε ένα συγκεκριμένο πλέγμα αναφέρεται στη βιβλιογραφία ως "Πρόβλημα Αναδίπλωσης Πρωτεΐνης Πάνω Σε Πλέγμα" (Protein Chain Lattice Fitting Problem (PCLF)) και πολλοί ερευνητές έχουν ασχοληθεί με την επίλυσή του όπως θα δούμε σε επόμενο κεφάλαιο.

Σε αυτή την εργασία ασχολούμαστε με την επίλυση του PCLF προβλήματος χρησιμοποιώντας τα κριτήρια CRMS και DRMS. Σχεδιάζουμε και υλοποιούμε αλγόριθμους που υπολογίζουν τη βέλτιστη εικόνα μιας πρωτεΐνης πάνω σε ένα πλέγμα. Τέλος εφαρμόζουμε τους αλγόριθμους που υλοποιήσαμε σε μια σειρά από πρωτεΐνες και πλέγματα και παίρνουμε τις πρώτες ενδείξεις για το κατά πόσον η εύρεση ενός 'καλού' πλέγματος για κάποια πρωτεΐνη εξαρτάται από τη χρήση του κριτηρίου CRMS ή του DRMS.

ΚΕΦΑΛΑΙΟ 1.

ΘΕΩΡΗΤΙΚΟ ΥΠΟΒΑΘΡΟ

Στο εισαγωγικό αυτό κεφάλαιο θα αναλύσουμε κάποιες βασικές έννοιες που βοηθάνε στην κατανόηση του προβλήματος, αλλά και των εργαλείων που χρησιμοποιήθηκαν στην παρούσα πτυχιακή εργασία.

1.1 ΕΙΣΑΓΩΓΗ ΣΤΗ ΒΙΟΛΟΓΙΑ

Αρχικά, ας δούμε κάποιες ευρέως γνωστές έννοιες για τη χημεία των μορίων και την δομή των πρωτεϊνών [1].

1.1.1 ΤΑ ΑΤΟΜΑ

Το μικρότερο σωματίδιο ενός στοιχείου που εξακολουθεί να διατηρεί τις χαρακτηριστικές χημικές ιδιότητές του είναι γνωστό ως άτομο (Εικόνα 1.1) .

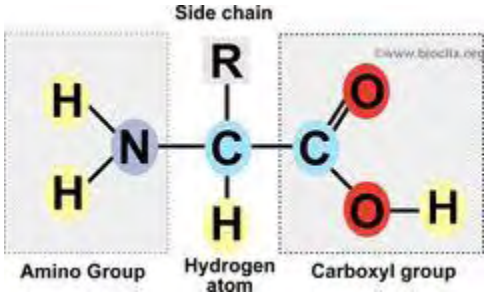
Εικόνα 1.1 : Άτομο

1.1.2 ΤΑ ΜΟΡΙΑ

Ένα μόριο είναι ένα άθροισμα ατόμων που συγκρατώνται μεταξύ τους με ομοιοπολικούς δεσμούς, οι οποίοι προκύπτουν από τη συνεισφορά και όχι τη μεταφορά ηλεκτρονίων ανάμεσα σε άτομα. Οι χημικοί αυτοί δεσμοί μπορεί να είναι ομοιοπολικοί δεσμοί, ετεροπολικοί δεσμοί, δεσμοί υδρογόνου, δυνάμεις Van der Waals κλπ. Τα κύτταρα περιέχουν τέσσερις κύριες οικογένειες μικρών οργανικών μορίων: τα σάκχαρα, τα λιπαρά οξέα, τα νουκλεοτίδια και τα αμινοξέα.

1.1.3 ΤΑ ΑΜΙΝΟΞΕΑ

Τα αμινοξέα είναι τα δομικά στοιχεία των πρωτεϊνών. Είναι ομάδα μορίων με μεγάλη ποικιλία, της οποίας τα μέλη έχουν μια καθοριστική ιδιότητα. Πιο συγκεκριμένα όλα περιέχουν μια όξινη καρβοξυλομάδα (carboxyl group) και μια αμινομάδα (amino group) οι οποίες συνδέονται με το ίδιο άτομο άνθρακα, που αποκαλείται α-άνθρακας. Η χημική ποικιλία προκύπτει από την πλευρική αλυσίδα (Side chain) που επίσης συνδέεται με τον α-άνθρακα (Εικόνα 1.2).



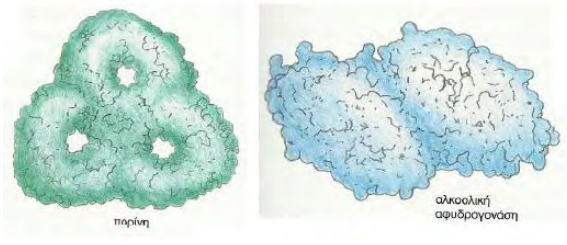
Εικόνα 1.2: Δομή αμινοξέος

Στις πρωτεΐνες συνήθως υπάρχουν 20 είδη αμινοξέων, το καθένα με μια διαφορετική πλευρική αλυσίδα συνδεδεμένη με το α-άτομο άνθρακα. Η χημική

ευελιξία που παρέχουν αυτά τα 20 αμινοξέα είναι ζωτικής σημασίας για τη λειτουργία των πρωτεϊνών. Πέντε από τα 20 αμινοξέα έχουν πλευρικές αλυσίδες ικανές να σχηματίσουν ιόντα όταν βρεθούν σε διάλυμα και επομένως έχουν φορτίο. Τα άλλα είναι μη φορτισμένα. Μερικά είναι πολικά και υδρόφιλα και άλλα μη πολικά και υδρόφοβα. Οι συλλογικές ιδιότητες των πλευρικών αλυσίδων των αμινοξέων ευθύνονται για όλες τις ποικίλες και σύνθετες λειτουργίες των πρωτεϊνών.

1.1.4 ΟΙ ΠΡΩΤΕΪΝΕΣ

Οι πρωτεΐνες είναι οι δομικοί λίθοι από τους οποίους συγκροτούνται τα κύτταρα και εμφανίζουν την μεγαλύτερη δομική και λειτουργική πολυπλοκότητα από όλα τα γνωστά μόρια. Οι πρωτεΐνες είναι πολυμερή αμινοξέων που συνδέονται το ένα με το άλλο σε μια μακριά αλυσίδα, η οποία ονομάζεται πολυπεπτιδική αλυσίδα. Το μέγεθος των πρωτεϊνών κυμαίνεται περίπου από 30 έως 10000 και πλέον αμινοξέα. Ωστόσο, στη μεγάλη πλειονότητά τους, οι πρωτεΐνες έχουν 50-2000 αμινοξέα. Οι πρωτεΐνες μπορεί να είναι σφαιρικές ή ινώδεις, να σχηματίζουν ινίδια, φύλλα, δακτύλιους ή σφαίρες (Εικόνα 1.3) .



Εικόνα 1.3 : Παραδείγματα 2 πρωτεϊνικών μορίων

ΠΤΥΧΩΣΗ ΠΡΩΤΕΙΝΩΝ

Κάθε είδος πρωτεΐνης έχει μια ιδιαίτερη τρισδιάστατη δομή, η οποία καθορίζεται από τη σειρά των αμινοξέων στην αλυσίδα της. Η τελική πτυχωμένη δομή, την οποία υιοθετεί μια πολυπεπτιδική αλυσίδα, καθορίζεται από ενεργειακές παραμέτρους και γενικά είναι εκείνη στην οποία η ελεύθερη ενέργεια είναι ελαχιστοποιημένη.

Μολονότι η συνολική διαμόρφωση κάθε πρωτεΐνης είναι μοναδική, δύο συνήθη πρότυπα πτύχωσης βρίσκονται συχνά σε ορισμένα τμήματα των πρωτεϊνικών αλυσίδων, η α-έλικα και το β-πτυχωτό φύλλο.

ΕΠΙΠΕΔΑ ΟΡΓΑΝΩΣΗΣ ΠΡΩΤΕΙΝΩΝ

Στη δομή των πρωτεϊνών υπάρχουν επίπεδα οργάνωσης. Τα επίπεδα αυτά δεν είναι ανεξάρτητα αλλά αποτελούν προέκταση το ένα του άλλου έως ότου καθοριστεί πλήρως η τρισδιάστατη δομή ολόκληρης της πρωτεΐνης και είναι τα εξής:

Πρωτοταγής Δομή: Η αλληλουχία των αμινοξέων τα οποία σχηματίζουν την πολυπεπτιδική αλυσίδα.

Δευτεροταγής Δομή: Τα τμήματα της πολυπεπτιδικής αλυσίδας που σχηματίζουν α-έλικες και β-πτυχωτά φύλλα.

Τριτοταγής Δομή: Η πλήρης τρισδιάστατη διαμόρφωση που σχηματίζεται από ολόκληρη την πολυπεπτιδική αλυσίδα.

Τεταρτοταγής Δομή: Η πλήρης δομή μιας πρωτεΐνης αν σχηματίζεται ως σύμπλοκο δύο ή περισσότερων πολυπεπτιδίων.

ΜΕΤΟΥΣΙΩΣΗ ΠΡΩΤΕΙΝΩΝ

Μία πρωτεΐνη μπορεί να ξεδιπλωθεί ή, όπως λέμε να μετουσιωθεί με την επίδραση ορισμένων διαλυτών που διασπούν τις μη ομοιοπολικές αλληλεπιδράσεις, οι οποίες συγκρατούν τη δομή της διπλωμένης αλυσίδας. Η μετουσίωση μετατρέπει την πρωτεΐνη σε μια εύκαμπτη πολυπεπτιδική αλυσίδα που έχει χάσει το φυσικό της σχήμα.

Όλες οι πληροφορίες που απαιτούνται για την πτύχωση μιας πολυπεπτιδικής αλυσίδας περιέχονται στην αλληλουχία των αμινοξέων της. Ωστόσο, δεν έχουμε ακόμα μάθει πώς να αξιοποιούμε αυτές τις πληροφορίες για να προβλέψουμε αυτή τη λεπτομερή τρισδιάστατη διαμόρφωση μιας πρωτεΐνης, δηλαδή τη διάταξη όλων των ατόμων της στο χώρο, από την αλληλουχία των αμινοξέων της. Προς το παρόν η ακριβής διαμόρφωση οποιασδήποτε πρωτεΐνης μπορεί να ανακαλυφθεί μόνο πειραματικά, χρησιμοποιώντας την τεχνική της κρυσταλλογραφίας με ακτίνες Χ ή τη μέθοδο του πυρηνικού μαγνητικού συντονισμού. Έως τώρα, με τις τεχνικές αυτές έχουν αναλυθεί πλήρως πάνω από 10000 πρωτεΐνες.

1.2 Η PDB (PROTEIN DATA BANK)

Για τις ανάγκες της παρούσας πτυχιακής εργασίας συλλέχθηκαν δεδομένα από τη γνωστή πρωτεϊνική βάση δεδομένων PDB (Protein Data Bank) για την οποία θα μιλήσουμε παρακάτω.

1.2.1 ΣΧΕΤΙΚΑ ΜΕ ΤΗΝ PDB (Protein Data Bank)

Η Protein Data Bank είναι μια διαδικτυακή βάση δεδομένων, τα αρχεία της οποίας περιέχουν πληροφορίες για τις τρισδιάστατες δομές μεγάλων βιολογικών μορίων. Η PDB ιδρύθηκε το 1971 στο Εθνικό Εργαστήριο του Brookhaven (Brookhaven National Laboratory) από τον Walter Hamilton και αρχικά περιείχε 7 δομές. Το 1998 υπεύθυνη για την διαχείριση της PDB έγινε η Research Collaboratory for Structural Bioinformatics (RCSB) που αφορά την έρευνα στην δομική Βιοπληροφορική. Το 2003 η ιστοσελίδα wwPDB δημιουργήθηκε για να διατηρεί ένα ενιαίο αρχείο των μακρομιριακών δομών ελεύθερα διαθέσιμο παγκοσμίως. Σήμερα η RCSB PDB περιέχει 105.339 βιολογικές δομές μακρομορίων οι περισσότερες από τις οποίες αφορούν πρωτεΐνες. Επιπλέον, η RCSB PDB στηρίζει μια ιστοσελίδα όπου οι επισκέπτες μπορούν να εκτελέσουν απλά και σύνθετα ερωτήματα σχετικά με τα δεδομένα, καθώς επίσης να αναλύσουν και να οπτικοποιήσουν τα αποτελέσματα [2].

Στην παρούσα πτυχιακή εργασία χρησιμοποιήθηκαν δεδομένα από αρχεία της PDB που αφορούν πρωτεΐνες που έχουν αναλυθεί με την μέθοδο της κρυσταλλογραφίας ακτινών Χ, οι οποίες έχουν καταταχθεί ανάλογα με την πτύχωσή τους στις εξής κατηγορίες :

- Alpha proteins: Πρωτεΐνες των οποίων η δομή σχηματίζεται κυρίως από α-έλικες.
- Beta proteins: Πρωτεΐνες των οποίων η δομή σχηματίζεται κυρίως από

β-φύλλα.

- Alpha and Beta proteins: Πρωτεΐνες των οποίων η δομή σχηματίζεται και από α-έλικες και από β-φύλλα.

- Άλλες: Πρωτεΐνες που δεν έχουν κατηγοριοποιηθεί σε κάποια από τις παραπάνω ομάδες.

Πιο συγκεκριμένα επιλέχθηκαν 80 πρωτεΐνες, 20 από κάθε κατηγορία.

1.3 ΒΑΣΙΚΕΣ ΕΝΝΟΙΕΣ ΑΠΟ ΤΗ ΘΕΩΡΙΑ ΓΡΑΦΗΜΑΤΩΝ

Στη συνέχεια αναφέρονται κάποιες γνωστές έννοιες από την θεωρία γραφημάτων καθώς επίσης και για τα πλέγματα και τα ιδιαίτερα χαρακτηριστικά τους [3] .

1.3.1 ΓΡΑΦΗΜΑΤΑ

Γράφος ή γράφημα είναι μια δομή που αποτελείται από ένα σύνολο κορυφών ή κόμβων ή σημείων που συνδέονται μεταξύ τους με ένα σύνολο ακμών ή γραμμών. Γενικά ένας γράφος συμβολίζεται ως G(V,E) ή G=(V,E) ή (V(G),E(G)), όπου V και E είναι τα σύνολα των κορυφών και των ακμών αντίστοιχα. Το πλήθος των κορυφών ενός γράφου συμβολίζεται με n=|V| και ονομάζεται τάξη του γράφου, ενώ το πλήθος των ακμών του γράφου συμβολίζεται με m=|E| και ονομάζεται μέγεθος του γράφου.

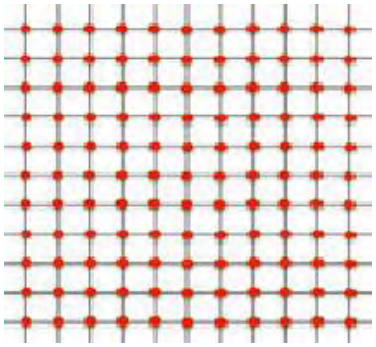
Κάθε ακμή προσδιορίζεται από δύο κορυφές που ονομάζονται τερματικά σημεία. Αν η ακμή e έχει τα u,v ως τερματικά σημεία, τότε η ακμή e ονομάζεται προσπίπτουσα στα σημεία u,v ή λέγεται ότι η e ενώνει τα σημεία u,v και συμβολίζεται με (u,v) ή (v,u). Αντίστοιχα ορίζεται ότι το σημείο u είναι γειτονικό του v και αντίστροφα.

Η γειτονιά μιας κορυφής v , N(v), είναι το σύνολο που ορίζεται από τη σχέση:

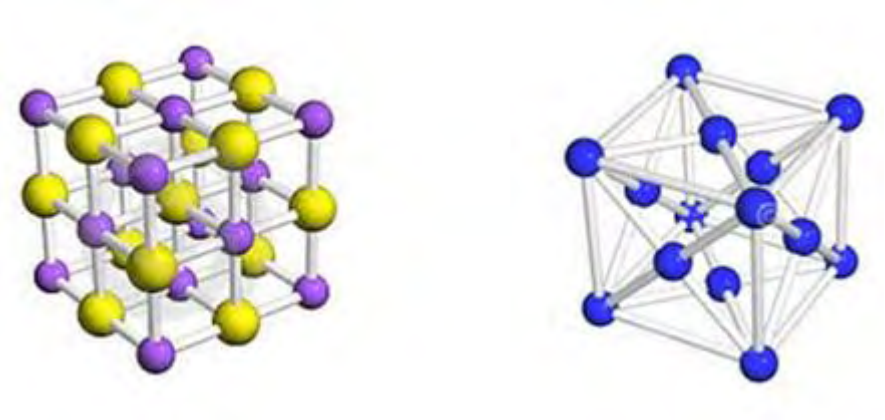
$N(v) = \{ u \in V(G) | (v,u) \in E(G) \}$

1.3.2 ΠΛΕΓΜΑ (LATTICE)

Τα πλέγματα είναι μια ειδική κατηγορία γραφημάτων που χρησιμοποιήσαμε στην παρούσα πτυχιακή εργασία. Τα πλέγματα μπορεί να είναι δισδιάστατα (Εικόνα 1.4), τρισδιάστατα (Εικόνα 1.5) ή ν-διάστατα.



Εικόνα 1.4: Παράδειγμα δισδιάστατου πλέγματος



Εικόνα 1.5: Παραδείγματα τρισδιάστατων πλεγμάτων

Σε ένα δισδιάστατο πλέγμα nxm κάθε κορυφή ορίζεται από ένα ζεύγος συντεταγμένων (u_x , u_y), όπου $1 \leq x \leq n$, $1 \leq y \leq m$. Αναλόγως γίνεται η γενίκευση στις τρεις διαστάσεις.

Στην παρούσα πτυχιακή εργασία χρησιμοποιήσαμε ένα σύνολο από τρισδιάστατα πλέγματα με διαφορετικά μήκη ακμών αλλά με το κοινό χαρακτηριστικό ότι όλα τα ζεύγη ακμών σχηματίζουν γωνίες 90^o.

1.4 ΜΑΘΗΜΑΤΙΚΟΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ

Το PCLF πρόβλημα μπορεί να διατυπωθεί χρησιμοποιώντας Μαθηματικό Προγραμματισμό. Πριν δώσουμε όμως μια τέτοια περιγραφή του προβλήματος, για λόγους πληρότητας της εργασίας και ευκολίας του αναγνώστη, θα κάνουμε σε αυτή την ενότητα μία σύντομη εισαγωγή στο Μαθηματικό Προγραμματισμό.

Ο Μαθηματικός Προγραμματισμός χρησιμοποιείται για να προσδιορισθεί η καλύτερη ή η άριστη λύση σ' ένα πρόβλημα που απαιτεί μία απόφαση ή ένα σύνολο αποφάσεων σχετικά με τη χρησιμοποίηση του συνόλου των περιορισμένων πόρων για την επίτευξη ενός αντικειμενικού στόχου.

Βήματα που απαρτίζουν το Μαθηματικό Προγραμματισμό:

- Μετατροπή ενός στατικού προβλήματος σε μαθηματικό μοντέλο που περιλαμβάνει όλα τα απαραίτητα στοιχεία του προβλήματος.
- Διερεύνηση των διαφορετικών λύσεων του προβλήματος.
- Εύρεση της πιο κατάλληλης ή της άριστης λύσης.[6]

Το γενικό πρόβλημα του μαθηματικού προγραμματισμού μπορεί να διατυπωθεί ως εξής:

Να βρεθούν οι τιμές των μεταβλητών $x_1,x_2,\ldots,x_n$ που ικανοποιούν m ανισότητες (ή ισότητες):

$$g_i(x_1,\ldots,x_n) \{\leq, =, \geq\} b_i \text{ , } i=1,2,\ldots,m \quad (1.4.1)$$

και που επιπλέον ελαχιστοποιούν ή μεγιστοποιούν τη συνάρτηση:

$$z= f(x_1,\ldots,x_n) \quad (1.4.2)$$

Οι συνθήκες (1.4.1) λέγονται περιορισμοί, η συνάρτηση (1.4.2) ονομάζεται αντικειμενική συνάρτηση, οι μεταβλητές x_i καλούνται μεταβλητές απόφασης, οι σταθερές b_i θεωρούνται γνωστές και για κάθε i ισχύει η ανισότητα ή η ισότητα. Κάθε $\underline{x} \in R^n$, $\underline{x} = (x_1,\ldots,x_n)^T$ που ικανοποιεί τους περιορισμούς καλείται εφικτή λύση του προβλήματος και κάθε εφικτή λύση που βελτιστοποιεί (μεγιστοποιεί ή ελαχιστοποιεί) την αντικειμενική συνάρτηση f, ονομάζεται βέλτιστη εφικτή λύση.

Το πρόβλημα μαθηματικού προγραμματισμού είναι κατά συνέπεια ένα πρόβλημα μεγιστοποίησης ή ελαχιστοποίησης. Ανάλογα με τα ιδιαιτέρα χαρακτηριστικά των συναρτήσεων g_i των περιορισμών (1.4.1) ή της αντικειμενικής συνάρτησης f (1.4.2), ο μαθηματικός προγραμματισμός μπορεί να χαρακτηρισθεί σαν ακέραιος, κυρτός, τετραγωνικός, δυναμικός, στοχαστικός, γραμμικός κλπ. [7]

Πιο συγκεκριμένα το PCLF πρόβλημα έχοντας ως κριτήριο το CRMS μπορεί να διατυπωθεί χρησιμοποιώντας γραμμικό προγραμματισμό, ενώ έχοντας ως κριτήριο το DRMS μπορεί να διατυπωθεί χρησιμοποιώντας μη γραμμικό προγραμματισμό.

1.4.1 ΓΡΑΜΜΙΚΟΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ

Στο υποκεφάλαιο αυτό θα κάνουμε μια σύντομη εισαγωγή στον Γραμμικό προγραμματισμό, μια από τις κατηγορίες του Μαθηματικού Προγραμματισμού.

Το μοντέλο του γραμμικού προγραμματισμού αποτελείται από γραμμικές συναρτήσεις και περιορισμούς, γεγονός που σημαίνει ότι οι μεταβλητές του μοντέλου έχουν μεταξύ τους αναλογικές σχέσεις. [6]

Το γενικό πρόβλημα του γραμμικού προγραμματισμού έχει τη μορφή:

$$z = \{\max, \min\} (c_1x_1 + c_2x_2 + \dots + c_nx_n)$$
$$a_{11} x_1 + a_{12} x_2 + \dots + a_{1n} x_n \leq, =, \geq b_1$$
$$a_{21} x_1 + a_{22} x_2 + \dots + a_{2n} x_n \leq, =, \geq b_2$$
$$\dots.$$
$$\dots.$$
$$a_{m1} x_1 + a_{m2} x_2 + \dots + a_{mn} x_n \leq, =, \geq b_m$$
$$x_1 , x_2 , \dots , x_n \geq 0$$

Αν όλοι οι περιορισμοί είναι εξισώσεις ή ανισώσεις της ίδιας φοράς, το παραπάνω πρόβλημα γραμμικού προγραμματισμού μπορεί να γραφεί πιο συνοπτικά με μορφή πινάκων:

$$z=\{\max,\min\} \underline{c}^T\underline{x}^T$$
$$A\underline{x} \{\leq, =, \geq\} \underline{b}$$
$$\underline{x} \geq \underline{0}$$

όπου

$$x=(x_1,\dots,x_n)^T , c=(c_1,\dots,c_n)^T , b=(b_1,\dots,b_m)^T , 0=(0,\dots,0)^T \quad \epsilon R^m$$

και

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}$$

[7]

Στη συνέχεια ακολουθεί ένα παράδειγμα περιγραφής ενός προβλήματος Γραμμικού Προγραμματισμού [8].

Παράδειγμα:

Να βρεθούν οι τιμές των y_1, y_2 που ελαχιστοποιούν τη συνάρτηση

$y_1 + y_2$

ενώ ισχύουν οι περιορισμοί:

$y_1 + 2y_2 \geq 3$

$2y_1 + y_2 \geq 5$

$y_2 \geq 0$

Στο παραπάνω πρόβλημα υπάρχουν δύο μεταβλητές και τρεις γραμμικοί περιορισμοί για αυτές. Ο τρίτος περιορισμός $y_2 \geq 0$ ανήκει σε μια ξεχωριστή κατηγορία περιορισμών που συχνά συναντάμε σε προβλήματα Γραμμικού Προγραμματισμού και λέγονται ‘ μη αρνητικοί περιορισμοί ’. Οι υπόλοιποι δύο περιορισμοί λέγονται βασικοί περιορισμοί. Η συνάρτηση που ζητάμε να ελαχιστοποιηθεί λέγεται αντικειμενική και είναι το άθροισμα $y_1 + y_2$.

Ένα μοντέλο γραμμικού προγραμματισμού, στο οποίο όλες οι μεταβλητές απόφασης πρέπει να έχουν ακέραιες τιμές ονομάζεται πρόβλημα ακέραιου προγραμματισμού.

Ένα πρόβλημα στο οποίο μόνο μερικές από τις μεταβλητές απόφασης πρέπει να έχουν ακέραιες τιμές, ονομάζεται πρόβλημα μικτού ακέραιου προγραμματισμού.

Όταν οι μεταβλητές απόφασης πρέπει να έχουν τιμή 0 ή 1, τότε τα προβλήματα ονομάζονται προβλήματα μηδέν-ένα ακέραιου προγραμματισμού.

Το PCLF πρόβλημα έχοντας ως κριτήριο το CRMS μπορεί να οριστεί ως πρόβλημα μηδέν - ένα ακέραιου Γραμμικού Προγραμματισμού.

1.4.2 ΜΗ ΓΡΑΜΜΙΚΟΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ

Στο υποκεφάλαιο αυτό θα κάνουμε μια σύντομη εισαγωγή στον Μη Γραμμικό προγραμματισμό, μια από τις κατηγορίες του Μαθηματικού Προγραμματισμού.

Στον Μη Γραμμικό Προγραμματισμού ασχολούμαστε με προβλήματα βελτιστοποίησης όπου οι αντικειμενικές συναρτήσεις ή/και οι περιορισμοί δεν είναι γραμμικές συναρτήσεις.

Κατηγορίες προβλημάτων Μ.Γ.Π. :

- Προβλήματα ΚΛΑΣΜΑΤΙΚΟΥ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ, όπου η αντικειμενική συνάρτηση είναι κλάσμα.
- Προβλήματα ΔΙΑΧΩΡΙΣΙΜΟΥ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ, όπως για παράδειγμα προβλήματα μεταφοράς προϊόντων, όπου το κόστος μεταφοράς ανά μονάδα προϊόντος δεν είναι σταθερό, αλλά μειώνεται όσο αυξάνεται η ποσότητα του μεταφερόμενου προϊόντος.
- Προβλήματα ΤΕΤΡΑΓΩΝΙΚΟΥ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ, όπου οι περιορισμοί είναι γραμμικοί και η αντικειμενική συνάρτηση είναι πολυώνυμο δευτέρου βαθμού. [9]

Το γενικό πρόβλημα του μη γραμμικού προγραμματισμού έχει τη μορφή:

$$\begin{aligned} \max/\min \quad & f(x1, \dots, xn) \\ \text{s.t.} \quad & g_1(x1, \dots, xn) \leq 0 \\ & \dots \\ & g_m(x1, \dots, xn) \leq 0 \\ & x \in \mathbb{R}^{n_1} \times \mathbb{Z}^{n_2} \end{aligned}$$

όπου,

$f, g1, \ldots, gm : \mathbb{R}^n \rightarrow \mathbb{R}$
[10]

Στη συνέχεια ακολουθεί ένα παράδειγμα περιγραφής ενός προβλήματος Μη Γραμμικού Προγραμματισμού [11].

Παράδειγμα:

Να βρεθούν οι τιμές των x_1, x_2 που μεγιστοποιούν τη συνάρτηση

$126x_1 - 9{x_1}^2 + 128x_2 - 13{x_2}^2$

ενώ ισχύουν οι περιορισμοί:

$x_1 \leq 4$

$2x_2 \leq 12$

$3x_1 + 2x_2 \leq 18$

$x_1, x_2 \geq 0$

Το PCLF πρόβλημα έχοντας ως κριτήριο το DRMS μπορεί να οριστεί ως πρόβλημα μηδέν - ένα ακέραιου Τετραγωνικού Προγραμματισμού.

ΚΕΦΑΛΑΙΟ 2.

ΣΧΕΤΙΚΗ ΠΡΟΗΓΟΥΜΕΝΗ ΕΡΕΥΝΑ

Το PCLF πρόβλημα κατατάσσεται σε μια περιοχή της υπολογιστικής βιολογίας, της οποίας ένα από τα πιο διάσημα σχετικά προβλήματα είναι το Protein Folding Problem, με το οποίο έχουν ασχοληθεί πολλοί ερευνητές. Σε αυτό το κεφάλαιο θα αναφερθούμε σε προηγούμενη έρευνα που σχετίζεται με το PCLF πρόβλημα.

Μία πρώτη απόπειρα να λυθεί το Protein Folding Problem έγινε το 1990 από τον Dill [12,37]. Ο Dill πρότεινε ένα μοντέλο που βασιζότανε στις υδροφοβικές αλληλεπιδράσεις που καθορίζουν την αναδίπλωση των πρωτεϊνικών αλυσίδων. Για το λόγο αυτό το μοντέλο του ονομάστηκε υδροφοβικό – πολικό μοντέλο (Hydrophobic – Polar (HP) Model). Το μοντέλο του Dill εφαρμόστηκε σε δισδιάστατα τετραγωνικά πλέγματα, όπου διαδοχικά αμινοξέα τοποθετήθηκαν σε διαδοχικές κορυφές των πλεγμάτων. Το κριτήριο της τοποθέτησης ήταν η μεγιστοποίηση των μη-γειτονικών υδρόφοβων αμινοξέων. Ο σκοπός της τοποθέτησης ήταν η τελική απεικόνιση στο πλέγμα να προβλέπει καλά την φυσική αναδίπλωση της αλυσίδας των αμινοξέων και άρα την τρισδιάστατη μορφή της πρωτεΐνης.

Στην έρευνά τους [13], οι A.Gupta, J.Manuch, L.Stacho σχεδίασαν θεωρητικές δισδιάστατες πρωτεΐνες και χρησιμοποιώντας το HP-Model, τοποθέτησαν τις αλυσίδες αμινοξέων πάνω σε δισδιάστατα πλέγματα.

Στην έρευνά τους [14], οι D.G. Covell και R.L. Jernigan, προτείνουν έναν αλγόριθμο ο οποίος απαριθμεί όλες τις δυνατές αναπαραστάσεις και επιλέγει την καλύτερη. Πιο συγκεκριμένα, χρησιμοποίησαν ένα σύνολο από κορυφές ενός πλέγματος, κάθε μια από τις οποίες αντιστοιχούσε σε ένα αμινοξύ της πρωτεΐνης. Στη συνέχεια μελέτησαν όλες τις πιθανές αναδιπλώσεις της πρωτεΐνης πάνω στις κορυφές αυτές του πλέγματος. Κάθε μια από αυτές απαριθμήθηκε με βάση το σύνολο των μη γειτονικών αμινοξέων που έρχονται σε επαφή κατά την τοποθέτηση. Η αναδίπλωση

που είχε τον μεγαλύτερο αριθμό μη γειτονικών αμινοξέων θεωρήθηκε η βέλτιστη, καθώς η δομή είχε τη μεγαλύτερη σταθερότητα. Αξίζει να σημειωθεί ότι ανάμεσα στις λύσεις υπήρχε πάντα η βέλτιστη αναδίπλωση.

Στην έρευνά τους [15], οι A. Godzik, A. Kolinski, και J. Skolnick, πρότειναν μια νέα μέθοδο στην οποία χωρίζουμε την πρωτεΐνη σε κομμάτια και μελετάμε το ελάχιστο CRMS κάθε τμήματος ξεχωριστά. Πιο συγκεκριμένα στην μέθοδο αυτή ξεκινάμε από το πρώτο αμινοξύ της πρωτεΐνης, το οποίο τοποθετείται στην κατάλληλη κορυφή του πλέγματος, έτσι ώστε το CRMS μιας μικρής περιοχής της ολικής απεικόνισης της πρωτεΐνης πάνω στο πλέγμα να ελαχιστοποιείται. Η κεντρική ιδέα της μεθόδου αυτής ήταν ότι η ολική βέλτιστη λύση μπορεί να προκύψει από μικρότερα τοπικά βέλτιστα.

Στην έρευνά τους [16], οι D.A. Hinds και M.Levitt, ανέπτυξαν μια μέθοδο της αναπαράστασης της πολυπεπτιδικής αλυσίδας, με την οποία μπορούμε να απαριθμήσουμε όλες τις πιθανές διαμορφώσεις των μικρών πρωτεϊνών. Στη συγκεκριμένη έρευνα επιλέχθηκαν πέντε μικρές πρωτεΐνες όπου η κάθε πρωτεΐνη αντιπροσωπεύεται από μια αναπόφευκτη διαδρομή πάνω σε τετραεδρικά πλέγματα. Το χαρακτηριστικό του μοντέλου αυτού είναι ότι δεν επιβάλλεται η ένα προς ένα αντιστοιχία μεταξύ των αμινοξέων και των κορυφών του πλέγματος. Παρόλο που με αυτή τη μέθοδο δεν είναι δυνατή η ακριβής απεικόνιση της πρωτεΐνης, βρέθηκε ότι ένα από αυτά τα μονοπάτια είναι επαρκώς ευέλικτο για να συμπεριλάβει το εύρος των πιθανών τοπολογιών ενός πολυπεπτιδικού σκελετού. Αυτή η μέθοδος χαρακτηρίζεται από γενικότητα και προβλεπτική ικανότητα καθώς δεν χρησιμοποιήθηκαν δομικά και ενεργειακά κριτήρια και δεν είναι απαραίτητη η εκ των προτέρων γνώση των δομικών λεπτομερειών.

Στην έρευνά [17], του ο Xiarong Huang όρισε το FPL πρόβλημα με τρεις διαφορετικούς τρόπους χρησιμοποιώντας ακέραιο προγραμματισμό και σχεδίασε αλγορίθμους που συνδυάζουν δυναμικό προγραμματισμό και επαναλαμβανόμενες τεχνικές με στόχο να μειώσει τον χώρο αναζήτησης της καλύτερης αναδίπλωσης εξασφαλίζοντας ταυτόχρονα τον εντοπισμό των βέλτιστων λύσεων. Τα πειράματα του σε μια τυχαία επιλεγμένη ομάδα πρωτεϊνών, έδειξαν ότι η βέλτιστη προσέγγιση σε κυβικά πλέγματα με μήκος πλευράς 3,8Å χρησιμοποιώντας ως κριτήριο το CRMS μπορεί να βρεθεί σε εφικτό χρονικό διάστημα για όλες τις πρωτεΐνες.

Στην έρευνά τους [18], οι B. H. Park and M. Levitt, πρότειναν έναν άπληστο αλγόριθμο που έδινε μια αναδίπλωση της πρωτεΐνης πάνω σε πλέγμα, πολύ κοντά στην φυσική τρισδιάστατη δομή της πρωτεΐνης. Δημιούργησαν μια καθολική καλή τοποθέτηση κρατώντας έναν μικρό αριθμό από τοπικά βέλτιστα (500 συνολικά). Συγκεκριμένα διάλεγαν καλές τοπικές τοποθετήσεις ώστε να εξασφαλίσουν την ελάχιστη CRMS απόκλιση από κάθε τμήμα της πρωτεϊνικής αλυσίδας. Σύμφωνα με αυτόν τον αλγόριθμο έδειξαν ότι η σχέση μεταξύ της πολυπλοκότητας (Complexity) και της ακρίβειας (Accuracy) της αναδίπλωσης ενός πολυπεπτιδικού σκελετού πάνω σε ένα πλέγμα, μπορεί να βρεθεί από τη σχέση(Accuracy)* $(\text{Complexity})^{-1/2}$.

