ΠΑΝΕΠΙΣΤΗΜΙΟ
ΙΩΑΝΝΙΝΩΝ

ΣΧΟΛΗ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

ΧΡΟΝΟΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ ΑΘΛΗΤΙΚΩΝ ΔΙΟΡΓΑΝΩΣΕΩΝ

Ευάγγελος Καραμπούλης

Επιβλέπων: Χρήστος Γκόγκος

Αναπληρωτής καθηγητής Πανεπιστημίου Ιωαννίνων

Άρτα,1 Νοεμβρίου, 2021

Sports Scheduling

Εγκρίθηκε από τριμελή εξεταστική επιτροπή

Άρτα, 1 Νοεμβρίου 2021

ΕΠΙΤΡΟΠΗ ΑΞΙΟΛΟΓΗΣΗΣ

1. Επιβλέπων καθηγητής

   Χρήστος Γκόγκος,

2. Μέλος επιτροπής

   Νικόλαος Αντωνιάδης,

3. Μέλος επιτροπής

   Νικόλαος Γιαννακέας,

Δήλωση μη λογοκλοπής

Δηλώνω υπεύθυνα και γνωρίζοντας τις κυρώσεις του Ν. 2121/1993 περί Πνευματικής Ιδιοκτησίας, ότι η παρούσα μεταπτυχιακή εργασία είναι εξ ολοκλήρου αποτέλεσμα δικής μου ερευνητικής εργασίας, δεν αποτελεί προϊόν αντιγραφής ούτε προέρχεται από ανάθεση σε τρίτους. Όλες οι πηγές που χρησιμοποιήθηκαν (κάθε είδους, μορφής και προέλευσης) για τη συγγραφή της περιλαμβάνονται στη βιβλιογραφία.

Καραμπούλης Ευάγγελος

Υπογραφή

Περίληψη

Η εργασία περιγράφει το πρόβλημα χρονοπρογραμματισμού αθλητικών διοργανώσεων. Θα αναλυθούν τρόποι κατασκευής ενός χρονοδιαγράμματος και κινήσεις αλλαγών για την τροποποίηση του, οι οποίες χρησιμοποιούνται με τυχαία επιλογή από τον αλγόριθμο του Hill Climbing για την εύρεση καλύτερης λύσης. Καλύτερη λύση θεωρείται η λύση με το μικρότερο κόστος παραβίασης περιορισμών με προτεραιότητα το κόστος παραβίασης σκληρών περιορισμών. Για τον υπολογισμό κόστους για την παραβίαση περιορισμού θα χρησιμοποιηθεί μία συνάρτηση κόστους για κάθε περιορισμό του προβλήματος. Τέλος, η εργασία συγκρίνει τα αποτελέσματα καλύτερης λύσης με αρχικοποίηση του χρονοδιαγράμματος από διαφορετικές μεθόδους κατασκευής, τα αποτελέσματα παράγονται από την εφαρμογή που δημιουργήθηκε για το σκοπό της πτυχιακής.

Πίνακας περιεχομένων

1. Εισαγωγή

Οι αθλητικές διοργανώσεις για το χρονοπρογραμματισμό του χρονοδιαγράμματος χρησιμοποιούν συχνά τη μορφή Round-Robin, υπάρχουν όμως και άλλες δημοφιλείς μορφές όπως πχ η μορφή knock-out όπου ζευγάρια ομάδων παίζουν αγώνες μεταξύ τους και η ηττημένη ομάδα αποκλείεται από το πρωτάθλημα. Στην εργασία αυτή θα μας απασχολήσει η μορφή double Round-Robin, φυσικά μπορούν να υπάρξουν τριπλά και τετραπλά. Υπάρχουν δύο τύποι round-robin, τα χρονικά περιορισμένα και τα χρονικά «άνετα» χρονοδιαγράμματα, ο τύπος που μας απασχολεί είναι τα χρονικά περιορισμένα χρονοδιαγράμματα.

2. Περιγραφή του προβλήματος

Το πρόβλημα χρονοπρογραμματισμού ενός χρονικά περιορισμένου διπλού Round-Robin χρονοδιαγράμματος σημαίνει πως κάθε ομάδα πρέπει να παίξει με όλες τις υπόλοιπες ομάδες του πρωταθλήματος δύο αγώνες, έναν αγώνα εντός έδρας και έναν αγώνα εκτός έδρας. Σε κάθε αγωνιστική υπάρχουν n αγώνες όπου $n = \frac{ομάδες}{2}$, ενώ για το πλήθος αγωνιστικών στην περίπτωση του double round robin ισούνται με $slots = 2(n-1)$. Χρονικά περιορισμένο χρονοδιάγραμμα σημαίνει ότι υποχρεωτικά παίζουν όλες οι ομάδες από μία φορά σε κάθε αγωνιστική. Για κάθε περίπτωση προβλήματος υπάρχουν δύο είδη περιορισμών, οι HARD περιορισμοί όπου πρέπει να ικανοποιηθούν οπωσδήποτε και οι SOFT περιορισμοί όπου εάν δεν ικανοποιούνται πλήρως αποδίδεται μία ποινή (penalty).

3. RobinXML

Το RobinXML είναι ένα πρότυπο βασισμένο στο XML και χωρίζεται σε τρία μέρη τα οποία ονομάζονται πεδία. Σκοπό έχει τη περιγραφή πολλών διαφορετικών πρωταθλημάτων με διαφορετικά δεδομένα.

3.1.Πεδίο A

Το πεδίο A περιέχει μόνο αυστηρούς κανόνες του πρωταθλήματος όπου δε μπορούν να παραβιαστούν.

3.1.1. Πρότυπο Πρωταθλήματος

Το πρότυπο πρωταθλήματος υποδηλώνει το τρόπο που θα διεξαχθεί το πρωτάθλημα. Οι συμβολισμοί είναι οι εξής:

- **kRR:** Πρωτάθλημα που διεξάγεται με γύρους σε k φάσεις. Όλες οι ομάδες αγωνίζονται και συναντιούνται μεταξύ τους k φορές.

- **kBRR:** Πρωτάθλημα στο οποίο το σύνολο των ομάδων χωρίζεται σε δύο ασύνδετα υποσύνολα Η διαφορά των δύο υποσυνόλων πρέπει να είναι μηδενική με περιθώριο μία ομάδα.

- **NRR:** Για όλες τις υπόλοιπες περιπτώσεις πρωταθλημάτων.

3.1.2. Συμπαγές

Συμπαγές αποκαλείται ένα πρωτάθλημα που το χρονοδιάγραμμα είναι:

- **Time-constrained (C):** Ένα χρονικά περιορισμένο χρονοδιάγραμμα σημαίνει ότι έχει κατασκευαστεί με το ελάχιστο πλήθος αγωνιστικών. Σε περίπτωση περιττού πλήθους ομάδων, η κατασκευή του χρονοδιαγράμματος πραγματοποιείται με μία προστιθέμενη υποθετική ομάδα, αλλιώς

- **Time-relaxed (R):** Το χρονοδιάγραμμα του πρωταθλήματος δε κατασκευάζεται με το ελάχιστο πλήθος αγωνιστικών και σε περίπτωση περιττού πλήθους ομάδων δε χρειάζεται η χρήση της υποθετικής ομάδας.

3.1.3. Ιδιότητες Συμμετρίας

Όταν σε ένα πρωτάθλημα υπάρχει ο κανόνας της συμμετρίας, τότε απαιτείται από το χρονοδιάγραμμα σε κάθε φάση του να περιέχει μία φορά αγώνες που συμμετέχουν οι ίδιες ομάδες. Ο κανόνας αυτός ονομάζεται κανονισμός φάσης.

Slot 1	Slot 2	Slot 3	Slot 4	Slot 5	Slot 6	Slot 7	Slot 8	Slot 9	Slot 10
1-6	1-5	6-1	1-3	3-1	1-4	5-1	4-1	1-2	2-1
2-5	6-4	5-2	4-2	2-4	5-3	4-6	3-5	3-6	6-3
3-4	2-3	4-3	5-6	6-5	6-2	3-2	2-6	4-5	5-4

Πίνακας 3.1 Χρονοδιάγραμμα χωρίς τη χρήση του κανονισμού φάσης

Slot 1	Slot 2	Slot 3	Slot 4	Slot 5	Slot 6	Slot 7	Slot 8	Slot 9	Slot 10
1-6	4-1	1-3	5-1	1-2	6-1	1-5	3-1	1-4	2-1
2-5	3-5	4-2	4-6	3-6	5-2	6-4	2-4	5-3	6-3
3-4	2-6	5-6	3-2	4-5	4-3	2-3	6-5	6-2	5-4

Πίνακας 3.2 Χρονοδιάγραμμα με τη χρήση του κανονισμού φάσης

Κανόνες συμμετρίας οι οποίοι μπορούν να χρησιμοποιηθούν:

- **Καθρεπτισμός:** Οι αγώνες της επόμενης φάσης είναι οι αγώνες της προηγούμενης φάσης με αντίστροφη κατάσταση εντός-εκτός έδρας.
- **Αναστροφή:** Οι αγωνιστικές της επόμενης φάσης δημιουργούνται με αντίστροφη σειρά και με τους αγώνες με αντίστροφη κατάσταση εντός-εκτός έδρας.
- **Αγγλικό Σύστημα:** Παρόμοιο με το καθρεπτισμό με την διαφορά της επόμενης φάσης όπου μετακινεί την τελευταία της αγωνιστική στην αρχή της.
- **Γαλλικό Σύστημα:** Παρόμοιο με το καθρεπτισμό με την διαφορά της επόμενης φάσης όπου μετακινεί την πρώτη της αγωνιστική στο τέλος της.

Slot 1	Slot 2	Slot 3	Slot 4	Slot 5	Slot 6	Slot 7	Slot 8	Slot 9	Slot 10
1-6	1-5	1-4	1-3	1-2	6-1	5-1	4-1	3-1	2-1
2-5	6-4	5-3	4-2	3-6	5-2	4-6	3-5	2-4	6-3
3-4	2-3	6-2	5-6	4-5	4-3	3-2	2-6	6-5	5-4

Πίνακας 3.3 Χρονοδιάγραμμα με τη χρήση του Καθρεπτισμού

Slot 1	Slot 2	Slot 3	Slot 4	Slot 5	Slot 6	Slot 7	Slot 8	Slot 9	Slot 10
1-6	1-5	1-4	1-3	1-2	2-1	3-1	4-1	5-1	6-1
2-5	6-4	5-3	4-2	3-6	6-3	2-4	3-5	4-6	5-2
3-4	2-3	6-2	5-6	4-5	5-4	6-5	2-6	3-2	4-3

Πίνακας 3.4 Χρονοδιάγραμμα με τη χρήση της Αναστροφής

Slot 1	Slot 2	Slot 3	Slot 4	Slot 5	Slot 6	Slot 7	Slot 8	Slot 9	Slot 10
1-6	1-5	1-4	1-3	1-2	2-1	6-1	5-1	4-1	3-1
2-5	6-4	5-3	4-2	3-6	6-3	5-2	4-6	3-5	2-4
3-4	2-3	6-2	5-6	4-5	5-4	4-3	3-2	2-6	6-5

Πίνακας 3.5 Χρονοδιάγραμμα με τη χρήση του Αγγλικού Συστήματος

Slot 1	Slot 2	Slot 3	Slot 4	Slot 5	Slot 6	Slot 7	Slot 8	Slot 9	Slot 10
1-6	1-5	1-4	1-3	1-2	5-1	4-1	3-1	2-1	6-1
2-5	6-4	5-3	4-2	3-6	4-6	3-5	2-4	6-3	5-2
3-4	2-3	6-2	5-6	4-5	3-2	2-6	6-5	5-4	4-3

Πίνακας 3.6 Χρονοδιάγραμμα με τη χρήση του Γαλλικού Συστήματος

3.2.Πεδίο Β

Στο πεδίο Β περιέχονται αυστηροί και ελαφριοί περιορισμοί του πρωταθλήματος. Στη διοργάνωση ενός πρωταθλήματος μπορούν να υπάρξουν αρκετά εμπόδια, με τη χρήση των περιορισμών αυτά τα εμπόδια μπορούν να ελαχιστοποιηθούν.

Οι αυστηροί περιορισμοί συνήθως αντιπροσωπεύουν γεγονότα, νόμους κράτους, διαθέσιμες ημερομηνίες, ενώ οι ελαφριοί περιορισμοί αντιπροσωπεύουν στο μεγαλύτερο μέρος τους επιθυμίες για παράδειγμα επιχειρήσεις, τηλεοπτικές μεταδόσεις, ομάδες και κτλ. Σε κάποιες περιπτώσεις αναλόγως με τη βαρύτητα της επιθυμίας μπορεί να καταταγεί σε αυστηρούς περιορισμούς.

Στο πεδίο περιλαμβάνονται περίπου 20 περιορισμοί οι οποίοι χωρίζονται σε 5 ομάδες περιορισμών, ανάλογα το σκοπό του κάθε περιορισμού. Ο κάθε ένας περιορισμός που περιλαμβάνεται καταχωρείται ως αυστηρός ή ελαφρύς.

3.2.1. Βασικοί Περιορισμοί

Οι βασικοί περιορισμοί έμμεσα θεωρούνται πάντα αυστηροί περιορισμοί και καθορίζουν τις βασικές λειτουργίες του πρωταθλήματος:

- Όλοι οι αγώνες πρέπει να κατασκευαστούν
- Κάθε ομάδα πρέπει να συμμετέχει το πολύ σε έναν αγώνα ανά αγωνιστική

Σε περίπτωση δοκιμαστικού πεδίου οι βασικοί περιορισμοί δεν εφαρμόζονται εκτός αν ανήκουν στη κατηγορία ελαφριών περιορισμών.

3.2.2. Περιορισμοί Περιεκτικότητας

Οι περιορισμοί περιεκτικότητας προσδιορίζουν το πλήθος των αγώνων που έχουν διεξαχθεί για ένα σύνολο ομάδων σε ένα σύνολο αγωνιστικών. Επίσης, οι περιορισμοί αυτοί έχουν τη δυνατότητα απαγόρευσης σε ένα σύνολο ομάδων τη συμμετοχή τους σε αγώνα για ένα σύνολο αγωνιστικών ως εντός ή εκτός έδρας.

3.2.3. Περιορισμοί Αγώνων

Οι περιορισμοί αγώνων εξαναγκάζουν ή απαγορεύουν την ανάθεση ενός συγκεκριμένου αγώνα σε ένα σύνολο αγωνιστικών.

3.2.4. Περιορισμοί Διαλειμμάτων

Οι περιορισμοί διαλειμμάτων θέτουν όριο πλήθους συνεχόμενων αγώνων εντός ή εκτός έδρας για ένα σύνολο ομάδων. Όταν μία ομάδα εντός έδρας ξεπεράσει το όριο τότε πραγματοποιείται διάλειμμα εντός έδρας (Home Break), σε περίπτωση που η ομάδα είναι εκτός έδρας τότε πραγματοποιείται διάλειμμα εκτός έδρας (Away Break).

Δε πρέπει να υπάρχουν συνεχόμενοι αγώνες πάνω από το όριο διότι μπορεί να μην είναι δυνατό για κάποιες ομάδες να ταξιδεύουν συνεχώς εκτός έδρας και συνεπώς είναι άδικο να διεξάγονται συνεχόμενοι αγώνες εντός έδρας για τις αντίπαλες ομάδες.

3.2.5. Περιορισμοί Δικαιοσύνης

Οι περιορισμοί δικαιοσύνης στοχεύουν την μεγιστοποίηση της ελκυστικότητας του πρωταθλήματος ως προς τους θεατές και της δικαιοσύνης ως προς τις ομάδες που συμμετέχουν στο πρωτάθλημα.

3.2.6. Περιορισμοί Διαχωρισμού

Οι περιορισμοί διαχωρισμού θέτουν όριο στην απόσταση των αγωνιστικών όπου συμμετέχουν οι ίδιες ομάδες σε έναν αγώνα. Με βάση τον περιορισμό οι δύο αγώνες θα διεξάγονται σε σύντομο χρονικό διάστημα είτε σε μεγαλύτερο.

3.3.Πεδίο Γ

Κάθε χρονοδιάγραμμα περιέχει μία τιμή infeasibility και μία τιμή objective. Η συνάρτηση κόστους φροντίζει τον υπολογισμό αυτών των δύο τιμών όπου infeasibility αντιπροσωπεύει το κόστος των αυστηρών περιορισμών και objective το κόστος των ελαφριών περιορισμών.

Το πεδίο Γ περιέχει ένα τύπο συνάρτησης κόστους ο οποίος αρμόζει στους στόχους του πρωταθλήματος. Οι στόχοι ενός πρωταθλήματος πιθανόν να διαφέρουν από πρωτάθλημα σε πρωτάθλημα.

Οι τύποι συναρτήσεων κόστους είναι οι εξής:

- **BR:** Κατασκευή του χρονοδιαγράμματος με ισοζυγισμένα ελάχιστα διαλείμματα
- **TR:** Κατασκευή του χρονοδιαγράμματος με την ελάχιστη απόσταση ταξιδιών.
- **CR:** Κατασκευή του χρονοδιαγράμματος με ελάχιστο οικονομικό κόστος βάση κάποιων εκτιμήσεων του κόστους ή των εσόδων
- **CO:** Κατασκευή του χρονοδιαγράμματος με το ελάχιστο carry-over effect
- **SC:** Κατασκευή του χρονοδιαγράμματος με την ελάχιστη τιμή της απόκλισης των ελαφριών περιορισμών (objective)

4. Περιορισμοί

Οι περιορισμοί αποτελούν ένα σημαντικό μέρος των αθλητικών χρονοδιαγραμμάτων και γι' αυτό το λόγο πρέπει να ικανοποιείται το μεγαλύτερο μέρος του Συνόλου των περιορισμών. Το Σύνολο των περιορισμών χωρίζεται σε δύο μέρη τα οποία είναι οι αυστηροί περιορισμοί (τύπου HARD) και οι ελαφριοί περιορισμοί (τύπου SOFT) όπου ο τύπος του κάθε περιορισμού υποδηλώνεται από το χαρακτηριστικό type. Οι HARD περιορισμοί απαγορεύεται να παραβιαστούν από το χρονοδιάγραμμα διότι αναπαριστούν βασικές ιδιότητες του, σε αντίθεση με τους SOFT περιορισμούς όπου μπορούμε να τους αναφέρουμε ως προαιρετικούς περιορισμούς δηλαδή αντιπροσωπεύουν προτιμήσεις όπου θα πρέπει να ικανοποιηθούν στο χρονοδιάγραμμα όταν αυτό είναι δυνατόν. Κάθε περιορισμός του Συνόλου $c \in C$ έχει ως έξοδο του ένα διάνυσμα ακέραιων αριθμών μεγέθους n, το διάνυσμα αυτό ονομάζεται διάνυσμα απόκλισης $D_c = [d_1\ d_2 \dots d_{n_c}]$. Σε περίπτωση όπου ένας περιορισμός τηρείται πλήρως, όλα τα στοιχεία του διανύσματος θα ισούνται με μηδέν. Αντίθετα, εάν ένας περιορισμός παραβιαστεί το διάνυσμα θα περιέχει ένα ή περισσότερα στοιχεία διάφορα του μηδενός.

4.1.CA1

Ο περιορισμός Περιεκτικότητας CA1 καθορίζει για μία ομάδα σε ένα ή περισσότερα slot την μέγιστη τιμή αγώνων εντός ή εκτός έδρας όπου μπορεί να συμμετέχει. Εάν η ομάδα συμμετέχει σε παραπάνω αγώνες εντός ή εκτός έδρας από την μέγιστη τιμή αγώνων σε κάθε slot, τότε υπολογίζεται το deviation όπου ισούται με τη διαφορά του πλήθους των αγώνων εντός ή εκτός έδρας και της μέγιστης τιμής αγώνων.

Η μορφή του περιορισμού στο αρχείο δίνεται ως εξής:

CA1 <CA1 teams="0" max="0" mode="H" slots="0" type="HARD"/>

Η ομάδα 0 μπορεί να αγωνιστεί σε 0 αγώνες εντός έδρας τη χρονική περίοδο 0.

Ο περιορισμός CA1 χρησιμοποιείται συχνά στα αθλητικά χρονοδιαγράμματα όπου και απαγορεύουν σε μία ομάδα να παίξει εντός ή εκτός έδρας σε μία δοσμένη χρονική περίοδο. Ο περιορισμός CA1 μπορεί επίσης να συμβάλει στην εξισορρόπηση της κατάστασης των παιχνιδιών εντός ή εκτός έδρας σε σχέση με το χρόνο και τις ομάδες.

$$\forall i \in T: di = \max(kmin - \sum j \in U \sum s \in Sxi, j, s; \sum j \in U \sum s \in Sxi, j, s - kmax; 0)$$

4.2.CA2

Ο περιορισμός Περιεκτικότητας CA2 καθορίζει για μία ομάδα με έναν ή περισσότερους αντιπάλους σε ένα ή περισσότερα Slot την μέγιστη τιμή αγώνων εντός, εκτός ή εντός-εκτός έδρας όπου μπορεί να συμμετέχει. Εάν η ομάδα συμμετέχει σε παραπάνω αγώνες εντός, εκτός ή εντός-εκτός έδρας αγώνες από την μέγιστη τιμή αγώνων με κάθε αντίπαλο κατά τη διάρκεια των slot, τότε υπολογίζεται το deviation το οποίο ισούται με τη διαφορά του πλήθους των αγώνων εντός, εκτός ή εντός-εκτός έδρας και της μέγιστης τιμής αγώνων.

Η μορφή περιορισμού στο αρχείο δίνεται ως εξής:

CA2 <CA2 teams1=”0” min=”0” max=”1” mode1=”HA” mode2=”GLOBAL” teams2=”1;2” slots=”0;1;2” type=”SOFT”/>

Η ομάδα 0 αγωνίζεται το πολύ σε 1 αγώνα εντός-εκτός έδρας ενάντια στην ομάδα 1 και 2 κατά τη διάρκεια των slot 0, 1 και 2.

Ο περιορισμός CA2 γενικεύει τον περιορισμό CA1 και εφαρμόζεται ώστε να απαγορεύει στις πιο αδύναμες ομάδες να συμμετέχουν σε αρχικούς αγώνες ενάντια στις δυνατές ομάδες.

$$\forall i \in T1: di = \max(kmin - \sum j \in T2 \sum s \in Sxi, j, s; \sum j \in T2 \sum s \in Sxi, j, s - kmax; 0)$$

4.3.CA3

Ο περιορισμός Περιεκτικότητας CA3 καθορίζει για μία ή παραπάνω ομάδες την μέγιστη τιμή αγώνων εντός, εκτός ή εντός-εκτός έδρας όπου μπορούν να συμμετέχουν σε μία ακολουθία intp(αριθμός συνεχόμενων slots) slots για κάθε έναν αντίπαλο. Εάν η κάθε ομάδα συμμετέχει σε παραπάνω αγώνες από την μέγιστη τιμή αγώνων σε intp συνεχόμενα slot με έναν ή περισσότερους αντιπάλους, τότε υπολογίζεται το deviation το οποίο ισούται με τη διαφορά του πλήθους των αγώνων με τη μέγιστη τιμή αγώνων.

Η μορφή περιορισμού στο αρχείο δίνεται ως εξής:

CA3 <CA3 teams1=”0” max=”2” mode1=”HA” teams2=”1;2;3” intp=”3” mode=”SLOTS” type=”SOFT”/>

Η ομάδα 0 συμμετέχει το πολύ σε 2 αγώνες σε 3 συνεχόμενες αγωνιστικές εναντίων στις ομάδες 1, 2 και 3.

Οι σκληροί περιορισμοί μειώνουν την παραμονή εντός έδρας, τα ταξίδια εκτός έδρας ή και τα δύο απαγορεύοντας συνεχόμενους αγώνες εντός, εκτός ή εντός-εκτός έδρας. Σε κάθε περίπτωση μπορούν να υπάρξουν το πολύ 2 σκληροί περιορισμοί. Οι ελαφριοί περιορισμοί μειώνουν το συνολικό αριθμό των συνεχόμενων αγώνων εναντίων μερικών δυνατών group ομάδων.

$$\forall i \in T1: di = |P| - k + 1 \sum l = 1(\max(kmin - \sum j \in T2 l + k - 1 \sum s = lxi, j, s; \sum j \in T2 l + k - 1 \sum s = lxi, j, s - kmax; 0))$$

4.4.CA4

Ο περιορισμός Περιεκτικότητας CA4 καθορίζει για μία ή περισσότερες ομάδες την μέγιστη τιμή αγώνων εντός, εκτός ή εντός-εκτός έδρας με έναν ή περισσότερους αντιπάλους κατά τη διάρκεια ενός ή παραπάνω slot (GLOBAL) ή κατά τη διάρκεια κάθε ενός ή παραπάνω slot (EVERY). Εάν η κάθε ομάδα συμμετέχει σε παραπάνω αγώνες από την μέγιστη τιμή αγώνων, τότε υπολογίζεται το deviation το οποίο ισούται με τη διαφορά του πλήθους των αγώνων με την μέγιστη τιμή αγώνων.