Σε διάφορες εργασίες έχουν παρουσιαστεί αλγόριθμοι που βασίζονται σε δυναμικό προγραμματισμό [19,20,21], αλλά και άπληστοι αλγόριθμοι [22].

Όσον αφορά την πολυπλοκότητα του PCLF προβλήματος, οι B. Berger και T. Leighton, έδειξαν ότι το HP-model σε κυβικά πλέγματα ανήκει στην κατηγορία των NP-complete προβλημάτων [23], οι J. Manuch and D. Gaur, έδειξαν ότι το PCLF πρόβλημα είναι NP-complete σε τρισδιάστατα πλέγματα με πλευρά 3.8Å χρησιμοποιώντας την απόκλιση cRMS [24], ενώ οι P. Crescenzi, D. Goldman, C. Papadimitriou, A. Piccolboni, M. Yannakakis έδειξαν ότι το HP-model στις δυο διαστάσεις ανήκει στην κατηγορία των NP-complete προβλημάτων [25]. Το πρόβλημα λύνεται βέλτιστα σε πολυωνυμικό χρόνο αν περιοριστούμε σε μονοδιάστατο πλέγμα όπως έδειξαν οι I. Emiris, E. Kranakis, E. Markou, E. Tsigaridas [26].

Τέλος, σημειώνουμε ότι πολλοί ακόμα ερευνητές έχουν ασχοληθεί με το παραπάνω πρόβλημα και ο αναγνώστης μπορεί να ανατρέξει στην σχετική βιβλιογραφία [27,28, 29, 30, 31, 32, 33, 34, 35, 36].

ΚΕΦΑΛΑΙΟ 3.
ΠΕΡΙΓΡΑΦΗ ΚΑΙ ΑΝΑΛΥΣΗ ΤΟΥ PCLF ΠΡΟΒΛΗΜΑΤΟΣ

3.1 ΠΕΡΙΓΡΑΦΗ ΤΟΥ ΠΡΟΒΛΗΜΑΤΟΣ

Το PCLF (Protein Chain Lattice Fitting problem) αναφέρεται στην αναδίπλωση μιας πρωτεΐνης πάνω σε ένα πλέγμα.

Πιο συγκεκριμένα έχοντας ως δεδομένα,

- για μια πρωτεΐνη:
  - την αλληλουχία των αμινοξέων της
  - το σχήμα της πρωτεΐνης στο χώρο όπως προκύπτει από τις συντεταγμένες των αμινοξέων της
- και για ένα πλέγμα:
  - το μήκος των ακμών του,

το PCLF πρόβλημα πραγματεύεται την τοποθέτηση των αμινοξέων πάνω στο πλέγμα με τέτοιον τρόπο ώστε η τελική απεικόνιση της πρωτεΐνης πάνω στο πλέγμα να είναι και η "πλησιέστερη" στην πραγματική απεικόνιση της πρωτεΐνης στο χώρο.

Φυσικά, η τοποθέτηση των αμινοξέων πάνω στο πλέγμα γίνεται με βάση κάποιους περιορισμούς σύμφωνα με την πραγματική αλληλουχία των αμινοξέων της πρωτεΐνης. Οι περιορισμοί αυτοί είναι οι εξής:

1. Κάθε αμινοξύ τοποθετείται σε μία μοναδική κορυφή του πλέγματος.
2. Σε κάθε κορυφή του πλέγματος μπορεί να τοποθετηθεί το πολύ ένα αμινοξύ.
3. Δύο διαδοχικά αμινοξέα τοποθετούνται σε γειτονικές κορυφές του πλέγματος.

Δύο διάσημα κριτήρια για το πότε η απεικόνιση της πρωτεΐνης πάνω στο πλέγμα είναι πλησιέστερη στην πραγματική απεικόνιση της πρωτεΐνης στο χώρο είναι τα εξής:

1. CRMS (Coordinates Root Mean Squared deviation):
   η ρίζα του μέσου όρου των τετραγωνικών αποκλίσεων μεταξύ των συντεταγμένων των αμινοξέων της πραγματικής απεικόνισης της πρωτεΐνης και των πιθανών κορυφών του πλέγματος. Το CRMS υπολογίζεται ως εξής:

$$\text{CRMS} = \sqrt{\frac{\sum_{p \epsilon P} \mathrm{d}^2(\mathrm{p},\mathrm{f}(\mathrm{p}))}{n}}$$

   όπου:

   p είναι η θέση του αμινοξέος στην τρισδιάστατη δομή,

   f(p) είναι η θέση του αμινοξέος p στο πλέγμα,

   d(p,f(p)) είναι η Ευκλείδια απόσταση μεταξύ του p και του f(p),

   n το πλήθος των αμινοξέων.

2. DRMS (Distance Root Mean Squared deviation):
   η ρίζα του μέσου όρου των τετραγωνικών αποκλίσεων μεταξύ των αποστάσεων δυο οποιονδήποτε αμινοξέων της πραγματικής απεικόνισης της πρωτεΐνης και δυο οποιονδήποτε πιθανών κορυφών του πλέγματος. Το DRMS υπολογίζεται ως εξής:

$$\text{DRMS} = \sqrt{\frac{\sum_{\forall i,j} (d(p_i, p_j) - d(q_i, q_j))^2}{\frac{n(n-1)}{2}}}$$

   όπου:

   $d(p_i,p_j)$ είναι ζεύγος αμινοξέων στην τρισδιάστατη δομή,

   $d(q_i,q_j)$ είναι ζεύγος αμινοξέων στο πλέγμα,

   n το πλήθος των αμινοξέων.

Και τα δύο κριτήρια μετράνε πόσο απέχει η πραγματική απεικόνιση της πρωτεΐνης με την τελική απεικόνιση της πρωτεΐνης πάνω στο πλέγμα. Οι μικρές τιμές και των δύο κριτηρίων είναι ένδειξη ότι η απεικόνιση της πρωτεΐνης πάνω στο πλέγμα είναι κοντά στην πραγματική απεικόνιση της πρωτεΐνης.

3.2 ΜΕΘΟΔΟΣ ΕΠΙΛΥΣΗΣ ΤΟΥ PCLF ΠΡΟΒΛΗΜΑΤΟΣ

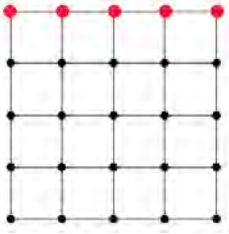
Για την επίλυση του προβλήματος ακολουθείται η εξής διαδικασία:

1) Παίρνουμε ένα σύνολο από πρωτεΐνες για τις οποίες γνωρίζουμε την αλληλουχία των αμινοξέων τους στην πρωτεϊνική αλυσίδα καθώς επίσης και τις συντεταγμένες του κάθε αμινοξέος.
2) Παίρνουμε ένα σύνολο από διαφορετικά μοντέλα πλεγμάτων.
3) Για κάθε μια πρωτεΐνη βρίσκουμε την αναπαράστασή της σε κάθε πλέγμα, η οποία τηρεί τους παραπάνω περιορισμούς και είναι τέτοια ώστε να ελαχιστοποιούνται τα κριτήρια CRMS ή DRMS.
4) Για κάθε πλέγμα βρίσκουμε τον μέσο όρο των CRMS και DRMS.
5) Το πλέγμα που δίνει τον μικρότερο μέσο όρο, ενδέχεται να είναι και το πλέγμα στο οποίο οι πρωτεΐνες αναδιπλώνονται καλύτερα. Δηλαδή η τελική τους απεικόνιση στο πλέγμα είναι πιο κοντά στην πραγματική τους απεικόνιση στον χώρο.

Για τις ανάγκες της υλοποίησης και επίλυσης του PCLF προβλήματος χρειάζεται να περιορίσουμε το μέγεθος του πλέγματος, για να είναι τέτοιο ώστε η πρωτεΐνη να χωράει ακόμα και στη χειρότερη περίπτωση, όπου η πρωτεϊνική αλυσίδα δεν αναδιπλώνεται και τα αμινοξέα παρατάσσονται σε ευθεία πάνω στο πλέγμα.

Φυσικά το μέγεθος του πλέγματος εξαρτάται άμεσα από τον τρόπο τοποθέτησης των αμινοξέων της πρωτεΐνης πάνω στο πλέγμα.

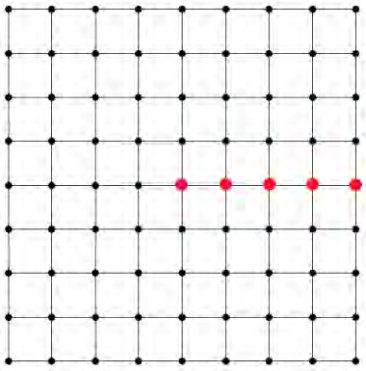
Για παράδειγμά, ένας τρόπος είναι να τοποθετηθεί το πρώτο αμινοξύ της πρωτεϊνικής αλυσίδας στην πρώτη κορυφή του πλέγματος. Έτσι το μέγεθος του πλέγματος θα είναι όσο και το πλήθος των αμινοξέων της πρωτεΐνης. Δηλαδή για μια πρωτεΐνη n αμινοξέων το μέγεθος του πλέγματος θα πρέπει να είναι τουλάχιστον nxn για δισδιάστατο πλέγμα (Εικόνα 3.1) και nxnxn για τρισδιάστατο.



Εικόνα 3.1: Παράδειγμα κατάλληλου πλέγματος πρωτεΐνη 5 αμινοξέων

Επειδή όμως στην πραγματικότητα η πρωτεϊνική δομή αναδιπλώνεται με οποιονδήποτε τρόπο και σε οποιαδήποτε κατεύθυνση, η λύση που θα δοθεί τοποθετώντας μια πρωτεΐνη με αυτόν τον τρόπο πάνω σε ένα πλέγμα, δε θα είναι η βέλτιστη και σίγουρα δε θα ανταποκρίνεται στην πραγματικότητα.

Ένας άλλος τρόπος τοποθέτησης της πρωτεΐνης πάνω στο πλέγμα είναι να μπει το πρώτο αμινοξύ στην κεντρική κορυφή του πλέγματος. Έτσι για να μπορεί η πρωτεΐνη να αναδιπλωθεί με οποιονδήποτε τρόπο και σε οποιαδήποτε κατεύθυνση, το μέγεθος του πλέγματος θα πρέπει να είναι τουλάχιστον (2n-1)x(2n-1) για δισδιάστατο πλέγμα (Εικόνα 3.2) και (2n-1)x(2n-1)x(2n-1) για τρισδιάστατο πλέγμα.



Εικόνα 3.2: Παράδειγμα κατάλληλου πλέγματος για πρωτεΐνη 5 αμινοξέων

Στην παρούσα πτυχιακή εργασία οι πρωτεΐνες τοποθετούνται πάνω στα πλέγματα με αυτόν τον τρόπο, γιατί έτσι μπορούν να αναδιπλωθούν πάνω στο πλέγμα με οποιονδήποτε τρόπο, όπως και στην πραγματικότητα, χωρίς να χάνεται πληροφορία.

Είναι ενδιαφέρον να σκεφτεί κανείς και άλλους τρόπους τοποθέτησης της πρωτεΐνης πάνω στο πλέγμα πάντα με την προϋπόθεση να μην χάνεται πληροφορία. Θα μπορούσαμε για παράδειγμα να τοποθετήσουμε το κεντρικό

αμινοξύ της πρωτεϊνικής αλυσίδας στην κεντρική κορυφή του πλέγματος. Με αυτόν τον τρόπο το μέγεθος του πλέγματος θα είναι (n+1)x(n+1) για δισδιάστατο πλέγμα και (n+1)x(n+1)x(n+1) για τρισδιάστατο πλέγμα. Έτσι, παρ' όλο που το μέγεθος του πλέγματος είναι σχεδόν το μισό, οι εξισώσεις που προκύπτουν από τους περιορισμούς του προβλήματος πολλαπλασιάζονται και έτσι δυσχεραίνεται η λύση του προβλήματος.

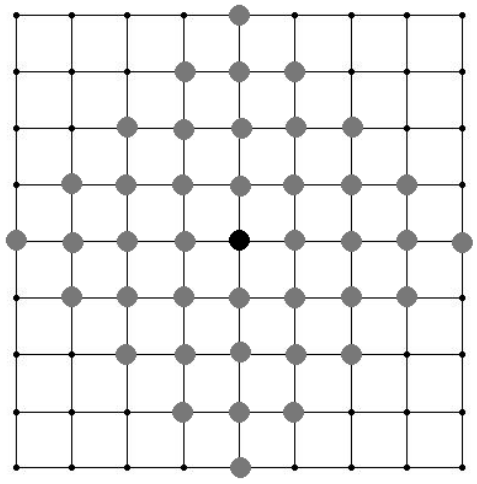
3.3 ΧΑΡΑΚΤΗΡΙΣΤΙΚΑ ΔΙΣΔΙΑΣΤΑΤΟΥ ΠΛΕΓΜΑΤΟΣ

Στο σημείο αυτό θα δείξουμε κάποια συμπεράσματα που προκύπτουν για τις κορυφές που χρησιμοποιούνται στην περίπτωση τοποθέτησης της πρωτεΐνης σε δισδιάστατο πλέγμα.

3.3.1 ΠΛΗΘΟΣ ΥΠΟΨΗΦΙΩΝ ΚΟΡΥΦΩΝ ΓΙΑ n ΑΜΙΝΟΞΕΑ

Όπως έχει ήδη αναφερθεί, έχοντας μια πρωτεΐνη με n αμινοξέα και τοποθετώντας το πρώτο αμινοξύ στην κεντρική κορυφή του πλέγματος, το μικρότερο πλέγμα που μπορεί να χρησιμοποιηθεί είναι διαστάσεων $(2n-1)x(2n-1)$, δηλαδή αποτελείται από $(2n-1)^2$ κορυφές.

Για παράδειγμα, αν υποθέσουμε ότι μια πρωτεΐνη αποτελείται από 5 αμινοξέα, το δισδιάστατο πλέγμα που θα μπορούσε να τοποθετηθεί θα ήταν το εξής (Εικόνα 3.1):



Εικόνα 3.1: Υποψήφιο δισδιάστατο πλέγμα για τοποθέτηση πρωτεΐνης 5 αμινοξέων

Με την μαύρη βούλα απεικονίζεται η κεντρική κορυφή του πλέγματος, όπου τοποθετείται το πρώτο αμινοξύ και με γκρι βούλες φαίνονται οι υποψήφιες κορυφές που μπορούν να τοποθετηθούν τα άλλα αμινοξέα. Είναι φανερό πως σε

πολλές από τις κορυφές του πλέγματος δεν μπορεί να τοποθετηθεί κανένα αμινοξύ με όποιον τρόπο και να αναδιπλωθεί η πρωτεΐνη.

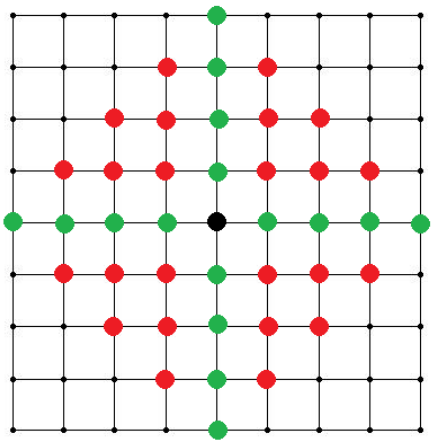
Έτσι λοιπόν το πλήθος των υποψήφιων κορυφών (έστω possible_points) ενός δισδιάστατου πλέγματος για μια πρωτεΐνη n αμινοξέων μπορεί να υπολογισθεί εύκολα από τον εξής τύπο:

total_possible_points =

$$4\sum_{i=1}^{n-2} i \; + \; 4(\mathrm{n}-1) \; + \; 1$$

όπου:

- $4\sum_{i=1}^{n-2} i$:Οι 4 περιοχές που βρίσκονται οι κορυφές του πλέγματος στην περίπτωση που τα αμινοξέα της πρωτεΐνης τοποθετούνται έτσι ώστε η πρωτεΐνη να αναδιπλώνεται και απεικονίζονται με κόκκινες βούλες (Εικόνα 3.2).
- $4(\mathrm{n}\text{-}1)$: Οι 4 περιοχές που βρίσκονται οι κορυφές του πλέγματος στην περίπτωση που τα αμινοξέα της πρωτεΐνης τοποθετούνται γραμμικά πάνω στο πλέγμα και απεικονίζονται με πράσινες βούλες (Εικόνα 3.2).
- 1 : Η κεντρική κορυφή του πλέγματος στην οποία τοποθετείται το πρώτο αμινοξύ που απεικονίζεται με μαύρη βούλα (Εικόνα 3.2).



Εικόνα 3.2: Υποψήφιο δισδιάστατο πλέγμα για τοποθέτηση πρωτεΐνης 5 αμινοξέων

Επίσης,

total_possible_points = $4\sum_{i=1}^{n-2} i + 4(\text{n-1}) + 1$

total_possible_points = $4\frac{(n-2)(n-1)}{2} + 4(\text{n-1}) + 1$

total_possible_points = $2\text{n}^2 - 2\text{n} - 1$

Για παράδειγμα,

για n=1 total_possible_points = 1

n=2 total_possible_points = 5

n=3 total_possible_points = 13

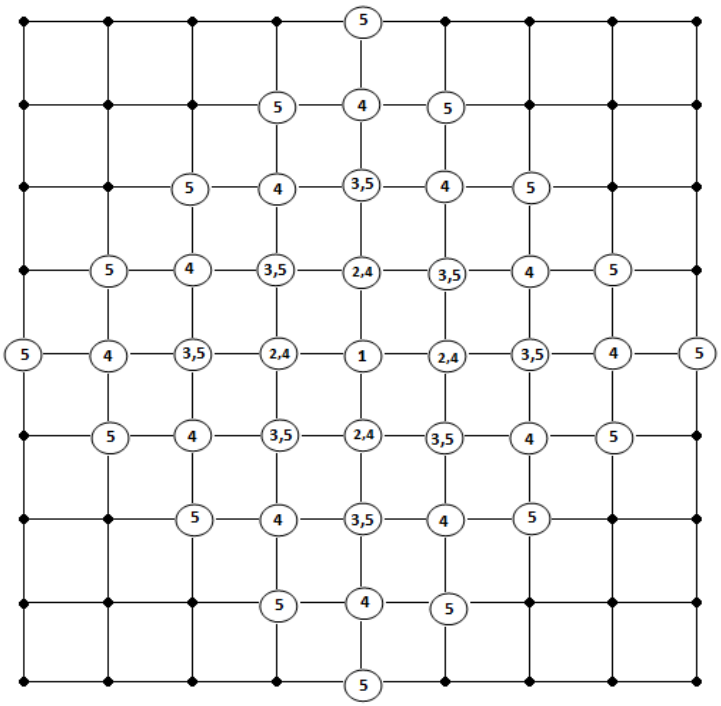
n=4 total_possible_points = 25

n=5 total_possible_points = 41

3.3.2 ΠΛΗΘΟΣ ΥΠΟΨΗΦΙΩΝ ΚΟΡΥΦΩΝ ΓΙΑ ΚΑΘΕ ΑΜΙΝΟΞΥ

Εκτός από τον συνολικό αριθμό των υποψήφιων κορυφών του πλέγματος που μπορούν να τοποθετηθούν τα αμινοξέα μιας πρωτεΐνης, μπορεί να υπολογισθεί και ο αριθμός των υποψήφιων κορυφών για κάθε αμινοξύ ξεχωριστά.

Για παράδειγμα, αν υποθέσουμε ότι μια πρωτεΐνη αποτελείται από 5 αμινοξέα, στο παρακάτω πλέγμα φαίνονται οι υποψήφιες κορυφές για κάθε αμινοξύ (Εικόνα 3.3).



Εικόνα 3.3: Υποψήφιο δισδιάστατο πλέγμα για τοποθέτηση πρωτεΐνης 5 αμινοξέων

Το $1^{ο}$ αμινοξύ μπαίνει στην κεντρική κορυφή του πλέγματος, στην οποία δεν μπορεί να τοποθετηθεί κανένα άλλο.

Το $2^{ο}$ αμινοξύ μπορεί να τοποθετηθεί στις 4 γειτονικές κορυφές της κεντρικής κορυφής.

Το $3^{ο}$ αμινοξύ μπορεί να τοποθετηθεί στις γειτονικές κορυφές των υποψήφιων κορυφών για το $2^{ο}$ αμινοξύ εκτός φυσικά από την κεντρική κορυφή κ.ο.κ.

Έστω ότι με k συμβολίζουμε το κάθε αμινοξύ, και με S(k) τη συνάρτηση που υπολογίζει το πλήθος των υποψήφιων κορυφών για το k-οστό αμινοξύ.

Το πρώτο αμινοξύ τοποθετείται στην κεντρική κορυφή.

Για κάθε αμινοξύ εκτός του πρώτου, αν η θέση του αμινοξέος στη πρωτεϊνική αλυσίδα είναι άρτιος αριθμός (έστω k), δηλαδή ($2^{ο}$, $4^{ο}$, $6^{ο}$ κλπ), τότε το πλήθος των υποψήφιων κορυφών είναι :

- $S(k) = 4 * \frac{k}{2} + 4\sum_{i=1}^{\frac{k-2}{2}} 2i = k^2$

  $(\ k{=}2,\ \ S(2)\ =\ 4$

$k=4,\ S(4) = 4 + 4 * 2 + S(2)$

$S(4) = 4 + 4 * 2 + 4 = 16$

$k=6,\ S(6) = 4 + 4 * 4 + S(4)$

$S(6) = 4 + 4 * 4 + 4 + 4 * 2 + S(2)$

$S(6) = 4 + 4 * 4 + 4 + 4 * 2 + 4 = 36$

$k=8,\ S(8) = 4 + 4 * 6 + S(6)$

$S(8) = 4 + 4 * 6 + 4 + 4 * 4 + S(4)$

$S(8) = 4 + 4 * 6 + 4 + 4 * 4 + 4 + 4 * 2 + S(2)$

$S(8) = 4 + 4 * 6 + 4 + 4 * 4 + 4 + 4 * 2 + 4 = 64$)

Ενώ αν η θέση του αμινοξέος στην πρωτεϊνική αλυσίδα είναι περιττός αριθμός, δηλαδή (3^o, 5^o,7^o κλπ), τότε το πλήθος των υποψήφιων κορυφών είναι :

- $S(k) = 4 * \frac{k-1}{2} + 4 \sum_{i=1}^{\frac{k-1}{2}} (2i - 1) = k^2-1$

($k=1,\ S(1) = 1$

$k=3,\ S(3) = 4 + 4 * 1 = 8$

$k=5,\ S(5) = 4 + 4 * 3 + S(3)$

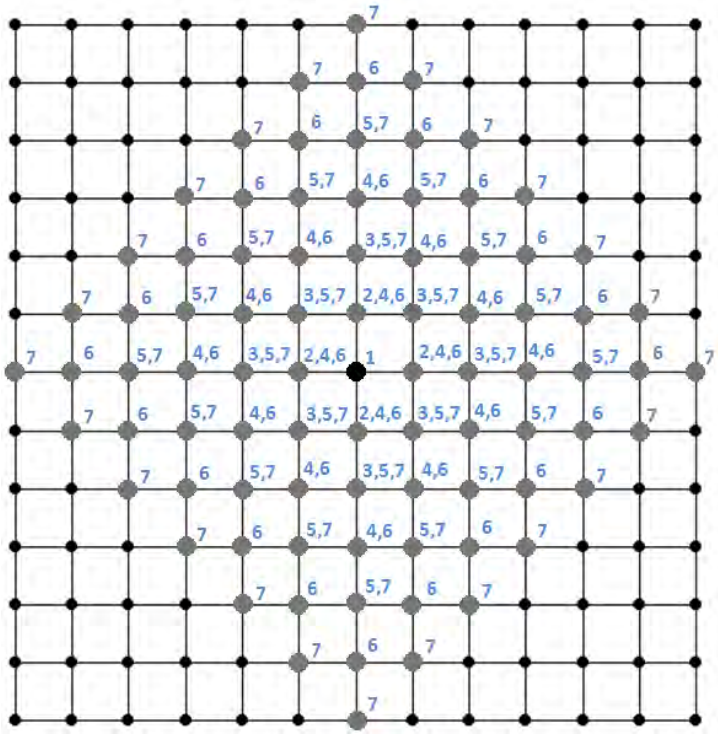
$S(5) = 4 + 4 * 3 + 4 + 4 * 1 = 24$

$k=7,\ S(7) = 4 + 4 * 5 + S(5)$

$S(7) = 4 + 4 * 3 + 4 + 4 * 3 + S(3)$

$S(7) = 4 + 4 * 3 + 4 + 4 * 3 + 4 + 4 * 1 = 48$)

Όπως φαίνεται και στην παρακάτω εικόνα (Εικόνα 3.4),



Εικόνα 3.4: Υποψήφιο δισδιάστατο πλέγμα για τοποθέτηση πρωτεΐνης 7 αμινοξέων

το $2^{ο}$ αμινοξύ μπορεί να τοποθετηθεί σε $2^2 = 4$ κορυφές,

το $3^{ο}$ αμινοξύ μπορεί να τοποθετηθεί σε $3^2 - 1 = 8$ κορυφές,

το $4^{ο}$ αμινοξύ μπορεί να τοποθετηθεί σε $4^2 = 16$ κορυφές,

το $5^{ο}$ αμινοξύ μπορεί να τοποθετηθεί σε $5^2 - 1 = 24$ κορυφές.

το $6^{ο}$ αμινοξύ μπορεί να τοποθετηθεί σε $6^2 = 36$ κορυφές,

το $7^{ο}$ αμινοξύ μπορεί να τοποθετηθεί σε $7^2 - 1 = 48$ κορυφές.

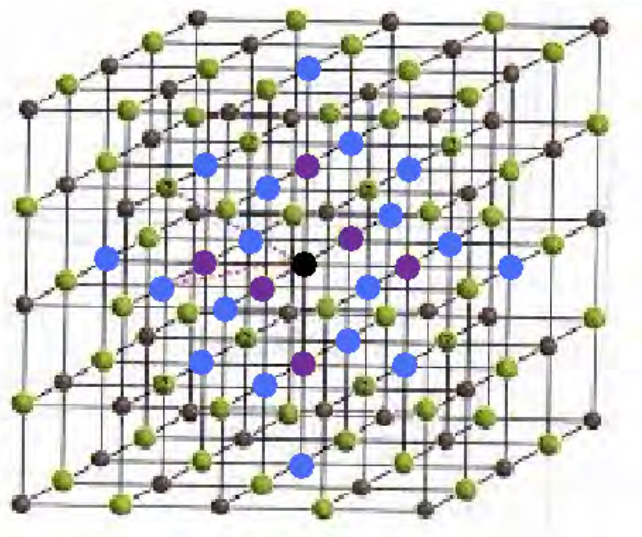
3.4 ΧΑΡΑΚΤΗΡΙΣΤΙΚΑ ΤΡΙΣΔΙΑΣΤΑΤΟΥ ΠΛΕΓΜΑΤΟΣ

Στο σημείο αυτό θα δείξουμε κάποια συμπεράσματα που προκύπτουν για τις κορυφές που χρησιμοποιούνται στην περίπτωση τοποθέτησης της πρωτεΐνης σε τρισδιάστατο πλέγμα με ανάλογο σκεπτικό όπως και στο δισδιάστατο πλέγμα.

3.4.1 ΠΛΗΘΟΣ ΥΠΟΨΗΦΙΩΝ ΚΟΡΥΦΩΝ ΓΙΑ n ΑΜΙΝΟΞΕΑ

Όπως έχει ήδη αναφερθεί, έχοντας μια πρωτεΐνη με n αμινοξέα και τοποθετώντας το πρώτο αμινοξύ στην κεντρική κορυφή του πλέγματος, το μικρότερο πλέγμα που μπορεί να χρησιμοποιηθεί είναι διαστάσεων (2n-1)x(2n-1)x(2n-1), δηλαδή αποτελείται από $(2n-1)^3$ κορυφές.

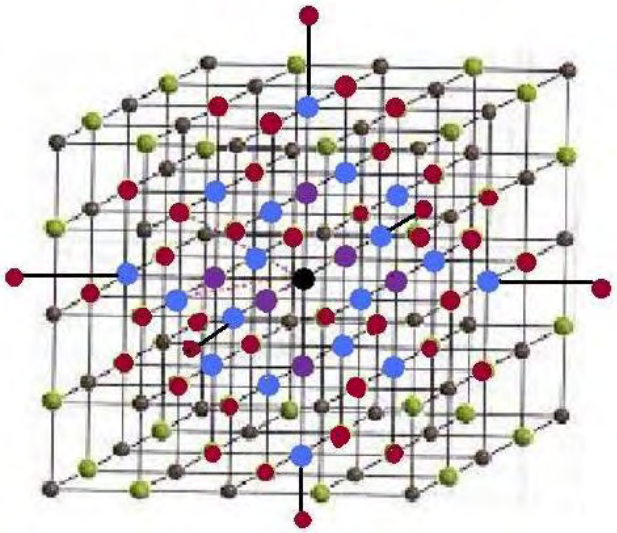
Για παράδειγμα, αν υποθέσουμε ότι η πρωτεΐνη αποτελείται από 3 αμινοξέα το τρισδιάστατο πλέγμα που θα μπορούσε να τοποθετηθεί θα ήταν το εξής (Εικόνα 3.5):



Εικόνα 3.5: Υποψήφιο τρισδιάστατο πλέγμα για τοποθέτηση πρωτεΐνης 3 αμινοξέων

Με την μαύρη βούλα απεικονίζεται η κεντρική κορυφή που τοποθετείται το πρώτο αμινοξύ. Με μοβ βούλες απεικονίζονται οι κορυφές που μπορεί να τοποθετηθεί το δεύτερο αμινοξύ και με μπλε οι υποψήφιες κορυφές για το τρίτο αμινοξύ.

Ενώ αν υποθέσουμε ότι η πρωτεΐνη αποτελείται από 4 αμινοξέα το τρισδιάστατο πλέγμα που θα μπορούσε να τοποθετηθεί θα ήταν το εξής (Εικόνα 3.6):



Εικόνα 3.6: Υποψήφιο τρισδιάστατο πλέγμα για τοποθέτηση πρωτεΐνης 4 αμινοξέων

Με την μαύρη βούλα απεικονίζεται η κεντρική κορυφή που τοποθετείται το πρώτο αμινοξύ, με μοβ βούλες απεικονίζονται οι κορυφές που μπορεί να τοποθετηθεί το δεύτερο αμινοξύ, με μπλε οι υποψήφιες κορυφές για το τρίτο αμινοξύ και τέλος με καφέ βούλες οι κορυφές για το $4^{ο}$ αμινοξύ, εκτός φυσικά από τις κοινές υποψήφιες κορυφές με το $2^{ο}$ αμινοξύ.

Όπως και στο δισδιάστατο χώρο, σε πολλές από τις κορυφές του πλέγματος δε μπορεί να τοποθετηθεί κανένα αμινοξύ, με όποιον τρόπο και να αναδιπλωθεί η πρωτεΐνη.

Έτσι λοιπόν το πλήθος των υποψήφιων κορυφών (έστω possible_points) ενός τρισδιάστατου πλέγματος για μια πρωτεΐνη n αμινοξέων μπορεί να υπολογισθεί εύκολα επεκτείνοντας τον αντίστοιχο τύπο για δισδιάστατο πλέγμα ως εξής:

total_possible_points =

$$4\sum_{i=1}^{n-2} i + 8 \sum_{i=1}^{n-2} i(n - (i + 1)) + 6(n-1) + 1$$

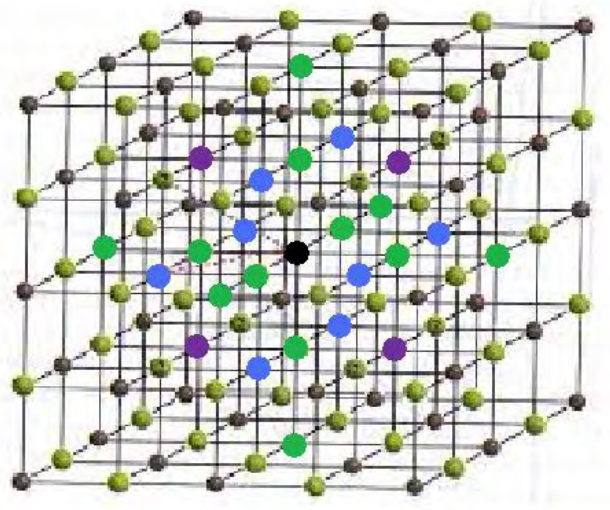
όπου:

1 : Η κεντρική κορυφή του πλέγματος στην οποία τοποθετείται το πρώτο αμινοξύ που απεικονίζεται με μαύρη βούλα (Εικόνα 3.7).

$6(\mathrm{n}\text{-}1)$: Οι 6 περιοχές του πλέγματος που βρίσκονται οι κορυφές στις οποίες τα αμινοξέα της πρωτεΐνης μπορούν να τοποθετηθούν γραμμικά και απεικονίζονται με πράσινες βούλες (Εικόνα 3.7).

$4\sum_{i=1}^{n-2} i$: Είναι οι κορυφές που βρίσκονται στις 4 περιοχές του πλέγματος που σχηματίζονται στο επίπεδο και απεικονίζονται με μοβ βούλες (Εικόνα 3.7).

$8\sum_{\mathrm{i}=1}^{\mathrm{n}-2}\mathrm{i}(\mathrm{n}-(\mathrm{i}+1))$: Είναι οι κορυφές που βρίσκονται στις περιοχές του πλέγματος που σχηματίζονται στον τρισδιάστατο χώρο και απεικονίζονται με μπλε βούλες (Εικόνα 3.7).



Εικόνα 3.7: Υποψήφιο τρισδιάστατο πλέγμα για τοποθέτηση πρωτεΐνης 3 αμινοξέων

Επίσης

$$\text{total\_possible\_points} = 4\sum_{\mathrm{i}=1}^{\mathrm{n}-2}\mathrm{i} + 8\sum_{\mathrm{i}=1}^{\mathrm{n}-2}\mathrm{i}(\mathrm{n}-(\mathrm{i}+1)) + 6(\mathrm{n}\text{-}1) + 1$$

$$\text{total\_possible\_points} = \frac{8}{6}\mathrm{n}^3 - 2\mathrm{n}^2 + \frac{16}{6}\mathrm{n} - 1$$

Για παράδειγμα,

για n=1 total_possible_points = 1

n=2 total_possible_points = 7

n=3 total_possible_points = 25

n=4 total_possible_points = 63

n=5 total_possible_points = 129

3.4.2 ΠΛΗΘΟΣ ΥΠΟΨΗΦΙΩΝ ΚΟΡΥΦΩΝ ΓΙΑ ΚΑΘΕ ΑΜΙΝΟΞΥ

Το $1^{ο}$ αμινοξύ μπαίνει στην κεντρική κορυφή του πλέγματος, στην οποία δε μπορεί να τοποθετηθεί κανένα άλλο.

Το $2^{ο}$ αμινοξύ μπορεί να τοποθετηθεί στις 6 γειτονικές κορυφές της κεντρικής κορυφής.