Η μορφή περιορισμού στο αρχείο δίνεται ως εξής:

CA4 <CA4 teams1=”0;1” max=”3” mode1=”H” teams2=”2;3” mode2=”GLOBAL” slots=”0;1” type=”HARD”/>

Οι ομάδες 0 και 1 συμμετέχουν το πολύ σε 3 αγώνες εντός έδρας εναντίων των ομάδων 2 και 3 κατά τη διάρκεια των χρονικών περιόδων 0 και 1.

$$d = \max(kmin - \sum i \in T1 \sum j \in T2 \sum s \in Sxi, j, s; \sum i \in T1 \sum j \in T2 \sum s \in Sxi, j, s - kmax; 0)$$

4.5.GA1

Ο περιορισμός Αγώνα GA1 αναγκάζει ή απαγορεύει την τοποθέτηση για έναν ή παραπάνω αγώνες σε ένα ή παραπάνω συγκεκριμένα slot. Εάν το πλήθος των αγώνων που έχουν τοποθετηθεί σε ένα απαγορευμένο slot είναι μεγαλύτερο από τη μέγιστη τιμή αγώνων ή μικρότερη από την ελάχιστη τιμή αγώνων, τότε υπολογίζεται το deviation το οποίο ισούται με τη διαφορά του πλήθους αγώνων με τη μέγιστη τιμή αγώνων ή την ελάχιστη τιμή αγώνων.

Η μορφή περιορισμού στο αρχείο δίνεται ως εξής:

GA1 <GA1 min=”0” max=”0” meetings=”0,1;1,2;” slots=”3” type=”HARD”/>

Οι αγώνες (0,1) και (1,2) δε μπορούν να διεξαχθούν κατά τη διάρκεια του τρίτου slot.

$$d = \max(kmin - \sum(i,j) \in G \sum s \in Sxi,j,s; \sum(i,j) \in G \sum s \in Sxi,j,s - kmax; 0)$$

4.6.BR1

Ο περιορισμός Διαλείμματος BR1 καθορίζει την μέγιστη τιμή διαλειμμάτων εντός, εκτός ή εντός-εκτός έδρας για μία ή παραπάνω ομάδες. Εάν το πλήθος των διαλειμμάτων είναι μεγαλύτερο από τη μέγιστη τιμή των διαλειμμάτων, τότε υπολογίζεται το deviation όπου ισούται με τη διαφορά του πλήθους των διαλειμμάτων με τη μέγιστη τιμή των διαλειμμάτων.

Η μορφή περιορισμού στο αρχείο δίνεται ως εξής:

BR1 <BR1 teams=”0” intp=”0” mode2=”HA” slots”1” type=”HARD”/>

Η ομάδα 0 δε μπορεί να κάνει διάλειμμα στο πρώτο slot.

Ο περιορισμός BR1 μπορεί να απαγορεύσει διαλείμματα στα αρχικά ή τελευταία slot, ή μπορεί να μειώσει το συνολικό αριθμό των διαλειμμάτων ανά ομάδα.

$$\forall i \in T: di = |k - \sum s \in Sbi, shi, s|$$

4.7.BR2

Ο περιορισμός Διαλείμματος BR2 καθορίζει την μέγιστη συνολική τιμή διαλειμμάτων εντός-εκτός έδρας για όλες τις ομάδες του συνόλου teams κατά τη διάρκεια ενός ή παραπάνω slot. Εάν η συνολική τιμή διαλειμμάτων είναι μεγαλύτερη από τη μέγιστη συνολική τιμή διαλειμμάτων, τότε υπολογίζεται το deviation όπου ισούται με τη διαφορά της συνολικής τιμής διαλειμμάτων και τη μέγιστη συνολική τιμή διαλειμμάτων.

Η μορφή περιορισμού στο αρχείο δίνεται ως εξής:

BR2 <BR2 homeMode=”HA” teams=”0;1” mode2=”LEQ” intp=”2” slots”0;1;2;3” type=”HARD”/>

Η ομάδα 0 και 1 δεν έχουν συνολικά περισσότερα από δύο διαλείμματα κατά τη διάρκεια των slot 0, 1, 2 και 3.

Ο περιορισμός BR2 εφαρμόζεται για να μειώσει τη συνολική τιμή διαλειμμάτων.

$$d = |k - \sum i \in T \sum s \in Sbi, shi, s|$$

4.8.FA2

Ο περιορισμός Δικαιοσύνης FA2 καθορίζει τη μέγιστη τιμή διαφορά αγώνων εντός έδρας για τους αγώνες που έχουν πραγματοποιηθεί σε κάθε αγωνιστική του συνόλου slots μεταξύ των ομάδων του συνόλου teams. Εάν η διαφορά αγώνων είναι μεγαλύτερη από τη μέγιστη τιμή διαφοράς αγώνων, τότε υπολογίζεται το deviation όπου ισούται με τη διαφορά της μέγιστης τιμής διαφοράς αγώνων και τη διαφορά αγώνων.

Η μορφή περιορισμού στο αρχείο δίνεται ως εξής:

FA2 <FA2 teams=”0;1;2” mode=”H” intp=”1” slots=”0;1;2;3” type=”HARD”/>

Η διαφορά των αγώνων εντός έδρας που έχουν διεξαχθεί ανάμεσα στις ομάδες 0, 1 και 2 δεν είναι μεγαλύτερη του ένα κατά τη διάρκεια των slot 0, 1, 2 και 3.

Ο περιορισμός FA2 εφαρμόζεται ώστε να είναι πιο ισορροπημένοι οι αγώνες εντός έδρας σε κάθε slot για κάθε ομάδα, δηλαδή να είναι όσο το δυνατόν μικρότερη η διαφορά των αγώνων εντός έδρας που έχουν διεξαχθεί ανάμεσα στις ομάδες σε κάθε slot.

$$\forall i, j \in T, i < j: di, j = maxs \in S(|\sum t \in T \sum 1 \leq p \leq s(xi, t, p - xj, t, p)| - k; 0)$$

4.9.SE1

Ο περιορισμός Διαχωρισμού SE1 καθορίζει την ελάχιστη τιμή διαφοράς slot κοινών αγώνων των ομάδων στο σύνολο teams. Το deviation υπολογίζεται με τη διαφορά των slot στους δύο κοινούς αγώνες μεταξύ των ομάδων.

Η μορφή περιορισμού στο αρχείο δίνεται ως εξής:

SE1 <SE1 teams=”0;1” min=”5” mode1=”SLOTS” type=”HARD”/>

Υπάρχει διαφορά τουλάχιστον 5 slot στους κοινούς αγώνες των ομάδων 0 και 1 δηλαδή στον αγώνα (0-1) και (1-0).

Ο περιορισμός SE1 χρησιμοποιείται για να χωρίσει δύο αγώνες με συμμετέχοντες τις ίδιες ομάδες το ελάχιστο min αγωνιστικές απόσταση μεταξύ τους.

$$\forall i, j \in T, i < j: d_{i,j} = \sum_{s1 \in P} \sum_{s1+1 \leq s2 \leq |P|} y_{i,j,s1,s2} \max(k - (s2 - s1); 0)$$

5. Δεδομένα Προβλήματος

Τα δεδομένα που χρησιμοποιήθηκαν στη προσέγγιση του προβλήματος είναι τα προβλήματα test του διαγωνισμού ITC2021 και δίνεται η πρόσβαση ώστε να τα κατεβάσετε σε αυτό το link https://www.sportscheduling.ugent.be/ITC2021/instances.php. Είναι τα πιο απλά και μικρά προβλήματα του διαγωνισμού αλλά είναι επαρκής για τη σωστή λειτουργία του προγράμματος. Τα ονόματα των αρχείων που χρησιμοποιήθηκαν είναι το ITC2021_Test1.xml μέχρι και το ITC2021_Test8.xml.

6. Προσέγγιση Προβλήματος

6.1.IDE & Java

Κατά την ανάπτυξη του κώδικα χρησιμοποιήθηκε η έκδοση της γλώσσας προγραμματισμού Java 8 σε συνδυασμό με το περιβάλλον της JetBrains IntelliJ IDEA Community Edition για τη δημιουργία και ανάπτυξη του Maven Project. Συγκεκριμένα δημιουργήθηκε η ανάγκη για το Maven Project για την χρήση του Junit 5 σε συνδυασμό με την Java 8.

6.2.Δομές αποθήκευσης

6.2.1. Κλάση Instance

Η κλάση Instance δημιουργήθηκε για τις ανάγκες αποθήκευσης και χρήσης της περίπτωσης προβλήματος. Η λογική της κλάσης είναι η μοντελοποίηση του XML αρχείου σε αντικείμενο ώστε να χρησιμοποιηθούν ως δεδομένα του προγράμματος. Ως πεδία ορισμού της κλάσης έχουν οριστεί οι ετικέτες του XML αρχείου, όπου όπως αναφέραμε παραπάνω έχουν ήδη μοντελοποιηθεί σε κλάσεις της Java. Από τα πεδία ορισμού τα δύο πιο βασικά είναι το αντικείμενο της κλάσης Resources και της κλάσης Constraints. Η κλάση Resources έχει ως πεδία ορισμού τρεις λίστες, μία για το πρωτάθλημα (league), μία για τις ομάδες (teams) και μία για τις αγωνιστικές (slots). Η κλάση Constraints έχει ως πεδία ορισμού εννέα λίστες για κάθε έναν από τους περιορισμούς όπου αναφέρθηκαν στο κεφάλαιο 3

7. RobinXML

Το RobinXML είναι ένα πρότυπο βασισμένο στο XML και χωρίζεται σε τρία μέρη τα οποία ονομάζονται πεδία. Σκοπό έχει τη περιγραφή πολλών διαφορετικών πρωταθλημάτων με διαφορετικά δεδομένα.

7.1.Πεδίο A

Το πεδίο A περιέχει μόνο αυστηρούς κανόνες του πρωταθλήματος όπου δε μπορούν να παραβιαστούν.

7.1.1. Πρότυπο Πρωταθλήματος

Το πρότυπο πρωταθλήματος υποδηλώνει το τρόπο που θα διεξαχθεί το πρωτάθλημα. Οι συμβολισμοί είναι οι εξής:

- **kRR:** Πρωτάθλημα που διεξάγεται με γύρους σε k φάσεις. Όλες οι ομάδες αγωνίζονται και συναντιούνται μεταξύ τους k φορές.

- **kBRR:** Πρωτάθλημα στο οποίο το σύνολο των ομάδων χωρίζεται σε δύο ασύνδετα υποσύνολα Η διαφορά των δύο υποσυνόλων πρέπει να είναι μηδενική με περιθώριο μία ομάδα.

- **NRR:** Για όλες τις υπόλοιπες περιπτώσεις πρωταθλημάτων.

7.1.2. Συμπαγές

Συμπαγές αποκαλείται ένα πρωτάθλημα που το χρονοδιάγραμμα είναι:

- **Time-constrained (C):** Ένα χρονικά περιορισμένο χρονοδιάγραμμα σημαίνει ότι έχει κατασκευαστεί με το ελάχιστο πλήθος αγωνιστικών. Σε περίπτωση περιττού πλήθους ομάδων, η κατασκευή του χρονοδιαγράμματος πραγματοποιείται με μία προστιθέμενη υποθετική ομάδα, αλλιώς

- **Time-relaxed (R):** Το χρονοδιάγραμμα του πρωταθλήματος δε κατασκευάζεται με το ελάχιστο πλήθος αγωνιστικών και σε περίπτωση περιττού πλήθους ομάδων δε χρειάζεται η χρήση της υποθετικής ομάδας.