Το $3^{ο}$ αμινοξύ μπορεί να τοποθετηθεί στις γειτονικές κορυφές των υποψήφιων κορυφών για το $2^{ο}$ αμινοξύ, εκτός φυσικά από την κεντρική κορυφή κ.ο.κ.

Έτσι λοιπόν το πλήθος των υποψήφιων κορυφών για κάθε αμινοξύ μπορεί να υπολογισθεί ως εξής:

Έστω ότι με k συμβολίζουμε το κάθε αμινοξύ, και με S(k) τη συνάρτηση που υπολογίζει το πλήθος των υποψήφιων κορυφών για το k-οστό αμινοξύ.

Για

$\mathrm{k}=1,\ \mathrm{S}(1) = 1$

$\mathrm{k}=2,\ \mathrm{S}(2) = 6$

$\mathrm{k}=3,\ \mathrm{S}(3) = 6 + 4 * 1 + 8 * (1) = 18$

$\mathrm{k}=4,\ \mathrm{S}(4) = 6 + 4 * 2 + 8 * (1 + 2) + \mathrm{S}(\mathrm{k} - 2)$

$\mathrm{S}(4) = 6 + 4 * 2 + 8 * (1 + 2) + \mathrm{S}(2)$

$\mathrm{S}(4) = 6 + 4 * 2 + 8 * (1 + 2) + 6 = 44$

$\mathrm{k}=5,\ \mathrm{S}(5) = 6 + 4 * 3 + 8(1 + 2 + 3) + \mathrm{S}(\mathrm{k} - 2)$

$\mathrm{S}(5) = 6 + 4 * 3 + 8(1 + 2 + 3) + \mathrm{S}(3)$

$$S(5) = \ 6 + 4 * 3 + 8(1 + 2 + 3) + \ 6 + 4 * 1 + 8 * (1) = 84$$

k=6, $S(6) = \ 6 + 4 * 4 + 8(1 + 2 + 3 + 4) + \ S(4)$

$$S(6) = \ 6 + 4 * 4 + 8(1 + 2 + 3 + 4) + \ 6 + 4 * 2 + 8 * (1 + 2) + S(2)$$

$$S(6) = \ 6 + 4 * 4 + 8(1 + 2 + 3 + 4) + \ 6 + 4 * 2 + 8 * (1 + 2) + 6 = 146$$

Έτσι, αν η θέση του αμινοξέος στη πρωτεϊνική αλυσίδα είναι άρτιος αριθμός (έστω k), δηλαδή (2^o , 4^o , 6^o κλπ), τότε το πλήθος των υποψήφιων κορυφών είναι

$$S(k) = 6 * \frac{k}{2} + 4 \sum_{i=1}^{\frac{k-2}{2}} 2i + 8 \sum_{i=1}^{\frac{k-2}{2}} \sum_{j=1}^{2i} j$$

$$S(k) = \ \frac{2}{3}k^3 + \frac{1}{3}k$$

Ενώ αν η θέση του αμινοξέος στην πρωτεϊνική αλυσίδα είναι περιττός αριθμός, δηλαδή (3^o , 5^o ,7^o κλπ), τότε το πλήθος των υποψήφιων κορυφών είναι

$$S(k) = 6 * \frac{k-1}{2} + 4 \sum_{i=1}^{\frac{k-1}{2}} (2i - 1) + 8 \sum_{i=1}^{\frac{k-1}{2}} \sum_{j=1}^{2i-1} j$$

$$S(k) = \ \frac{2}{3}k^3 + \frac{1}{3}k - 1$$

ΚΕΦΑΛΑΙΟ 4.
ΜΑΘΗΜΑΤΙΚΗ ΜΟΝΤΕΛΟΠΟΙΗΣΗ ΤΟΥ PCLF ΠΡΟΒΛΗΜΑΤΟΣ

Ένας συνηθισμένος τρόπος μοντελοποίησης προβλημάτων είναι ο μαθηματικός προγραμματισμός. Έτσι, για να λύσουμε το PCLF πρόβλημα μπορούμε να το ορίσουμε ως πρόβλημα μαθηματικού προγραμματισμού. Πιο συγκεκριμένα, έχοντας ως κριτήριο το CRMS, το PCLF πρόβλημα μπορεί να οριστεί ως πρόβλημα ακέραιου γραμμικού προγραμματισμού, ενώ έχοντας ως κριτήριο το DRMS, το PCLF πρόβλημα μπορεί να οριστεί ως πρόβλημα ακέραιου μη-γραμμικού προγραμματισμού.

4.1 ΜΑΘΗΜΑΤΙΚΗ ΠΕΡΙΓΡΑΦΗ ΤΟΥ PCLF ΠΡΟΒΛΗΜΑΤΟΣ

Έστω μια πρωτεΐνη που αποτελείται από n αμινοξέα.

Θεωρούμε,

- Ένα διατεταγμένο σύνολο P που αντιστοιχεί στα αμινοξέα της πρωτεϊνικής αλυσίδας και ισχύει:

  $p_i \in P$,

  όπου $i=1,2,3,\ldots,n$ και p_i το i-οστό αμινοξύ της πρωτεΐνης.

- Ένα διατεταγμένο σύνολο L που αντιστοιχεί στις κορυφές του πλέγματος που πρόκειται να τοποθετηθούν τα αμινοξέα της πρωτεϊνικής αλυσίδας και ισχύει:

$l_j \in L$,

όπου $j=1,2,\ldots,(2n^2-2n+1)$ και l_j η j-οστή κορυφή του πλέγματος.

Οι μεταβλητές του προβλήματος είναι της μορφής $X_{i,j}$ και είναι δυαδικές. Δηλαδή οι τιμές που μπορούν να πάρουν είναι ή 0 ή 1. Κάθε μεταβλητή $X_{i,j}$ παίρνει τιμή 1 αν το αμινοξύ p_i τοποθετείται στην κορυφή l_j και τιμή 0 αν στην κορυφή l_j δεν τοποθετείται κανένα αμινοξύ.

4.1.1 ΑΝΤΙΚΕΙΜΕΝΙΚΗ ΣΥΝΑΡΤΗΣΗ ΤΟΥ ΠΡΟΒΛΗΜΑΤΟΣ

Έχοντας ως κριτήριο το CRMS, η αντικειμενική συνάρτηση που θέλουμε να ελαχιστοποιήσουμε είναι γραμμική και είναι η εξής:

$$\sum_{p_i \in P, l_j \in L} (d^2(p_i, l_j) * X_{i,j})$$

Έχοντας ως κριτήριο το DRMS, η αντικειμενική συνάρτηση που θέλουμε να ελαχιστοποιήσουμε είναι μη γραμμική και είναι η εξής:

$$\sum_{p_k, p_l \in P, l_m l_n \in L} ((d(p_k, p_l) - d(l_m, l_n))^2 * X_{k,m} * X_{l,n}$$

4.1.2 ΠΕΡΙΟΡΙΣΜΟΙ ΤΟΥ ΠΡΟΒΛΗΜΑΤΟΣ

Στο προηγούμενο κεφάλαιο αναλύθηκαν οι περιορισμοί του PCLF προβλήματος, οι οποίοι αφορούν την σωστή τοποθέτηση των αμινοξέων πάνω στο πλέγμα και είναι

οι ίδιοι είτε έχοντας ως κριτήριο το CRMS, είτε το DRMS. Οι περιορισμοί αυτοί μπορούν να περιγραφούν με μαθηματικό τρόπο όπως φαίνεται παρακάτω:

Περιορισμός 1: Ο περιορισμός αυτός εγγυάται ότι κάθε αμινοξύ p_i τοποθετείται σε μία μοναδική κορυφή l_j του πλέγματος.

$$\forall p_i \in P : \sum_{l_j \epsilon L} X_{i,j} = 1$$

Δηλαδή για τυχαίο p_i υπάρχει $X_{i,k} = 1$ και $X_{i,j} = 0, \forall j \neq k$

Περιορισμός 2: Ο περιορισμός αυτός εγγυάται ότι σε κάθε κορυφή του πλέγματος μπορεί να τοποθετηθεί το πολύ ένα αμινοξύ.

$$\forall l_j \in L : \sum_{p_i \epsilon P} X_{i,j} \leq 1$$

Περιορισμός 3: Ο περιορισμός αυτός εγγυάται ότι δυο οποιαδήποτε διαδοχικά αμινοξέα τοποθετούνται σε γειτονικές κορυφές του πλέγματος.

$$\forall l_j \in L, \forall p_i \in P : X_{i,j} + \sum_{l_k \epsilon L - N(l_j)} X_{i+1,k} \leq 1$$

Δηλαδή, σε ένα πλέγμα αν μια κορυφή l_j καταληφθεί από ένα αμινοξύ p_i (δηλαδή $X_{i,j} = 1$), τότε το επόμενο στην αλυσίδα αμινοξύ p_{i+1} δεν μπορεί να τοποθετηθεί σε καμία άλλη κορυφή του πλέγματος εκτός του συνόλου $N(l_j)$ στο οποίο ανήκουν οι γειτονικές κορυφές της l_j.

4.2 ΠΑΡΑΔΕΙΓΜΑ

Για την καλύτερη κατανόηση του προβλήματος αλλά και της μαθηματικής περιγραφής του, ακολουθεί ένα παράδειγμα τοποθέτησης μιας πρωτεΐνης που αποτελείται από 3 αμινοξέα σε ένα δισδιάστατο πλέγμα με μήκος ακμής 3.8Å. Επιλέχθηκε παράδειγμα πρωτεΐνης μόνο με 3 αμινοξέα λόγω μικρότερου όγκου δεδομένων αλλά και δισδιάστατο πλέγμα για ευκολότερη οπτικοποίηση και κατανόηση του προβλήματος.

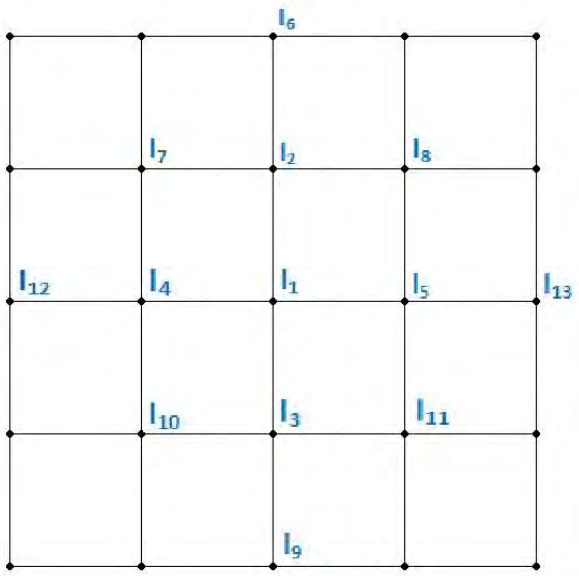
4.2.1 ΔΕΔΟΜΕΝΑ ΠΡΟΒΛΗΜΑΤΟΣ

Από την PDB συλλέχθηκαν τα δεδομένα για τις συντεταγμένες των τριών πρώτων αμινοξέων της πρωτεΐνης 1HK9, μια δεσμευτική πρωτεΐνη του RNA. Έτσι λοιπόν ο πίνακας των συντεταγμένων είναι ο εξής:

	x	y
1^{o} αμινοξύ (p_1)	-1,777	21,184
2^{o} αμινοξύ (p_2)	-2,312	22,761
3^{o} αμινοξύ (p_3)	0,609	21,091

$$P = < p_1 , p_2 , p_3 >$$

Το απαιτούμενο πλέγμα με τις υποψήφιες κορυφές είναι το εξής (Εικόνα 4.1):



Εικόνα 4.1: Δισδιάστατο πλέγμα όπου αναγράφονται οι υποψήφιες κορυφές για το συγκεκριμένο παράδειγμα

$L = < l_1 , l_2 , l_3 , l_4 , l_5 , l_6 , l_7 , l_8 , l_9 , l_{10} , l_{11} , l_{12} , l_{13} >$

Σύμφωνα με αυτά που είπαμε, το πρώτο αμινοξύ μπαίνει στην κεντρική κορυφή του lattice, άρα η κορυφή l_1 παίρνει τις συντεταγμένες του πρώτου αμινοξέος.

$l_1 : (-1{,}777 \ , \ 21{,}184)$

Έχοντας τις συντεταγμένες της κεντρικής κορυφής μπορούν εύκολα να υπολογιστούν και οι συντεταγμένες των υπόλοιπων κορυφών.

Η κορυφή l_2 έχει την ίδια τετμημένη με την κορυφή l_1 στον άξονα x, ενώ οι τεταγμένες τους διαφέρουν μια ακμή του πλέγματος στον άξονα y. Άρα, οι συντεταγμένες της κορυφής l_2 , θα είναι:

$l_2 : (-1{,}777 \ , \ (21{,}184 + 3{,}8))$

$l_2 : (-1{,}777 \ , \ 24{,}984)$

Με τον ίδιο τρόπο υπολογίζονται και οι συντεταγμένες των υπόλοιπων κορυφών του πλέγματος που είναι υποψήφιες για την τοποθέτηση κάποιου αμινοξέος.

$l_3 : (-1{,}777 \ , \ 17{,}384)$

$l_4 : (-5{,}777 \ , \ 21{,}184)$

$l_5 : (\ 2{,}023 \ , \ 21{,}184)$

l_6 : (-1,777 , 28,784)

l_7 : (-5,577 , 24,984)

l_8 : (2,023 , 24,984)

l_9 : (-1,777 , 13,584)

l_{10} : (-5,777 , 17,384)

l_{11} : (2,023 , 17,384)

l_{12} : (-9,377 , 21,184)

l_{13} : (5,823 , 21,184)

Επίσης στο πρόβλημα χρησιμοποιούνται 13 δυαδικές μεταβλητές της μορφής:

$$X_{i,j} \text{ , όπου } i=\{1,2,3\} \text{ και } j=\{1,2,\ldots,13\} \text{ ,}$$

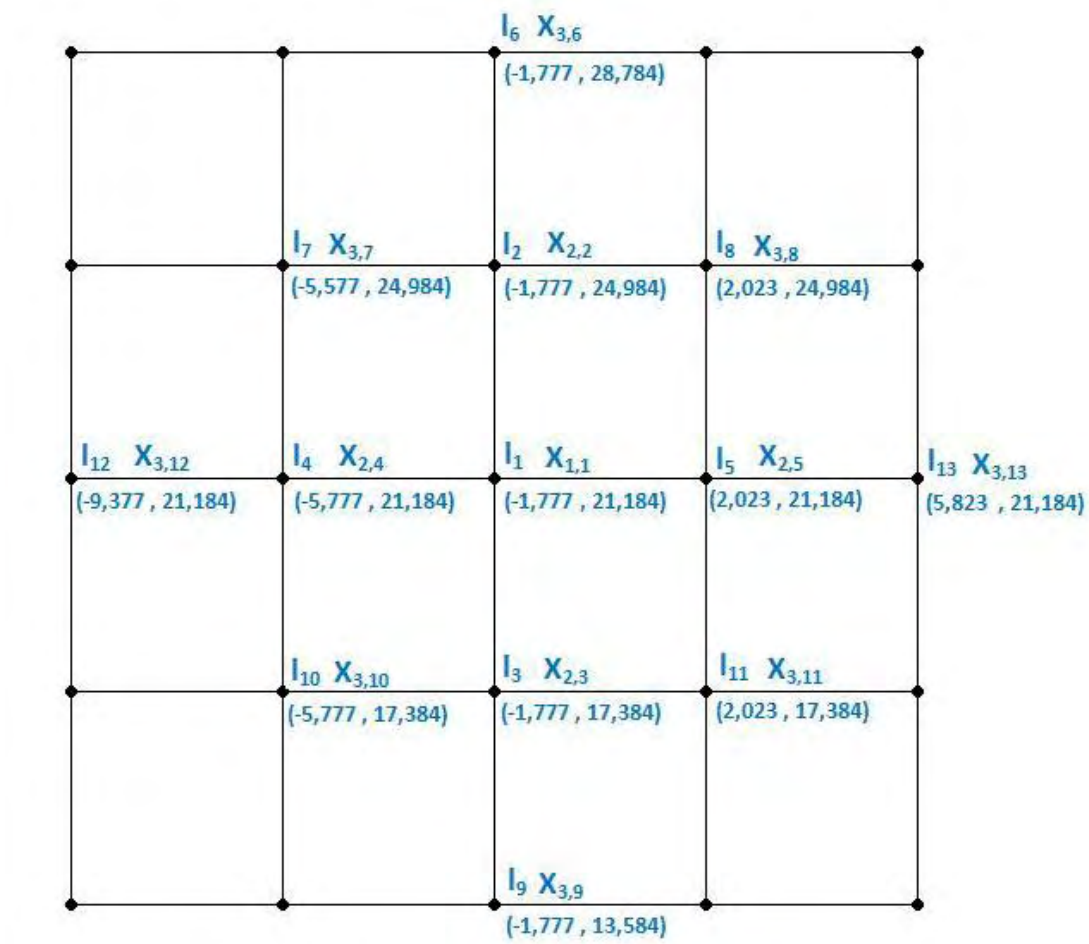
οι οποίες παίρνουν τιμή 1 όταν το αμινοξύ p_i να τοποθετηθεί στην l_j κορυφή του πλέγματος.

Οι μεταβλητές αυτές είναι οι εξής:

- $X_{1,1}$
  (το πρώτο αμινοξύ μπαίνει στην κεντρική κορυφή του lattice)
- $X_{2,2}$, $X_{2,3}$, $X_{2,4}$, $X_{2,5}$
  (4 υποψήφιες κορυφές για το 2^o αμινοξύ)
- $X_{3,6}$, $X_{3,7}$, $X_{3,8}$, $X_{3,9}$, $X_{3,10}$, $X_{3,11}$, $X_{3,12}$, $X_{3,13}$
  (8 υποψήφιες κορυφές για το 3^o αμινοξύ)

Παρατηρείστε ότι όπως αναφέραμε παραπάνω οι υποψήφιες κορυφές του πλέγματος είναι λιγότερες από το συνολικό πλήθος των κορυφών του και αυτό έχει σαν αποτέλεσμα την μείωση των μεταβλητών.

Στο παρακάτω πλέγμα φαίνονται οι υποψήφιες κορυφές μαζί με τις μεταβλητές που αντιστοιχούν σε κάθε κορυφή, καθώς επίσης και οι συντεταγμένες της κάθε κορυφής (Εικόνα 4.2).



Εικόνα 4.2: Δισδιάστατο πλέγμα όπου αναγράφονται οι υποψήφιες κορυφές, οι συντεταγμένες τους και οι μεταβλητές για το συγκεκριμένο παράδειγμα

4.2.2 ΑΝΤΙΚΕΙΜΕΝΙΚΗ ΣΥΝΑΡΤΗΣΗ ΜΕ ΚΡΙΤΗΡΙΟ ΤΟ CRMS

Η αντικειμενική συνάρτηση με κριτήριο το CRMS μετράει τη ρίζα του μέσου όρου των τετραγωνικών αποκλίσεων μεταξύ των συντεταγμένων των αμινοξέων της πραγματικής απεικόνισης της πρωτεΐνης και των πιθανών κορυφών του πλέγματος. Η αντικειμενική συνάρτηση που ζητείται να ελαχιστοποιηθεί είναι η εξής:

Minimize

$$\sum_{p_i \epsilon P, l_j \epsilon L} (\mathrm{d}^2(\mathrm{p_i}, \mathrm{l_j}) * X_{i,j})$$

Το πλήθος των όρων της αντικειμενικής συνάρτησης (έστω s) θα είναι:

$s = (a_1 + a_2 + \ldots + a_n)$

$s = 1+4+8=13$

Το πρώτο αμινοξύ μπαίνει στην κεντρική κορυφή του πλέγματος η οποία παίρνει και τις συντεταγμένες του αμινοξέος, άρα η μεταξύ τους διαφορά είναι 0. Έτσι ο πρώτος όρος της αντικειμενικής συνάρτησης θα είναι

$0{,}000000X_{1,1}$.

Στη συνέχεια η αντικειμενική συνάρτηση μετράει την απόκλιση μεταξύ των πιθανών κορυφών για το 2^o αμινοξύ και των πραγματικών συντεταγμένων του.

2^o αμινοξύ p_2 : (-2,312 , 22,761)

κορυφή l_2 : (-1,777 , 24,984)

$d^2(p_2,l_2) = \sqrt{(-2{,}312 + 1{,}777)^2 + (22{,}761 - 24{,}984)^2}^2 = 5{,}227954$

από όπου προκύπτει ο όρος $5{,}227954X_{2,2}$

κορυφή l_3 : (-1,777 , 17,384)

$d^2(p_2,l_3) = \sqrt{(-2{,}312 + 1{,}777)^2 + (22{,}761 - 17{,}384)^2}^2 = 29{,}198354$

από όπου προκύπτει ο όρος $29{,}198354X_{2,3}$

κορυφή l_4 : (-5,577 , 21,184)

$d^2(p_2,l_4) = \sqrt{(-2{,}312 + 5{,}577)^2 + (22{,}761 - 21{,}184)^2}^2 = 13{,}147154$

από όπου προκύπτει ο όρος $13{,}147154X_{2,4}$

κορυφή l_5 : (2,023 , 21,184)

$d^2(p_2,l_5) = \sqrt{(-2{,}312 - 2{,}023)^2 + (22{,}761 - 21{,}184)^2}^2 = 21{,}279154$

από όπου προκύπτει ο όρος $21{,}279154X_{2,5}$

Στη συνέχεια η αντικειμενική συνάρτηση μετράει την απόκλιση των πιθανών κορυφών για το $3^{ο}$ αμινοξύ με τις πραγματικές συντεταγμένες του αμινοξέος.

$3^{ο}$ αμινοξύ p_3 : (0,609 , 21,091)

κορυφή l_6 : (-1,777 , 28,784)

$d^2(p_3,l_6) = \sqrt{(0,609+1,777)^2 + (21,091-28,784)^2}^{-2} = 64,875245$

από όπου προκύπτει ο όρος $64,875245X_{3,6}$

κορυφή l_7 : (-5,577 , 24,984)

$d^2(p_3,l_7) = \sqrt{(0,609+5,577)^2 + (21,091-24,984)^2}^{-2} = 53,422045$

από όπου προκύπτει ο όρος $53,422045X_{3,7}$

κορυφή l_8 : (2,023 , 24,984)

$d^2(p_3,l_8) = \sqrt{(0,609-2,023)^2 + (21,091-24,984)^2}^{-2} = 17,154845$

από όπου προκύπτει ο όρος $17,154845X_{3,8}$

κορυφή l_9 : (-1,777 , 13,584)

$d^2(p_3,l_9) = \sqrt{(0,609+1,777)^2 + (21,091-13,584)^2}^{-2} = 62,048045$

από όπου προκύπτει ο όρος $62,048045X_{3,9}$

κορυφή l_{10} : (-5,577 , 17,384)

$d^2(p_3,l_{10}) = \sqrt{(0,609+5,577)^2 + (21,091-17,384)^2}^{-2} = 52,008445$

από όπου προκύπτει ο όρος $52,008445X_{3,10}$

κορυφή l_{11} : (2,023 , 17,384)

$d^2(p_3,l_{11}) = \sqrt{(0,609-2,023)^2 + (21,091-17,384)^2}^{-2} = 15,741245$

από όπου προκύπτει ο όρος $15,741245X_{3,11}$

κορυφή l_{12} : (-9,377 , 21,184)

$d^2(p_3,l_{12}) = \sqrt{(0{,}609 + 9{,}377)^2 + (21{,}091 - 21{,}184)^2}^2 = 99{,}728845$

από όπου προκύπτει ο όρος $99{,}728845X_{3,12}$

κορυφή l_{13} : (5,823 , 21,184)

$d^2(p_3,l_{13}) = \sqrt{(0{,}609 - 5{,}823)^2 + (21{,}091 - 21{,}184)^2}^2 = 27{,}194445$

από όπου προκύπτει ο όρος $27{,}194445X_{3,13}$

Άρα η αντικειμενική συνάρτηση για το CRMS θα είναι:

$0{,}000000X_{1,1} + 5{,}227954X_{2,2} + 29{,}198354X_{2,3} + 13{,}147154X_{2,4} + 21{,}279154X_{2,5}$
$+ 64{,}875245X_{3,6} + 53{,}422045X_{3,7} + 17{,}154845X_{3,8} + 62{,}048045X_{3,9} +$
$52{,}008445X_{3,10} + 15{,}741245X_{3,11} + 99{,}728845X_{3,12} + 27{,}194445X_{3,13}$

4.2.3 ΑΝΤΙΚΕΙΜΕΝΙΚΗ ΣΥΝΑΡΤΗΣΗ ΜΕ ΚΡΙΤΗΡΙΟ ΤΟ DRMS

Η αντικειμενική συνάρτηση με κριτήριο το DRMS μετράει τη ρίζα του μέσου όρου των τετραγωνικών αποκλίσεων μεταξύ των αποστάσεων δυο οποιονδήποτε αμινοξέων της πραγματικής απεικόνισης της πρωτεΐνης και δυο οποιονδήποτε πιθανών κορυφών του πλέγματος.

Η αντικειμενική συνάρτηση που ζητείται να ελαχιστοποιηθεί είναι η εξής:

Minimize

$$\sum_{p_k,p_l \in P, l_m l_n \in L} ((d(p_k,p_l) - d(l_m,l_n))^2 * X_{k,m} * X_{l.n}$$

Όπως θα δούμε παρακάτω, το πλήθος των όρων της αντικειμενικής συνάρτησης είναι 44.

$s = (1(4+8) + 4*8) = 44.$

Αρχικά υπολογίζονται οι αποστάσεις μεταξύ των αμινοξέων στην πρωτεϊνική αλυσίδα.

$p_1 = (\,-1{,}777\ ,\ 21{,}184\,)$

$p_2 = (\,-2{,}312\ ,\ 22{,}761\,)$

$p_3 = (\ 0{,}609\ ,\ 21{,}091\,)$

$$d(p_1,p_2) = \sqrt{(-1{,}777 + 2{,}312)^2 + (21{,}184 - 22{,}761)^2} = 1{,}665279$$

$$d(p_1,p_3) = \sqrt{(-1{,}777 - 0{,}609)^2 + (21{,}184 - 21{,}091)^2} = 2{,}387812$$

$$d(p_2,p_3) = \sqrt{(-2{,}312 - 0{,}609)^2 + (22{,}761 - 21{,}091)^2} = 3{,}364690$$

Στη συνέχεια υπολογίζονται οι αποστάσεις μεταξύ των κορυφών του πλέγματος και η διαφορά τους με τις αντίστοιχες πραγματικές αποστάσεις στην πρωτεϊνική αλυσίδα.

Για τις αποστάσεις μεταξύ του $1^{ου}$ και του $2^{ου}$ αμινοξέος και των υποψήφιων κορυφών που μπορούν να τοποθετηθούν έχουμε

$d(l_1,l_2) = d(l_1,l_3) = d(l_1,l_4) = d(l_1,l_5) = 3{,}8$

$(d(p_1,p_2) - d(l_1,l_2))^2 = (1{,}665279 - 3{,}8)^2 = 4{,}557034$

Άρα οι 4 πρώτοι όροι στην αντικειμενική συνάρτηση με κριτήριο το DRMS είναι

$$4{,}557034X_{1,1}X_{2,2} + 4{,}557034X_{1,1}X_{2,3} + 4{,}557034X_{1,1}X_{2,4} + 4{,}557034X_{1,1}X_{2,5}$$

Για τις αποστάσεις μεταξύ του $1^{ου}$ και του $3^{ου}$ αμινοξέος και των υποψήφιων κορυφών που μπορούν να τοποθετηθούν έχουμε

$d(l_1,l_6) = d(l_1,l_{12}) = d(l_1,l_{13}) = d(l_1,l_9) = 7{,}6$

$(d(p_1,p_3) - d(l_1,l_6))^2 = (2{,}387812 - 7{,}6)^2 = 27{,}166906$

Άρα η αντικειμενική συνάρτηση συνεχίζεται

$$27{,}166906X_{1,1}X_{3,6} + 27{,}166906X_{1,1}X_{3,12} + 27{,}166906X_{1,1}X_{3,13} + 27{,}166906X_{1,1}X_{3,9}$$

$$d(l_1,l_7) = d(l_1,l_8) = d(l_1,l_{10}) = d(l_1,l_{12}) = 5{,}374011$$

$(d(p_1,p_3) - d(l_1,l_7))^2 = (2{,}387812 - 5{,}374011)^2 = 8{,}917389$

Άρα η αντικειμενική συνάρτηση συνεχίζεται

$8{,}917389X_{1,1}X_{3,7} + 8{,}917389X_{1,1}X_{3,8} + 8{,}917389X_{1,1}X_{3,10} + 8{,}917389X_{1,1}X_{3,11}$

Στη συνέχεια υπολογίζονται οι αποστάσει μεταξύ της κορυφής l_2 με όλες τις πιθανές κορυφές για το 3^o αμινοξύ.

$d(l_2,l_7) = d(l_1,l_8) = d(l_1,l_6) = 3{,}8$

$(d(p_2,p_3) - d(l_2,l_6))^2 = (3{,}364690 - 3{,}8)^2 = 0{,}189495$

Άρα η αντικειμενική συνάρτηση συνεχίζεται

$0{,}189495X_{2,2}X_{3,7} + 0{,}189495X_{2,2}X_{3,8} + 0{,}189495X_{1,1}X_{3,6}$

Με τον ίδιο τρόπο υπολογίζονται και οι υπόλοιποι όροι της αντικειμενικής συνάρτησης, η οποία τελικά είναι:

$4{,}557034X_{1,1}X_{2,2} + 4{,}557034X_{1,1}X_{2,3} + 4{,}557034X_{1,1}X_{2,4} + 4{,}557034X_{1,1}X_{2,5} +$

$27{,}166906X_{1,1}X_{3,6} + 27{,}166906X_{1,1}X_{3,12} + 27{,}166906X_{1,1}X_{3,13} + 27{,}166906X_{1,1}X_{3,9} +$

$8{,}917389X_{1,1}X_{3,7} + 8{,}917389X_{1,1}X_{3,8} + 8{,}917389X_{1,1}X_{3,10} + 8{,}917389X_{1,1}X_{3,11} +$

$0{,}189495X_{2,2}X_{3,7} + 0{,}189495X_{2,2}X_{3,8} + 0{,}189495X_{2,2}X_{3,6} + 64{,}566202X_{2,2}X_{3,9} +$

$26{,}341201X_{2,2}X_{3,10} + 26{,}341201X_{2,2}X_{3,11} + 26{,}341201X_{2,2}X_{3,12} + 26{,}341201X_{2,2}X_{3,13} +$

$0{,}189495X_{2,3}X_{3,9} + 0{,}189495X_{2,3}X_{3,10} + 0{,}189495X_{2,3}X_{3,11} + 64{,}566202X_{2,3}X_{3,6} +$

$26{,}341201X_{2,3}X_{3,7} + 26{,}341201X_{2,3}X_{3,8} + 26{,}341201X_{2,3}X_{3,12} + 26{,}341201X_{2,3}X_{3,13} +$

$0{,}189495X_{2,4}X_{3,7} + 0{,}189495X_{2,4}X_{3,10} + 0{,}189495X_{2,4}X_{3,12} + 64{,}566202X_{2,4}X_{3,13} +$

$26{,}341201X_{2,4}X_{3,6} + 26{,}341201X_{2,4}X_{3,8} + 26{,}341201X_{2,4}X_{3,9} + 26{,}341201X_{2,4}X_{3,11} +$

$0{,}189495X_{2,5}X_{3,8} + 0{,}189495X_{2,5}X_{3,11} + 0{,}189495X_{2,5}X_{3,13} + 64{,}566202X_{2,5}X_{3,12} +$

$26{,}341201X_{2,5}X_{3,6} + 26{,}341201X_{2,5}X_{3,7} + 26{,}341201X_{2,5}X_{3,9} + 26{,}341201X_{2,5}X_{3,10} +$

$0{,}189495X_{2,5}X_{3,11} + 64{,}566202X_{2,5}X_{3,12} + 0{,}189495X_{2,5}X_{3,13}$

4.2.4 ΠΕΡΙΟΡΙΣΜΟΙ

Σύμφωνα με τον πρώτο περιορισμό του προβλήματος, κάθε αμινοξύ p_i τοποθετείται σε μία μοναδική κορυφή l_j του πλέγματος. Έτσι προκύπτουν οι εξής εξισώσεις:

- $X_{1,1} = 1$
- $X_{2,2} + X_{2,3} + X_{2,4} + X_{2,5} = 1$
- $X_{3,6} + X_{3,7} + X_{3,8} + X_{3,9} + X_{3,10} + X_{3,11} + X_{3,12} + X_{3,13} = 1$

Σύμφωνα με τον δεύτερο περιορισμό του προβλήματος, σε κάθε κορυφή του πλέγματος μπορεί να τοποθετηθεί το πολύ ένα αμινοξύ. Έτσι προκύπτουν οι εξής ανισώσεις:

- $X_{1,1} \leq 1$
- $X_{2,2} \leq 1$
- $X_{2,3} \leq 1$
- $X_{2,4} \leq 1$
- $X_{2,5} \leq 1$
- $X_{3,6} \leq 1$
- $X_{3,7} \leq 1$
- $X_{3,8} \leq 1$
- $X_{3,9} \leq 1$
- $X_{3,10} \leq 1$
- $X_{3,11} \leq 1$
- $X_{3,12} \leq 1$
- $X_{3,13} \leq 1$

Παρατηρούμε ότι, λόγω της μικρής πρωτεϊνικής αλυσίδας που έχουμε στο συγκεκριμένο παράδειγμα, σε κάθε υποψήφια κορυφή του πλέγματος μπορεί να τοποθετηθεί μόνο ένα αμινοξύ.

Σύμφωνα με τον τρίτο περιορισμό του προβλήματος, δύο διαδοχικά αμινοξέα τοποθετούνται σε γειτονικές κορυφές του πλέγματος. Έτσι προκύπτουν οι εξής ανισώσεις:

- $X_{1,1} \leq 1$ (Αλλά ήδη από τον 1^{o} περιορισμό $X_{1,1} = 1$)
- $X_{2,2} + X_{3,9} + X_{3,10} + X_{3,11} + X_{3,12} + X_{3,13} \leq 1$
- $X_{2,3} + X_{3,6} + X_{3,7} + X_{3,8} + X_{3,12} + X_{3,13} \leq 1$
- $X_{2,4} + X_{3,6} + X_{3,8} + X_{3,9} + X_{3,11} + X_{3,13} \leq 1$
- $X_{2,5} + X_{3,6} + X_{3,7} + X_{3,9} + X_{3,10} + X_{3,12} \leq 1$

ΚΕΦΑΛΑΙΟ 5.

ΑΝΑΛΥΣΗ ΠΡΟΓΡΑΜΜΑΤΟΣ

Η επίλυση του PCLF προβλήματος στην παρούσα πτυχιακή εργασία χωρίζεται σε 2 μέρη. Στο πρώτο μέρος, με το οποίο θα ασχοληθούμε σε αυτό το κεφάλαιο, φτιάξαμε ένα πρόγραμμα το οποίο παίρνει ως είσοδο τις συντεταγμένες των αμινοξέων μιας πρωτεΐνης και ένα πλέγμα και δημιουργεί τον ορισμό του προβλήματος. Στο δεύτερο μέρος, με το οποίο θα ασχοληθούμε στο επόμενο κεφάλαιο, βάζουμε τον ορισμό του προβλήματος ως είσοδο σε ένα εμπορικό πρόγραμμα και αυτό λύνει το πρόβλημα και μας επιστρέφει την τιμή της αντικειμενικής συνάρτησης καθώς επίσης και τις τιμές όλων των μεταβλητών.

Παρακάτω αναλύουμε το πρόγραμμα που δημιουργήθηκε στο πρώτο μέρος.

5.1 ΣΚΟΠΟΣ ΤΟΥ ΠΡΟΓΡΑΜΜΑΤΟΣ

Το πρόγραμμα που δημιουργήσαμε παίρνει ένα αρχείο από την PDB, το οποίο περιέχει πληροφορίες για κάποια πρωτεΐνη, το διαβάζει και αποθηκεύει σε έναν πίνακα τις συντεταγμένες των κεντρικών ανθράκων των αμινοξέων της πρωτεΐνης. Στη συνέχεια δημιουργεί κάποια αρχεία που περιέχουν τον ορισμό του προβλήματος σε μορφή μαθηματικού προγραμματισμού, ώστε να μπορούν να μπουν ως είσοδος στο πρόγραμμα CPLEX, το οποίο θα τα επεξεργαστεί και θα δώσει λύση στο πρόβλημα.

Πιο συγκεκριμένα, τα αρχεία αυτά φροντίζουμε να έχουν τέτοια μορφή, ώστε να μπορούν να μπουν στο εμπορικό πρόγραμμα CPLEX (για το οποίο θα μιλήσουμε αργότερα), που θα τα επεξεργαστεί και θα μας δώσει τη λύση του προβλήματος.

5.2 ΒΑΣΙΚΕΣ ΔΟΜΕΣ ΤΟΥ ΠΡΟΓΡΑΜΜΑΤΟΣ

Οι σημαντικότερες δομές του προγράμματος που υλοποιήθηκε είναι οι εξής:

- Ο πίνακας protein_courdinates:
  Είναι ένας δισδιάστατος πίνακας πραγματικών αριθμών μέσα στον οποίο αποθηκεύονται οι συντεταγμένες των κεντρικών ανθράκων των αμινοξέων της πρωτεΐνης. Ο αριθμός των γραμμών είναι ίσος με το πλήθος των αμινοξέων που θα τοποθετηθούν στο πλέγμα και ο αριθμός στων στηλών είναι 3 και αντιστοιχεί στις x,y,z συντεταγμένες του κάθε αμινοξέος.
- Ο πίνακας possible_points:
  Είναι ένας μονοδιάστατος πίνακας τόσων θέσεων, όσα και τα αμινοξέα που θα τοποθετηθούν στο πλέγμα. Ο πίνακας possible_points είναι τύπου ARRAY_NODE, δηλαδή κάθε στοιχείο του πίνακα αντιστοιχεί σε ένα αμινοξύ και αποτελείται από δύο πεδία. Το πρώτο πεδίο είναι ένας ακέραιος, ο οποίος αφορά το πλήθος των υποψήφιων κορυφών που μπορεί να τοποθετηθεί το αμινοξύ. Το δεύτερο πεδίο είναι ένας δείκτης, ο οποίος δείχνει στην κορυφή μιας λίστας, οι κόμβοι της οποίας αντιστοιχούν στις πιθανές κορυφές του πλέγματος που μπορεί να τοποθετηθεί το αμινοξύ.
- Ο πίνακας lattice_array:
  Ο πίνακας lattice_array είναι ένας τρισδιάστατος πίνακας τύπου LATTICE_POINT, κάθε στοιχείο του οποίου αντιστοιχεί σε μια κορυφή του πλέγματος που τοποθετούνται τα αμινοξέα και περιλαμβάνει τις αντίστοιχες συντεταγμένες της κορυφής. Δηλαδή, κάθε στοιχείο του πίνακα περιλαμβάνει 3 πεδία με τις συντεταγμένες της κορυφής.

5.3 ΣΤΑΘΕΡΕΣ ΤΟΥ ΠΡΟΓΡΑΜΜΑΤΟΣ

- Lx :
  Το μήκος της ακμής που συνδέει δύο κορυφές του lattice στον άξονα x
- Ly :
  Το μήκος της ακμής που συνδέει δύο κορυφές του lattice στον άξονα y
- Lz :
  Το μήκος της ακμής που συνδέει δύο κορυφές του lattice στον άξονα z
- PROTEIN_POINTS N:
  Το πλήθος των αμινοξέων
- MIDDLE PROTEIN_POINTS-1:
  Η κεντρική κορυφή του πλέγματος, στην οποία τοποθετείται το πρώτο αμινοξύ με βάση κάποια αρίθμηση των κορυφών του πλέγματος που επιλέγουμε.
- LATTICE_DIMENSION 2*PROTEIN_POINTS-1:
  Το μέγεθος της διάστασης του πίνακα που αντιστοιχεί στο μέγεθος της διάστασης του πλέγματος.

5.4 ΜΕΤΑΒΛΗΤΕΣ ΤΟΥ ΠΡΟΓΡΑΜΜΑΤΟΣ ΠΟΥ ΑΝΤΙΣΤΟΙΧΟΥΝ ΣΤΙΣ ΜΕΤΑΒΛΗΤΕΣ ΤΟΥ ΠΡΟΒΛΗΜΑΤΟΣ

Οι μεταβλητές που χρησιμοποιούνται στο πρόγραμμά μας για την επίλυση του PCLF προβλήματος, οι οποίες γράφονται στα αρχεία που περιέχουν την αντικειμενική συνάρτηση και τους περιορισμούς του προβλήματος, έχουν την εξής μορφή:

$X(\alpha 1, (\alpha 2, \alpha 3, \alpha 4))$, όπου

α1: Ακέραιος που αντιστοιχεί κάποιο αμινοξύ.

(α2, α3, α4): Είναι οι συντεταγμένες κάποιας υποψήφιας κορυφής του πλέγματος.

(Δεν εννοούμε τις πραγματικές συντεταγμένες αλλά τις συντεταγμένες
Με βάση κάποια διάταξη των σημείων του πλέγματος).

5.5 Η ΣΥΝΑΡΤΗΣΗ main

Μέσα στο κυρίως πρόγραμμα καλούνται, η μια μετά την άλλη, οι εννιά συναρτήσεις που υλοποιήθηκαν με την εξής σειρά:

- get_protein_coordinates_from_pdb_file:
  Διαβάζει από ένα αρχείο της PDB τις συντεταγμένες των κεντρικών ανθράκων των αμινοξέων που θα τοποθετηθούν στο πλέγμα και τις αποθηκεύει στον δισδιάστατο πίνακα protein_coordinates.
- create_list:
  Δημιουργεί τον πίνακα possible_points.
- create_lattice:
  Δημιουργεί τον πίνακα lattice_array.
- crms_function:
  Δημιουργεί ένα αρχείο που περιέχει την αντικειμενική συνάρτηση με κριτήριο το CRMS.
- drms_function:
  Δημιουργεί ένα αρχείο που περιέχει την αντικειμενική συνάρτηση με κριτήριο το DRMS.
- constraint1:
  Δημιουργεί το αρχείο που περιέχει τον πρώτο περιορισμό του προβλήματος.
- constraint2:
  Δημιουργεί το αρχείο που περιέχει τον δεύτερο περιορισμό του προβλήματος.
- constraint3:
  Δημιουργεί το αρχείο που περιέχει τον τρίτο περιορισμό του προβλήματος.
- binaries:
  Δημιουργεί το αρχείο που περιέχει τις δυαδικές μεταβλητές του προβλήματος.

5.6 Η ΣΥΝΑΡΤΗΣΗ get_protein_coordinate_from _pdb_file

Η συνάρτηση get_protein_coordinate_from _pdb_file παίρνει είσοδο δύο ορίσματα, ένα όνομα αρχείου (file_name) και το δυσδιάστατο πίνακα protein_coordinates.

Το αρχείο file_name αντιστοιχεί σε κάποιο αρχείο της PDB και περιέχει τις πληροφορίες για την πρωτεΐνη. Μετά την κλήση της συνάρτησης, ο πίνακας protein_coordinates θα περιέχει τις συντεταγμένες των κεντρικών ανθράκων των αμινοξέων της πρωτεΐνης που θα τοποθετηθούν στο lattice.

Για να γίνει αυτό η συνάρτηση εκτελεί τα εξής βήματα:

- Ανοίγει το αρχείο file_name για διάβασμα
- Διαβάζει το αρχείο γραμμή-γραμμή και μόλις συναντήσει τον κεντρικό άνθρακα του αμινοξέος αντιγράφει τους χαρακτήρες που αντιστοιχούν στην συντεταγμένη x σε μια αλφαριθμητική μεταβλητή και στη συνέχεια μετατρέπει την αλφαριθμητική μεταβλητή σε πραγματικό αριθμό και την αποθηκεύει στην αντίστοιχη θέση του πίνακα protein_coordinates. Το ίδιο γίνεται και με τις συντεταγμένες y και z.

5.7 Η ΣΥΝΑΡΤΗΣΗ create_list

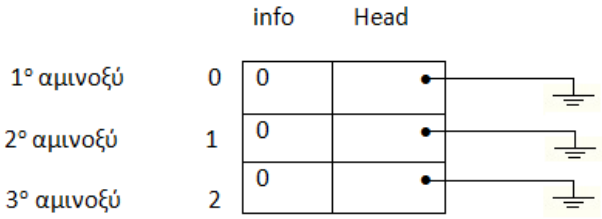
Η συνάρτηση create_list δημιουργεί τον πίνακα possible_points. Όπως έχει αναφερθεί, ο πίνακας possible_points “κρατάει” το πλήθος των πιθανών κορυφών που μπορεί να τοποθετηθεί ένα αμινοξύ, καθώς επίσης και τη λίστα με αυτές τις κορυφές.

Πιο συγκεκριμένα, η συνάρτηση ξεκινώντας από την κεντρική κορυφή του πλέγματος, όπου τοποθετείται το πρώτο αμινοξύ, ψάχνει και αποθηκεύει σε έναν

πίνακα όλες τις γειτονικές κορυφές τις οποίες και βάζει στην λίστα των υποψήφιων κορυφών του πρώτου αμινοξέος. Στη συνέχεια, για όλα τα επόμενα αμινοξέα διατρέχει τη λίστα του προηγούμενου αμινοξέος και για κάθε κόμβο αποθηκεύει τις συντεταγμένες των γειτονικών κορυφών σε έναν πίνακα, ελέγχει τη λίστα του τρέχοντος αμινοξέος και αν ο κόμβος δεν υπάρχει, τότε τον προσθέτει.

Παρακάτω αναλύουμε τα βήματα της συνάρτησης.

Πρώτα η συνάρτηση αρχικοποιεί τον πίνακα. Έτσι, αν για παράδειγμα, υπάρχουν τρία αμινοξέα για να τοποθετηθούν στο πλέγμα, ο πίνακας possible_points μετά την αρχικοποίηση του θα έχει την εξής μορφή (Σχήμα 5.1):

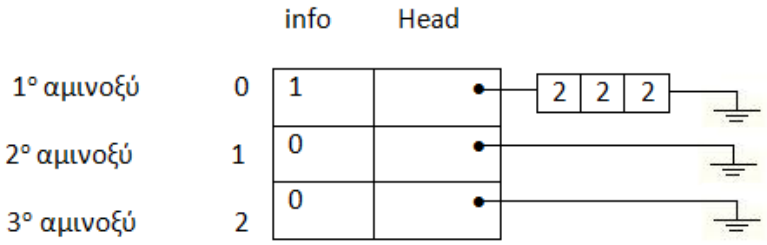


Σχήμα 5.1: Ο πίνακας possible_points αρχικοποιημένος

Στη συνέχεια, ο πίνακας possible_points, παίρνει τιμές για το πρώτο αμινοξύ, που είναι γνωστό ότι θα μπει στην κεντρική κορυφή του πλέγματος. Έτσι, ο πίνακας διαμορφώνεται ως εξής (Σχήμα 5.2):

MIDDLE= PROTEIN_POINTS – 1 = 3-1=2

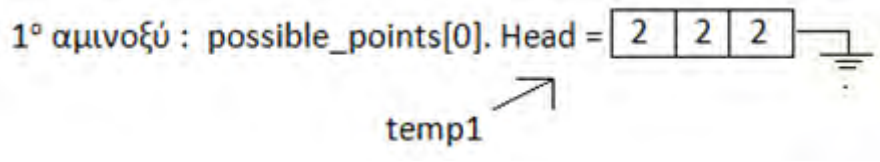
Possible_Points



Σχήμα 5.2: Ο πίνακας possible_points με την λίστα του 1ου αμινοξέος

Στη συνέχεια η συνάρτηση ψάχνει να βρει τις υποψήφιες κορυφές και για τα υπόλοιπα αμινοξέα. Σύμφωνα με τον τρίτο περιορισμό του προβλήματος, δυο διαδοχικά αμινοξέα μπορούν να τοποθετηθούν σε δυο διαδοχικές κορυφές του πλέγματος. Έτσι οι κορυφές που μπορεί να τοποθετηθεί κάθε αμινοξύ είναι οι γειτονικές κορυφές των υποψήφιων κορυφών που μπορεί να τοποθετηθεί το προηγούμενο αμινοξύ. Για να γίνει αυτό, το πρόγραμμα χρησιμοποιεί για κάθε αμινοξύ, έναν δείκτη που δείχνει στη λίστα των πιθανών κορυφών του αμινοξέος και έναν δείκτη που διατρέχει την λίστα των πιθανών κορυφών του προηγούμενου αμινοξέος. Για κάθε κόμβο της λίστας του προηγούμενου αμινοξέος, αποθηκεύει σε έναν πίνακα adj_points τις γειτονικές κορυφές και τις βάζει στη λίστα του τρέχοντος αμινοξέος εκτός από αυτές που υπάρχουν ήδη στη λίστα ή την κεντρική κορυφή, που είναι πιασμένη από το πρώτο αμινοξύ.

Για το δεύτερο αμινοξύ λοιπόν, υπάρχει ένας δείκτης, ο οποίος δείχνει στη λίστα του πρώτου (temp1) (Σχήμα 5.3)



Σχήμα 5.3: Ο δείκτης temp1 δείχνει στη λίστα του 1ου αμινοξέος

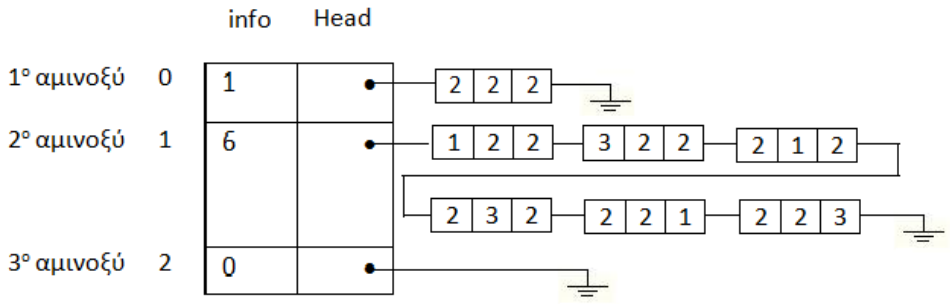
Στον πίνακα adj_points τοποθετούνται οι γειτονικές κορυφές της κεντρικής κορυφής [2][2][2].

adj_points

1	2	2
3	2	2
2	1	2
2	3	2
2	2	1
2	2	3

Η λίστα για το δεύτερο αμινοξύ είναι αρχικά κενή. Επίσης, όλες οι γειτονικές κορυφές της κεντρικής κορυφής του lattice που τοποθετείται το πρώτο αμινοξύ, είναι πιθανές κορυφές για το δεύτερο αμινοξύ, έτσι ο πίνακας possible_points θα έχει την εξής μορφή (Σχήμα 5.4):

possible_points



Σχήμα 5.4: Ο πίνακας possible_points με τις λίστες του 1ου και 2ου αμινοξέος

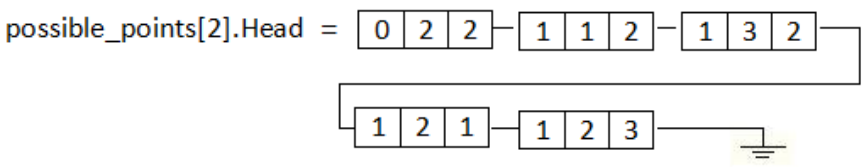
Για το τρίτο αμινοξύ, διατρέχεται η λίστα του δεύτερου αμινοξέος.

Αρχικά στον πίνακα adj_points τοποθετούνται οι γειτονικές κορυφές της [1][2][2]

adj_points

0	2	2
2	2	2
1	1	2
1	3	2
1	2	1
1	2	3

Όλες οι κορυφές τοποθετούνται στη λίστα του τρίτου αμινοξέος εκτός από την [2][2][2] που είναι η κεντρική κορυφή (Σχήμα 5.5).



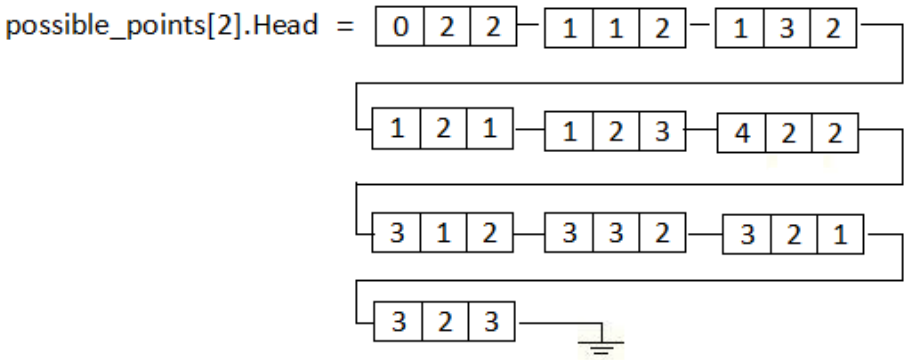
Σχήμα 5.5: Η λίστα του 3ου αμινοξέος

Στη συνέχεια ο δείκτης που διατρέχει την λίστα του δεύτερου αμινοξέος δείχνει την κορυφή [3][2][2] και ο πίνακας adj_points θα έχει τις γειτονικές κορυφές της [3][2][2].

adj_points

2	2	2
4	2	2
3	1	2
3	3	2
3	2	1
3	2	3

Η κορυφή [2][2][2] είναι η κεντρική κορυφή και άρα δεν μπαίνει στη λίστα (Σχήμα 5.6)



Σχήμα 5.6: Η λίστα του 3ου αμινοξέος

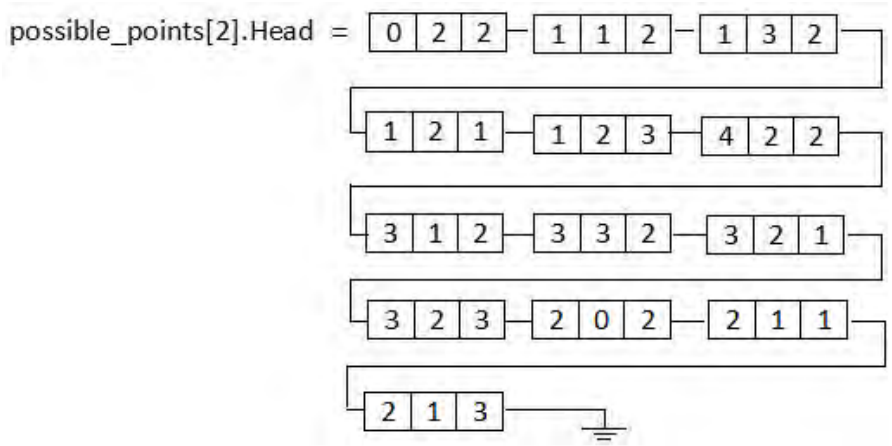
Μετά, ο δείκτης temp1, δείχνει στην κορυφή [2][1][2], και ο πίνακας adj_points που περιέχει τις γειτονικές της κορυφές είναι:

adj_points

2	2	2

4	2	2
3	1	2
3	3	2
3	2	1
3	2	3

Οι κορυφές [1][1][2] και [3][1][2] υπάρχουν ήδη στη λίστα και άρα δεν θα μπουν, όπως επίσης και η κορυφή [2][2][2] που είναι η κεντρική κορυφή (Σχήμα 5.7).



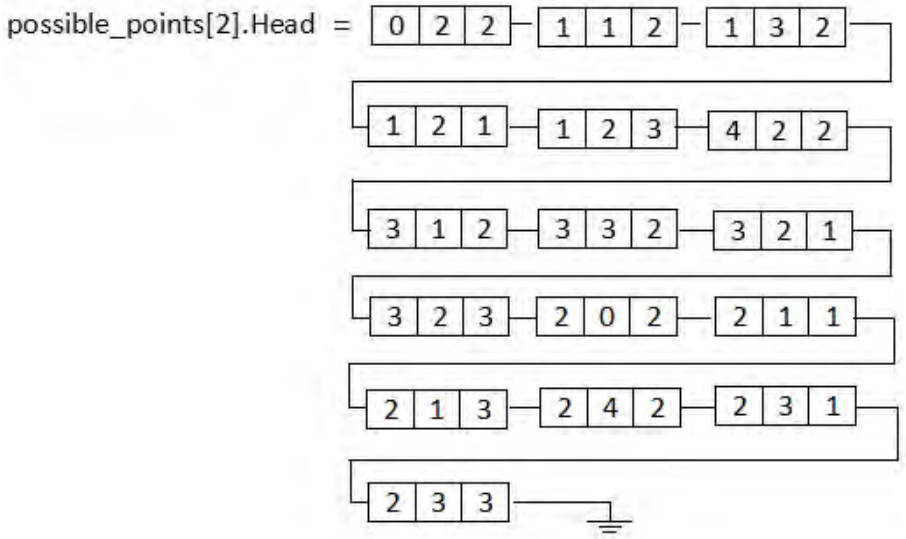
Σχήμα 5.7: Η λίστα του 3ου αμινοξέος

Τώρα, ο temp1 δείχνει στην κορυφή [2][3][2] και ο πίνακας των γειτονικών κορυφών είναι:

adj_points

1	3	2
3	3	2
2	2	2
2	4	2
2	3	1
2	3	3

Οι κορυφές [1][3][2] και [3][3][2] υπάρχουν ήδη στη λίστα και απορρίπτονται καθώς επίσης και η κορυφή [2][2][2] που είναι η κεντρική κορυφή (Σχήμα 5.8).



Σχήμα 5.8: Η λίστα του 3ου αμινοξέος

Για την κορυφή [2][2][1] ο πίνακας των γειτονικών κορυφών είναι ο εξής:

adj_points

1	2	1
3	2	1
2	1	1
2	3	1
2	2	0
2	2	2

Μόνο η κορυφή [2][2][0] θα μπει στη λίστα.

Τέλος, για την κορυφή [2][2][3] ο πίνακας των γειτονικών κορυφών θα είναι

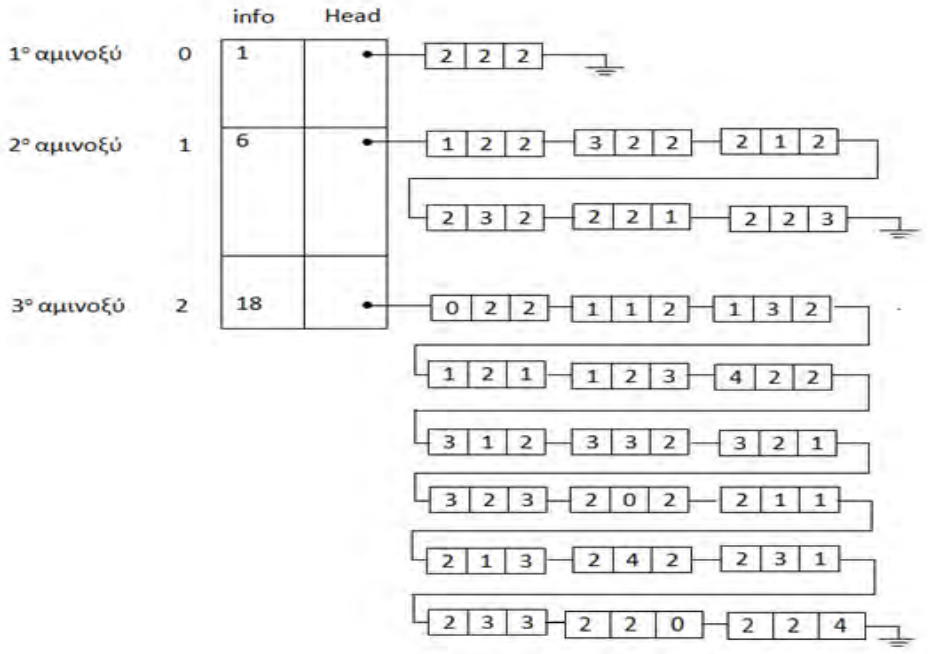
adj_points

1	2	1
3	2	1
2	1	1
2	3	1
2	2	0
2	2	2

και μόνο η κορυφή [2][2][4] θα μπει στη λίστα και θα είναι η τελευταία.

Έτσι, ο πίνακας possible_points διαμορφώνεται ως εξής (Σχήμα 5.9):

possible_points



Σχήμα 5.9: Ο πίνακας possible_points με τις λίστες του 1ου ,2ου και 3ου αμινοξέος

Η συνάρτηση create_list λειτουργεί με αυτόν τον τρόπο για οποιοδήποτε πλήθος αμινοξέων.

5.8 Η ΣΥΝΑΡΤΗΣΗ create_lattice

Η συνάρτηση create_lattice δημιουργεί έναν πίνακα lattice_array, ο οποίος αντιστοιχεί στις κορυφές του πλέγματος που τοποθετούνται τα αμινοξέα. Πιο συγκεκριμένα, για κάθε αμινοξύ η συνάρτηση βρίσκει από την αντίστοιχη λίστα του πίνακα possible_points τις πιθανές κορυφές που μπορεί να τοποθετηθεί και ανάλογα με την απόσταση που απέχει η κάθε κορυφή από την κεντρική, της οποίας οι συντεταγμένες είναι ήδη γνωστές, βρίσκει και αποθηκεύει τις συντεταγμένες για αυτό το στοιχείο.

Παρακάτω αναλύουμε τα βήματα της συνάρτησης.

Αρχικά, η συνάρτηση αρχικοποιεί τον πίνακα lattice_array θέτοντας σε κάθε κορυφή τις συντεταγμένες x, y, z ίσες με 0.0.

Όπως έχει αναφερθεί, στην κεντρική κορυφή του πλέγματος τοποθετείται το πρώτο αμινοξύ της πρωτεϊνικής αλυσίδας, του οποίου οι συντεταγμένες είναι τα τρία πρώτα στοιχεία του πίνακα protein_coordinates. Γι' αυτό και στη συνέχεια η συνάρτηση αναθέτει τις συντεταγμένες αυτές στο στοιχείο του πίνακα που αντιστοιχεί στην κεντρική κορυφή.

Ο πίνακας lattice_array έχει LATTICE_DIMENSION^3 στοιχεία. Επειδή όμως υπάρχουν κορυφές, οι οποίες σύμφωνα με τους περιορισμούς του προβλήματος δεν είναι υποψήφιες για την τοποθέτηση κάποιου αμινοξέος, η συνάρτηση βρίσκει και αποθηκεύει μόνο τις συντεταγμένες στα στοιχεία του πίνακα lattice_array που είναι υποψήφιες κορυφές για κάποιο αμινοξύ.

Για να γίνει αυτό χρησιμοποιείται ο πίνακας possible_points που περιέχει τις πιθανές κορυφές που μπορεί να τοποθετηθεί το κάθε αμινοξύ. Η λειτουργία της συνάρτησης γίνεται πιο κατανοητή με το παρακάτω παράδειγμα.

Η κεντρική κορυφή του lattice παίρνει τις συντεταγμένες του πρώτου αμινοξέος, οι οποίες είναι αποθηκευμένες στον πίνακα protein_coordinates.

Οι συντεταγμένες κάθε άλλης υποψήφιας κορυφής του πλέγματος βρίσκονται ανάλογα με την απόστασή της από την κεντρική κορυφή. Επίσης κάθε κορυφή του πλέγματος συνδέεται με την γειτονική της μέσω μιας ακμής, το μήκος της οποίας καθορίζεται από τις σταθερές Lx, Ly, Lz.

Για κάθε αμινοξύ χρησιμοποιείται ένας βοηθητικός δείκτης temp, ο οποίος δείχνει στη λίστα των υποψήφιων κορυφών του πίνακα possible_points. Έτσι τα στοιχεία temp->x, temp->y, temp->z δείχνουν τις αντίστοιχες θέσεις της κορυφής στον πίνακα lattice_array.

Η συνάρτηση χρησιμοποιεί τρεις μεταβλητές (foundx, foundy, foundz), στις οποίες αποθηκεύεται ο αριθμός των ακμών που απέχει η υποψήφια κορυφή που δείχνει ο temp από την κεντρική κορυφή.

Τέλος, η συνάρτηση υπολογίζει την απόσταση και για τις τρεις συντεταγμένες (foundx*Lx, foundy*Ly, foundz*Lz για x, y, z αντίστοιχα) και την προσθέτει με την αντίστοιχη της κεντρικής κορυφής.

5.9 Η ΣΥΝΑΡΤΗΣΗ crms_function

Η συνάρτηση crms_function δημιουργεί ένα αρχείο με όνομα input_cplex_crms, μέσα στο οποίο γράφει την αντικειμενική συνάρτηση του προβλήματος με κριτήριο το CRMS. Το αρχείο που δημιουργείται είναι τύπου lp για να μπορεί να μπει ως είσοδος στο CPLEX.

Όπως έχει αναφερθεί ήδη, το CRMS μετράει τη ρίζα του μέσου όρου των τετραγωνικών αποκλίσεων μεταξύ των συντεταγμένων των αμινοξέων της πραγματικής απεικόνισης της πρωτεΐνης και των πιθανών κορυφών του πλέγματος.

$$\text{CRMS}=\sqrt{\frac{\sum \mathrm{d}^2(\mathrm{p},\mathrm{f}(\mathrm{p}))}{n}}$$

Μέσα στο αρχείο γράφεται μόνο η αντικειμενική συνάρτηση, δηλαδή το άθροισμα στον αριθμητή.

Πιο συγκεκριμένα η συνάρτηση διατρέχει τις λίστες όλων των αμινοξέων και υπολογίζει την Ευκλείδεια απόσταση μεταξύ της πραγματικής θέσης του αμινοξέος και της υποψήφιας κορυφής του πλέγματος.

Παρακάτω αναλύουμε τη λειτουργία της συνάρτησης.

Η συνάρτηση crms_function για να υπολογίσει και να γράψει μέσα στο αρχείο την αντικειμενική συνάρτηση παίρνει ως ορίσματα τους τρεις βασικούς πίνακες του προγράμματός μας, τον πίνακα lattice_array, τον πίνακα protein_coordinates και τον πίνακα possible_points.

Για κάθε αμινοξύ, χρησιμοποιείται ένας βοηθητικός δείκτης temp ο οποίος δείχνει στην αρχή της λίστας των υποψήφιων κορυφών του αμινοξέος. Στη συνέχεια η συνάρτηση διατρέχει τη λίστα και μέχρι να φτάσει στο τέλος της, υπολογίζει την Ευκλείδεια απόσταση μεταξύ της πραγματικής θέσης του αμινοξέος, οι συντεταγμένες της οποίας βρίσκονται στον πίνακα protein_coordinates και της υποψήφιας κορυφής του πλέγματος, οι συντεταγμένες της οποίας βρίσκονται αποθηκευμένες στον πίνακα lattice_arrray.

Μόλις υπολογισθεί η απόσταση αυτή, γράφεται στο αρχείο ακολουθούμενη από την αντίστοιχη μεταβλητή και η συνάρτηση συνεχίζει στον επόμενο κόμβο της λίστας. Η ίδια διαδικασία ακολουθείται για όλα τα αμινοξέα που θέλουμε να τοποθετηθούν στο πλέγμα.

Μόλις η αντικειμενική συνάρτηση γραφεί στο αρχείο, η συνάρτηση crms_function κλείνει το αρχείο και τερματίζει.

Η λειτουργία της συνάρτησης γίνεται πιο κατανοητή από το παρακάτω παράδειγμα.

Έστω ότι θέλουμε να τοποθετήσουμε στο πλέγμα τα τρία πρώτα αμινοξέα της πρωτεΐνης 1HK9, μια δεσμευτική πρωτεΐνη του RNA.

Ο πίνακας protein_coordinates είναι ο εξής:

protein_coordinates

-1,777	21,184	-12,541
-2,312	22,761	-9,108
0,609	21,091	-7,428

Για την τοποθέτηση τριών αμινοξέων απαιτείται ένας πίνακας 5x5x5

lattice_array[5][5][5]

Το πρώτο αμινοξύ τοποθετείται στην κεντρική κορυφή του πίνακα

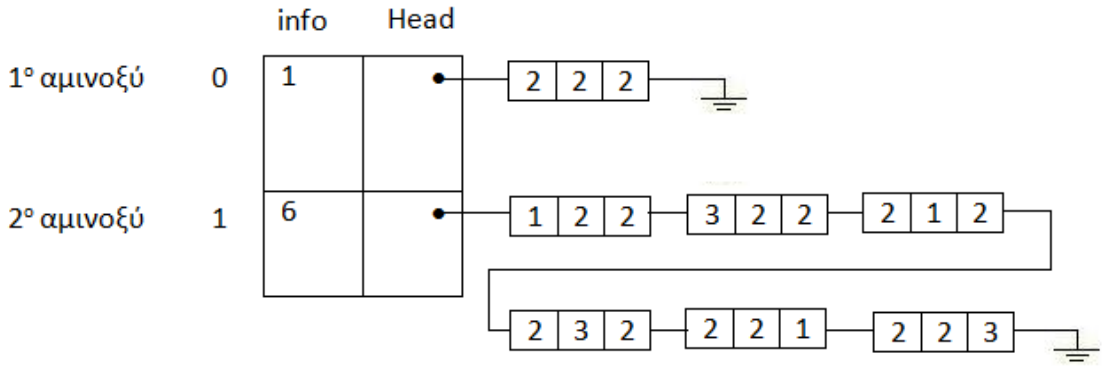
lattice_array[2][2][2].x = -1,777

lattice_array[2][2][2].y = 21,184

lattice_array[2][2][2].z = -12,541

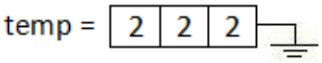
Ο πίνακας possible_points για τα δυο πρώτα αμινοξέα είναι ο εξής (Σχήμα 5.10):

possible_points



Σχήμα 5.10: Ο πίνακας possible_points για τα δυο πρώτα αμινοξέα

Στη συνάρτηση crms_function αρχικά ο δείκτης temp δείχνει στη λίστα του πρώτου αμινοξέος

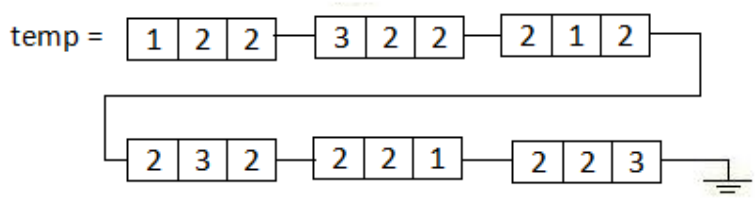


Επειδή όμως το πρώτο αμινοξύ τοποθετείται στην κεντρική κορυφή του πλέγματος και άρα οι συντεταγμένες της κορυφής συμπίπτουν με τις πραγματικές συντεταγμένες του πρώτου αμινοξέος, η Ευκλείδεια απόσταση μεταξύ τους είναι 0.00 .