7.1.3. Ιδιότητες Συμμετρίας

Όταν σε ένα πρωτάθλημα υπάρχει ο κανόνας της συμμετρίας, τότε απαιτείται από το χρονοδιάγραμμα σε κάθε φάση του να περιέχει μία φορά αγώνες που συμμετέχουν οι ίδιες ομάδες. Ο κανόνας αυτός ονομάζεται κανονισμός φάσης.

Slot 1	Slot 2	Slot 3	Slot 4	Slot 5	Slot 6	Slot 7	Slot 8	Slot 9	Slot 10
1-6	1-5	6-1	1-3	3-1	1-4	5-1	4-1	1-2	2-1
2-5	6-4	5-2	4-2	2-4	5-3	4-6	3-5	3-6	6-3
3-4	2-3	4-3	5-6	6-5	6-2	3-2	2-6	4-5	5-4

Πίνακας 3.1 Χρονοδιάγραμμα χωρίς τη χρήση του κανονισμού φάσης

Slot 1	Slot 2	Slot 3	Slot 4	Slot 5	Slot 6	Slot 7	Slot 8	Slot 9	Slot 10
1-6	4-1	1-3	5-1	1-2	6-1	1-5	3-1	1-4	2-1
2-5	3-5	4-2	4-6	3-6	5-2	6-4	2-4	5-3	6-3
3-4	2-6	5-6	3-2	4-5	4-3	2-3	6-5	6-2	5-4

Πίνακας 3.2 Χρονοδιάγραμμα με τη χρήση του κανονισμού φάσης

Κανόνες συμμετρίας οι οποίοι μπορούν να χρησιμοποιηθούν:

- **Καθρεπτισμός:** Οι αγώνες της επόμενης φάσης είναι οι αγώνες της προηγούμενης φάσης με αντίστροφη κατάσταση εντός-εκτός έδρας.
- **Αναστροφή:** Οι αγωνιστικές της επόμενης φάσης δημιουργούνται με αντίστροφη σειρά και με τους αγώνες με αντίστροφη κατάσταση εντός-εκτός έδρας.
- **Αγγλικό Σύστημα:** Παρόμοιο με το καθρεπτισμό με την διαφορά της επόμενης φάσης όπου μετακινεί την τελευταία της αγωνιστική στην αρχή της.
- **Γαλλικό Σύστημα:** Παρόμοιο με το καθρεπτισμό με την διαφορά της επόμενης φάσης όπου μετακινεί την πρώτη της αγωνιστική στο τέλος της.

Slot 1	Slot 2	Slot 3	Slot 4	Slot 5	Slot 6	Slot 7	Slot 8	Slot 9	Slot 10
1-6	1-5	1-4	1-3	1-2	6-1	5-1	4-1	3-1	2-1
2-5	6-4	5-3	4-2	3-6	5-2	4-6	3-5	2-4	6-3
3-4	2-3	6-2	5-6	4-5	4-3	3-2	2-6	6-5	5-4

Πίνακας 3.3 Χρονοδιάγραμμα με τη χρήση του Καθρεπτισμού

Slot 1	Slot 2	Slot 3	Slot 4	Slot 5	Slot 6	Slot 7	Slot 8	Slot 9	Slot 10
1-6	1-5	1-4	1-3	1-2	2-1	3-1	4-1	5-1	6-1
2-5	6-4	5-3	4-2	3-6	6-3	2-4	3-5	4-6	5-2
3-4	2-3	6-2	5-6	4-5	5-4	6-5	2-6	3-2	4-3

Πίνακας 3.4 Χρονοδιάγραμμα με τη χρήση της Αναστροφής

Slot 1	Slot 2	Slot 3	Slot 4	Slot 5	Slot 6	Slot 7	Slot 8	Slot 9	Slot 10
1-6	1-5	1-4	1-3	1-2	2-1	6-1	5-1	4-1	3-1
2-5	6-4	5-3	4-2	3-6	6-3	5-2	4-6	3-5	2-4
3-4	2-3	6-2	5-6	4-5	5-4	4-3	3-2	2-6	6-5

Πίνακας 3.5 Χρονοδιάγραμμα με τη χρήση του Αγγλικού Συστήματος

Slot 1	Slot 2	Slot 3	Slot 4	Slot 5	Slot 6	Slot 7	Slot 8	Slot 9	Slot 10
1-6	1-5	1-4	1-3	1-2	5-1	4-1	3-1	2-1	6-1
2-5	6-4	5-3	4-2	3-6	4-6	3-5	2-4	6-3	5-2
3-4	2-3	6-2	5-6	4-5	3-2	2-6	6-5	5-4	4-3

Πίνακας 3.6 Χρονοδιάγραμμα με τη χρήση του Γαλλικού Συστήματος

7.2.Πεδίο Β

Στο πεδίο Β περιέχονται αυστηροί και ελαφριοί περιορισμοί του πρωταθλήματος. Στη διοργάνωση ενός πρωταθλήματος μπορούν να υπάρξουν αρκετά εμπόδια, με τη χρήση των περιορισμών αυτά τα εμπόδια μπορούν να ελαχιστοποιηθούν.

Οι αυστηροί περιορισμοί συνήθως αντιπροσωπεύουν γεγονότα, νόμους κράτους, διαθέσιμες ημερομηνίες, ενώ οι ελαφριοί περιορισμοί αντιπροσωπεύουν στο μεγαλύτερο μέρος τους επιθυμίες για παράδειγμα επιχειρήσεις, τηλεοπτικές μεταδόσεις, ομάδες και κτλ. Σε κάποιες περιπτώσεις αναλόγως με τη βαρύτητα της επιθυμίας μπορεί να καταταγεί σε αυστηρούς περιορισμούς.

Στο πεδίο περιλαμβάνονται περίπου 20 περιορισμοί οι οποίοι χωρίζονται σε 5 ομάδες περιορισμών, ανάλογα το σκοπό του κάθε περιορισμού. Ο κάθε ένας περιορισμός που περιλαμβάνεται καταχωρείται ως αυστηρός ή ελαφρύς.

7.2.1. Βασικοί Περιορισμοί

Οι βασικοί περιορισμοί έμμεσα θεωρούνται πάντα αυστηροί περιορισμοί και καθορίζουν τις βασικές λειτουργίες του πρωταθλήματος:

- Όλοι οι αγώνες πρέπει να κατασκευαστούν
- Κάθε ομάδα πρέπει να συμμετέχει το πολύ σε έναν αγώνα ανά αγωνιστική

Σε περίπτωση δοκιμαστικού πεδίου οι βασικοί περιορισμοί δεν εφαρμόζονται εκτός αν ανήκουν στη κατηγορία ελαφριών περιορισμών.

7.2.2. Περιορισμοί Περιεκτικότητας

Οι περιορισμοί περιεκτικότητας προσδιορίζουν το πλήθος των αγώνων που έχουν διεξαχθεί για ένα σύνολο ομάδων σε ένα σύνολο αγωνιστικών. Επίσης, οι περιορισμοί αυτοί έχουν τη δυνατότητα απαγόρευσης σε ένα σύνολο ομάδων τη συμμετοχή τους σε αγώνα για ένα σύνολο αγωνιστικών ως εντός ή εκτός έδρας.

7.2.3. Περιορισμοί Αγώνων

Οι περιορισμοί αγώνων εξαναγκάζουν ή απαγορεύουν την ανάθεση ενός συγκεκριμένου αγώνα σε ένα σύνολο αγωνιστικών.

7.2.4. Περιορισμοί Διαλειμμάτων

Οι περιορισμοί διαλειμμάτων θέτουν όριο πλήθους συνεχόμενων αγώνων εντός ή εκτός έδρας για ένα σύνολο ομάδων. Όταν μία ομάδα εντός έδρας ξεπεράσει το όριο τότε πραγματοποιείται διάλειμμα εντός έδρας (Home Break), σε περίπτωση που η ομάδα είναι εκτός έδρας τότε πραγματοποιείται διάλειμμα εκτός έδρας (Away Break).

Δε πρέπει να υπάρχουν συνεχόμενοι αγώνες πάνω από το όριο διότι μπορεί να μην είναι δυνατό για κάποιες ομάδες να ταξιδεύουν συνεχώς εκτός έδρας και συνεπώς είναι άδικο να διεξάγονται συνεχόμενοι αγώνες εντός έδρας για τις αντίπαλες ομάδες.

7.2.5. Περιορισμοί Δικαιοσύνης

Οι περιορισμοί δικαιοσύνης στοχεύουν την μεγιστοποίηση της ελκυστικότητας του πρωταθλήματος ως προς τους θεατές και της δικαιοσύνης ως προς τις ομάδες που συμμετέχουν στο πρωτάθλημα.

7.2.6. Περιορισμοί Διαχωρισμού

Οι περιορισμοί διαχωρισμού θέτουν όριο στην απόσταση των αγωνιστικών όπου συμμετέχουν οι ίδιες ομάδες σε έναν αγώνα. Με βάση τον περιορισμό οι δύο αγώνες θα διεξάγονται σε σύντομο χρονικό διάστημα είτε σε μεγαλύτερο.

7.3.Πεδίο Γ

Κάθε χρονοδιάγραμμα περιέχει μία τιμή infeasibility και μία τιμή objective. Η συνάρτηση κόστους φροντίζει τον υπολογισμό αυτών των δύο τιμών όπου infeasibility αντιπροσωπεύει το κόστος των αυστηρών περιορισμών και objective το κόστος των ελαφριών περιορισμών.

Το πεδίο Γ περιέχει ένα τύπο συνάρτησης κόστους ο οποίος αρμόζει στους στόχους του πρωταθλήματος. Οι στόχοι ενός πρωταθλήματος πιθανόν να διαφέρουν από πρωτάθλημα σε πρωτάθλημα.

Οι τύποι συναρτήσεων κόστους είναι οι εξής:

- **BR:** Κατασκευή του χρονοδιαγράμματος με ισοζυγισμένα ελάχιστα διαλείμματα
- **TR:** Κατασκευή του χρονοδιαγράμματος με την ελάχιστη απόσταση ταξιδιών.
- **CR:** Κατασκευή του χρονοδιαγράμματος με ελάχιστο οικονομικό κόστος βάση κάποιων εκτιμήσεων του κόστους ή των εσόδων
- **CO:** Κατασκευή του χρονοδιαγράμματος με το ελάχιστο carry-over effect
- **SC:** Κατασκευή του χρονοδιαγράμματος με την ελάχιστη τιμή της απόκλισης των ελαφριών περιορισμών (objective)

Περιορισμοί. Η εισαγωγή δεδομένων του XML αρχείου στην κλάση πραγματοποιείται με τη χρήση της JAXB.

7.3.1. Κλάση Timetable

Η κλάση Timetable δημιουργήθηκε με το σκοπό αποθήκευσης του αθλητικού χρονοδιαγράμματος (timetable) όπου κατασκευάζεται με τη βοήθεια των μεθόδων κατασκευής χρονοδιαγράμματος που θα αναφερθούν παρακάτω. Στην κλάση Timetable υπάρχουν συνολικά τέσσερα πεδία ορισμού.

Το πεδίο scheduledMatches, αντιπροσωπεύει μία λίστα από κατασκευασμένους αγώνες του χρονοδιαγράμματος όπου σκοπό έχει την μετάβαση της στην κλάση Solution.

Το πεδίο hashMapSchedule τύπου HashMap με key τύπου Pair και value τύπου Integer, οι τύποι του key και του value στο Pair είναι Integer. Ως key χρησιμοποιεί το ζευγάρι από δύο ομάδες δηλαδή ένας αγώνας και ως τιμή επιστρέφει το slot όπου πραγματοποιείται ο αγώνας.

Το πεδίο hashMapSlot τύπου HashMap με key τύπου Pair και value τύπου Integer, οι τύποι του key και του value στο Pair είναι Integer. Ως key χρησιμοποιεί ένα ζευγάρι που αποτελείται από μία ομάδα και ένα slot και επιστρέφει ως value τον αντίπαλο της ομάδας στο συγκεκριμένο slot.

Το πεδίο hashMapStatus τύπου HashMap με key τύπου Pair και value τύπου Boolean, οι τύποι του key και του value στο Pair είναι Integer. Ως key χρησιμοποιεί ένα ζευγάρι που αποτελείται από μία ομάδα και ένα slot και επιστρέφει ως value True εάν η ομάδα αγωνίζεται εντός έδρας στο συγκεκριμένο slot και False εάν αγωνίζεται εκτός έδρας.