Έτσι ο πρώτος όρος της αντικειμενικής συνάρτησης θα είναι

0,000000X(1,(2,2,2))

Στη συνέχεια, ο δείκτης temp δείχνει στη λίστα των υποψήφιων κορυφών για το δεύτερο αμινοξύ (Σχήμα 5.11).



Σχήμα 5.11: Ο δείκτης temp δείχνει στη λίστα του 2ου αμινοξέος

Από την συνάρτηση create_lattice έχουν υπολογιστεί οι συντεταγμένες της κορυφής [1|2|2]

lattice_array[1][2][2].x = -5,577

lattice_array[1][2][2].y = 21,184

lattice_array[1][2][2].z = -12,541

Οι συντεταγμένες του 2ου αμινοξέος από τον πίνακα protein_coordinates είναι

(-2,312 , 22,761 , -9,108)

Έτσι η Ευκλείδεια απόσταση υπολογίζεται ως εξής:

$|-2,312 + 5,577|^2 + |22,761 – 21,184|^2 + |-9,108 + 12,541|^2 =$

$3,265^2 + 1,577^2 + 3,433^2 =$

10,660225 + 2,486929 + 11,785489 =

24,932643

Άρα η αντικειμενική συνάρτηση συνεχίζεται

0,000000X(1,(2,2,2)) + 24,932643X(2,(1,2,2))

Στη συνέχεια ο δείκτης temp δείχνει στην επόμενη υποψήφια κορυφή, δηλαδή την [3|2|2] και ακολουθείται η ίδια διαδικασία μέχρι το τέλος της λίστας. Μόλις τελειώσει η λίστα του 2ου αμινοξέος, ο δείκτης temp δείχνει στην λίστα των υποψήφιων κορυφών για το 3ο αμινοξύ κ.ο.κ.

5.10 Η ΣΥΝΑΡΤΗΣΗ drms_function

Η συνάρτηση drms_function δημιουργεί ένα αρχείο με όνομα input_cplex_drms, μέσα στο οποίο γράφει την αντικειμενική συνάρτηση του προβλήματος με κριτήριο το DRMS. Το αρχείο που δημιουργείται είναι τύπου lp για να μπορεί να μπει ως είσοδος στο CPLEX.

Όπως έχει αναφερθεί ήδη, το DRMS μετράει πόσο καλά μπορεί να παρουσιαστεί ολόκληρη η δομή της πρωτεΐνης στο πλέγμα και υπολογίζεται ως εξής:

$$\text{DRMS}=\sqrt{\frac{\sum_{i-1}^{n-1}\sum_{j=i+1}^{n}(d(p_i,p_j)-d(q_i,q_j))^2}{\frac{n(n-1)}{2}}}$$

Μέσα στο αρχείο γράφεται μόνο η αντικειμενική συνάρτηση, δηλαδή το άθροισμα των γινομένων του αριθμητή.

Πιο συγκεκριμένα η συνάρτηση διατρέχει τις λίστες όλων των αμινοξέων και για κάθε κορυφή υπολογίζει την Ευκλείδεια απόσταση μεταξύ αυτής και όλων των κορυφών του πλέγματος που είναι υποψήφιες για κάποιο αμινοξύ. Επίσης υπολογίζει την Ευκλείδεια απόσταση μεταξύ όλων των αμινοξέων της διάταξης της πρωτεϊνικής αλυσίδας. Τέλος υπολογίζει την διαφορά μεταξύ όλων των αποστάσεων μεταξύ δύο διαδοχικών αμινοξέων της αλυσίδας και δύο αντίστοιχων υποψήφιων κορυφών του πλέγματος.

Παρακάτω αναλύουμε τη λειτουργία της συνάρτησης.

Η συνάρτηση drms_function για να υπολογίσει και να γράψει μέσα στο αρχείο την αντικειμενική συνάρτηση παίρνει ως ορίσματα τους τρεις βασικούς πίνακες του προγράμματός μας, τον πίνακα lattice_array, τον πίνακα protein_coordinates και τον πίνακα possible_points.

Η συνάρτηση χρησιμοποιεί τρείς βοηθητικές πραγματικές μεταβλητές, τις distance1, distance2 και total_distance.

Η μεταβλητή distance1 χρησιμοποιείται για τον υπολογισμό των πραγματικών αποστάσεων μεταξύ των αμινοξέων της πρωτεΐνης, οι συντεταγμένες των οποίων βρίσκονται στον πίνακα protein_points.

Η μεταβλητή distance2 χρησιμοποιείται για τον υπολογισμό των αποστάσεων μεταξύ των υποψήφιων κορυφών του lattice, οι συντεταγμένες των οποίων βρίσκονται στον πίνακα lattice_array.

Η μεταβλητή total_distance χρησιμοποιείται για τον υπολογισμό της απόστασης μεταξύ των αποστάσεων distance1 και distance2 που είναι και το ζητούμενο σύμφωνα με το κριτήριο DRMS.

Επίσης χρησιμοποιούνται και δυο βοηθητικές μεταβλητές-δείκτες, οι temp1 και temp2 που δείχνουν στις λίστες των υποψήφιων κορυφών για κάθε ζευγάρι αμινοξέων για τον υπολογισμό της απόστασης distance2.

Αρχικά, ο δείκτης temp1 δείχνει στη λίστα του $1^{ου}$ αμινοξέος που αντιστοιχεί στην κεντρική κορυφή του lattice και ο δείκτης temp2 δείχνει στη λίστα του επόμενου αμινοξέος.

Αφού υπολογιστεί η πραγματική απόσταση μεταξύ του $1^{ου}$ και του $2^{ου}$ αμινοξέος (distance1), ο δείκτης temp2 διατρέχει την λίστα και για κάθε υποψήφια κορυφή υπολογίζει την απόστασή της από την κορυφή που δείχνει ο temp1 (distance2) και στη συνέχεια υπολογίζεται η συνολική απόσταση (total_distance) η οποία και γράφεται στο αρχείο. Όταν ο temp2 φτάσει στο τέλος της λίστας, συνεχίζει με την λίστα του επόμενου αμινοξέος, και μόλις διατρέξει τις λίστες των αμινοξέων του πίνακα possible_points, τότε ο δείκτης temp1 συνεχίζει στον επόμενο κόμβο (αν υπάρχει) ή στην επόμενη λίστα για να ξαναρχίσει η ίδια διαδικασία.

Η συνάρτηση τελειώνει μόλις υπολογιστούν και γραφούν στο αρχείο οι αποστάσεις μεταξύ όλων των ζευγαριών των πραγματικών αμινοξέων της πρωτεΐνης και των υποψήφιων κορυφών του πλέγματος.

Η λειτουργία της συνάρτησης γίνεται πιο κατανοητή από το παρακάτω παράδειγμά.

Έστω ότι θέλουμε να τοποθετήσουμε στο πλέγμα τα τρία πρώτα αμινοξέα της πρωτεΐνης 1HK9.

Ο πίνακας protein_coordinates είναι ο εξής:

protein_coordinates

-1,777	21,184	-12,541
-2,312	22,761	-9,108
0,609	21,091	-7,428

Για την τοποθέτηση τριών αμινοξέων απαιτείται ένας πίνακας 5X5X5

lattice_array[5][5][5]

Το πρώτο αμινοξύ τοποθετείται στην κεντρική κορυφή του πίνακα

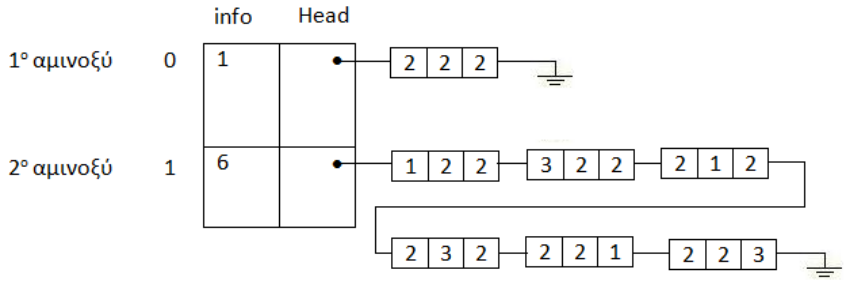
lattice_array[2][2][2].x = -1,777

lattice_array[2][2][2].y = 21,184

lattice_array[2][2][2].z = -12,541

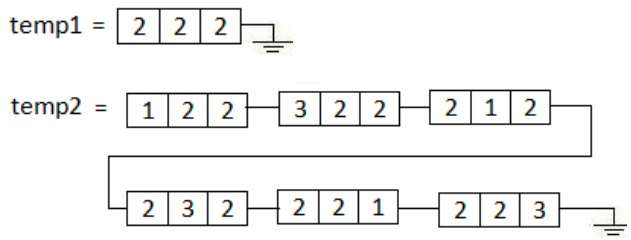
Ο πίνακας possible_points για τα δυο πρώτα αμινοξέα είναι ο εξής (Σχήμα 5.12):

possible_points



Σχήμα 5.12: Ο πίνακας possible_points για τα 2 πρώτα αμινοξέα

Αρχικά ο δείκτης temp1 δείχνει στη λίστα του 1ου αμινοξέος και ο temp2 στη λίστα του 2ου (Σχήμα 5.13)



Σχήμα 5.13: Οι δείκτες temp1 και temp2 δείχνουν στις λίστες του 1ου και του 2ου αμινοξέος αντίστοιχα

Σύμφωνα με τον πίνακα protein_coordinates η πραγματική απόσταση μεταξύ των δυο πρώτων αμινοξέων υπολογίζεται και αποθηκεύεται στην μεταβλητή distance1

distance1=

$$\sqrt{(-1{,}777 + 2{,}323)^2 + (21{,}184 - 22{,}761)^2 + (-12{,}541 + 9{,}108)^2}$$

distance1 = $\sqrt{0{,}286225 + 24{,}86929 + 11{,}785489}$

distance1 = 3,815579

Οι συντεταγμένες της κορυφής [2][2][2] που δείχνει ο temp1 και της κορυφής [1][2][2] που δείχνει ο temp2 είναι γνωστές και αποθηκευμένες στον πίνακα lattice_array και η μεταξύ τους απόσταση υπολογίζεται και αποθηκεύεται στην μεταβλητή distance2.

lattice_array[2][2][2].x = -1,777

lattice_array[2][2][2].y = 21,184

lattice_array[2][2][2].z = -12,541

lattice_array[1][2][2].x = -5,577

lattice_array[1][2][2].y = 21,184

lattice_array[1][2][2].z = -12,541

distance2 = $\sqrt{(-1{,}777 + 5{,}577)^2 + (21{,}184 - 21{,}184)^2 + (-12{,}541 + 12{,}541)^2}$

distance2 = $\sqrt{(3{,}8)^2}$

distance2 = 3,8

Στη συνέχεια υπολογίζεται η πραγματική απόσταση total_distance

total_distance = $|\text{distance1} - \text{distance2}|^2$

total_distance = $|3{,}815578 - 3{,}8|^2$

total_distance = $0{,}015578^2$

total_distance = 0,000243

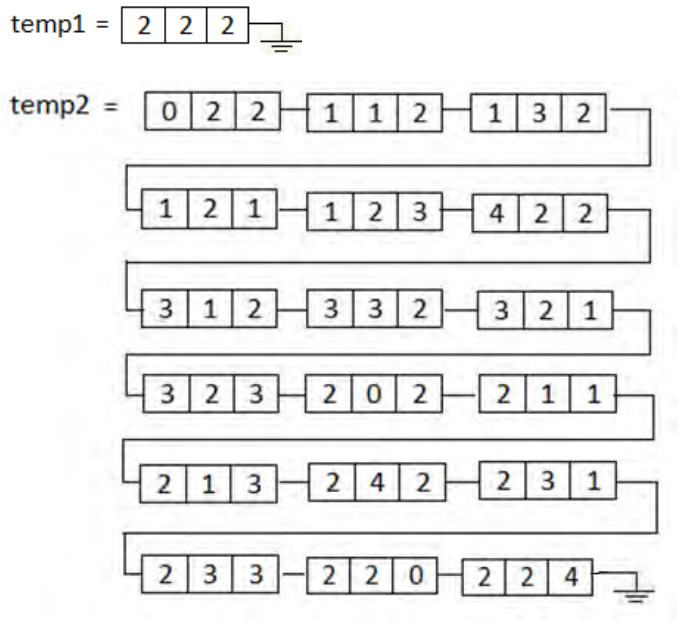
Άρα ο πρώτος όρος της αντικειμενικής συνάρτησης για το DRMS θα είναι

[0,000243X(1,(2,2,2))*X(2,(1,2,2))

Στη συνέχεια ο temp2 δείχνει στην κορυφή [3][2][2] η οποία απέχει και αυτή μια κορυφή από την κεντρική, άρα η συνολική απόσταση total_distance θα είναι ίδια και η αντικειμενική συνάρτηση θα είναι

[0,000243X(1,(2,2,2))*X(2,(1,2,2)) + 0,000243X(1,(2,2,2))*X(2,(3,2,2))

Ο temp2 συνεχίζει να διατρέχει την λίστα για το 2^{ο} αμινοξύ γράφοντας παράλληλα στο αρχείο τις αποστάσεις που υπολογίζονται. Μόλις τελειώσει τη λίστα του 2^{ου} αμινοξέος ο temp2 δείχνει στη λίστα του 3^{ου} αμινοξέος. Το νέο distance1 που θα υπολογιστεί θα είναι η πραγματική απόσταση μεταξύ του 1^{ου} και του 3^{ου} αμινοξέος από τον πίνακα protein_coordinates.



Σχήμα 5.14: Οι δείκτες temp1 και temp2 δείχνουν στις λίστες του 1^{ου} και του 2^{ου} αμινοξέος αντίστοιχα

distance1=

$$\sqrt{(-1{,}777-0{,}609)^2+\ (21{,}84-21{,}091)^2+\ (-12{,}541+7{,}428)^2}$$

distance1 = $\sqrt{(-2{,}386)^2+\ (0{,}093)^2+\ (-5{,}113)^2}$

distance1 = $\sqrt{5{,}692996+0{,}008649+26{,}142769}$

distance1 = 5,643085

Από τις συντεταγμένες που βρίσκονται αποθηκευμένες στον πίνακα lattice_array βρίσκουμε το distance2

lattice_array[2][2][2].x = -1,777

lattice_array[2][2][2].y = 21,184

lattice_array[2][2][2].z = -12,541

lattice_array[0][2][2].x = -9,377

lattice_array[0][2][2].y = 21,184

lattice_array[0][2][2].z = -12,541

distance2 = $\sqrt{(-1{,}777 + 9{,}377)^2}$ = 7.6

total_distance = $|5{,}643085 - 7{,}6|^2 = 1{,}956915^2 = 3{,}829514$

Οι πρώτοι 6 όροι της αντικειμενικής συνάρτησης έχουν την ίδια απόσταση, την 0.000243 και ο $7^{ος}$ όρος θα είναι ο 3.829514. Άρα η αντικειμενική συνάρτηση μέχρι το σημείο αυτό θα είναι

[0,000243X(1,(2,2,2))*X(2,(1,2,2)) + 0,000243X(1,(2,2,2))*X(2,(3,2,2)) +
0,000243X(1,(2,2,2))*X(2,(2,1,2)) + 0,000243X(1,(2,2,2))*X(2,(2,3,2)) +
0,000243X(1,(2,2,2))*X(2,(2,2,1)) + 0,000243X(1,(2,2,2))*X(2,(2,2,3)) +
3,829514X(1,(2,2,2)*X(3,(0,2,2)) + …

Στη συνέχεια ο temp2 θα δείχνει στην επόμενη υποψήφια κορυφή για το $3^{ο}$ αμινοξύ και θα συνεχιστεί η ίδια διαδικασία μέχρι να φτάσει στο τέλος της λίστας. Όταν γίνει αυτό, ο temp1 θα δείχνει στη λίστα του $2^{ου}$ αμινοξέος και ο temp2 στη λίστα του $3^{ου}$.

Η ίδια διαδικασία θα συνεχιστεί μέχρι να υπολογιστούν και να γραφούν στο αρχείο οι αποστάσεις μεταξύ όλων των ζευγών των πιθανών κορυφών σύμφωνα με την αντικειμενική συνάρτηση για το DRMS.

5.11 Η ΣΥΝΑΡΤΗΣΗ constraint1

Η συνάρτηση constraint1 δημιουργεί ένα αρχείο με όνομα con1 και μέσα γράφει τον πρώτο περιορισμό του προβλήματος που αφορά τη σωστή τοποθέτηση των αμινοξέων πάνω στο lattice. Το αρχείο που δημιουργείται είναι τύπου lp για να μπορεί να μπει ως είσοδος στο CPLEX.

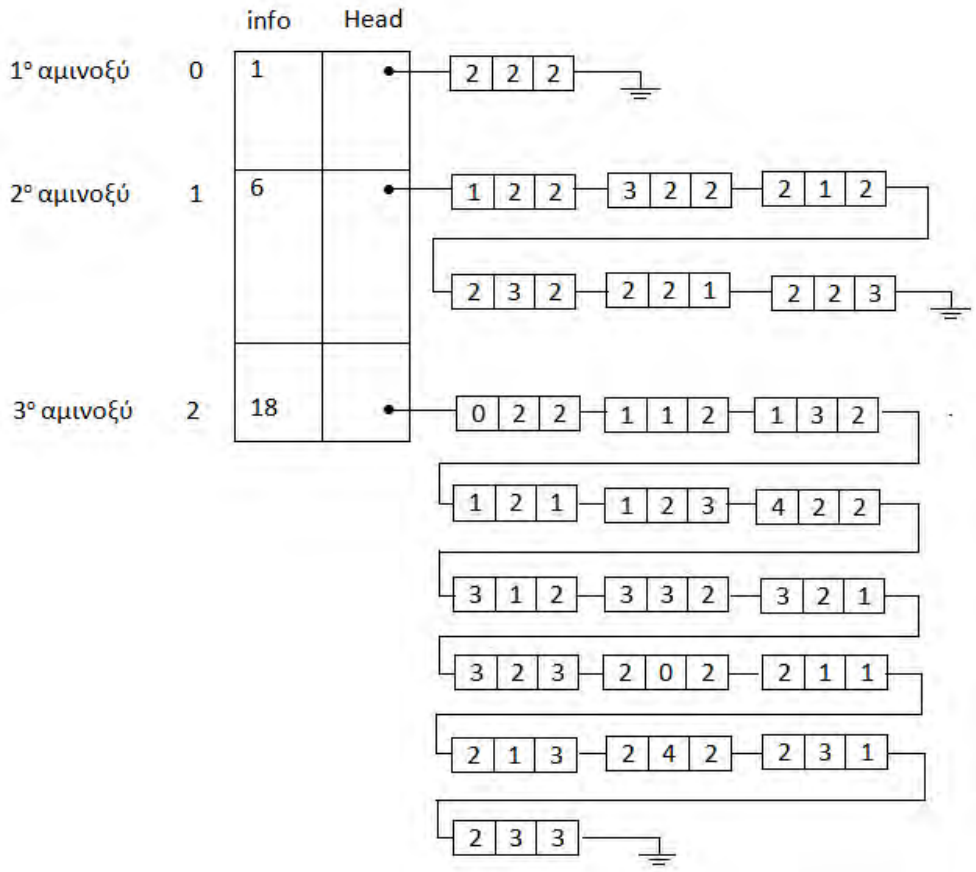
Ο πρώτος περιορισμός του PCLF προβλήματος είναι ο εξής:

Κάθε αμινοξύ p_i τοποθετείται σε μια μοναδική κορυφή q_j του lattice. Δηλαδή το άθροισμα των μεταβλητών που αντιστοιχούν στις υποψήφιες κορυφές για κάθε αμινοξύ ισούται με 1.

Πιο συγκεκριμένα η συνάρτηση διατρέχει τις λίστες όλων των αμινοξέων και για κάθε κορυφή γράφεται η αντίστοιχη μεταβλητή στο αρχείο.

Για να γίνει αυτό, η συνάρτηση constraint1 παίρνει ως είσοδο μόνο τον πίνακα protein_points. Χρησιμοποιείται πάλι μια βοηθητική μεταβλητή temp που διατρέχει την λίστα των υποψήφιων κορυφών όλων των αμινοξέων και ανάλογα με τον κόμβο που βρίσκεται, γράφεται η αντίστοιχη μεταβλητή στο αρχείο σε μορφή αθροίσματος. Μόλις ο δείκτης temp φτάσει στο τέλος της λίστας, γράφεται στο αρχείο "=1" και ο δείκτης temp προχωράει στη λίστα του επόμενου αμινοξέος.

Χρησιμοποιώντας το ίδιο παράδειγμα για τα 3 πρώτα αμινοξέα ο πίνακας possible_points είναι ο εξής (Σχήμα 5.15)



Σχήμα 5.15: Ο πίνακας possible_points για τα 3 πρώτα αμινοξέα

Αρχικά ο temp δείχνει την λίστα του 1ου αμινοξέος, άρα στο αρχείο γράφεται η μεταβλητή X(1,(2,2,2)) και επειδή η λίστα δεν έχει άλλη υποψήφια κορυφή, πριν ο temp προχωρήσει στη λίστα του 2ου αμινοξέος στο αρχείο γράφεται το "=1" και αλλάζει η γραμμή

X(1,(2,2,2)) = 1

Στη συνέχεια ο temp δείχνει στη λίστα του 2ου αμινοξέος και προτού ο temp προχωρήσει σε επόμενο κόμβο, στο αρχείο γράφεται το "+" εκτός αν ο temp έχει φτάσει στο τέλος της λίστας οπότε και γράφεται το "=1".

Άρα η τρίτη γραμμή του αρχείου είναι η εξής

X(2,(1,2,2)) + X(2,(3,2,2)) + X(2,(2,1,2)) + X(2,(2,3,2)) + X(2,(2,2,1)) + X(2,(2,2,3)) = 1

Η συνάρτηση συνεχίζεται με τον ίδιο τρόπο μέχρι ο temp να φτάσει στο τέλος της τελευταίας λίστας του πίνακα possible_points. Μόλις γίνει αυτό, η συνάρτηση constraint1 κλείνει το αρχείο και τερματίζει.

5.12 Η ΣΥΝΑΡΤΗΣΗ constraint2

Η συνάρτηση constraint2 δημιουργεί ένα αρχείο με όνομα con2 και μέσα γράφει το δεύτερο περιορισμό του προβλήματος, ο οποίος αφορά τη σωστή τοποθέτηση των αμινοξέων πάνω στο lattice. Το αρχείο που δημιουργείται είναι τύπου lp για να μπορεί να μπει ως είσοδος στο CPLEX.

Ο δεύτερος περιορισμός του PCLF προβλήματος είναι ο εξής:

Σε κάθε κορυφή του lattice μπορεί να τοποθετηθεί το πολύ ένα αμινοξύ. Δηλαδή, το άθροισμα των μεταβλητών όλων των αμινοξέων που μπορούν να τοποθετηθούν σε μια κορυφή είναι <=1.

Οι υποψήφιες κορυφές ενός ζυγού αμινοξέος (π.χ. $2^{ο}$, $4^{ο}$, $6^{ο}$ κλπ αμινοξύ) είναι και υποψήφιες κορυφές για όλα τα επόμενα ζυγά αμινοξέα, αντίστοιχα και οι υποψήφιες κορυφές ενός μονού αμινοξέος (π.χ. $3^{ο}$, $5^{ο}$, $7^{ο}$ κλπ αμινοξύ) είναι και υποψήφιες κορυφές για όλα τα επόμενα μονά αμινοξέα.

Στη συνάρτηση constraint2, μια βοηθητική μεταβλητή temp1 διατρέχει τις λίστες των υποψήφιων κορυφών από το $2^{ο}$ αμινοξύ και μετά γιατί το πρώτο έχει τοποθετηθεί ήδη στην κεντρική κορυφή.

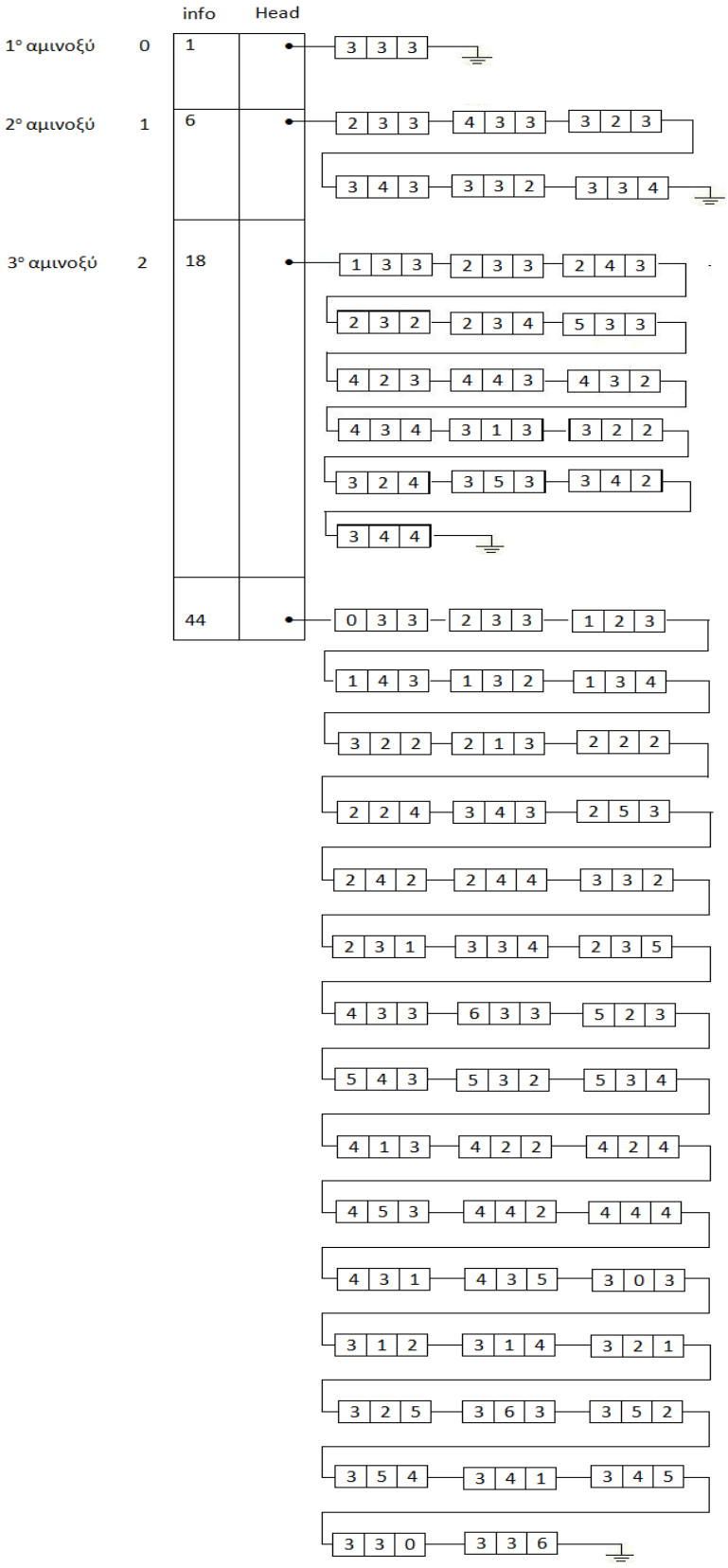
Αρχικά ο δείκτης temp1 διατρέχει τη λίστα του $2^{ου}$ αμινοξέος και για κάθε κόμβο γράφει στο αρχείο την αντίστοιχη μεταβλητή και προσθέτει και τις αντίστοιχες μεταβλητές για όλα τα ζυγά αμινοξέα.

Το ίδιο συμβαίνει και για τη λίστα του $3^{ου}$ αμινοξέος.

Στην περίπτωση του $4^{ου}$ αμινοξέος και για όλα τα υπόλοιπα, χρησιμοποιείται και μια δεύτερη βοηθητική μεταβλητή temp2 που δείχνει στη λίστα του προηγούμενου

ζυγού αμινοξέος (ή μονού για τα αντίστοιχα μονά). Καθώς ο temp1 διατρέχει την λίστα του 4^{ου} αμινοξέος, ο temp2 διατρέχει τη λίστα του 2^{ου} αμινοξέος, για να μη ξαναγραφούν οι ίδιες κορυφές, ενώ για τις κορυφές που δεν έχουν ξαναγραφεί, προστίθενται και οι αντίστοιχες μεταβλητές για τα επόμενα μονά ή ζυγά κ.ο.κ.

Για το ίδιο παράδειγμα, αλλά για 4 αμινοξέα αυτή την φορά, ο πίνακας possible_points θα είναι ο εξής (Σχήμα 5.16):



Σχήμα 5.16: Ο πίνακας possible_points για 4 αμινοξέα

Έτσι στο αρχείο θα γραφούν οι εξής εξισώσεις για τον $2^{ο}$ περιορισμό:

X(2,(2,3,3)) + X(4,(2,3,3)) <=1

X(2,(4,3,3)) + X(4,(4,3,3)) <=1

X(2,(3,2,3)) + X(4,(3,2,3)) <=1

X(2,(3,4,3)) + X(4,(3,4,3)) <=1

X(2,(3,3,2)) + X(4,(3,3,2)) <=1

X(2,(3,3,4)) + X(4,(3,3,4)) <=1

X(3,(1,3,3)) <=1

X(3,(2,2,3)) <=1

…..

…..

…..

X(3,(3,3,1)) <=1

X(3,(3,3,5)) <=1

X(4,(0,3,3)) <=1

X(4,(1,2,3)) <=1

…….

…….

……..

X(4,(3,3,0)) <=1

X(4,(3,3,6)) <=1

5.13 Η ΣΥΝΑΡΤΗΣΗ constraint3

Η συνάρτηση constraint3 δημιουργεί ένα αρχείο με όνομα con3 και μέσα γράφει τον τρίτο περιορισμό του προβλήματος, ο οποίος αφορά τη σωστή τοποθέτηση των αμινοξέων πάνω στο πλέγμα. Το αρχείο που δημιουργείται είναι τύπου lp για να μπορεί να μπει ως είσοδος στο CPLEX.

Ο τρίτος περιορισμός του PCLF προβλήματος, όπως έχει ήδη αναφερθεί, είναι ο εξής:

Δύο διαδοχικά αμινοξέα τοποθετούνται σε δυο διαφορετικές κορυφές του lattice. Δηλαδή, για κάθε υποψήφια κορυφή το άθροισμα της μεταβλητής που αντιστοιχεί σε αυτήν την κορυφή συν το άθροισμα όλων των μεταβλητών που αντιστοιχούν στις υποψήφιες κορυφές του επόμενου αμινοξέος, εκτός από αυτές που είναι γειτονικές της κορυφής, πρέπει να είναι “<=1” .

Πιο συγκεκριμένα, η συνάρτηση διατρέχει ταυτόχρονα τη λίστα κάθε αμινοξέος και του επόμενου του στην πρωτεϊνική αλυσίδα. Για κάθε κορυφή γράφει την αντίστοιχη μεταβλητή στο αρχείο σε μορφή αθροίσματος μαζί με τις υποψήφιες κορυφές που αντιστοιχούν στο επόμενο αμινοξύ, εκτός από αυτές που είναι γειτονικές της.

Παρακάτω αναλύουμε τη λειτουργία της συνάρτησης.

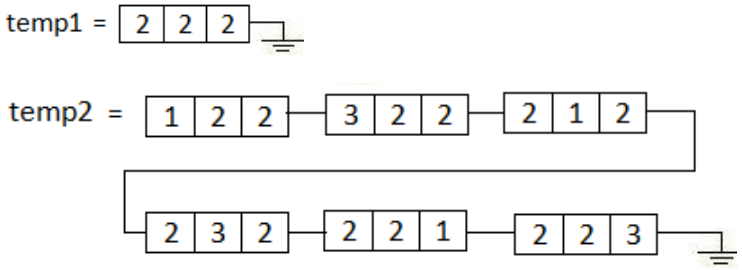
Για να γίνει αυτό η συνάρτηση constraint3 παίρνει ως είσοδο μόνο τον πίνακα possible_points.

Χρησιμοποιούνται δυο βοηθητικές μεταβλητές, οι temp1 και temp2.

Για κάθε αμινοξύ, ο temp1 διατρέχει τη λίστα των υποψήφιων κορυφών του αμινοξέος και ο temp2 διατρέχει τη λίστα των υποψήφιων κορυφών του επόμενου αμινοξέος.

Αρχικά γράφεται στο αρχείο σε μορφή αθροίσματος η μεταβλητή που δείχνει ο δείκτης temp1. Ο δείκτης temp2 διατρέχει την λίστα των υποψήφιων κορυφών του επόμενου αμινοξέος και γράφει στο αρχείο την μεταβλητή που αντιστοιχεί στην υποψήφια κορυφή που δείχνει, εκτός και αν αυτή είναι κάποια από τις γειτονικές κορυφές αυτής που δείχνει ο temp1.

Η συνάρτηση τελειώνει όταν ο temp1 δείχνει στην τελευταία κορυφή της λίστας του προτελευταίου αμινοξέος και ο temp2 διατρέξει την λίστα του τελευταίου.
Με βάση το ίδιο παράδειγμα και για τρία αμινοξέα, αρχικά ο temp1 δείχνει στη λίστα του 1ου αμινοξέος και ο temp2 στη λίστα του 2ου αμινοξέος (Σχήμα 5.17).

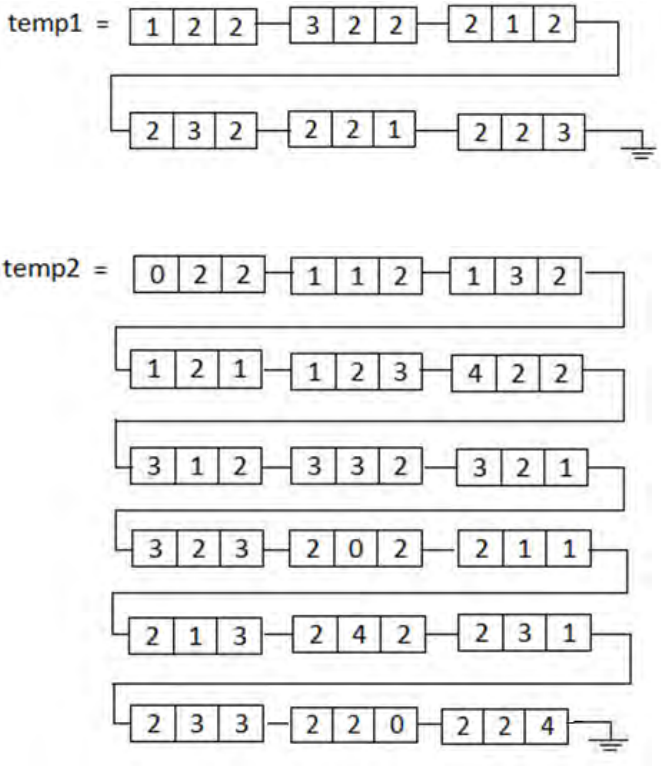


Σχήμα 5.17: Οι δείκτες temp1 και temp2 δείχνουν στις λίστες του 1ου και 2ου αμινοξέος αντίστοιχα

Επειδή όμως, όλες οι κορυφές που δείχνει ο temp2 είναι γειτονικές της temp1, η πρώτη γραμμή που γράφεται στο αρχείο con3 είναι

X(1,(2,2,2)) <= 1

Στη συνέχεια ο temp1 δείχνει στη λίστα του 2ου και ο temp2 στη λίστα του 3ου αμινοξέος (Σχήμα 5.18).



Σχήμα 5.178: Οι δείκτες temp1 και temp2 δείχνουν στις λίστες του 2ου και 3ου αμινοξέος αντίστοιχα

Καθώς ο temp2 διατρέχει τη λίστα, δε γράφει στο αρχείο τις μεταβλητές που αντιστοιχούν στις υποψήφιες κορυφές που είναι γειτονικές της temp1, δηλαδή
οι [0][2][2], [1][1][2], [1][3][2], [1][2][1], [1][2][3], [3][1][2]
Άρα η $2^{η}$ γραμμή του αρχείου θα είναι:
X(2,(1,2,2)) + X(3,(4,2,2)) + X(3,(3,1,2)) + X(3,(3,3,2)) + X(3,(3,2,1)) + X(3,(3,2,3)) + X(3,(2,0,2)) + X(3,(2,1,1)) + X(3,(2,1,3)) + X(3,(2,4,2)) + X(3,(2,3,1)) + X(3,(2,3,3)) + X(3,(2,2,0)) + X(3,(2,2,4)) <= 1

Στη συνέχεια ο temp1 προχωράει στον επόμενο κόμβο, δηλαδή τον [3|2|2] και ο temp2 ξαναδιατρέχει την λίστα των υποψήφιων κορυφών για το $3^{ο}$ αμινοξύ, για να αποκλείσει τις γειτονικές κορυφές της [3|2|2] από το άθροισμα του επόμενου περιορισμού.

Η ίδια διαδικασία συνεχίζεται για όλες τις υποψήφιες κορυφές. Μόλις ολοκληρωθεί ο τρίτος περιορισμός, η συνάρτηση κλείνει το αρχείο και τερματίζει.

5.14 Η ΣΥΝΑΡΤΗΣΗ binaries

Η συνάρτηση binaries δημιουργεί ένα αρχείο με όνομα binaries και μέσα γράφει τις μεταβλητές του προβλήματος, καθώς επίσης και τα όρια των μεταβλητών. Το αρχείο που δημιουργείται είναι τύπου lp για να μπορεί να μπει ως είσοδος στο CPLEX.

Η συνάρτηση binaries παίρνει ως είσοδο τον πίνακα possible_points και χρησιμοποιώντας μια βοηθητική μεταβλητή temp, διατρέχει τις λίστες των υποψήφιων κορυφών όλων των αμινοξέων και γράφει στο αρχείο την αντίστοιχη μεταβλητή με τα όριά της.

ΚΕΦΑΛΑΙΟ 6.

ΤΟ ΠΡΟΓΡΑΜΜΑ CPLEX

Όπως αναφέραμε και στο προηγούμενο κεφάλαιο, στο $2^{ο}$ μέρος της επίλυσης του PCLF προβλήματος, το πρόγραμμα CPLEX παίρνει τα αρχεία που δημιουργήθηκαν στο πρόγραμμα της C και περιέχουν ορισμένο το PCLF πρόβλημα σε μορφή μαθηματικού προγραμματισμού και μας επιστρέφει τη λύση του προβλήματος. Πιο συγκεκριμένα, το πρόγραμμα CPLEX επιστρέφει τη τιμή της αντικειμενικής συνάρτησης, καθώς επίσης και τις τιμές όλων των μεταβλητών του προβλήματος.

Στο κεφάλαιο αυτό θα μιλήσουμε πιο αναλυτικά για το πρόγραμμα CPLEX, τους λόγους που το επιλέξαμε και με ποιον τρόπο το χρησιμοποιήσαμε.

6.1 ΤΟ ΠΡΟΓΡΑΜΜΑ IBM ILOG CPLEX

Το πρόγραμμα IBM ILOG CPLEX είναι ένα εμπορικό πρόγραμμα που χρησιμοποιείται ευρύτατα για τη λύση προβλημάτων μαθηματικού προγραμματισμού. Το πρόγραμμα αναπτύσσεται από την εταιρία IBM, στην οποία ανήκουν τα πνευματικά δικαιώματα του προγράμματος και την οποία ευχαριστούμε για τη δωρεάν παραχώρησή του για ερευνητικούς σκοπούς.

Το ILOG CPLEX είναι ένα εργαλείο για την επίλυση μαθηματικών προβλημάτων βελτιστοποίησης, που συνήθως αναφέρονται ως προβλήματα Γραμμικού Προγραμματισμού (Linear Programming LP). Επίσης μπορεί να λύσει προβλήματα Μη-Γραμμικού προγραμματισμού, καθώς επίσης και προβλήματα Μεικτού Ακέραιου Προγραμματισμού (Mixed Integer Programming MIP), όπου κάποιες ή όλες οι μεταβλητές (LP ή QP) παίρνουν ακέραιες τιμές.

Το πρόγραμμα έχει πολύ περισσότερες δυνατότητες από αυτές που το χρησιμοποιήσαμε εμείς και ο ενδιαφερόμενος μπορεί να ανατρέξει στο manual αυτού του πακέτου που αναλύει όλες του τις δυνατότητες. [6]

6.2 Η ΧΡΗΣΗ ΤΟΥ CPLEX ΣΤΗΝ ΠΑΡΟΥΣΑ ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Ο λόγος που επιλέξαμε το πρόγραμμα ILOG CPLEX είναι το ότι μπορεί να επιλύσει τόσο προβλήματα γραμμικού προγραμματισμού, όπως είναι ορισμένο το PCLF πρόβλημα με κριτήριο το CRMS, όσο και προβλήματα μη- γραμμικού προγραμματισμού, όπως είναι ορισμένο το PCLF πρόβλημα με κριτήριο το DRMS. Επίσης είναι αρκετά εύκολο στη χρήση του και δέχεται αρχεία σε διάφορες μορφές που μπορούν να παραχθούν από άλλα προγράμματα. Επιπλέον είναι ένα πρόγραμμα που δεν έχει περιορισμούς όσον αφορά το πλήθος των μεταβλητών και των εξισώσεων. Τέλος, πολύ σημαντικό κριτήριο είναι το ότι το πρόγραμμα CPLEX μπορεί να επιλέξει αυτόματα τον καλύτερο αλγόριθμο για την επίλυση ενός συγκεκριμένου προβλήματος, με βάση τη δομή του.

Στην παρούσα πτυχιακή εργασία έγινε χρήση του CPLEX Interactive Optimizer που περιλαμβάνει τις εξής εντολές:

- add: προσθέτει περιορισμούς στο πρόβλημα
- baropt: λύνει το πρόβλημα χρησιμοποιώντας τον αλγόριθμο barrier
- change: αλλάζει το πρόβλημα
- conflict: βελτιώνει κάποιο ανέφικτο πρόβλημα, για παράδειγμα, σε περίπτωση που αντικρούονται κάποιοι περιορισμοί
- display: παρουσιάζει το πρόβλημα, τη λύση, τις μεταβλητές, τις παραμέτρους κλπ
- enter: εισάγει ένα καινούριο πρόβλημα
- feasopt: γίνεται χαλάρωση στους περιορισμούς σε ένα ανέφικτο πρόβλημα
- help: προβάλει πληροφορίες για τις εντολές του CPLEX
- mipopt: λύνει ένα πρόβλημα μεικτού ακέραιου προγραμματισμού
- optimize: λύνει το πρόβλημα
- primopt: λύνει το πρόβλημα χρησιμοποιώντας τη μέθοδο primal
- quit: γίνεται έξοδος από το CPLEX
- read: διαβάζει το πρόβλημα από αρχείο
- set: θέτει παραμέτρους
- tranopt: λύνει το πρόβλημα χρησιμοποιώντας τη μέθοδο dual
- tune: δοκιμάζει διάφορες παραμέτρους
- write: γράφει το πρόβλημα και τη λύση του σε κάποιο αρχείο

- xecute: εκτελεί κάποια εντολή του συστήματος

Όπως έχει αναφερθεί στο προηγούμενο κεφάλαιο, το πρόγραμμα που δημιουργήθηκε στο $1^{ο}$ μέρος της επίλυσης του PCLF προβλήματος, επιστρέφει τα εξής 6 αρχεία:

1. input_cplex_crms: περιλαμβάνει την αντικειμενική συνάρτηση του PCLF προβλήματος με κριτήριο το CRMS.
2. input_cplex_drms: περιλαμβάνει την αντικειμενική συνάρτηση του PCLF προβλήματος με κριτήριο το DRMS.
3. con1: περιλαμβάνει τον πρώτο περιορισμό του PCLF προβλήματος.
4. con2: περιλαμβάνει τον δεύτερο περιορισμό του PCLF προβλήματος.
5. con3: περιλαμβάνει τον τρίτο περιορισμό του PCLF προβλήματος.
6. binaries: περιλαμβάνει τις δυαδικές μεταβλητές του PCLF προβλήματος.

Τα αρχεία αυτά είναι τύπου .lp για να μπορούν να διαβαστούν από το CPLEX.

Αρχικά με την εντολή read διαβάζουμε την αντικειμενική συνάρτηση (cRMS ή dRMS).

- read input_cplex_crms.lp
- read input_cplex_drms.lp

Στη συνέχεια προσθέτουμε τα αρχεία με τους περιορισμούς και τις μεταβλητές

- add con1
- add con2
- add con3
- add binaries

Μόλις ολοκληρωθεί η διαδικασία όπου το CPLEX διαβάζει τα αρχεία, δηλαδή τα δεδομένα του προβλήματος, λύνουμε το πρόβλημα

- optimize

Τέλος, μόλις το CPLEX ολοκληρώσει την διαδικασία βελτιστοποίησης, αποθηκεύουμε τη λύση

- write lysi (δημιουργείται ένα αρχείο που περιλαμβάνει το πρόβλημα)
- write lysi.sol (δημιουργείται ένα αρχείο που περιλαμβάνει την τιμή της αντικειμενικής συνάρτησης αλλά και όλων των μεταβλητών)

ΚΕΦΑΛΑΙΟ 7.

ΠΕΙΡΑΜΑΤΙΚΑ ΑΠΟΤΕΛΕΣΜΑΤΑ

Για την εξαγωγή των αποτελεσμάτων χρησιμοποιήθηκαν δεδομένα από την PDB.

Πιο συγκεκριμένα, τα πειραματικά μας αποτελέσματά προέρχονται από συνολικά 80 πρωτεΐνες, χωρισμένες σε 4 κατηγορίες σύμφωνα με την SCOP που είναι οι εξής:

- 20 πρωτεΐνες που ανήκουν στην κατηγορία ALPHA
- 20 πρωτεΐνες που ανήκουν στην κατηγορία BETA
- 20 πρωτεΐνες που ανήκουν και στην κατηγορία ALPHA και στην κατηγορία Beta
- 20 άλλες πρωτεΐνες που δεν ανήκουν ούτε στην κατηγορία ALPHA ούτε στην κατηγορία BETA

Δυστυχώς η επίλυση προβλημάτων μη-γραμμικού προγραμματισμού δεν απαιτεί μόνο πολύ χρόνο αλλά και τεράστια μνήμη. Έτσι, με το υπολογιστικό σύστημα που είχαμε στη διάθεση μας, μπορέσαμε να υπολογίσουμε τη λύση του προβλήματος μόνο για πρωτεϊνικές αλυσίδες 5 αμινοξέων. Παρ' όλα αυτά, στόχος ήταν η ανάπτυξη μιας μεθόδου που λύνει το πρόβλημα για οποιοδήποτε μέγεθος.

Τα πειραματικά αποτελέσματα προέκυψαν από την τοποθέτηση των παραπάνω πρωτεϊνών σε 5 διαφορετικά πλέγματα.

7.1 ΠΕΙΡΑΜΑΤΙΚΑ ΑΠΟΤΕΛΕΣΜΑΤΑ ΠΛΕΓΜΑΤΟΣ ΜΕΓΕΘΟΥΣ (3.8 – 3.8 – 3.8)

7.1.1 ΠΡΩΤΕΪΝΕΣ ΠΟΥ ΑΝΗΚΟΥΝ ΣΤΗΝ ΚΑΤΗΓΟΡΙΑ ALPHA

Στον παρακάτω πίνακα (Πίνακας 7.1) φαίνονται τα πειραματικά αποτελέσματα για τις 20 πρωτεΐνες που ανήκουν στην κατηγορία ALPHA.

PDB ID	CLASSIFICATION	CRMS	DRMS
1B0B	OXYGEN STORAGE/TRANSPORT	2,239	0,577
3DLY	OXYGEN STORAGE/TRANSPORT	2,621	0,859
1H97	OXYGEN TRANSPORT	2,214	0,678
1LHS	OXYGEN STORAGE	1,999	0,574
1LHT	OXYGEN STORAGE	1,974	0,595
1MBW	OXYGEN TRANSPORT	2,332	0,806
1UVX	OXYGEN STORAGE/TRANSPORT	2,644	0,779
1UVY	OXYGEN STORAGE/TRANSPORT	2,305	0,89
2RD6	TRANSFERASE	1,894	0,546
2RJV	STRUCTURAL PROTEIN	2,191	0,619
2VAY	METAL TRANSPORT	2,657	**0,447**
2ZB5	VIRAL PROTEIN	2,223	0,892
3B75	TRANSPORT PROTEIN, OXYGEN BINDING	2,388	0,599
3C1B	STRUCTURAL PROTEIN / DNA	2,135	0,803
3DIK	VIRAL PROTEIN	1,623	0,72
3DOG	TRANSFERASE, CELL CYCLE	2,048	0,692
3DPY	TRANSFERASE	2,067	0,625
3DU7	CELL CYCLE	**3,782**	0,646
3DY6	TRANSCRIPTION	2,574	**0,907**
3DYN	HYDROLASE	**1,511**	0,636
Μέσος Όρος		**2,271**	**0,695**

Πίνακας 7.1: Πίνακας πειραματικών αποτελεσμάτων για τις πρωτεΐνες που ανήκουν στην κατηγορία ALPHA

Σύμφωνα με τον πίνακα των αποτελεσμάτων για τις πρωτεΐνες που ανήκουν στην κατηγορία ALPHA, έχοντας ως κριτήριο το CRMS, ο μέσος όρος είναι 2,271, η μεγαλύτερη τιμή είναι 3,782, η μικρότερη 1,511 και η διαφορά τους 2,271. Έχοντας ως κριτήριο το DRMS, ο μέσος όρος είναι 0,695, η μεγαλύτερη τιμή είναι 0,907, η μικρότερη 0,447 και η διαφορά τους 0,46.

7.1.2 ΠΡΩΤΕΪΝΕΣ ΠΟΥ ΑΝΗΚΟΥΝ ΣΤΗΝ ΚΑΤΗΓΟΡΙΑ BETA

Στον παρακάτω πίνακα (Πίνακας 7.2) φαίνονται τα πειραματικά αποτελέσματα για τις 20 πρωτεΐνες που ανήκουν στην κατηγορία BETA.

PDB ID	CLASSIFICATION	CRMS	DRMS
1AYO	MACROGLOBULIN	2,404	0,579
1BV8	PROTEIN BINDING	2,236	0,701
1BW8	PEPTIDE BINDING PROTEIN	2,693	1,28
1BXX	ENDOCYTOSIS/EXOCYTOSIS	2,669	**1,287**
1EDY	PROTEIN BINDING	2,163	0,57
1HES	ENDOCYTOSIS/EXOCYTOSIS	2,572	1,283
2BP5	ENDOCYTOSIS	2,484	1,279
2PR9	ENDOCYTOSIS	2,33	1,248
2VO8	HYDROLASE	1,798	**0,499**
2Z7F	HYDROLASE/HYDROLASE INHIBITOR	2,107	1,105
2Z7U	OXIDOREDUCTASE	**1,796**	1,185
3BGB	HYDROLASE	2,302	0,944
3BN9	HYDROLASE	2,807	1,117
2C2S	IMMUNE SYSTEM	**2,868**	0,776
3CCB	HYDROLASE	2,467	0,714
3D7P	TRANSPORT PROTEIN	2,13	1,06
3D93	LYASE	2,304	0,77
3DUH	IMMUNE SYSTEM/CYTOKINE	2,112	0,646
3DWH	LIGASE	2,522	0,851
3ELN	OXIDOREDUCTASE	2,207	1,212
Μέσος Όρος		**2,348**	**0,955**

Πίνακας 7.2: Πίνακας πειραματικών αποτελεσμάτων για τις πρωτεΐνες που ανήκουν στην κατηγορία BETA

Σύμφωνα με τον πίνακα των αποτελεσμάτων για τις πρωτεΐνες που ανήκουν στην κατηγορία BETA, έχοντας ως κριτήριο το CRMS, ο μέσος όρος είναι 2,348, η μεγαλύτερη τιμή είναι 2,868, η μικρότερη 1,796 και η διαφορά τους 1,072. Έχοντας

ως κριτήριο το DRMS, ο μέσος όρος είναι 0,955, η μεγαλύτερη τιμή είναι 1,287, η μικρότερη 0,499 και η διαφορά τους 0,788.

7.1.3 ΠΡΩΤΕΪΝΕΣ ΠΟΥ ΑΝΗΚΟΥΝ ΚΑΙ ΣΤΗΝ ΚΑΤΗΓΟΡΙΑ ALPHA ΚΑΙ ΣΤΗΝ ΚΑΤΗΓΟΡΙΑ BETA

Στον παρακάτω πίνακα (Πίνακας 7.3) φαίνονται τα πειραματικά αποτελέσματα για τις 20 πρωτεΐνες που ανήκουν και στην κατηγορία ALPHA στην κατηγορία BETA.

PDB ID	CLASSIFICATION	CRMS	DRMS
1ZQ1	LYASE	2,36	0,787
2AEH	TRANSFERASE	2,783	0,821
2AL6	TRANSFERASE	2,305	0,804
2BSI	CHAPERONE	2,307	0,828
2C35	TRANSFERASE	2,838	0,504
2E2H	TRANSCRIPTION,TRANSFERASE/DNA RNA HYBRID	2,162	1,295
2ELB	PROTEIN BINDING	2,747	0,626
2ERJ	IMMUNE SYSTEM/CYTOKINE	1,582	0,625
2F5Q	HYDROLASE, DNA	2,589	0,630
2F7Q	HYDROLASE	1,976	**1,564**
2GZW	TRANSCRIPTION	**1,108**	0,663
2HUE	DNA BINDING PROTEIN	2,499	0,583
2NZ8	SIGNALING PROTEIN, CELL CYCLE	1,791	1,01
2OCE	STRUCTURAL GENOMICS, UNKNOWN FUNCTION	2,251	0,88
2ZKM	HYDROLASE	2,364	0,581
3BXX	OXIDOREDUCTASE	2,771	0,869
3CC4	RIBOSOME	2,22	0,681
3CZN	HYDROLASE	1,913	**0,49**
3E5U	TRANSCRIPTION REGULATION	2,256	0,738
3E6C	TRANSCRIPTION REGULATION/DNA	**3,209**	0,938
Μέσος Όρος		**2,302**	**0,938**

Πίνακας 7.3: Πίνακας πειραματικών αποτελεσμάτων για τις πρωτεΐνες που ανήκουν και στην κατηγορία ALPHA και στην κατηγορία BETA

Σύμφωνα με τον πίνακα των αποτελεσμάτων για τις πρωτεΐνες που ανήκουν και στην κατηγορία ALPHA και στην κατηγορία BETA, έχοντας ως κριτήριο το CRMS, ο μέσος όρος είναι 2,302, η μεγαλύτερη τιμή είναι 3,209, η μικρότερη 1,108 και η διαφορά τους 2,101. Έχοντας ως κριτήριο το DRMS, ο μέσος όρος είναι 0,938, η μεγαλύτερη τιμή είναι 1,564, η μικρότερη 0,49 και η διαφορά τους 1,074.

7.1.4 ΠΡΩΤΕΪΝΕΣ ΠΟΥ ΔΕΝ ΑΝΗΚΟΥΝ ΟΥΤΕ ΣΤΗΝ ΚΑΤΗΓΟΡΙΑ ALPHA, ΟΥΤΕ ΣΤΗΝ ΚΑΤΗΓΟΡΙΑ BETA

Στον παρακάτω πίνακα (Πίνακας 7.4) φαίνονται τα πειραματικά αποτελέσματα για τις 20 πρωτεΐνες που δεν ανήκουν ούτε στην κατηγορία ALPHA, ούτε στην κατηγορία BETA.

PDB ID	CLASSIFICATION	CRMS	DRMS
1BB1	DE NOVO PROTEIN DESIGN	2,096	0,817
1BKV	STRUCTURAL PROTEIN	**5,33**	**1,936**
1FUL	CELL ADHESION	2,363	**0,324**
1QP6	DE NOVO PROTEIN	1,933	0,822
1RH4	COILED COIL	2,265	0,757
1YIW	TRANSCRIPTION, TRANSFERASE/DNA RNA HYBRID	1,948	0,936
2A3D	THREE-HELIX BUNDLE	2,486	0,724
2EL5	TRANSCRIPTION	2,674	1,143
2ENT	TRANSCRIPTION	**1,54**	0,758
2FCM	STRUCTURAL PROTEIN	2,499	0,878
2JUA	DE NOVO PROTEIN	2,612	0,606
2JVF	DE NOVO PROTEIN	2,931	1,116
2O6N	DE NOVO PROTEIN	2,087	0,709
2OP7	LIGASE	2,608	0,783
2QYZ	STRUCTURAL GENOMICS, UNKNOWN FUNCTION	2,515	0,815
1YMZ	UNKNOWN FUNCTION	2,662	0,746
2YRH	CELL CYCLE	1,887	0,633
2ZCB	SIGNALING PROTEIN	2,203	0,915
2ZGG	DE NOVO PROTEIN	2,433	0,78
3CSR	PROTEIN BINDING	2,113	0,918
Μέσος Όρος		**2,459**	**0,859**

Πίνακας 7.4: Πίνακας πειραματικών αποτελεσμάτων για τις πρωτεΐνες που δεν ανήκουν ούτε στην κατηγορία ALPHA ούτε στην κατηγορία BETA

Σύμφωνα με τον πίνακα των αποτελεσμάτων για τις πρωτεΐνες που δεν ανήκουν ούτε στην κατηγορία ALPHA ούτε στην κατηγορία BETA, έχοντας ως κριτήριο το

CRMS, ο μέσος όρος είναι 2,459, η μεγαλύτερη τιμή είναι 5,33, η μικρότερη 1,54 και η διαφορά τους 3,79. Έχοντας ως κριτήριο το DRMS, ο μέσος όρος είναι 0,859, η μεγαλύτερη τιμή είναι 1,936, η μικρότερη 0,324 και η διαφορά τους 1,612.

7.2 ΠΕΙΡΑΜΑΤΙΚΑ ΑΠΟΤΕΛΕΣΜΑΤΑ ΠΛΕΓΜΑΤΟΣ ΜΕΓΕΘΟΥΣ (2 – 3 – 1)

7.2.1 ΠΡΩΤΕΪΝΕΣ ΠΟΥ ΑΝΗΚΟΥΝ ΣΤΗΝ ΚΑΤΗΓΟΡΙΑ ALPHA

Στον παρακάτω πίνακα (Πίνακας 7.5) φαίνονται τα πειραματικά αποτελέσματα για τις 20 πρωτεΐνες που ανήκουν στην κατηγορία ALPHA.

PDB ID	CLASSIFICATION	CRMS	DRMS
1B0B	OXYGEN STORAGE/TRANSPORT	2,509	1,699
3DLY	OXYGEN STORAGE/TRANSPORT	2,209	1,25
1H97	OXYGEN TRANSPORT	4,612	1,814
1LHS	OXYGEN STORAGE	5,398	1,958
1LHT	OXYGEN STORAGE	5,276	1,932
1MBW	OXYGEN TRANSPORT	2,55	1,279
1UVX	OXYGEN STORAGE/TRANSPORT	2,195	1,291
1UVY	OXYGEN STORAGE/TRANSPORT	2,499	**1,212**
2RD6	TRANSFERASE	3,717	1,907
2RJV	STRUCTURAL PROTEIN	4,348	1,842
2VAY	METAL TRANSPORT	5,591	2,133
2ZB5	VIRAL PROTEIN	4,452	1,693
3B75	TRANSPORT PROTEIN, OXYGEN BINDING	3,149	1,975
3C1B	STRUCTURAL PROTEIN / DNA	4,155	2,383
3DIK	VIRAL PROTEIN	2,348	2,404
3DOG	TRANSFERASE, CELL CYCLE	3,667	1,535
3DPY	TRANSFERASE	**1,455**	2,062
3DU7	CELL CYCLE	**5,944**	**2,771**
3DY6	TRANSCRIPTION	2,578	1,261
3DYN	HYDROLASE	2,913	1,971
Μέσος Όρος		**3,578**	**1,819**

Πίνακας 7.5: Πίνακας πειραματικών αποτελεσμάτων για τις πρωτεΐνες που ανήκουν στην κατηγορία ALPHA

Σύμφωνα με τον πίνακα των αποτελεσμάτων για τις πρωτεΐνες που ανήκουν στην κατηγορία ALPHA, έχοντας ως κριτήριο το CRMS, ο μέσος όρος είναι 3,578, η μεγαλύτερη τιμή είναι 5,944, η μικρότερη 1,455 και η διαφορά τους 4,489. Έχοντας ως κριτήριο το DRMS, ο μέσος όρος είναι 1,819, η μεγαλύτερη τιμή είναι 2,771, η μικρότερη 1,212 και η διαφορά τους 1,559.

7.2.2 ΠΡΩΤΕΪΝΕΣ ΠΟΥ ΑΝΗΚΟΥΝ ΣΤΗΝ ΚΑΤΗΓΟΡΙΑ BETA

Στον παρακάτω πίνακα (Πίνακας 7.6) φαίνονται τα πειραματικά αποτελέσματα για τις 20 πρωτεΐνες που ανήκουν στην κατηγορία BETA.

PDB ID	CLASSIFICATION	CRMS	DRMS
1AYO	MACROGLOBULIN	4,358	1,791
1BV8	PROTEIN BINDING	4,306	1,436
1BW8	PEPTIDE BINDING PROTEIN	2,673	1,361
1BXX	ENDOCYTOSIS/EXOCYTOSIS	2,666	1,324
1EDY	PROTEIN BINDING	3,237	1,881
1HES	ENDOCYTOSIS/EXOCYTOSIS	2,618	**1,295**
2BP5	ENDOCYTOSIS	2,472	1,33
2PR9	ENDOCYTOSIS	3,554	1,428
2VO8	HYDROLASE	5,442	2,284
2Z7F	HYDROLASE/HYDROLASE INHIBITOR	2,903	1,718
2Z7U	OXIDOREDUCTASE	4,412	1,791
3BGB	HYDROLASE	**5,798**	2,281
3BN9	HYDROLASE	3,751	1,655
2C2S	IMMUNE SYSTEM	5,603	2,15
3CCB	HYDROLASE	5,35	2,302
3D7P	TRANSPORT PROTEIN	**2,395**	1,772
3D93	LYASE	5,145	1,954
3DUH	IMMUNE SYSTEM/CYTOKINE	4,08	**2,33**
3DWH	LIGASE	3,98	1,396
3ELN	OXIDOREDUCTASE	2,626	1,579
Μέσος Όρος		**3,868**	**1,753**

Πίνακας 7.6: Πίνακας πειραματικών αποτελεσμάτων για τις πρωτεΐνες που ανήκουν στην κατηγορία BETA

Σύμφωνα με τον πίνακα των αποτελεσμάτων για τις πρωτεΐνες που ανήκουν στην κατηγορία ΒΕΤΑ, έχοντας ως κριτήριο το CRMS, ο μέσος όρος είναι 3,868, η μεγαλύτερη τιμή είναι 5,798, η μικρότερη 2,395 και η διαφορά τους 3,403. Έχοντας ως κριτήριο το DRMS, ο μέσος όρος είναι 1,753, η μεγαλύτερη τιμή είναι 2,33, η μικρότερη 1,295 και η διαφορά τους 1,035.

7.2.3 ΠΡΩΤΕΪΝΕΣ ΠΟΥ ΑΝΗΚΟΥΝ ΚΑΙ ΣΤΗΝ ΚΑΤΗΓΟΡΙΑ ALPHA ΚΑΙ ΣΤΗΝ ΚΑΤΗΓΟΡΙΑ BETA

Στον παρακάτω πίνακα (Πίνακας 7.7) φαίνονται τα πειραματικά αποτελέσματα για τις 20 πρωτεΐνες που ανήκουν και στην κατηγορία ALPHA στην κατηγορία BETA.

PDB ID	CLASSIFICATION	CRMS	DRMS
1ZQ1	LYASE	2,389	1,369
2AEH	TRANSFERASE	6,009	**2,665**
2AL6	TRANSFERASE	4,429	2,071
2BSI	CHAPERONE	**2,252**	1,29
2C35	TRANSFERASE	5,745	2,187
2E2H	TRANSCRIPTION,TRANSFERASE/DNA RNA HYBRID	2,831	1,328
2ELB	PROTEIN BINDING	4,391	1,847
2ERJ	IMMUNE SYSTEM/CYTOKINE	2,542	2,166
2F5Q	HYDROLASE, DNA	3,328	1,74
2F7Q	HYDROLASE	4,071	1,806
2GZW	TRANSCRIPTION	2,5	1,534
2HUE	DNA BINDING PROTEIN	3,833	1,938
2NZ8	SIGNALING PROTEIN, CELL CYCLE	5,775	2,426
2OCE	STRUCTURAL GENOMICS, UNKNOWN FUNCTION	3,005	**1,223**
2ZKM	HYDROLASE	4,692	2,079
3BXX	OXIDOREDUCTASE	4,495	2,299
3CC4	RIBOSOME	5,017	2,143
3CZN	HYDROLASE	4,029	1,788
3E5U	TRANSCRIPTION REGULATION	4,812	2,174
3E6C	TRANSCRIPTION REGULATION/DNA	**6,102**	2,6
Μέσος Όρος		**4,112**	**1,934**

Πίνακας 7.7: Πίνακας πειραματικών αποτελεσμάτων για τις πρωτεΐνες που ανήκουν και στην κατηγορία ALPHA και στην κατηγορία BETA

Σύμφωνα με τον πίνακα των αποτελεσμάτων για τις πρωτεΐνες που ανήκουν και στην κατηγορία ALPHA και στην κατηγορία BETA, έχοντας ως κριτήριο το CRMS, ο μέσος όρος είναι 4,112, η μεγαλύτερη τιμή είναι 6,102, η μικρότερη 2,252 και η

διαφορά τους 3,85. Έχοντας ως κριτήριο το DRMS, ο μέσος όρος είναι 1,934, η μεγαλύτερη τιμή είναι 2,665, η μικρότερη 1,223 και η διαφορά τους 1,442.

7.2.4 ΠΡΩΤΕΪΝΕΣ ΠΟΥ ΔΕΝ ΑΝΗΚΟΥΝ ΟΥΤΕ ΣΤΗΝ ΚΑΤΗΓΟΡΙΑ ALPHA, ΟΥΤΕ ΣΤΗΝ ΚΑΤΗΓΟΡΙΑ BETA

Στον παρακάτω πίνακα (Πίνακας 7.8) φαίνονται τα πειραματικά αποτελέσματα για τις 20 πρωτεΐνες που δεν ανήκουν ούτε στην κατηγορία ALPHA, ούτε στην κατηγορία BETA.

PDB ID	CLASSIFICATION	CRMS	DRMS
1BB1	DE NOVO PROTEIN DESIGN	2,409	1,277
1BKV	STRUCTURAL PROTEIN	**8,146**	**4,944**
1FUL	CELL ADHESION	3,505	1,723
1QP6	DE NOVO PROTEIN	2,361	1,618
1RH4	COILED COIL	**2,14**	1,323
1YIW	TRANSCRIPTION, TRANSFERASE/DNA RNA HYBRID	5,941	2,462
2A3D	THREE-HELIX BUNDLE	2,18	1,327
2EL5	TRANSCRIPTION	2,648	1,532
2ENT	TRANSCRIPTION	2,324	1,809
2FCM	STRUCTURAL PROTEIN	6,17	2,574
2JUA	DE NOVO PROTEIN	4,357	1,475
2JVF	DE NOVO PROTEIN	3,031	1,615
2O6N	DE NOVO PROTEIN	2,164	1,305
2OP7	LIGASE	3,678	1,421
2QYZ	STRUCTURAL GENUMICS, UNKNOWN FUNCTION	2,81	1,311
1YMZ	UNKNOWN FUNCTION	5,578	2,269
2YRH	CELL CYCLE	4,312	1,834
2ZCB	SIGNALING PROTEIN	5,288	2,516
2ZGG	DE NOVO PROTEIN	2,718	**1,276**
3CSR	PROTEIN BINDING	3,441	2,171
Μέσος Όρος		**3,76**	**1,889**

Πίνακας 7.8: Πίνακας πειραματικών αποτελεσμάτων για τις πρωτεΐνες που δεν ανήκουν ούτε στην κατηγορία ALPHA ούτε στην κατηγορία BETA

Σύμφωνα με τον πίνακα των αποτελεσμάτων για τις πρωτεΐνες που δεν ανήκουν ούτε στην κατηγορία ALPHA ούτε στην κατηγορία BETA, έχοντας ως κριτήριο το CRMS, ο μέσος όρος είναι 3,76, η μεγαλύτερη τιμή είναι 8,146, η μικρότερη 2,14 και η διαφορά τους 6,006. Έχοντας ως κριτήριο το DRMS, ο μέσος όρος είναι 1,889, η μεγαλύτερη τιμή είναι 4,944, η μικρότερη 1,276 και η διαφορά τους 3,332.

7.3 ΠΕΙΡΑΜΑΤΙΚΑ ΑΠΟΤΕΛΕΣΜΑΤΑ ΠΛΕΓΜΑΤΟΣ ΜΕΓΕΘΟΥΣ (4 – 4 – 4)

7.3.1 ΠΡΩΤΕΪΝΕΣ ΠΟΥ ΑΝΗΚΟΥΝ ΣΤΗΝ ΚΑΤΗΓΟΡΙΑ ALPHA

Στον παρακάτω πίνακα (Πίνακας 7.9) φαίνονται τα πειραματικά αποτελέσματα για τις 20 πρωτεΐνες που ανήκουν στην κατηγορία ALPHA.

PDB ID	CLASSIFICATION	CRMS	DRMS
1B0B	OXYGEN STORAGE/TRANSPORT	2,356	**0,618**
3DLY	OXYGEN STORAGE/TRANSPORT	2,658	0,851
1H97	OXYGEN TRANSPORT	2,203	0,652
1LHS	OXYGEN STORAGE	2,003	0,66
1LHT	OXYGEN STORAGE	1,992	0,697
1MBW	OXYGEN TRANSPORT	2,54	0,939
1UVX	OXYGEN STORAGE/TRANSPORT	2,723	0,778
1UVY	OXYGEN STORAGE/TRANSPORT	2,404	0,804
2RD6	TRANSFERASE	1,98	0,744
2RJV	STRUCTURAL PROTEIN	2,06	0,74
2VAY	METAL TRANSPORT	2,544	0,791
2ZB5	VIRAL PROTEIN	2,165	**0,946**
3B75	TRANSPORT PROTEIN, OXYGEN BINDING	2,4	0,777
3C1B	STRUCTURAL PROTEIN / DNA	2,052	0,84
3DIK	VIRAL PROTEIN	1,764	0,797
3DOG	TRANSFERASE, CELL CYCLE	2,159	0,785
3DPY	TRANSFERASE	3,266	0,667
3DU7	CELL CYCLE	**3,61**	0,906
3DY6	TRANSCRIPTION	2,659	0,881
3DYN	HYDROLASE	**1,624**	0,644
Μέσος Όρος		**2,358**	**0,776**

Πίνακας 7.9: Πίνακας πειραματικών αποτελεσμάτων για τις πρωτεΐνες που ανήκουν στην κατηγορία ALPHA

Σύμφωνα με τον πίνακα των αποτελεσμάτων για τις πρωτεΐνες που ανήκουν στην κατηγορία ALPHA, έχοντας ως κριτήριο το CRMS, ο μέσος όρος είναι 2,358, η μεγαλύτερη τιμή είναι 3,61, η μικρότερη 1,624 και η διαφορά τους 1,986. Έχοντας ως κριτήριο το DRMS, ο μέσος όρος είναι 0,776, η μεγαλύτερη τιμή είναι 0,946, η μικρότερη 0,618 και η διαφορά τους 0,328.