Η κλάση Timetable εκτός από τα πεδία ορισμού, πραγματοποιεί λειτουργίες μέσω των μεθόδων της όπως τον υπολογισμό κόστους παραβίασης των περιορισμών, εκτύπωση του χρονοδιαγράμματος και κινήσεις αλλαγών στο χρονοδιάγραμμα. Για κάθε ένα περιορισμό έχει υλοποιηθεί μία μέθοδος υπολογισμού κόστους παραβίασης του και όλες τους χρησιμοποιούνται στη μέθοδο υπολογισμού συνολικού κόστους της λύσης. Η μέθοδος εκτύπωσης, πραγματοποιεί την έξοδο του χρονοδιαγράμματος στην οθόνη και εκχωρεί τους κατασκευασμένους αγώνες στη λίστα scheduledMatches όπου αναφέραμε παραπάνω. Τέλος για κάθε τελεστή γειτνίασης του χρονοδιαγράμματος, η κλάση Timetable υλοποιεί την αντίστοιχη μέθοδο. Κάθε μέθοδος τελεστή γειτνίασης έχει ως λειτουργία της την εφαρμογή του τελεστή γειτνίασης όταν αυτή καλεστεί από τον αλγόριθμο του Hill Climbing.

7.3.2. Κλάση Solution

Η κλάση Solution δημιουργήθηκε για τις ανάγκες αποθήκευσης, χρήσης και εξόδου της λύσης του προβλήματος σε αρχείο XML. Τα πεδία ορισμού της κλάσης είναι δύο, μία λίστα για τους κατασκευασμένους αγώνες και ένα αντικείμενο MetaData. Μέσα στο MetaData το πιο σημαντικό πεδίο ορισμού είναι το ObjectiveValue το οποίο με τη σειρά του περιέχει τις δύο τιμές κόστους παραβίασης περιορισμών για τους μαλακούς περιορισμούς (objective) και τους σκληρούς περιορισμούς (infeasibility). Τέλος με χρήση της JAXB γίνεται μοντελοποίηση της κλάσης Solution σε XML αρχείο.

7.4. JUnit Tests

Στο πρόγραμμα χρησιμοποιήθηκε το δωρεάν ανοικτού κώδικα framework JUnit 5, το οποίο μας επιτρέπει να δημιουργούμε unit tests για τον έλεγχο κάποιου μέρος του κώδικα ως προς τη λειτουργία του σε ελάχιστο χρόνο. Στο αρχείο TimetableTest.java δημιουργήθηκε η αντίστοιχη κλάση όπου περιέχει μία συνάρτηση setup για την δημιουργία του timetable. Για κάθε μέθοδο υπολογισμού κόστους παραβίασης περιορισμών της Timetable έχει υλοποιηθεί ένα αντίστοιχο test unit στο οποίο γίνεται έλεγχος εάν οι τιμές που επιστρέφει η συνάρτηση στο αντικείμενο ObjectiveValue είναι οι αναμενόμενες. Σε περίπτωση που οι τιμές δεν είναι οι αναμενόμενες, τότε το test δεν είναι επιτυχής και εμφανίζεται μήνυμα λάθους. Επίσης δημιουργήθηκε test για τον έλεγχο της μεθόδου υπολογισμού συνολικού κόστους παραβίασης περιορισμών με τον ίδιο τρόπο που χρησιμοποιήθηκε στα υπόλοιπα tests.

7.5. Μέθοδοι Κατασκευής Χρονοδιαγράμματος

7.5.1. Circle Method

Ο κυκλικός αλγόριθμος χωρίζει τις ομάδες σε δύο ίσα μέρη, όπου το πρώτο μέρος είναι η πρώτη γραμμή και το δεύτερο μέρος είναι η δεύτερη γραμμή σε ένα δισδιάστατο πίνακα με 2 γραμμές και στήλες ίσες με το πλήθος των αγώνων της συγκεκριμένης αγωνιστικής. Σε όλες τις αγωνιστικές υπάρχει μία και μόνο σταθερή ομάδα, η οποία κατά προτίμηση είναι η πρώτη ομάδα στη λίστα των ομάδων, ενώ οι υπόλοιπες ομάδες κάνουν μία κυκλική κίνηση με τη φορά του ρολογιού για $2(n-1)$ επαναλήψεις στο 2RR. Η πρώτη γραμμή απεικονίζει τις ομάδες όπου αγωνίζονται εντός έδρας και η δεύτερη γραμμή απεικονίζει τις ομάδες που αγωνίζονται εκτός έδρας για έως $n-1$ επαναλήψεις, από $n-1$ έως $2(n-1)$ ισχύει το αντίστροφο.

Στην υλοποίηση του αλγορίθμου χρησιμοποιήθηκε μία συνδεδεμένη λίστα όπου κρατούσε τις ομάδες εντός έδρας, μία συνδεδεμένη λίστα που κρατούσε τις ομάδες εκτός έδρας και ένα αντικείμενο Team όπου κρατάει την σταθερή ομάδα. Δημιουργήθηκε ένας δισδιάστατος πίνακας με αντικείμενα Match για την ανάγκη κατασκευής του χρονοδιαγράμματος και στο τέλος κάθε αγωνιστικής (επανάληψη στο πρόγραμμα) προσθέτω το τελευταίο στοιχείο της home λίστας στο τέλος της away λίστας και το πρώτο στοιχείο της away λίστας στην αρχή της home λίστας, αντίστοιχα αφαιρούνται τα στοιχεία από κάθε λίστα μετά από τη μεταφορά τους. Στο δεύτερο μισό των αγωνιστικών απλά ορίζουμε αντίστροφα τις home και away λίστες στα αντικείμενα Match ώστε να καλυφτεί το Double Round-Robin.

Phase 1					**Phase 2**				
Slot 1	**Slot 2**	**Slot 3**	**Slot 4**	**Slot 5**	**Slot 6**	**Slot 7**	**Slot 8**	**Slot 9**	**Slot 10**
1-6	1-5	1-4	1-3	1-2	6-1	5-1	4-1	3-1	2-1
2-5	6-4	5-3	4-2	3-6	5-2	4-6	3-5	2-4	6-3
3-4	2-3	6-2	5-6	4-5	4-3	3-2	2-6	6-5	5-4

Πίνακας 7.1 Double Round Robin με τη χρήση της Circle Method για 6 ομάδες

Phase 1							**Phase 2**						
Slot 1	**Slot 2**	**Slot 3**	**Slot 4**	**Slot 5**	**Slot 6**	**Slot 7**	**Slot 8**	**Slot 9**	**Slot 10**	**Slot 11**	**Slot 12**	**Slot 13**	**Slot 14**
1-8	1-7	1-6	1-5	1-4	1-3	1-2	8-1	7-1	6-1	5-1	4-1	3-1	2-1
2-7	8-6	7-5	6-4	5-3	4-2	3-8	7-2	6-8	5-7	4-6	3-5	2-4	8-3
3-6	2-5	8-4	7-3	6-2	5-8	4-7	6-3	5-2	4-8	3-7	2-6	8-5	7-4
4-5	3-4	2-3	8-2	7-8	6-7	5-6	5-4	4-3	3-2	2-8	8-7	7-6	6-5

Πίνακας 7.2 Double Round Robin με τη χρήση της Circle Method για 8 ομάδες

7.5.2. Berger Method

Ο αλγόριθμος του Berger είναι παρόμοιος με τον κυκλικό αλγόριθμο, δηλαδή η δημιουργία των αγώνων στο χρονοδιάγραμμα για κάθε slot πραγματοποιείται χρησιμοποιώντας σα σταθερή ομάδα την τελευταία ομάδα (ομάδα n) και με τη κυκλική κίνηση των υπόλοιπων ομάδων. Η κατασκευή αγώνων χωρίζεται σε δύο μέρη όπου είναι τα άρτια και τα περιττά slot. Στα άρτια slot η σταθερή ομάδα χρησιμοποιείται ως ομάδα εκτός έδρας στην κατασκευή του αγώνα της, στο πρώτο άρτιο slot αρχικοποιούνται η ομάδα 1 έως την ομάδα n/2 ως εντός έδρας ομάδες και η ομάδα n-1 έως την ομάδα (n/2)+1 ως εκτός έδρας ομάδες, έτσι σε κάθε επόμενο άρτιο slot πραγματοποιείται μία δεξιά κυκλική κίνηση μεταξύ των ομάδων που χρησιμοποιήθηκαν ως ομάδες εντός και εκτός έδρας στην κατασκευή αγώνων. Στα περιττά slot η σταθερή ομάδα χρησιμοποιείται ως ομάδα εντός έδρας στην κατασκευή του αγώνα της, στο πρώτο περιττό slot αρχικοποιούνται η ομάδα 1 και (ομάδα (n/2)+1 έως την ομάδα n-1) ως εντός έδρας ομάδες και η ομάδα 2 έως την ομάδα n/2 ως εκτός έδρας ομάδες, έτσι σε κάθε επόμενο περιττό slot πραγματοποιείται μία δεξιά κυκλική κίνηση μεταξύ των ομάδων που χρησιμοποιήθηκαν ως ομάδες εντός και εκτός έδρας στην κατασκευή αγώνων.

Στην υλοποίηση του αλγορίθμου χρησιμοποιήθηκαν τέσσερις συνδεδεμένες λίστες, δύο από τις οποίες αντιπροσωπεύουν τις ομάδες εντός (evenHomeTeams) και εκτός έδρας (evenAwayTeams) για τα άρτια slots και οι υπόλοιπες δύο (oddHomeTeams και oddAwayTeams) για τα περιττά slots. Μία μεταβλητή (constantTeam) κρατάει τη σταθερή ομάδα, δηλαδή την ομάδα n. Στη συνέχεια, για κάθε slot έγινε έλεγχος εάν είναι άρτιο ή περιττό. Κάθε slot έχει teams/2 periods, στη πρώτη περίοδο χρησιμοποιεί τη σταθερή ομάδα, αλλιώς χρησιμοποιεί τις δύο συνδεδεμένες λίστες αντίστοιχα για το άρτιο ή το περιττό slot.

Phase 1					**Phase 2**				
Slot 1	**Slot 2**	**Slot 3**	**Slot 4**	**Slot 5**	**Slot 6**	**Slot 7**	**Slot 8**	**Slot 9**	**Slot 10**
1-6	6-4	2-6	6-5	3-6	6-1	4-6	6-2	5-6	6-3
2-5	5-3	3-1	1-4	4-2	5-2	3-5	1-3	4-1	2-4
3-4	1-2	4-5	2-3	5-1	4-3	2-1	5-4	3-2	1-5

Πίνακας 7.3 Double Round Robin με τη χρήση της Berger Method για 6 ομάδες

Phase 1							Phase 2						
Slot 1	**Slot 2**	**Slot 3**	**Slot 4**	**Slot 5**	**Slot 6**	**Slot 7**	**Slot 8**	**Slot 9**	**Slot 10**	**Slot 11**	**Slot 12**	**Slot 13**	**Slot 14**
1-8	8-5	2-8	8-6	3-8	8-7	4-8	8-1	5-8	8-2	6-8	8-3	7-8	8-4
2-7	6-4	3-1	7-5	4-2	1-6	5-3	7-2	4-6	1-3	5-7	2-4	6-1	3-5
3-6	7-3	4-7	1-4	5-1	2-5	6-2	6-3	3-7	7-4	4-1	1-5	5-2	2-6
4-5	1-2	5-6	2-3	6-7	3-4	7-1	5-4	2-1	6-5	3-2	7-6	4-3	1-7

Πίνακας 7.4 Double Round Robin με τη χρήση της Berger Method για 8 ομάδες

7.6. Τελεστές Γειτνίασης

Στην λύση χρησιμοποιούνται τέσσερις τελεστές γειτνίασης οι οποίοι κατά την εφαρμογή τους επηρεάζουν την παραβίαση των περιορισμών θετικά ή αρνητικά. Οι τελεστές αυτοί χρησιμοποιούνται από τον αλγόριθμο του Hill Climbing κατά την προσπάθεια του εύρεσης καλύτερης λύσης.

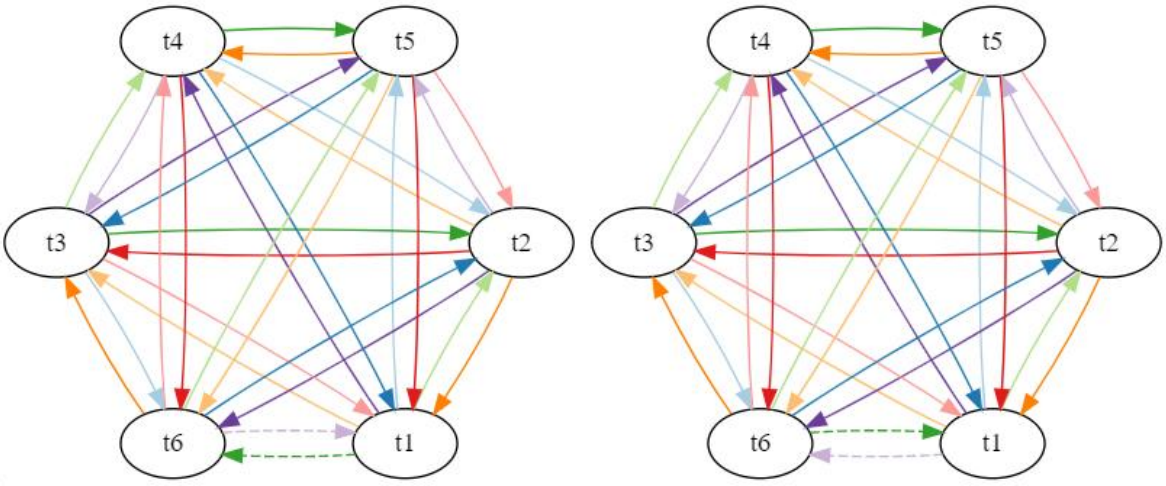
```
public void patch(int home, int away, int slot) {
    this.hashMapSchedule.replace(new Pair<>(home, away), slot);

    this.hashMapSlot.replace(new Pair<>(home, slot), away);
    this.hashMapSlot.replace(new Pair<>(away, slot), home);

    this.hashMapStatus.replace(new Pair<>(home, slot), true);
    this.hashMapStatus.replace(new Pair<>(away, slot), false);
}
```

Εικόνα 7.1 Μέθοδος για την ενημέρωση των αλλαγών του χρονοδιαγράμματος

Η μέθοδος patch στην **Εικόνα 7.1** εφαρμόζει τη μέθοδος replace του HashMap για να πραγματοποιηθούν όλες οι αντικαταστάσεις στο χρονοδιάγραμμα και να γίνει επιτυχώς η ενημέρωση του. Όλες οι μέθοδοι των τελεστών γειτνίασης χρησιμοποιούν την μέθοδο patch για την ενημέρωση του χρονοδιαγράμματος.

7.6.1. Ανταλλαγή Ρεβάνς

Πρώτος τελεστής γειτνίασης είναι η ανταλλαγή ρεβάνς. Κάθε ζευγάρι ομάδων πραγματοποιεί δύο αγώνες όπου στον ένα αγώνα η πρώτη ομάδα αγωνίζεται εντός έδρας και η δεύτερη ομάδα εκτός έδρας και στον δεύτερο αγώνα η πρώτη ομάδα αγωνίζεται εκτός έδρας και η δεύτερη ομάδα εντός έδρας. Ο τελεστής αυτός εντοπίζει τους δύο αγώνες μεταξύ του ζεύγους ομάδων που επιλέγονται και ανταλλάζει το slot διεξαγωγής τους.



Εικόνα 7.2 Ανταλλαγή ρεβάνς για την ομάδα t1 και t6

Phase 1					Phase 2				
Slot 1	**Slot 2**	**Slot 3**	**Slot 4**	**Slot 5**	**Slot 6**	**Slot 7**	**Slot 8**	**Slot 9**	**Slot 10**
1-6	1-5	1-4	1-3	1-2	6-1	5-1	4-1	3-1	2-1
2-5	6-4	5-3	4-2	3-6	5-2	4-6	3-5	2-4	6-3
3-4	2-3	6-2	5-6	4-5	4-3	3-2	2-6	6-5	5-4

Πίνακας 7.5 Χρονοδιάγραμμα πριν την εφαρμογή της ανταλλαγής ρεβάνς για την ομάδα 1 και 6

Phase 1					Phase 2				
Slot 1	**Slot 2**	**Slot 3**	**Slot 4**	**Slot 5**	**Slot 6**	**Slot 7**	**Slot 8**	**Slot 9**	**Slot 10**
6-1	1-5	1-4	1-3	1-2	1-6	5-1	4-1	3-1	2-1
2-5	6-4	5-3	4-2	3-6	5-2	4-6	3-5	2-4	6-3
3-4	2-3	6-2	5-6	4-5	4-3	3-2	2-6	6-5	5-4

Πίνακας 7.6 Χρονοδιάγραμμα μετά την εφαρμογή της ανταλλαγής ρεβάνς για την ομάδα 1 και 6

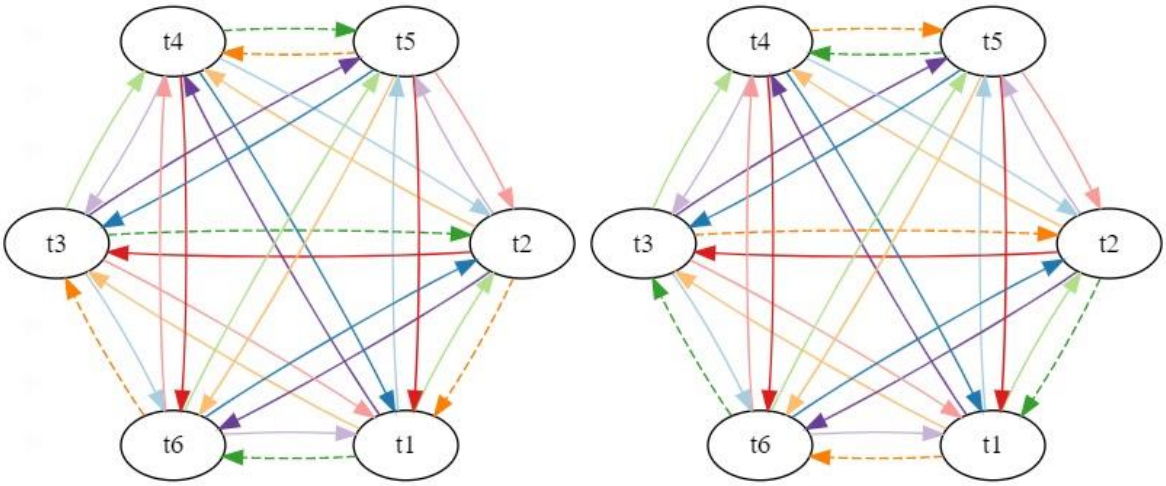
```
public void swapRematch(int team1, int team2) {
    int slot1 = hashMapSchedule.get(new Pair<>(team1, team2));
    int slot2 = hashMapSchedule.get(new Pair<>(team2, team1));

    patch(team1, team2, slot2);
    patch(team2, team1, slot1);
}
```

Εικόνα 7.3 Μέθοδος υλοποίησης ανταλλαγής ρεβάνς

Η μέθοδος swapRematch στην **Εικόνα 7.3** δέχεται ως παραμέτρους δύο ακέραιες τιμές που αναπαριστούν τις δύο ομάδες. Δημιουργεί δύο ακέραιες μεταβλητές στις οποίες αποθηκεύει αντίστοιχα το slot του αγώνα και της ρεβάνς του. Στη μέθοδο patch ως παράμετροι χρησιμοποιείται ο αγώνας με το slot της ρεβάνς και αντιστρόφως για τη ρεβάνς με το slot του αγώνα, έτσι πραγματοποιείται ο τελεστής γειτνίασης για την ανταλλαγή ρεβάνς.

7.6.2. Ανταλλαγή Αγωνιστικών

Δεύτερος τελεστής γειτνίασης είναι η ανταλλαγή αγωνιστικών (slots). Κατά την εφαρμογή του τελεστή αυτού πραγματοποιείται ανταλλαγή όλων των αγώνων μεταξύ δύο επιλεγμένων slot.



Εικόνα 7.4 Ανταλλαγή δύο αγωνιστικών που απεικονίζονται με τα χρώματα πράσινο και πορτοκαλί

Phase 1					Phase 2				
Slot 1	**Slot 2**	**Slot 3**	**Slot 4**	**Slot 5**	**Slot 6**	**Slot 7**	**Slot 8**	**Slot 9**	**Slot 10**
1-6	1-5	1-4	1-3	1-2	6-1	5-1	4-1	3-1	2-1
2-5	6-4	5-3	4-2	3-6	5-2	4-6	3-5	2-4	6-3
3-4	2-3	6-2	5-6	4-5	4-3	3-2	2-6	6-5	5-4

Πίνακας 7.7 Χρονοδιάγραμμα πριν την εφαρμογή της ανταλλαγής αγωνιστικών Slot 2 και 7

Phase 1					Phase 2				
Slot 1	**Slot 2**	**Slot 3**	**Slot 4**	**Slot 5**	**Slot 6**	**Slot 7**	**Slot 8**	**Slot 9**	**Slot 10**
1-6	5-1	1-4	1-3	1-2	6-1	1-5	4-1	3-1	2-1
2-5	4-6	5-3	4-2	3-6	5-2	6-4	3-5	2-4	6-3
3-4	3-2	6-2	5-6	4-5	4-3	2-3	2-6	6-5	5-4

Πίνακας 7.8 Χρονοδιάγραμμα μετά την εφαρμογή της ανταλλαγής αγωνιστικών Slot 2 και 7

```java
public void swapSlots(int slot1, int slot2) {

    if (isPhasedViolatedCheckBySlots(slot1, slot2))
        return;

    HashMap<Pair<Integer, Integer>, Integer> hashMapSlot = getHashMapSlot();
    HashMap<Pair<Integer, Integer>, Boolean> hashMapStatus = getHashMapStatus();

    Map<Pair<Integer, Integer>, Boolean> swappedMatches = new HashMap<>();
    for (int team1 = 0; team1 < instance.getResources().getTeams().size(); team1++) {
        int team2Slot1 = hashMapSlot.get(new Pair<>(team1, slot1));
        boolean slot1Status = hashMapStatus.get(new Pair<>(team1, slot1));
        int team2Slot2 = hashMapSlot.get(new Pair<>(team1, slot2));
        boolean slot2Status = hashMapStatus.get(new Pair<>(team1, slot2));
        if (slot1Status) {
            if (swappedMatches.get(new Pair<>(team1,team2Slot1)) == null) {
                patch(team1, team2Slot1, slot2);
                swappedMatches.put(new Pair<>(team1, team2Slot1), true);
            }
        } else {
            if (swappedMatches.get(new Pair<>(team2Slot1,team1)) == null) {
                patch(team2Slot1, team1, slot2);
                swappedMatches.put(new Pair<>(team2Slot1, team1), true);
            }
        }
        if (slot2Status) {
            if (swappedMatches.get(new Pair<>(team1,team2Slot2)) == null) {
                patch(team1, team2Slot2, slot1);
                swappedMatches.put(new Pair<>(team1, team2Slot2), true);
            }
        } else {
            if (swappedMatches.get(new Pair<>(team2Slot2, team1)) == null) {
                patch(team2Slot2, team1, slot1);
                swappedMatches.put(new Pair<>(team2Slot2, team1), true);
            }
        }
    }
}
```

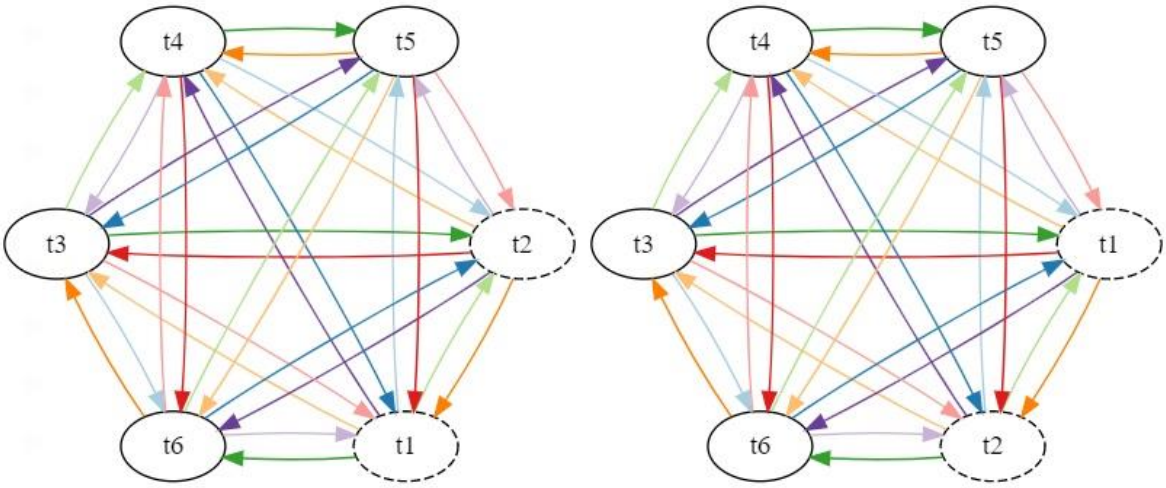
Εικόνα 7.5 Μέθοδος υλοποίησης ανταλλαγής αγωνιστικών