7.3.2 ΠΡΩΤΕΪΝΕΣ ΠΟΥ ΑΝΗΚΟΥΝ ΣΤΗΝ ΚΑΤΗΓΟΡΙΑ BETA

Στον παρακάτω πίνακα (Πίνακας 7.10) φαίνονται τα πειραματικά αποτελέσματα για τις 20 πρωτεΐνες που ανήκουν στην κατηγορία BETA.

PDB ID	CLASSIFICATION	CRMS	DRMS
1AYO	MACROGLOBULIN	2,429	0,734
1BV8	PROTEIN BINDING	2,072	0,728
1BW8	PEPTIDE BINDING PROTEIN	2,685	1,161
1BXX	ENDOCYTOSIS/EXOCYTOSIS	2,664	1,179
1EDY	PROTEIN BINDING	2,281	0,642
1HES	ENDOCYTOSIS/EXOCYTOSIS	2,702	1,18
2BP5	ENDOCYTOSIS	2,52	1,226
2PR9	ENDOCYTOSIS	2,346	1,235
2VO8	HYDROLASE	**1,702**	**0,522**
2Z7F	HYDROLASE/HYDROLASE INHIBITOR	2,196	1,146
2Z7U	OXIDOREDUCTASE	1,758	**1,259**
3BGB	HYDROLASE	2,287	0,855
3BN9	HYDROLASE	2,785	1,144
3C2S	IMMUNE SYSTEM	**2,801**	0,673
3CCB	HYDROLASE	2,267	0,579
3D7P	TRANSPORT PROTEIN	2,256	1,153
3D93	LYASE	2,381	0,813
3DUH	IMMUNE SYSTEM/CYTOKINE	1,888	0,772
3DWH	LIGASE	2,705	0,793

3ELN	OXIDOREDUCTASE	2,265	1,232
Μέσος Όρος		**2,35**	**0,951**

Πίνακας 7.10: Πίνακας πειραματικών αποτελεσμάτων για τις πρωτεΐνες που ανήκουν στην κατηγορία BETA

Σύμφωνα με τον πίνακα των αποτελεσμάτων για τις πρωτεΐνες που ανήκουν στην κατηγορία BETA, έχοντας ως κριτήριο το CRMS, ο μέσος όρος είναι 2,35, η μεγαλύτερη τιμή είναι 2,801, η μικρότερη 1,702 και η διαφορά τους 1,099. Έχοντας ως κριτήριο το DRMS, ο μέσος όρος είναι 0,951, η μεγαλύτερη τιμή είναι 1,259, η μικρότερη 0,522 και η διαφορά τους 0,737.

7.3.3 ΠΡΩΤΕΪΝΕΣ ΠΟΥ ΑΝΗΚΟΥΝ ΚΑΙ ΣΤΗΝ ΚΑΤΗΓΟΡΙΑ ALPHA ΚΑΙ ΣΤΗΝ ΚΑΤΗΓΟΡΙΑ BETA

Στον παρακάτω πίνακα (Πίνακας 7.11) φαίνονται τα πειραματικά αποτελέσματα για τις 20 πρωτεΐνες που ανήκουν και στην κατηγορία ALPHA στην κατηγορία BETA.

PDB ID	CLASSIFICATION	CRMS	DRMS
1ZQ1	LYASE	2,404	0,694
2AEH	TRANSFERASE	2,679	0,769
2AL6	TRANSFERASE	2,106	0,762
2BSI	CHAPERONE	2,304	0,784
2C35	TRANSFERASE	2,716	0,736
2E2H	TRANSCRIPTION,TRANSFERASE/DNA RNA HYBRID	2,332	**1,178**
2ELB	PROTEIN BINDING	2,693	0,627
2ERJ	IMMUNE SYSTEM/CYTOKINE	1,579	**0,498**
2F5Q	HYDROLASE, DNA	2,61	0,74
2F7Q	HYDROLASE	1,95	0,598
2GZW	TRANSCRIPTION	**1,025**	0,747
2HUE	DNA BINDING PROTEIN	2,602	0,655
2NZ8	SIGNALING PROTEIN, CELL CYCLE	1,665	0,808
2OCE	STRUCTURAL GENOMICS, UNKNOWN FUNCTION	2,441	0,854
2ZKM	HYDROLASE	2,312	0,542
3BXX	OXIDOREDUCTASE	2,579	0,859
3CC4	RIBOSOME	2,165	0,603
3CZN	HYDROLASE	1,889	0,607
3E5U	TRANSCRIPTION REGULATION	2,189	0,691
3E6C	TRANSCRIPTION REGULATION/DNA	**3,137**	0,708
Μέσος Όρος		**2,269**	**0,723**

Πίνακας 7.11: Πίνακας πειραματικών αποτελεσμάτων για τις πρωτεΐνες που ανήκουν και στην κατηγορία ALPHA και στην κατηγορία BETA

Σύμφωνα με τον πίνακα των αποτελεσμάτων για τις πρωτεΐνες που ανήκουν και στην κατηγορία ALPHA και στην κατηγορία BETA, έχοντας ως κριτήριο το CRMS, ο μέσος όρος είναι 2,269, η μεγαλύτερη τιμή είναι 3,137, η μικρότερη 1,025 και η διαφορά τους 2,112. Έχοντας ως κριτήριο το DRMS, ο μέσος όρος είναι 0,723, η μεγαλύτερη τιμή είναι 1,178, η μικρότερη 0,498 και η διαφορά τους 0,68.

7.3.4 ΠΡΩΤΕΪΝΕΣ ΠΟΥ ΔΕΝ ΑΝΗΚΟΥΝ ΟΥΤΕ ΣΤΗΝ ΚΑΤΗΓΟΡΙΑ ALPHA, ΟΥΤΕ ΣΤΗΝ ΚΑΤΗΓΟΡΙΑ BETA

Στον παρακάτω πίνακα (Πίνακας 7.12) φαίνονται τα πειραματικά αποτελέσματα για τις 20 πρωτεΐνες που δεν ανήκουν ούτε στην κατηγορία ALPHA, ούτε στην κατηγορία BETA.

PDB ID	CLASSIFICATION	CRMS	DRMS
1BB1	DE NOVO PROTEIN DESIGN	2,188	0,822
1BKV	STRUCTURAL PROTEIN	**5,127**	**1,581**
1FUL	CELL ADHESION	2,358	0,711
1QP6	DE NOVO PROTEIN	2,023	0,762
1RH4	COILED COIL	2,31	0,758
1YIW	TRANSCRIPTION, TRANSFERASE/DNA RNA HYBRID	1,837	0,936
2A3D	THREE-HELIX BUNDLE	2,559	0,693
2EL5	TRANSCRIPTION	2,681	1,056
2ENT	TRANSCRIPTION	**1,717**	0,87
2FCM	STRUCTURAL PROTEIN	2,268	0,898
2JUA	DE NOVO PROTEIN	2,691	**0,557**
2JVF	DE NOVO PROTEIN	2,894	1,086
2O6N	DE NOVO PROTEIN	2,153	0,738
2OP7	LIGASE	2,742	0,768
2QYZ	STRUCTURAL GENOMICS, UNKNOWN FUNCTION	2,568	0,749
1YMZ	UNKNOWN FUNCTION	2,583	0,585
2YRH	CELL CYCLE	1,718	0,838

2ZCB	SIGNALING PROTEIN	2,022	0,945
2ZGG	DE NOVO PROTEIN	2,514	0,773
3CSR	PROTEIN BINDING	2,221	0,944
Μέσος Όρος		**2,459**	**0,854**

Πίνακας 7.12: Πίνακας πειραματικών αποτελεσμάτων για τις πρωτεΐνες που δεν ανήκουν ούτε στην κατηγορία ALPHA ούτε στην κατηγορία BETA

Σύμφωνα με τον πίνακα των αποτελεσμάτων για τις πρωτεΐνες που δεν ανήκουν ούτε στην κατηγορία ALPHA, ούτε στην κατηγορία BETA, έχοντας ως κριτήριο το CRMS, ο μέσος όρος είναι 2,459, η μεγαλύτερη τιμή είναι 5,127, η μικρότερη 1,717 και η διαφορά τους 3,41. Έχοντας ως κριτήριο το DRMS, ο μέσος όρος είναι 0,854, η μεγαλύτερη τιμή είναι 1,581, η μικρότερη 0,557 και η διαφορά τους 1,024.

7.4 ΠΕΙΡΑΜΑΤΙΚΑ ΑΠΟΤΕΛΕΣΜΑΤΑ ΠΛΕΓΜΑΤΟΣ ΜΕΓΕΘΟΥΣ (3.6 – 3.6 – 3.6)

7.4.1 ΠΡΩΤΕΪΝΕΣ ΠΟΥ ΑΝΗΚΟΥΝ ΣΤΗΝ ΚΑΤΗΓΟΡΙΑ ALPHA

Στον παρακάτω πίνακα (Πίνακας 7.13) φαίνονται τα πειραματικά αποτελέσματα για τις 20 πρωτεΐνες που ανήκουν στην κατηγορία ALPHA.

PDB ID	CLASSIFICATION	CRMS	DRMS
1B0B	OXYGEN STORAGE/TRANSPORT	2,129	0,666
3DLY	OXYGEN STORAGE/TRANSPORT	2,603	0,936
1H97	OXYGEN TRANSPORT	2,282	0,579
1LHS	OXYGEN STORAGE	2,065	0,49
1LHT	OXYGEN STORAGE	2,029	0,502
1MBW	OXYGEN TRANSPORT	2,157	0,719
1UVX	OXYGEN STORAGE/TRANSPORT	2,57	0,859
1UVY	OXYGEN STORAGE/TRANSPORT	2,232	0,953
2RD6	TRANSFERASE	1,882	0,547
2RJV	STRUCTURAL PROTEIN	2,355	0,689
2VAY	METAL TRANSPORT	2,805	0,279
2ZB5	VIRAL PROTEIN	2,263	0,88
3B75	TRANSPORT PROTEIN, OXYGEN BINDING	2,436	0,356
3C1B	STRUCTURAL PROTEIN/DNA	2,278	0,68
3DIK	VIRAL PROTEIN	1,594	0,697
3DOG	TRANSFERASE, CELL CYCLE	1,987	0,55
3DPY	TRANSFERASE	1,825	0,523
3DU7	CELL CYCLE	**3,958**	**0,277**
3DY6	TRANSCRIPTION	2,513	**0,969**
3DYN	HYDROLASE	**1,478**	0,503
Μέσος Όρος		**2,272**	**0,633**

Πίνακας 7.13: Πίνακας πειραματικών αποτελεσμάτων για τις πρωτεΐνες που ανήκουν στην κατηγορία ALPHA

Σύμφωνα με τον πίνακα των αποτελεσμάτων για τις πρωτεΐνες που ανήκουν στην κατηγορία ALPHA, έχοντας ως κριτήριο το CRMS, ο μέσος όρος είναι 2,272, η μεγαλύτερη τιμή είναι 3,958, η μικρότερη 1,478 και η διαφορά τους 2,48. Έχοντας ως κριτήριο το DRMS, ο μέσος όρος είναι 0,633, η μεγαλύτερη τιμή είναι 0,969, η μικρότερη 0,277 και η διαφορά τους 0,692.

7.4.2 ΠΡΩΤΕΪΝΕΣ ΠΟΥ ΑΝΗΚΟΥΝ ΣΤΗΝ ΚΑΤΗΓΟΡΙΑ BETA

Στον παρακάτω πίνακα (Πίνακας 7.14) φαίνονται τα πειραματικά αποτελέσματα για τις 20 πρωτεΐνες που ανήκουν στην κατηγορία BETA.

PDB ID	CLASSIFICATION	CRMS	DRMS
1AYO	MACROGLOBULIN	2,367	0,582
1BV8	PROTEIN BINDING	1,991	0,679
1BW8	PEPTIDE BINDING PROTEIN	2,527	1,235
1BXX	ENDOCYTOSIS/EXOCYTOSIS	2,55	**1,255**
1EDY	PROTEIN BINDING	2,099	0,575
1HES	ENDOCYTOSIS/EXOCYTOSIS	2,379	1,181
2BP5	ENDOCYTOSIS	2,378	1,192
2PR9	ENDOCYTOSIS	2,34	1,172
2VO8	HYDROLASE	1,94	0,695
2Z7F	HYDROLASE/HYDROLASE INHIBITOR	2,061	0,899
2Z7U	OXIDOREDUCTASE	**1,885**	1,161
3BGB	HYDROLASE	2,34	0,762
3BN9	HYDROLASE	2,721	0,918
3C2S	IMMUNE SYSTEM	**2,973**	1,001
3CCB	HYDROLASE	2,688	0,97
3D7P	TRANSPORT PROTEIN	2,059	1,083
3D93	LYASE	2,289	**0,522**
3DUH	IMMUNE SYSTEM/CYTOKINE	2,366	0,736
3DWH	LIGASE	2,379	0,919
3ELN	OXIDOREDUCTASE	2,2	1,179
Μέσος Όρος		**2,327**	**0,936**

Πίνακας 7.14: Πίνακας πειραματικών αποτελεσμάτων για τις πρωτεΐνες που ανήκουν στην κατηγορία BETA

Σύμφωνα με τον πίνακα των αποτελεσμάτων για τις πρωτεΐνες που ανήκουν στην κατηγορία BETA, έχοντας ως κριτήριο το CRMS, ο μέσος όρος είναι 2,327, η μεγαλύτερη τιμή είναι 2,973, η μικρότερη 1,885 και η διαφορά τους 1,088. Έχοντας ως κριτήριο το DRMS, ο μέσος όρος είναι 0,936, η μεγαλύτερη τιμή είναι 1,255, η μικρότερη 0,522 και η διαφορά τους 0,733.

7.4.3 ΠΡΩΤΕΪΝΕΣ ΠΟΥ ΑΝΗΚΟΥΝ ΚΑΙ ΣΤΗΝ ΚΑΤΗΓΟΡΙΑ ALPHA ΚΑΙ ΣΤΗΝ ΚΑΤΗΓΟΡΙΑ BETA

Στον παρακάτω πίνακα (Πίνακας 7.15) φαίνονται τα πειραματικά αποτελέσματα για τις 20 πρωτεΐνες που ανήκουν και στην κατηγορία ALPHA στην κατηγορία BETA.

PDB ID	CLASSIFICATION	CRMS	DRMS
1ZQ1	LYASE	2,343	0,94
2AEH	TRANSFERASE	2,932	**0,458**
2AL6	TRANSFERASE	2,527	0,889
2BSI	CHAPERONE	2,193	0,883
2C35	TRANSFERASE	2,994	0,507
2E2H	TRANSCRIPTION,TRANSFERASE/DNA RNA HYBRID	2,024	**1,273**
2ELB	PROTEIN BINDING	2,84	0,735
2ERJ	IMMUNE SYSTEM/CYTOKINE	1,684	0,683
2F5Q	HYDROLASE, DNA	2,593	0,637
2F7Q	HYDROLASE	2,065	0,623
2GZW	TRANSCRIPTION	**1,264**	0,59
2HUE	DNA BINDING PROTEIN	2,421	0,677
2NZ8	SIGNALING PROTEIN, CELL CYCLE	1,983	0,608
2OCE	STRUCTURAL GENOMICS, UNKNOWN FUNCTION	2,091	0,814
2ZKM	HYDROLASE	2,411	0,798
3BXX	OXIDOREDUCATE	2,984	0,724
3CC4	RIBOSOME	2,33	0,905
3CZN	HYDROLASE	2,002	0,605
3E5U	TRANSCRIPTION REGULATION	2,376	0,932
3E6C	TRANSCRIPTION REGULATION/DNA	**3,287**	0,724
Μέσος Όρος		**2,367**	**0,75**

Πίνακας 7.15: Πίνακας πειραματικών αποτελεσμάτων για τις πρωτεΐνες που ανήκουν και στην κατηγορία ALPHA και στην κατηγορία BETA

Σύμφωνα με τον πίνακα των αποτελεσμάτων για τις πρωτεΐνες που ανήκουν και στην κατηγορία ALPHA και στην κατηγορία BETA, έχοντας ως κριτήριο το CRMS, ο μέσος όρος είναι 2,367, η μεγαλύτερη τιμή είναι 3,287, η μικρότερη 1,264 και η

διαφορά τους 2,023. Έχοντας ως κριτήριο το DRMS, ο μέσος όρος είναι 0,75, η μεγαλύτερη τιμή είναι 1,273, η μικρότερη 0,458 και η διαφορά τους 0,815.

7.4.4 ΠΡΩΤΕΪΝΕΣ ΠΟΥ ΔΕΝ ΑΝΗΚΟΥΝ ΟΥΤΕ ΣΤΗΝ ΚΑΤΗΓΟΡΙΑ ALPHA, ΟΥΤΕ ΣΤΗΝ ΚΑΤΗΓΟΡΙΑ BETA

Στον παρακάτω πίνακα (Πίνακας 7.16) φαίνονται τα πειραματικά αποτελέσματα για τις 20 πρωτεΐνες που δεν ανήκουν ούτε στην κατηγορία ALPHA, ούτε στην κατηγορία BETA.

PDB ID	CLASSIFICATION	CRMS	DRMS
1BB1	DE NOVO PROTEIN DESIGN	2,031	0,872
1BKV	STRUCTURAL PROTEIN	**5,546**	**2,323**
1FUL	CELL ADHESION	2,409	0,592
1QP6	DE NOVO PROTEIN	1,89	0,679
1RH4	COILED COIL	2,206	0,836
1YIW	TRANSCRIPTION, TRANSFERASE/DNA RNA HYBRID	2,123	0,578
2A3D	THREE-HELIX BUNDLE	2,438	0,834
2EL5	TRANSCRIPTION	2,686	1,282
2ENT	TRANSCRIPTION	**1,442**	0,804
2FCM	STRUCTURAL PROTEIN	2,73	**0,477**
2JUA	DE NOVO PROTEIN	2,542	0,704
2JVF	DE NOVO PROTEIN	2,852	1,139
2O6N	DE NOVO PROTEIN	2,051	0,875
2OP7	LIGASE	2,513	0,761
2QYZ	STRUCTURAL GENOMICS, UNKNOWN FUNCTION	2,486	0,925
1YMZ	UNKNOWN FUNCTION	2,785	0,858
2YRH	CELL CYCLE	2,088	0,578
2ZCB	SIGNALING PROTEIN	2,424	0,508
2ZGG	DE NOVO PROTEIN	2,377	0,855
3CSR	PROTEIN BINDING	1,966	0,924
Μέσος Όρος		**2,479**	**0,87**

Πίνακας 7.16: Πίνακας πειραματικών αποτελεσμάτων για τις πρωτεΐνες που δεν ανήκουν ούτε στην κατηγορία ALPHA ούτε στην κατηγορία BETA

Σύμφωνα με τον πίνακα των αποτελεσμάτων για τις πρωτεΐνες που δεν ανήκουν ούτε στην κατηγορία ALPHA, ούτε στην κατηγορία BETA, έχοντας ως κριτήριο το CRMS, ο μέσος όρος είναι 2,479, η μεγαλύτερη τιμή είναι 5,546, η μικρότερη 1,442 και η διαφορά τους 4,104. Έχοντας ως κριτήριο το DRMS, ο μέσος όρος είναι 0,87, η μεγαλύτερη τιμή είναι 2,323, η μικρότερη 0,477 και η διαφορά τους 1,846.

7.5 ΠΕΙΡΑΜΑΤΙΚΑ ΑΠΟΤΕΛΕΣΜΑΤΑ ΠΛΕΓΜΑΤΟΣ ΜΕΓΕΘΟΥΣ (3.6 – 3.8 – 4)

7.5.1 ΠΡΩΤΕΪΝΕΣ ΠΟΥ ΑΝΗΚΟΥΝ ΣΤΗΝ ΚΑΤΗΓΟΡΙΑ ALPHA

Στον παρακάτω πίνακα (Πίνακας 7.17) φαίνονται τα πειραματικά αποτελέσματα για τις 20 πρωτεΐνες που ανήκουν στην κατηγορία ALPHA.

PDB ID	CLASSIFICATION	CRMS	DRMS
1B0B	OXYGEN STORAGE/TRANSPORT	2,334	0,501
3DLY	OXYGEN STORAGE/TRANSPORT	2,752	0,79
1H97	OXYGEN TRANSPORT	2,245	0,573
1LHS	OXYGEN STORAGE	1,991	0,476
1LHT	OXYGEN STORAGE	1,955	0,49
1MBW	OXYGEN TRANSPORT	2,423	0,739
1UVX	OXYGEN STORAGE/TRANSPORT	2,744	0,509
1UVY	OXYGEN STORAGE/TRANSPORT	2,33	0,823
2RD6	TRANSFERASE	1,898	0,474
2RJV	STRUCTURAL PROTEIN	2,326	0,559
2VAY	METAL TRANSPORT	2,663	**0,223**
2ZB5	VIRAL PROTEIN	2,196	0,849
3B75	TRANSPORT PROTEIN, OXYGEN BINDING	2,4	0,342
3C1B	STRUCTURAL PROTEIN/DNA	2,29	0,68
3DIK	VIRAL PROTEIN	1,765	0,697
3DOG	TRANSFERASE, CELL CYCLE	1,974	0,566
3DPY	TRANSFERASE	2,163	0,463
3DU7	CELL CYCLE	**3,822**	0,277
3DY6	TRANSCRIPTION	2,657	**0,903**
3DYN	HYDROLASE	**1,526**	0,581
Μέσος Όρος		**2,323**	**0,576**

Πίνακας 7.17: Πίνακας πειραματικών αποτελεσμάτων για τις πρωτεΐνες που ανήκουν στην κατηγορία ALPHA

Σύμφωνα με τον πίνακα των αποτελεσμάτων για τις πρωτεΐνες που ανήκουν στην κατηγορία ALPHA, έχοντας ως κριτήριο το CRMS, ο μέσος όρος είναι 2.323, η μεγαλύτερη τιμή είναι 3.822, η μικρότερη 1.526 και η διαφορά τους 2.296. Έχοντας ως κριτήριο το DRMS, ο μέσος όρος είναι 0.576, η μεγαλύτερη τιμή είναι 0.903, η μικρότερη 0.223 και η διαφορά τους 0.68.

7.5.2 ΠΡΩΤΕΪΝΕΣ ΠΟΥ ΑΝΗΚΟΥΝ ΣΤΗΝ ΚΑΤΗΓΟΡΙΑ BETA

Στον παρακάτω πίνακα (Πίνακας 7.18) φαίνονται τα πειραματικά αποτελέσματα για τις 20 πρωτεΐνες που ανήκουν στην κατηγορία BETA.

PDB ID	CLASSIFICATION	CRMS	DRMS
1AYO	MACROGLOBULIN	2,428	**0,484**
1BV8	PROTEIN BINDING	2,029	0,702
1BW8	PEPTIDE BINDING PROTEIN	2,649	1,206
1BXX	ENDOCYTOSIS/EXOCYTOSIS	2,678	**1,223**
1EDY	PROTEIN BINDING	2,017	0,499
1HES	ENDOCYTOSIS/EXOCYTOSIS	2,501	1,144
2BP5	ENDOCYTOSIS	2,521	1,146
2PR9	ENDOCYTOSIS	2,343	1,112
2VO8	HYDROLASE	**1,846**	0,631
2Z7F	HYDROLASE/HYDROLASE INHIBITOR	2,142	0,955
2Z7U	OXIDOREDUCTASE	1,878	1,069
3BGB	HYDROLASE	2,3	0,762
3BN9	HYDROLASE	2,766	0,987
2C25	LECTIN	**2,88**	0,855
3CCB	HYDROLASE	2,46	0,768
3D7P	TRANSPORT PROTEIN	2,285	0,954
3D93	LYASE	2,327	0,51
3DUH	IMMUNE SYSTEM/CYTOKINE	2,327	0,631
3DWH	LIGASE	2,562	0,836
3ELN	OXIDOREDUCTASE	2,15	1,086
Μέσος Όρος		**2,354**	**0,878**

Πίνακας 7.18: Πίνακας πειραματικών αποτελεσμάτων για τις πρωτεΐνες που ανήκουν στην κατηγορία BETA

Σύμφωνα με τον πίνακα των αποτελεσμάτων για τις πρωτεΐνες που ανήκουν στην κατηγορία BETA, έχοντας ως κριτήριο το CRMS, ο μέσος όρος είναι 2,354, η μεγαλύτερη τιμή είναι 2,88, η μικρότερη 1,846 και η διαφορά τους 1,034. Έχοντας ως κριτήριο το DRMS, ο μέσος όρος είναι 0,878, η μεγαλύτερη τιμή είναι 1,223, η μικρότερη 0,484 και η διαφορά τους 0,739.

7.5.3 ΠΡΩΤΕΪΝΕΣ ΠΟΥ ΑΝΗΚΟΥΝ ΚΑΙ ΣΤΗΝ ΚΑΤΗΓΟΡΙΑ ALPHA ΚΑΙ ΣΤΗΝ ΚΑΤΗΓΟΡΙΑ BETA

Στον παρακάτω πίνακα (Πίνακας 7.19) φαίνονται τα πειραματικά αποτελέσματα για τις 20 πρωτεΐνες που ανήκουν και στην κατηγορία ALPHA στην κατηγορία BETA.

PDB ID	CLASSIFICATION	CRMS	DRMS
1ZQ1	LYASE	2,485	0,723
2AEH	TRANSFERASE	2,687	**0,458**
2AL6	TRANSFERASE	2,206	0,81
2BSI	CHAPERONE	2,217	0,766
2C35	TRANSFERASE	2,88	0,432
2E2H	TRANSCRIPTION,TRANSFERASE/DNA-RNA HYBRID	1,99	**1,23**
2ELB	PROTEIN BINDING	2,751	0,64
2ERJ	IMMUNE SYSTEM/CYTOKINE	1,547	0,613
2F5Q	HYDROLASE, DNA	2,543	0,519
2F7Q	HYDROLASE	2,03	0,491
2GZW	TRANSCRIPTION	**1,026**	0,597
2HUE	DNA BINDING PROTEIN	2,479	0,527
2NZ8	SIGNALING PROTEIN, CELL CYCKE	1,744	0,608
2OCE	STRUCTURAL GENOMICS, UNKNOWN FUNCTION	2,2	0,679
2ZKM	HYDROLASE	2,368	0,651
3BXX	OXIDOREDUCTASE	2,85	0,724
3CC4	RIBOSOME	2,28	0,753
3CZN	HYDROLASE	1,97	0,479
3E5U	TRANSCRIPTION REGULATION	2,176	0,791
3E6C	TRANSCRIPTION REGULATION/DNA	**3,181**	0,724
Μέσος Όρος		**2,281**	**0,661**

Πίνακας 7.19: Πίνακας πειραματικών αποτελεσμάτων για τις πρωτεΐνες που ανήκουν και στην κατηγορία ALPHA και στην κατηγορία BETA

Σύμφωνα με τον πίνακα των αποτελεσμάτων για τις πρωτεΐνες που ανήκουν και στην κατηγορία ALPHA και στην κατηγορία BETA, έχοντας ως κριτήριο το CRMS, ο

μέσος όρος είναι 2,281, η μεγαλύτερη τιμή είναι 3,181, η μικρότερη 1,026 και η διαφορά τους 2,155. Έχοντας ως κριτήριο το DRMS, ο μέσος όρος είναι 0,661, η μεγαλύτερη τιμή είναι 1,23, η μικρότερη 0,458 και η διαφορά τους 0,772.

7.5.4 ΠΡΩΤΕΪΝΕΣ ΠΟΥ ΔΕΝ ΑΝΗΚΟΥΝ ΟΥΤΕ ΣΤΗΝ ΚΑΤΗΓΟΡΙΑ ALPHA, ΟΥΤΕ ΣΤΗΝ ΚΑΤΗΓΟΡΙΑ BETA

Στον παρακάτω πίνακα (Πίνακας 7.20) φαίνονται τα πειραματικά αποτελέσματα για τις 20 πρωτεΐνες που δεν ανήκουν ούτε στην κατηγορία ALPHA, ούτε στην κατηγορία BETA.

PDB ID	CLASSIFICATION	CRMS	DRMS
1BB1	DE NOVO PROTEIN DESIGN	2,158	0,798
1BKV	STRUCTURAL PROTEIN	**5,364**	**2,323**
1FUL	CELL ADHESION	2,334	0,509
1QP6	DE NOVO PROTEIN	1,822	0,709
1RH4	COILED COIL	2,253	0,701
1YIW	TRANSCRIPTION, TRANSFERASE/DNA RNA HYBRID	1,979	0,578
2A3D	THREE-HELIX BUNDLE	2,467	0,66
2EL5	TRANSCRIPTION	2,711	1,112
2ENT	TRANSCRIPTION	**1,761**	0,676
2FCM	STRUCTURAL PROTEIN	2,505	**0,477**
2JUA	DE NOVO PROTEIN	2,566	0,65
2JVF	DE NOVO PROTEIN	2,868	1,058
2O6N	DE NOVO PROTEIN	2,14	0,71
2OP7	LIGASE	2,574	0,749
2QYZ	STRUCTURAL GENOMICS, UNKNOWN FUNCTION	2,488	0,758
1YMZ	UNKNOWN FUNCTION	2,657	0,858
2YRH	CELL CYCLE	2,053	0,51
2ZCB	SIGNALING PROTEIN	2,078	0,508
2ZGG	DE NOVO PROTEIN	2,554	0,73

3CSR	PROTEIN BINDING	1,91	0,688
Μέσος Όρος		**2,462**	**0,788**

Πίνακας 7.20: Πίνακας πειραματικών αποτελεσμάτων για τις πρωτεΐνες που δεν ανήκουν ούτε στην κατηγορία ALPHA ούτε στην κατηγορία BETA

Σύμφωνα με τον πίνακα των αποτελεσμάτων για τις πρωτεΐνες που δεν ανήκουν ούτε στην κατηγορία ALPHA, ούτε στην κατηγορία BETA, έχοντας ως κριτήριο το CRMS, ο μέσος όρος είναι 2,462, η μεγαλύτερη τιμή είναι 5,364, η μικρότερη 1,761 και η διαφορά τους 3,603. Έχοντας ως κριτήριο το DRMS, ο μέσος όρος είναι 0,788, η μεγαλύτερη τιμή είναι 2,323, η μικρότερη 0,477 και η διαφορά τους 1,846.

7.6 ΑΝΑΛΥΣΗ ΠΕΙΡΑΜΑΤΙΚΩΝ ΑΠΟΤΕΛΕΣΜΑΤΩΝ

Στον παρακάτω πίνακα (Πίνακας 7.21) φαίνονται συνοπτικά τα πειραματικά αποτελέσματα της έρευνάς μας.

	(4-4-4)	**(2-3-1)**	**(3.8-3.8-3.8)**	**(3.6-3.6-3.6)**	**(3.6-3.8-4)**
CRMS(ALPHA)	2,358	3,578	2,271	2,272	2,323
DRMS(ALPHA)	0,776	1,819	0,695	0,633	0,576
CRMS(BETA)	2,35	3,868	2,348	2,327	2,354
DRMS(BETA)	0,951	1,753	0,955	0,936	0,878
CRMS(ALPHA-BETA)	2,269	4,112	2,302	2,367	2,281
DRMS(ALPHA-BETA)	0,723	1,934	0,938	0,75	0,661
CRMS(OTHER)	2,459	3,76	2,459	2,479	2,462
DRMS(OTHER)	0,854	1,889	0,859	0,87	0,788
M.O CRMS	**2,359**	**3,83**	**2,345**	**2,361**	**2,355**
M.O DRMS	**0,826**	**1,849**	**0,862**	**0,797**	**0,726**

Πίνακας 7.21: Πίνακας πειραματικών αποτελεσμάτων

Σύμφωνα με τα πειραματικά αποτελέσματα προκύπτουν τα εξής συμπεράσματα:

- Το καλύτερο πλέγμα που επιλύει το PCLF πρόβλημα έχοντας ως κριτήριο το CRMS, είναι το κυβικό πλέγμα με διαστάσεις (3.8 - 3.8 - 3.8).
- Το καλύτερο πλέγμα που επιλύει το PCLF πρόβλημα έχοντας ως κριτήριο το DRMS, είναι το κυβικό πλέγμα με διαστάσεις (3.6 - 3.8 - 4).
- Το χειρότερο πλέγμα που επιλύει το PCLF πρόβλημα και για τα δυο κριτήρια, είναι το (2 - 3 - 1).
- Οι πρωτεΐνες που φαίνεται να τοποθετούνται καλύτερα πάνω στα πλέγματα έχοντας ως κριτήριο το CRMS, είναι οι πρωτεΐνες που ανήκουν στην ομάδα ALPHA, καθώς έχουν καλύτερο Μ.Ο. σε τρία από τα πέντε πλέγματα, ενώ στα άλλα δύο καλύτερο Μ.Ο. έχουν οι πρωτεΐνες που ανήκουν και στην ομάδα ALPHA και στην ομάδα BETA.
- Οι πρωτεΐνες που φαίνεται να τοποθετούνται καλύτερα πάνω στα πλέγματα έχοντας ως κριτήριο το DRMS, είναι η πρωτεΐνες που ανήκουν στην ομάδα ALPHA, καθώς έχουν καλύτερο Μ.Ο σε τρία από τα πέντε πλέγματα, ενώ μόνο οι πρωτεΐνες που ανήκουν στην ομάδα BETA δεν έχουν καλύτερο Μ.Ο σε κανένα πλέγμα.
- Οι πρωτεΐνες που φαίνεται να τοποθετούνται χειρότερα πάνω στα πλέγματα έχοντας ως κριτήριο το CRMS, είναι η πρωτεΐνες που δεν ανήκουν ούτε στην κατηγορία ALPHA, ούτε στην κατηγορία BETA, καθώς έχουν χειρότερο Μ.Ο. σε τέσσερα από τα πέντε πλέγματα, ενώ στο πέμπτο πλέγμα χειρότερο Μ.Ο έχουν οι πρωτεΐνες που ανήκουν και στην ομάδα ALPHA και στην ομάδα BETA.
- Οι πρωτεΐνες που φαίνεται να τοποθετούνται χειρότερα πάνω στα πλέγματα έχοντας ως κριτήριο το DRMS,, είναι η πρωτεΐνες που ανήκουν στην κατηγορία BETA, καθώς έχουν χειρότερο Μ.Ο. σε τρία από τα πέντε

πλέγματα, ενώ στα άλλα δύο πλέγματα χειρότερο Μ.Ο έχουν οι πρωτεΐνες που ανήκουν και στην κατηγορία ALPHA και στην κατηγορία BETA.

ΚΕΦΑΛΑΙΟ 8.

ΣΥΜΠΕΡΑΣΜΑΤΑ – ΠΡΟΤΑΣΕΙΣ

Στο τελευταίο αυτό κεφάλαιο θα κάνουμε μια σύντομη ανασκόπηση της πτυχιακής αυτής εργασίας, όσον αφορά τους στόχους και τα συμπεράσματα. Επίσης θα αναφερθούμε σε προβλήματα που παραμένουν ανοιχτά και σε προτάσεις για προτεινόμενη έρευνα.