Η μέθοδος swapSlots για την υλοποίηση του τελεστή της ανταλλαγής αγωνιστικών στην **Εικόνα 7.5** χρησιμοποιεί ως παραμέτρους δύο ακέραιες μεταβλητές οι οποίες αναπαριστούν τα δύο slot (αγωνιστικές) τα οποία ανταλλάσσουν όλους τους αγώνες μεταξύ τους. Αρχικά πραγματοποιείται έλεγχος για τη παραβίαση του Phase περιορισμού εάν εφαρμοστεί ο τελεστής γειτνίασης ανταλλαγή αγωνιστικών για τα δύο συγκεκριμένα slot. Εφόσον δεν έχει

παραβιαστεί το Phase, η μέθοδος συνεχίζει τη λειτουργία της και δημιουργούνται δύο τοπικά αντίγραφα HashMap και ένα κενό HashMap swappedMatches για την προσθήκη των αγώνων όπου έχει πραγματοποιηθεί η ανταλλαγή. Για κάθε ομάδα αποθηκεύουμε την αντίπαλη της και τη κατάσταση εντός-εκτός έδρας ξεχωριστά και για τα δύο slot. Εάν ο αγώνας δεν εμπεριέχεται στο swappedMatches τότε προστίθεται ο αγώνας σε αυτό και γίνεται ενημέρωση με τη μέθοδο patch.

7.6.3. Ανταλλαγή Ομάδων

Τρίτος τελεστής γειτνίασης είναι η ανταλλαγή ομάδων. Κατά την εφαρμογή του συγκεκριμένου τελεστή η κατάσταση εντός-εκτός έδρας των αντιπάλων παραμένει σταθερή και γίνεται αλλαγή μεταξύ μόνο των δύο ομάδων. Στην περίπτωση όπου ο αγώνας διεξάγεται μεταξύ των δύο επιλεγμένων ομάδων τότε πραγματοποιείται ανταλλαγή θέσης και κατάστασης εντός-εκτός έδρας.



Εικόνα 7.6 Αναπαράσταση ανταλλαγής ομάδων μεταξύ των ομάδων t1 και t2 σε γράφημα

Phase 1					Phase 2				
Slot 1	**Slot 2**	**Slot 3**	**Slot 4**	**Slot 5**	**Slot 6**	**Slot 7**	**Slot 8**	**Slot 9**	**Slot 10**
1-6	1-5	1-4	1-3	1-2	6-1	5-1	4-1	3-1	2-1
2-5	6-4	5-3	4-2	3-6	5-2	4-6	3-5	2-4	6-3
3-4	2-3	6-2	5-6	4-5	4-3	3-2	2-6	6-5	5-4

Πίνακας 7.9 Χρονοδιάγραμμα πριν την εφαρμογή της ανταλλαγής ομάδων για την ομάδα 1 και 2

Phase 1					Phase 2				
Slot 1	**Slot 2**	**Slot 3**	**Slot 4**	**Slot 5**	**Slot 6**	**Slot 7**	**Slot 8**	**Slot 9**	**Slot 10**
2-6	2-5	2-4	2-3	2-1	6-2	5-2	4-2	3-2	1-2
1-5	6-4	5-3	4-1	3-6	5-1	4-6	3-5	1-4	6-3
3-4	1-3	6-1	5-6	4-5	4-3	3-1	1-6	6-5	5-4

Πίνακας 7.10 Χρονοδιάγραμμα μετά την εφαρμογή της ανταλλαγής ομάδων για την ομάδα 1 και 2

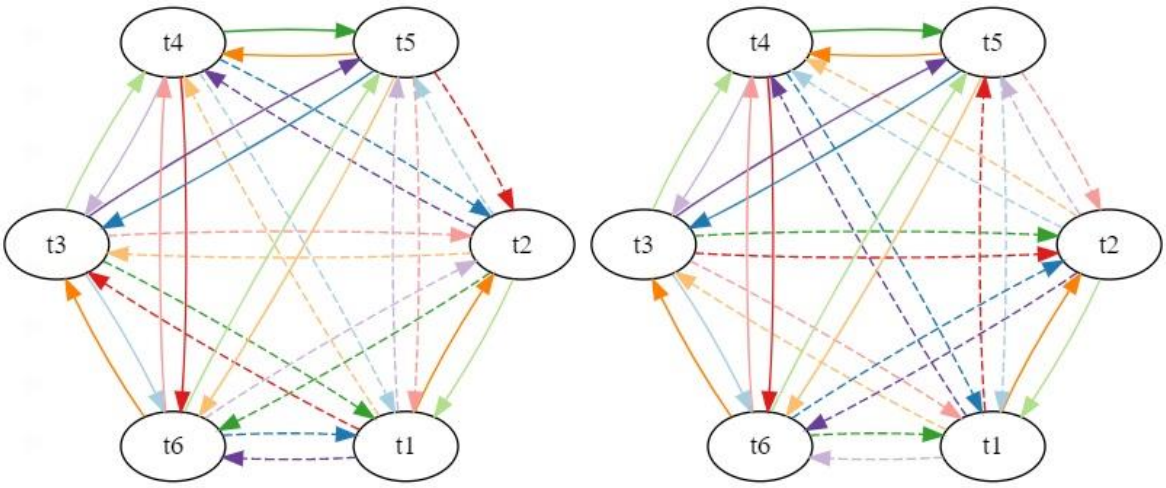
```java
public void swapTeams(int team1, int team2) {
    HashMap<Pair<Integer, Integer>, Integer> hashMapSlot = getHashMapSlot();
    HashMap<Pair<Integer, Integer>, Boolean> hashMapStatus = getHashMapStatus();
    for (int slot = 0; slot < instance.getResources().getSlots().size(); slot++) {
        int team1Opponent = hashMapSlot.get(new Pair<>(team1, slot));
        boolean team1Status = hashMapStatus.get(new Pair<>(team1, slot));
        int team2Opponent = hashMapSlot.get(new Pair<>(team2, slot));
        boolean team2Status = hashMapStatus.get(new Pair<>(team2, slot));
        if (team1Opponent == team2) {
            if (team1Status) {
                patch(team2, team1, slot);
            } else {
                patch(team1, team2, slot);
            }
        } else {
            if (team1Status) {
                patch(team2, team1Opponent, slot);
            } else {
                patch(team1Opponent, team2, slot);
            }
            if (team2Status) {
                patch(team1, team2Opponent, slot);
            } else {
                patch(team2Opponent, team1, slot);
            }
        }
    }
}
```

Εικόνα 7.7 Μέθοδος υλοποίησης ανταλλαγής ομάδων

Για την υλοποίηση του τελεστή γειτνίασης ανταλλαγής ομάδων έχει δημιουργηθεί η μέθοδος swapTeams που βλέπουμε στην **Εικόνα 7.7** και ως παραμέτρους χρησιμοποιεί δύο ακέραιες μεταβλητές για την αναπαράσταση των δύο ομάδων όπου θα αλλάξουν θέσεις μεταξύ τους σε κάθε slot. Για κάθε slot αποθηκεύεται η ομάδα-αντίπαλος και η κατάσταση εντός-εκτός έδρας για τις δύο ομάδες. Εάν οι δύο ομάδες αγωνίζονται μεταξύ τους θα ανταλλάξουν κατάσταση εντός-εκτός έδρας, αλλιώς ανταλλάζουν μεταξύ τους θέση στους δύο αγώνες.

7.6.4. Ανταλλαγή Αντιπάλων

Τέταρτος τελεστής γειτνίασης είναι η ανταλλαγή αντιπάλων. Κατά την εφαρμογή αυτού του τελεστή η κατάσταση εντός-εκτός έδρας των δύο ομάδων που δίνονται παραμένει ίδια και ανταλλάσσουν τους αντιπάλους τους. Στην περίπτωση όπου οι δύο ομάδες είναι αντίπαλες πραγματοποιείται ανταλλαγή θέσης και κατάστασης εντός-εκτός έδρας.



Εικόνα 7.8 Αναπαράσταση ανταλλαγής αντιπάλων μεταξύ των ομάδων t1 και t2 σε γράφημα

Phase 1					Phase 2				
Slot 1	**Slot 2**	**Slot 3**	**Slot 4**	**Slot 5**	**Slot 6**	**Slot 7**	**Slot 8**	**Slot 9**	**Slot 10**
1-6	1-5	1-4	1-3	1-2	6-1	5-1	4-1	3-1	2-1
2-5	6-4	5-3	4-2	3-6	5-2	4-6	3-5	2-4	6-3
3-4	2-3	6-2	5-6	4-5	4-3	3-2	2-6	6-5	5-4

Πίνακας 7.11 Χρονοδιάγραμμα πριν την εφαρμογή της ανταλλαγής αντιπάλων για την ομάδα 1 και 2

Phase 1					Phase 2				
Slot 1	**Slot 2**	**Slot 3**	**Slot 4**	**Slot 5**	**Slot 6**	**Slot 7**	**Slot 8**	**Slot 9**	**Slot 10**
1-5	1-3	1-6	1-4	2-1	5-1	3-1	6-1	4-1	1-2
2-6	6-4	5-3	3-2	3-6	6-2	4-6	3-5	2-3	6-3
3-4	2-5	4-2	5-6	4-5	4-3	5-2	2-4	6-5	5-4

Πίνακας 7.12 Χρονοδιάγραμμα μετά την εφαρμογή της ανταλλαγής αντιπάλων για την ομάδα 1 και 2

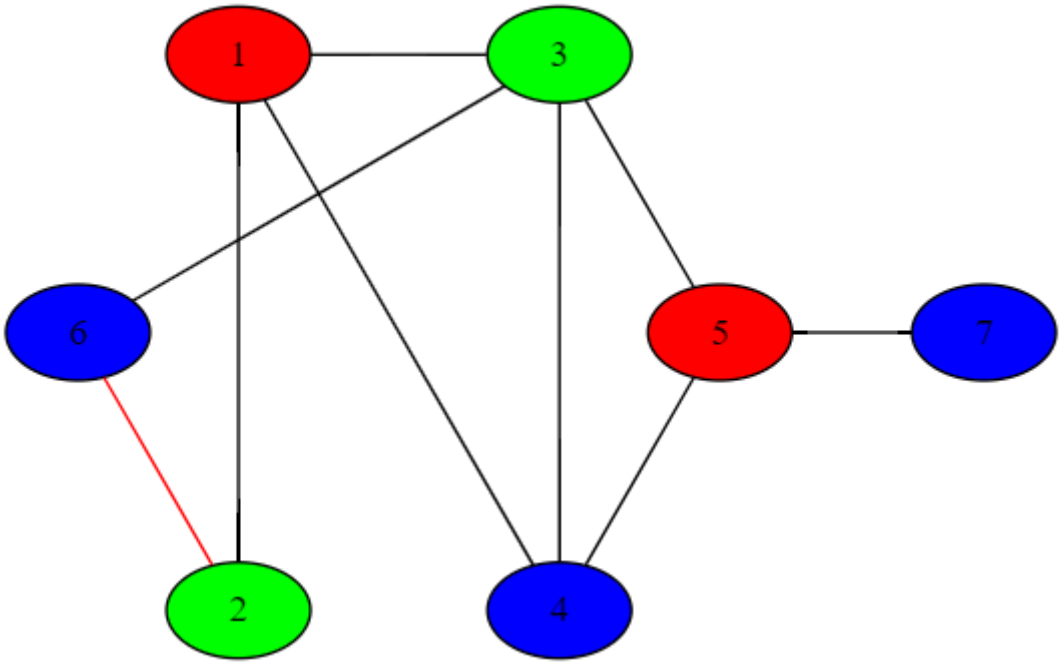
```
public void swapOpponents(int team1, int team2) {
    HashMap<Pair<Integer, Integer>, Integer> hashMapSlot = getHashMapSlot();
    HashMap<Pair<Integer, Integer>, Boolean> hashMapStatus = getHashMapStatus();
    for (int slot = 0; slot < instance.getResources().getSlots().size(); slot++) {
        int team1Opponent = hashMapSlot.get(new Pair<>(team1, slot));
        boolean team1Status = hashMapStatus.get(new Pair<>(team1, slot));
        int team2Opponent = hashMapSlot.get(new Pair<>(team2, slot));
        boolean team2Status = hashMapStatus.get(new Pair<>(team2, slot));
        if (team1Opponent == team2) {
            if (team1Status) {
                patch(team2, team1, slot);
            } else {
                patch(team1, team2, slot);
            }
        } else {
            if (team1Status) {
                patch(team1, team2Opponent, slot);
            } else {
                patch(team2Opponent, team1, slot);
            }
            if (team2Status) {
                patch(team2, team1Opponent, slot);
            } else {
                patch(team1Opponent, team2, slot);
            }
        }
    }
}
```