8.1 ΣΥΜΠΕΡΑΣΜΑΤΑ

Ο στόχος της πτυχιακής αυτής εργασίας ήταν η σχεδίαση και η υλοποίηση μιας ολοκληρωμένης μεθόδου, η οποία επιλύει το PCLF πρόβλημα, είτε έχοντας ως κριτήριο το CRMS, είτε έχοντας ως κριτήριο το DRMS. Η εφαρμογή αυτής της μεθόδου έγινε σε μια σειρά διάφορων πρωτεϊνών και πλεγμάτων, των οποίων η επιλογή δεν έγινε τυχαία.

Ουσιαστικά αυτό που επιχειρήσαμε ήταν να δώσουμε απάντηση στο ερώτημα: «Κατά πόσο πρωτεΐνες που κατατάσσονται στην ίδια ομάδα λόγω κοινών χαρακτηριστικών τους, δίνουν βέλτιστες αναδιπλώσεις σε όχι πολύ διαφορετικά πλέγματα».

Όσον αφορά το σύνολο των πρωτεϊνών, επιλέχθηκαν 80 πρωτεΐνες, 20 από 4 διαφορετικέ ομάδες πρωτεϊνών, των οποίων η κατηγοριοποίηση γίνεται σύμφωνα με κοινά χαρακτηριστικά, τα οποία έχουν να κάνουν με την πτύχωσή τους.

Όσον αφορά το σύνολο των πλεγμάτων, επιλέχθηκαν 5 διαφορετικά πλέγματα, τα οποία διαφέρουν στα μήκη των ακμών τους. Σύμφωνα με έρευνες που έχουν γίνει από πολλούς ερευνητές, το πλέγμα με μήκος ακμών (3,8-3,8-3,8) δίνει ‘καλές’ αναδιπλώσεις πρωτεϊνών συγκριτικά με την πραγματική τους αναδίπλωση και έτσι

επιλέξαμε αυτό το πλέγμα και άλλα τέσσερα πλέγματα με μήκη ακμών, που δεν απέχουν πολύ από αυτό.

Σύμφωνα με τα πειραματικά μας αποτελέσματα λοιπόν, το πλέγμα που δίνει καλύτερα αποτελέσματα έχοντας ως κριτήριο το CRMS, είναι το (3,8-3,8-3,8), ενώ το πλέγμα που δίνει καλύτερα αποτελέσματα έχοντας ως κριτήριο το DRMS, είναι το (3,6-3,8-4). Από την άλλη, οι πρωτεΐνες που φαίνεται να δίνουν καλύτερα αποτελέσματα έχοντας ως κριτήριο είτε το CRMS είτε το DRMS είναι οι πρωτεΐνες που ανήκουν στην ομάδα ALPHA.

8.2 ΠΡΟΤΑΣΕΙΣ - ΑΝΟΙΧΤΑ ΠΡΟΒΛΗΜΑΤΑ

Επεκτείνοντας τις έρευνες που έχουν γίνει για το PCLF πρόβλημα, ενδιαφέρον παρουσιάζουν οι παρακάτω προτάσεις, για τις οποίες θα μπορούσαν να γίνουν περεταίρω έρευνες και μελέτες:

- Η υλοποίηση ενός προγράμματος που οπτικοποιεί την αναδίπλωση μιας πρωτεΐνης σε ένα πλέγμα. Κάτι τέτοιο θα μπορούσε να βοηθήσει στο ερώτημα σχετικά με το πόσο κοντά “οπτικά” είναι η αναδίπλωση της πρωτεΐνης στο πλέγμα με την πραγματική αναδίπλωση της πρωτεΐνης στη φύση.
- Η μελέτη του PCLF προβλήματος με βάση διαφορετικά κριτήρια από τα CRMS και DRMS ή/και με διαφορετικούς περιορισμούς.
- Η μελέτη του προβλήματος σε διαφορετικούς τύπους πλεγμάτων (π.χ. μη-ορθογώνια) και η σύγκριση των αποτελεσμάτων με ήδη γνωστά αποτελέσματα σε ορθογώνια πλέγματα.

ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] B. Alberts, D. Bray, K. Hopkin, A. Johnson, J. Lewis, M. Raff, K. Roberts, P. Walters. Essential Cell Biology. (N. Ανάγνου, Π. Παπαζαφείρη, Ι. Παπαματθαιάκης, Κ. Σταματόπουλος. Βασικές Αρχές Κυτταρικής Βιολογίας Μεταφ.). Ιατρικές Εκδόσεις Ε.Π.Ε Τόμος 1. Δεύτερη Έκδοση (2006)

[2]
http://www.rcsb.org/pdb/static.do?p=general_information/about_pdb/index.html

[3] Μανωλόπουλοος Γ. (1996), ΜΑΘΗΜΑΤΑ ΘΕΩΡΙΑΣ ΓΡΑΦΩΝ Θεμελιώσεις-Αλγόριθμοι-Εφαρμογές, Αθήνα, ΕΚΔΟΣΕΙΣ ΝΕΩΝ ΤΕΧΝΟΛΟΓΙΩΝ.

[5] CPLEX Tutorial Handout Retrieved from
www.me.utexas.edu/~bard/LP/LP%20Handouts/CPLEX%20Tutorial%20Handout.pdf

[6] Β. Κώστογλου. ΓΡΑΜΜΙΚΟΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ (Linear Programming)
http://aetos.it.teithe.gr/~vkostogl/files/Epixeirisiaki/Linear%20programming_GR_teliko.pdf

[7] Ε. Ιωαννίδης. Γραμμικός Προγραμματισμός
http://myria.math.aegean.gr/epeaek/pdfs/linear-programming.pdf

[8] Thomas S. Ferguson. LINEAR PROGRAMMING. A Concise Introduction
http://www.math.ucla.edu/~tom/LP.pdf

[9] Β. Κούτρας. ΕΠΙΧΕΙΡΗΣΙΑΚΗ ΕΡΕΥΝΑ ΙΙ. Σημειώσεις Μη Γραμμικού Προγραμματισμού. ΧΙΟΣ 2010.
http://www.fme.aegean.gr/sites/default/files/cn/simeioseis_mi_grammikoy_programmatismoy.pdf

[10] Raymond Hemmecke, Matthias Köppe, Jon Lee, Robert Weismantel. Nonlinear Integer Programming. (2009).
http://arxiv.org/pdf/0906.5171.pdf

[11] College of Management, NCTU. Chap 12 Nonlinear Programming. Spring, 2009.
http://ocw.nctu.edu.tw/upload/classbfs121001561576097.pdf

[12] K.A Dill. Theory for the folding and stability of globular proteins. Biochemistry, 1985. 24(6): p. 1501-1509.

[13] A. Gupta, J. Manuch, and L. Stacho. Structure-approximating inverse protein folding problem in the 2D HP model. Journal of Computational Biology, 2005. 12(10): p. 1328–1345.

[14] D.G. Covell and R.L. Jernigan. Conformations of folded proteins in restricted spaces. Biochemistry, 1990. 29(13): p. 3287-3294.

[15] Godzik, A. Kolinski, J. Skolnick. Lattice representations of globular proteins: How good are they? Journal of Computational Chemistry, 1993. 14(10): p. 1194-1202, 1993.

[16] D.A. Hinds, M. Levitt. A lattice model for protein structure prediction at low resolution. Proceedings of the National Academy of Science of the USA, 1992. 89(7): p. 2536–2540.

[17] X. Huang. Fitting protein chains to lattice using integer programming approach. M.Sc. thesis. School of Computing Science, Simon Fraser University, 2007.

[18] B. H. Park, M. Levitt. The complexity and accuracy of discrete state models of protein structure. Journal of Molecular Biology, 1995. 249(2): p. 493-507.

[19] A. Rabow and H. A. Sheraga. Improved genetic algorithm for the protein folding problem by use of a Cartesian combination operator. Protein Science, 1996. 5(9): p. 1800-1815.

[20] B. A. Reva, D. S. Rykunov, A. J. Olson, and A. V. Finkelstein. Constructing lattice models of protein chains with side groups. Journal of Computational Biology, 1995. 2(4): p. 527-535.

[21] Rykunov DS, Reva BA, Finkelstein AV: Accurate general method for lattice approximation of three-dimensional structure of a chain molecule. Proteins, 1995. 22(2): p. 100-109.

[22] Mead, C. R., Manuch, J., Huang, X., Bhattacharyya, B., Stacho, L., and Gupta, A. Investigating Lattice Structure for Inverse Protein Folding. 30th FEBS Congrese & 9th IUBM Conference, Budapest, Hungary, (2005). Full Version in Preparation.

[23] B. Berger and T. Leighton. Protein folding in the hydrophobic-hydrophilic (HP) model is NP-complete. Journal of Computational Biology, 1998. 5(1): p. 27-40.

[24] J. Manuch, D. Gaur. Fitting protein chains to cubic lattice is NP-complete. Journal of Bioinformatics and Computational Biology, 2008. 6(1): p. 93-106.

[25] P. Crescenzi, D. Goldman, C. Papadimitriou, A. Piccolboni, and M. Yannakakis. On the complexity of protein folding. Journal of Computational Biology, 1998. 5(3): p. 423-465.

[26] I. Emiris, E. Kranakis, E. Markou, E. Tsigaridas. Finding the best on-lattice fit, manuscript.

[27] W.E. Hart, S.Istrail. Fast protein folding in the hydrophobic-hydrophilic model within three-eights of optimal. Journal of Computational Biology, 1996. 3(1): p. 53-96.

[28] A. Newman. A new algorithm for protein folding in the HP model, in Procceedings of the 13th ACM-SIAM, Symposium on Discrete Algorithms, 876-884,2002

[29] P. Koehl, M. Delrue. Building protein lattice models using self-consistent mean field theory. The Journal Of Chemical Physics, 1998. 108(22): p. 9540-9549.

[30]R. Kolodny, P.Koehl, L. Guibas and M.Levitt. Small Libraries of Protein Fragments Model Native Protein Structures Accurately. Journal of Molecular Biology, 2002. 323(2): p. 297 – 307.

[31] Y. Ponty, R. Istrate, E. Porcelli, and P. Clote. LocalMove: computing on-lattice fits for biopolymers. Nucleic Acids Research, 2008. 36(2): p. 216–222.

[32]
M. Mann, A. Dal Palu. Lattice model refinement of protein structures, 2010.

[33] M. Mann, R. Saunders, C. Smith, R. Backofen, and C. M. Deane. LatFit - producing high accuracy lattice models from protein atomic coordinates including side chains. 2010.

[34] F. Schwarzer, I. Lotan. Approximation of Protein Structure for Fast Similarity Measures. Journal of Computational Biology, 2004. 11(2-3): p. 299-317.

[35] T. Dallas. Algorithms & Experiments For The Protein Chain Lattice Fitting Problem, University of Lethbridge, 2006.

[36] V. Cutello, G. Nicosia, M. Pavone, J. Timmis, An Immune Algorithm for Protein Structure Prediction on Lattice Models. IEEE Transactions on Evolutionary Computation, 2007. 11: p. 101-117.

[37] K.A Dill. Dominant forces in protein folding. Biochemistry, 1990. 29(31): p. 7133-7155.

ΠΑΡΑΡΤΗΜΑ

ΤΟ ΠΡΟΓΡΑΜΜΑ ΣΤΗ ΓΛΩΣΣΑ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ C

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

#define Lx 3.8
#define Ly 3.8
#define Lz 3.8

#define PROTEIN_POINTS 5
#define MIDDLE PROTEIN_POINTS-1
#define LATTICE_DIMENSION 2*PROTEIN_POINTS-1

typedef struct node
{
   int x;
   int y;
   int z;
   struct node *next;
}NODE;

typedef struct array_node
{
   int info;
   NODE *Head;
}ARRAY_NODE;

typedef struct lattice_point
{
```

```
   double x;
   double y;
   double z;
}LATTICE_POINT;


void get_protein_coordinates_from_pdb_file( char *file_name, double
protein_coordinates[PROTEIN_POINTS][3] );

void create_list( ARRAY_NODE *possible_points );

void create_lattice( LATTICE_POINT
lattice_array[LATTICE_DIMENSION][LATTICE_DIMENSION][LATTICE_DIMENSION], double
protein_coordinates[PROTEIN_POINTS][3], ARRAY_NODE *possible_points );

void crms_function( LATTICE_POINT
lattice_array[LATTICE_DIMENSION][LATTICE_DIMENSION][LATTICE_DIMENSION], double
protein_coordinates[PROTEIN_POINTS][3], ARRAY_NODE possible_points[PROTEIN_POINTS] );

void drms_function( LATTICE_POINT
lattice_array[LATTICE_DIMENSION][LATTICE_DIMENSION][LATTICE_DIMENSION], double
protein_coordinates[PROTEIN_POINTS][3], ARRAY_NODE possible_points[PROTEIN_POINTS] );

void constraint1( ARRAY_NODE possible_points[PROTEIN_POINTS] );

void constraint2( ARRAY_NODE possible_points[PROTEIN_POINTS] );

void constraint3( ARRAY_NODE possible_points[PROTEIN_POINTS] );

void binaries( ARRAY_NODE possible_points[PROTEIN_POINTS] );


int main(int argc, char *argv[])
{
   int  i, j;
   char *file_name;
   double protein_coordinates[PROTEIN_POINTS][3];
```

```
LATTICE_POINT lattice_array[LATTICE_DIMENSION][LATTICE_DIMENSION][LATTICE_DIMENSION];
ARRAY_NODE possible_points[PROTEIN_POINTS];

file_name = "PR_1LHT.txt";

get_protein_coordinates_from_pdb_file( file_name, protein_coordinates );

create_list( possible_points );

create_lattice( lattice_array, protein_coordinates, possible_points );

crms_function( lattice_array, protein_coordinates, possible_points );

drms_function( lattice_array, protein_coordinates, possible_points );

constraint1( possible_points );

constraint2( possible_points );

constraint3( possible_points );

binaries( possible_points );


printf( "\n\nO Pinakas Protein_Coordinates\n\n" );
for( i=0; i<PROTEIN_POINTS; i++ )
{
    for( j=0; j<3; j++ )
    {
        printf( "%f   ", protein_coordinates[i][j] );
    }
    printf( "\n\n" );
}

printf( "O Pinakas Possible_Points.info\n" );
for( i=0; i<PROTEIN_POINTS; i++ )
{
    printf( "Possible_Points[%d].info=%d \n", i, possible_points[i].info );
```

```
   }
   printf( "\n" );


 system("PAUSE");
 return 0;
}

/* Η συνάρτηση get_protein_coordinates_from_pdb_file παίρνει σαν όρισμα ένα αρχείο από την PDB, διαβάζει τις συντεταγμένες των κεντρικών ανθράκων των αμινοξέων και τις αποθηκεύει στον πίνακα protein_coordinates. */
void get_protein_coordinates_from_pdb_file( char *file_name, double
protein_coordinates[PROTEIN_POINTS][3] )
{
   FILE *fl;
   int i;
   char coordinates[8];
   char initial_elements[100];

   fl = fopen(  file_name ,"r" );
   if( fl == NULL )
   {
      perror( "fopen" );
   }

   i = 0;

    while( !feof( fl ) && ( i < PROTEIN_POINTS )  )
    {
      fgets( initial_elements, 100, fl );
      if( ( strncmp( initial_elements, "ATOM", 4 ) == 0 ) &&
        ( strncmp( initial_elements+13, "CA", 2 )  == 0 ) )
      {
         strncpy( coordinates, initial_elements + 31, 8 );
         protein_coordinates[i][0] = atof( coordinates );
         strncpy( coordinates, initial_elements + 39, 8 );
         protein_coordinates[i][1] = atof( coordinates );
         strncpy( coordinates, initial_elements + 47, 8 );
```

```
            protein_coordinates[i][2] = atof( coordinates );
            i++;
        }
    }

    fclose( fl );
}

/*Η συνάρτηση create_list παίρνει σαν όρισμα των αρχικοποιημένο πίνακα possible_points και των επιστρέφει έχοντας δημιουργήσει τις λίστες των υποψήφιων κορυφών για κάθε αμινοξύ.*/
void create_list( ARRAY_NODE *possible_points )
{
    int i, k;
    int adj_points[6][3];
    NODE  *temp1, *temp2, *temp3, *new_node;

   for( i=0; i<PROTEIN_POINTS; i++ )
   {
       possible_points[i].info = 0;
       possible_points[i].Head = NULL;
   }

    possible_points[0].info = 1;
    new_node = malloc( sizeof( struct node ) );
    new_node->x = MIDDLE;
    new_node->y = MIDDLE;
    new_node->z = MIDDLE;
    new_node->next = NULL;
    possible_points[0].Head = new_node;

    for( i=1; i<PROTEIN_POINTS; i++ )
    {
       printf("Oi Pithanes koryfes gia to %d aminoksy einai:\n", i+1 );
       temp1 = possible_points[i-1].Head;
       temp2 = possible_points[i].Head;
       while( temp1 != NULL )
       {
           adj_points[0][0] = temp1->x-1;
```

```
adj_points[0][1] = temp1->y;
adj_points[0][2] = temp1->z;
adj_points[1][0] = temp1->x+1;
adj_points[1][1] = temp1->y;
adj_points[1][2] = temp1->z;
adj_points[2][0] = temp1->x;
adj_points[2][1] = temp1->y-1;
adj_points[2][2] = temp1->z;
adj_points[3][0] = temp1->x;
adj_points[3][1] = temp1->y+1;
adj_points[3][2] = temp1->z;
adj_points[4][0] = temp1->x;
adj_points[4][1] = temp1->y;
adj_points[4][2] = temp1->z-1;
adj_points[5][0] = temp1->x;
adj_points[5][1] = temp1->y;
adj_points[5][2] = temp1->z+1;

k = 0;
while( k <= 5 )
{
   if( temp2 == NULL )
   {
      if( !( ( adj_points[k][0] == MIDDLE ) && ( adj_points[k][1] == MIDDLE )
         && ( adj_points[k][2] == MIDDLE ) ) )
      {
         new_node = malloc( sizeof( struct node ) );
         new_node->x = adj_points[k][0];
         new_node->y = adj_points[k][1];
         new_node->z = adj_points[k][2];
         new_node->next = NULL;
         printf("%d %d %d\n", new_node->x, new_node->y, new_node->z );
         temp2 = new_node;
         possible_points[i].info = possible_points[i].info + 1;

      }
   }
   else
```

```
    {
        if( !( ( adj_points[k][0] == MIDDLE ) && ( adj_points[k][1] == MIDDLE )
            && ( adj_points[k][2] == MIDDLE ) ) )
        {
            temp3 = temp2;
            while( temp3 != NULL )
            {
                if( ( temp3->x == adj_points[k][0] ) && ( temp3->y ==
                    adj_points[k][1] ) && ( temp3->z == adj_points[k][2] ) )
                {
                    temp3 = NULL;
                }
                else
                {
                    if( temp3->next == NULL )
                    {
                        new_node = malloc( sizeof( struct node ) );
                        new_node->x = adj_points[k][0];
                        new_node->y = adj_points[k][1];
                        new_node->z = adj_points[k][2];
                        new_node->next=NULL;
                        printf( "%d %d %d\n", new_node->x, new_node->y,
                                new_node->z );
                        temp3->next = new_node;
                        temp3 = NULL;
                        possible_points[i].info = possible_points[i].info + 1;
                    }
                    else
                    {
                        temp3 = temp3->next;
                    }
                }
            }
        }
        k++;
    }
    printf("\n");
```

```
            temp1 = temp1->next;
        }
        possible_points[i].Head=temp2;
    }
}

/* Η συνάρτηση create_lattice παίρνει ως ορίσματα τους πίνακες lattice_array, protein_coordinates
και possible_points και δημιουργεί τον πίνακα lattice_array που έχει τις συντεταγμένες των κορυφών
του πλέγματος. */
void create_lattice( LATTICE_POINT
lattice_array[LATTICE_DIMENSION][LATTICE_DIMENSION][LATTICE_DIMENSION], double
protein_coordinates[PROTEIN_POINTS][3], ARRAY_NODE *possible_points )
{
    int i, j, k, foundx, foundy, foundz;
    NODE *temp;

    for( i=0; i<LATTICE_DIMENSION; i++)
    {
        for( j=0; j<LATTICE_DIMENSION; j++)
        {
            for( k=0; k<LATTICE_DIMENSION; k++ )
            {
                lattice_array[i][j][k].x = 0.0;
                lattice_array[i][j][k].y = 0.0;
                lattice_array[i][j][k].z = 0.0;
            }
        }
    }

    lattice_array[MIDDLE][MIDDLE][MIDDLE].x = protein_coordinates[0][0];
    lattice_array[MIDDLE][MIDDLE][MIDDLE].y = protein_coordinates[0][1];
    lattice_array[MIDDLE][MIDDLE][MIDDLE].z = protein_coordinates[0][2];

    for( i=1; i<PROTEIN_POINTS; i++ )
    {
        temp = possible_points[i].Head;
        while( temp != NULL )
        {
```

```
            foundx = temp->x - (MIDDLE);
            foundy = temp->y - (MIDDLE);
            foundz = temp->z - (MIDDLE);
            lattice_array[temp->x][temp->y][temp->z].x =
            lattice_array[MIDDLE][MIDDLE][MIDDLE].x + (foundx*Lx);
            lattice_array[temp->x][temp->y][temp->z].y =
            lattice_array[MIDDLE][MIDDLE][MIDDLE].y + (foundy*Ly);
            lattice_array[temp->x][temp->y][temp->z].z =
            lattice_array[MIDDLE][MIDDLE][MIDDLE].z + (foundz*Lz);

            temp = temp->next;
        }
    }

    printf( "\n" );
}

/* Η συνάρτηση crms_function παίρνει ως ορίσματα τους πίνακες lattice_array, protein_coordinates
και possible_points και δημιουργεί ένα αρχείο που περιέχει την αντικειμενική συνάρτηση με
κριτήριο το CRMS */
void crms_function( LATTICE_POINT
lattice_array[LATTICE_DIMENSION][LATTICE_DIMENSION][LATTICE_DIMENSION], double
protein_coordinates[PROTEIN_POINTS][3], ARRAY_NODE possible_points[PROTEIN_POINTS] )
{
    int i;
    double distance;
    NODE *temp;
    FILE *crms_file;

    crms_file = fopen( "input_cplex_crms.lp", "w" );

    printf( "\n\n ********CRMS********\n" );
    fprintf( crms_file, "Minimize\n" );
    fprintf( crms_file, "Obj:\n" );


    for( i=0; i<PROTEIN_POINTS; i++ )
    {
```

```
        temp = possible_points[i].Head;
        while( temp != NULL )
        {
            distance=( ( pow( fabs( protein_coordinates[i][0]- lattice_array[temp->x]
                              [temp->y][temp->z].x ), 2 ) ) +
                     ( pow( fabs( protein_coordinates[i][1]- lattice_array[temp->x]
                              [temp->y][temp->z].y ), 2 ) ) +
                     ( pow( fabs( protein_coordinates[i][2]- lattice_array[temp->x]
                              [temp>y][temp->z].z ), 2 ) ) );

            if( ( i == PROTEIN_POINTS-1 ) && ( temp->next == NULL ) )
            {
                fprintf( crms_file, "%fX(%d,(%d,%d,%d))  ",
                         distance, i+1, temp->x, temp->y, temp->z );
            }
            else
            {
                fprintf( crms_file, "%fX(%d,(%d,%d,%d)) + ",
                         distance, i+1, temp->x, temp->y, temp->z );
            }


            temp = temp->next;
        }
    }

    fprintf( crms_file, "\n\n" );
    fclose( crms_file );

}

/* Η συνάρτηση drms_function παίρνει ως ορίσματα τους πίνακες lattice_array, protein_coordinates
και possible_points και δημιουργεί ένα αρχείο που περιέχει την αντικειμενική συνάρτηση με
κριτήριο το DRMS */
void drms_function( LATTICE_POINT
lattice_array[LATTICE_DIMENSION][LATTICE_DIMENSION][LATTICE_DIMENSION], double
protein_coordinates[PROTEIN_POINTS][3], ARRAY_NODE possible_points[PROTEIN_POINTS] )
{
```

```
int  i,j;
double distance1, distance2, total_distance;
NODE *temp1, *temp2;
FILE *drms_file;



drms_file = fopen( "input_cplex_drms.lp", "w" );

printf( "\n\n ********DRMS********\n" );
fprintf( drms_file, "Minimize\n" );
fprintf( drms_file, "Obj:\n" );
fprintf( drms_file, "[ " );

i = 0;

while( i < PROTEIN_POINTS-1 )
{
    j = i+1;
    temp1 = possible_points[i].Head;
    while( temp1 != NULL )
    {
        j = i+1;
        while( j < PROTEIN_POINTS )
        {
            if( temp1 == possible_points[i].Head )
            {
                distance1 = sqrt( pow( ( protein_coordinates[i][0] -
                                         protein_coordinates[j][0]), 2 ) +
                                         pow( ( protein_coordinates[i][1] -
                                         protein_coordinates[j][1]), 2 ) +
                                         pow( ( protein_coordinates[i][2] –
                                         protein_coordinates[j][2]), 2 ) );
            }
            temp2 = possible_points[j].Head;
            while( temp2 != NULL )
            {
                distance2 = sqrt( pow( ( lattice_array[temp1->x][temp1->y][temp1>z].x)
                                 - ( lattice_array[temp2->x][temp2->y][temp2->z].x ), 2 ) +
```

```
                                pow( ( lattice_array[temp1->x][temp1->y][temp1->z].y )
                                - ( lattice_array[temp2->x][temp2->y][temp2->z].y ), 2 ) +
                               pow( ( lattice_array[temp1->x][temp1->y][temp1->z].z )
                               - ( lattice_array[temp2->x][temp2->y][temp2->z].z ), 2 ) );

            total_distance = pow( ( fabs( distance1-distance2 ) ), 2 );

            if( ( temp1->next == NULL ) && ( temp2->next == NULL )
              && ( i == PROTEIN_POINTS-2 ) && ( j = PROTEIN_POINTS-1 ) )
            {
               fprintf( drms_file, "%fX(%d,(%d,%d,%d))*X(%d,(%d,%d,%d))  ",
                           total_distance, i+1, temp1->x, temp1->y, temp1->z,
                                              j+1, temp2->x, temp2->y, temp2->z );
            }
            else
            {
               fprintf( drms_file, "%fX(%d,(%d,%d,%d))*X(%d,(%d,%d,%d)) + ",
                           total_distance, i+1, temp1->x, temp1->y, temp1->z,
                                            j+1, temp2->x, temp2->y, temp2->z );
            }

            temp2 = temp2->next;
         }
         j++;
      }
      temp1 = temp1->next;
   }
   i++;
}

fprintf( drms_file, " ]" );
fclose( drms_file );

}
```

```
/* Η συνάρτηση constraint1 παίρνει ως όρισμα τον πίνακα  possible_points και δημιουργεί ένα αρχείο που περιέχει τον πρώτο περιορισμό του προβλήματος */
void constraint1( ARRAY_NODE possible_points[PROTEIN_POINTS] )
{
   int i,j;
   NODE *temp;
   FILE *fl;

   fl = fopen( "con1.lp", "w" );

   fprintf( fl, "Subject To\n" );

   for( j=0; j<PROTEIN_POINTS; j++ )
   {
      temp = possible_points[j].Head;
      while( temp != NULL )
      {
         fprintf( fl, "X(%d,(%d,%d,%d)) ", j+1, temp->x, temp->y, temp->z );
         temp = temp->next;
         if( temp != NULL )
         {
            fprintf( fl, "+  " );
         }
         else
         {
            fprintf( fl, " = 1\n" );
         }
      }
   }
   fprintf( fl, "\n" );
   fclose( fl );

}
```

```
/* Η συνάρτηση constraint2 παίρνει ως όρισμα τον πίνακα  possible_points και δημιουργεί ένα αρχείο που περιέχει τον δεύτερο περιορισμό του προβλήματος */
void constraint2( ARRAY_NODE possible_points[PROTEIN_POINTS] )
{
   int i, j, check;
   NODE *temp1, *temp2;
   FILE *fl;

   fl = fopen( "con2.lp", "w" );

   for( i=1; i<PROTEIN_POINTS; i++ )
   {
      temp1 = possible_points[i].Head;
      while( temp1 != NULL )
      {
         if( i >= 3 )
         {
            temp2 = possible_points[i-2].Head;
            check = 0;
            while( ( check == 0 ) && ( temp2 != NULL ) )
            {
               if( ( temp1->x == temp2->x ) && ( temp1->y == temp2->y )
                           && ( temp1->z == temp2->z ) )
               {
                  check = 1;
               }
               else
               {
                  temp2 = temp2->next;
               }
            }
         }

         if( check == 1 )
         {
            temp1 = temp1->next;
         }
         else
```

```
            {
                j = i;
                while( j < PROTEIN_POINTS )
                {
                    fprintf( fl, "X(%d,(%d,%d,%d)) ", j+1, temp1->x, temp1->y, temp1->z );
                    j = j+2;
                    if( j < PROTEIN_POINTS )
                    {
                        fprintf( fl, " + " );
                    }
                }
                fprintf( fl, "<=1\n" );
                temp1 = temp1->next;
            }
        }
    }
    fprintf( fl, "\n" );
    fclose( fl );
}

/* Η συνάρτηση constraint3 παίρνει ως όρισμα τον πίνακα  possible_points και δημιουργεί ένα αρχείο που περιέχει τον τρίτο περιορισμό του προβλήματος */
void constraint3( ARRAY_NODE possible_points[PROTEIN_POINTS] )
{
    int i,j;
    NODE  *temp1, *temp2;
    FILE *fl;

    fl = fopen( "con3.lp", "w" );

    for( i=0; i<PROTEIN_POINTS-1; i++ )
    {
        temp1 = possible_points[i].Head;
        while( temp1 != NULL )
        {
            fprintf( fl, "X(%d,(%d,%d,%d)) ", i+1, temp1->x, temp1->y, temp1->z );
            j = i+1;
            temp2 = possible_points[j].Head;
```

```
            while( temp2 != NULL )
            {
                if( ( ( ( temp2->x == temp1->x-1 ) || ( temp2->x == temp1->x+1 ) )
                    && ( temp2->y == temp1->y ) && ( temp2->z == temp1->z ) ) ||
                  ( ( ( temp2->y == temp1->y-1 ) || ( temp2->y == temp1->y+1 ) )
                    && ( temp2->x == temp1->x ) && ( temp2->z == temp1->z ) ) ||
                  ( ( ( temp2->z == temp1->z-1 ) || ( temp2->z == temp1->z+1 ) )
                  && ( temp2->y == temp1->y ) && ( temp2->x == temp1->x ) ) )
                {
                    temp2 = temp2->next;
                }
                else
                {
                    fprintf( fl, "+ X(%d,(%d,%d,%d)) ", j+1, temp2->x, temp2->y, temp2->z );
                    temp2 = temp2->next;
                }

                if( temp2 == NULL )
                {
                    fprintf( fl, " <=1\n" );
                }
            }
            temp1 = temp1->next;
        }
    }

    fprintf( fl, "\n" );
    fclose( fl );
}

/* Η συνάρτηση binaries παίρνει ως όρισμα τον πίνακα  possible_points και δημιουργεί ένα αρχείο
που περιέχει τις μεταβλητές του προβλήματος με τα όριά τους. */
void binaries( ARRAY_NODE possible_points[PROTEIN_POINTS] )
{
    int i, j;
    NODE *temp, *temp1, *temp2;
    FILE *fl;
```

```
    fl = fopen( "binaries.lp", "w" );


    fprintf( fl, "Bounds\n" );

    for( i=0; i<PROTEIN_POINTS; i++ )
    {
        temp = possible_points[i].Head;
        while( temp!= NULL )
        {
            fprintf( fl, "0 <= X(%d,(%d,%d,%d)) <= 1 \n ", i+1, temp->x,
                        temp->y, temp->z );
            temp = temp->next;
        }
        fprintf( fl, "\n" );
    }

    fprintf( fl, "Binaries\n" );

    for( i=0; i<PROTEIN_POINTS; i++ )
    {
        temp = possible_points[i].Head;
        while( temp!= NULL )
        {
            fprintf( fl, "X(%d,(%d,%d,%d))\n ", i+1, temp->x, temp->y, temp->z );
            temp = temp->next;
        }
        fprintf( fl, "\n" );
    }


    fprintf( fl, "\n" );
    fprintf( fl, "End\n" );
    fclose( fl );
}
```

Με ατομική μου ευθύνη και γνωρίζοντας τις κυρώσεις [(1)], που προβλέπονται από της διατάξεις της παρ. 6 του άρθρου 22 του Ν. 1599/1986, δηλώνω ότι:

1. *Δεν παραθέτω κομμάτια βιβλίων ή άρθρων ή εργασιών άλλων αυτολεξεί* ***χωρίς να τα περικλείω σε εισαγωγικά*** *και χωρίς να αναφέρω το συγγραφέα, τη χρονολογία, τη σελίδα. Η αυτολεξεί παράθεση χωρίς εισαγωγικά χωρίς αναφορά στην πηγή, είναι λογοκλοπή. Πέραν της αυτολεξεί παράθεσης, λογοκλοπή θεωρείται και η παράφραση εδαφίων από έργα άλλων, συμπεριλαμβανομένων και έργων συμφοιτητών μου, καθώς και η παράθεση στοιχείων που άλλοι συνέλεξαν ή επεξεργάσθηκαν, χωρίς αναφορά στην πηγή. Αναφέρω πάντοτε με πληρότητα την πηγή κάτω από τον πίνακα ή σχέδιο, όπως στα παραθέματα.*

2. *Δέχομαι ότι η αυτολεξεί* ***παράθεση χωρίς εισαγωγικά****, ακόμα κι αν συνοδεύεται από αναφορά στην πηγή σε κάποιο άλλο σημείο του κειμένου ή στο τέλος του, είναι αντιγραφή. Η αναφορά στην πηγή στο τέλος π.χ. μιας παραγράφου ή μιας σελίδας, δεν δικαιολογεί συρραφή εδαφίων έργου άλλου συγγραφέα, έστω και παραφρασμένων, και παρουσίασή τους ως δική μου εργασία.*

3. *Δέχομαι ότι υπάρχει επίσης περιορισμός στο μέγεθος και στη συχνότητα των παραθεμάτων που μπορώ να εντάξω στην εργασία μου εντός εισαγωγικών. Κάθε μεγάλο παράθεμα (π.χ. σε πίνακα ή πλαίσιο, κλπ), προϋποθέτει ειδικές ρυθμίσεις, και όταν δημοσιεύεται προϋποθέτει την άδεια του συγγραφέα ή του εκδότη. Το ίδιο και οι πίνακες και τα σχέδια*

4. *Δέχομαι όλες τις συνέπειες σε περίπτωση λογοκλοπής ή αντιγραφής.*

Ημερομηνία: …19…/..5…/2015……

ΠΑΠΑΔΟΠΟΥΛΟΥ ΜΑΡΙΑ.

(Υπογραφή)

(1) «Όποιος εν γνώσει του δηλώνει ψευδή γεγονότα ή αρνείται ή αποκρύπτει τα αληθινά με έγγραφη υπεύθυνη δήλωση του άρθρου 8 παρ. 4 Ν. 1599/1986 τιμωρείται με φυλάκιση τουλάχιστον τριών μηνών. Εάν ο υπαίτιος αυτών των πράξεων σκόπευε να προσπορίσει στον εαυτόν του ή σε άλλον περιουσιακό όφελος βλάπτοντας τρίτον ή σκόπευε να βλάψει άλλον, τιμωρείται με κάθειρξη μέχρι 10 ετών.