Εικόνα 7.9 Μέθοδος υλοποίησης ανταλλαγής αντιπάλων

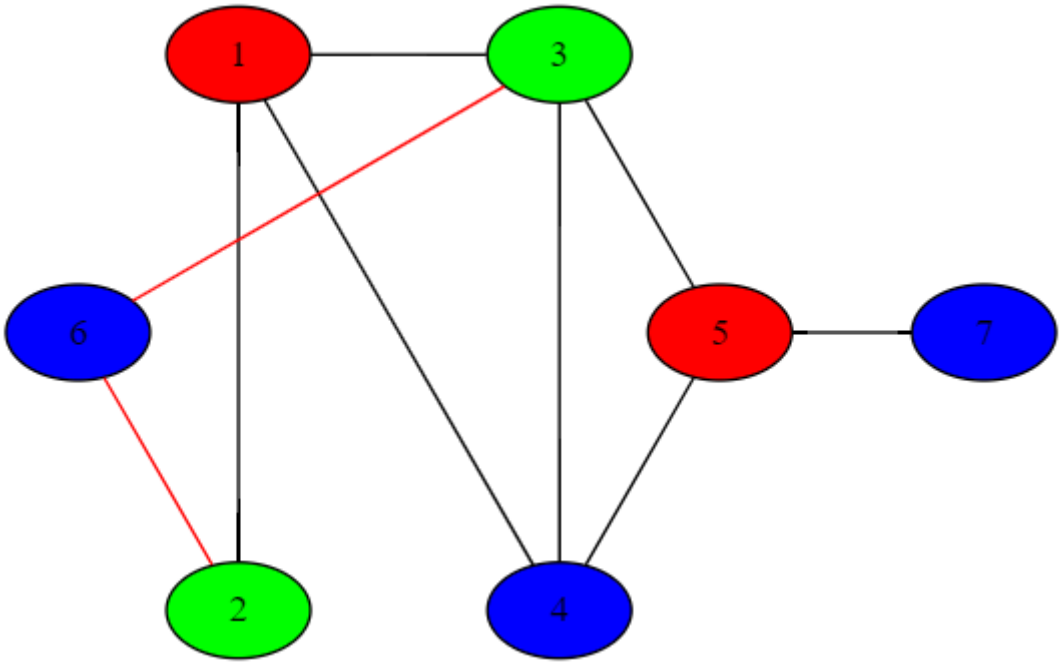
Η μέθοδος swapOpponents στην **Εικόνα 7.9** υλοποιεί το τελεστή γειτνίασης ανταλλαγής αντιπάλων και δέχεται ως παραμέτρους δύο ακέραιες τιμές οι οποίες αναπαριστούν δύο ομάδες. Για κάθε slot αποθηκεύεται η αντίπαλη ομάδα και η κατάσταση εντός-εκτός έδρας της ομάδας και για τις δύο ομάδες. Εάν ο αγώνας διεξάγεται ανάμεσα στις δύο ομάδες-παραμέτρους, τότε ανταλλάζουν κατάσταση εντός-εκτός έδρας. Αλλιώς, οι δύο ομάδες ανταλλάσσουν αντιπάλους μεταξύ τους από τους δύο διεξάγοντες αγώνες.

5.5.5. Αλυσίδα Kempe

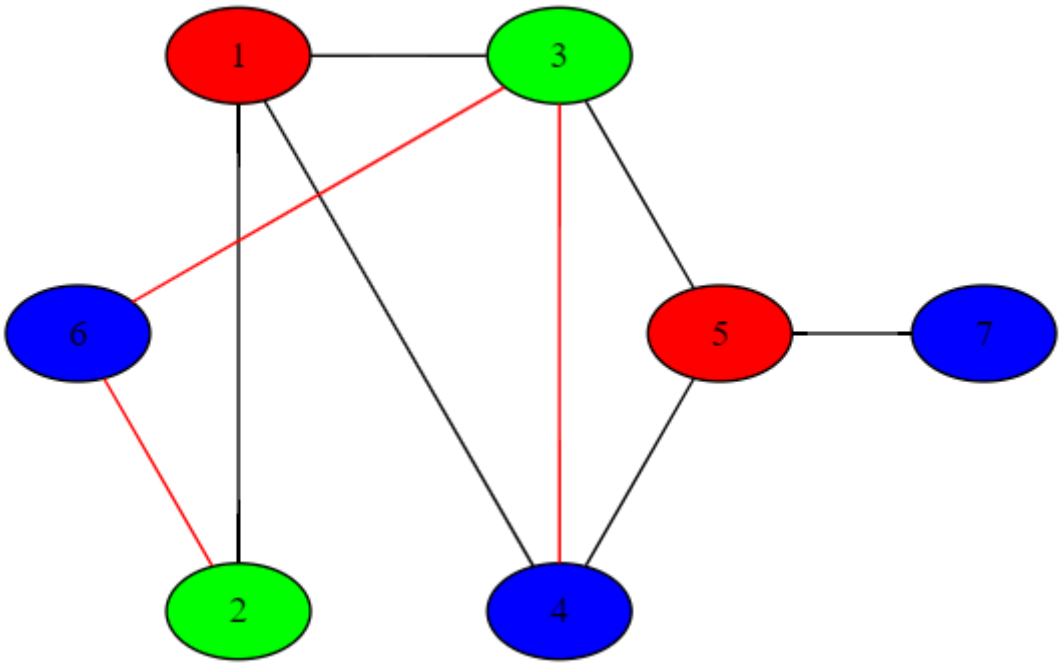
Ο τελεστής γειτνίασης αλυσίδα Kempe εφαρμόζεται σε χρωματισμένα γραφήματα. Επιλέγει τυχαία μία κορυφή του γραφήματος ως αρχική και στη συνέχεια ένα τυχαίο χρώμα για την αντικατάσταση του αρχικού χρώματος. Στη συνέχεια δημιουργείται ένα υπό-γράφημα στο οποίο οι κορυφές του είναι χρωματισμένες μόνο με τα δύο επιλεγμένα χρώματα και υπάρχει μονοπάτι από την αρχική κορυφή μέσω των χρωματισμένων κορυφών με το αρχικό ή το τυχαίο χρώμα. Σε αυτό το υπό-γράφημα κάθε κορυφή του μπορεί να έχει μόνο γειτονική κορυφή αντιστρόφου χρώματος ώστε να μπορεί να πραγματοποιηθεί αντιστροφή των χρωμάτων που έχει ανατεθεί στη κάθε μία.



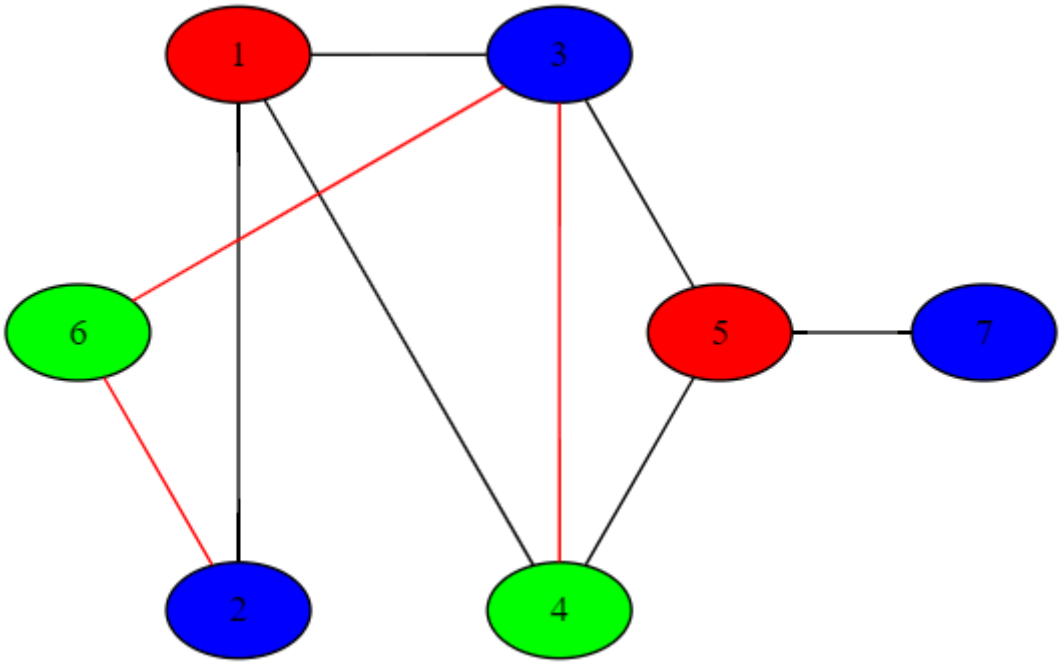
Εικόνα 7.10 Kempe-chain βήμα 1



Εικόνα 7.11 Kempe-chain βήμα 2



Εικόνα 7.12 Kempe-chain βήμα 3



Εικόνα 7.13 Kempe-chain βήμα 4

Στις εικόνες **Εικόνα *7.10***, **Εικόνα *7.11***, **Εικόνα *7.12*** και **Εικόνα *7.13*** απεικονίζονται τα βήματα που εκτελούνται για τον τελεστή γειτνίασης Kempe-chain, τα βήματα έχουν ως εξής:

1. Επιλέγεται τυχαία ως αρχική κορυφή η κορυφή 2 με τυχαίο χρώμα το μπλε.
2. Η κορυφή 2 βρίσκει γειτονική κορυφή με το χρώμα μπλε την κορυφή 6, **Εικόνα 7.10**
3. Η κορυφή 6 βρίσκει γειτονική κορυφή με το χρώμα πράσινο την κορυφή 3, **Εικόνα 7.11**
4. Η κορυφή 3 βρίσκει γειτονική κορυφή με το χρώμα μπλε την κορυφή 4, **Εικόνα 7.12**
5. Εφόσον η κορυφή 4 δε βρίσκει γειτονική κορυφή με πράσινο χρώμα τότε πραγματοποιείται η αλλαγή χρωμάτων κάθε κορυφής με το αντίστροφο χρώμα. Δηλαδή οι κορυφές 2 και 3 θα χρωματιστούν μπλε και οι κορυφές 6 και 4 θα χρωματιστούν πράσινες, **Εικόνα 7.13**

7.7.Αλγόριθμος Hill Climbing

Ο αλγόριθμος Hill Climbing ανήκει στην κατηγορία αλγόριθμων ευριστικής αναζήτησης και χρησιμοποιείται σε απλά προβλήματα, διότι δεν είναι εφικτό να εφαρμοστεί εξαντλητική αναζήτηση σε μεγαλύτερα και πιο περίπλοκα προβλήματα. Επίσης, εκτός από τον απλό αλγόριθμο του Hill Climbing υπάρχουν ποικίλες τροποποιημένες εκδόσεις του αλγορίθμου οι οποίες δεν θα χρησιμοποιηθούν στην υλοποίηση της εφαρμογής.

Η λογική του αλγορίθμου αποτελείται από 5 βήματα τα οποία έχουν ως εξής:

1. Αρχικοποίηση της «καλύτερης λύσης»
2. Δημιούργησε μία «προσωρινή καλύτερη λύση» με τη χρήση της «καλύτερης λύσης»
3. Σύγκρινε την «προσωρινή καλύτερη λύση» με τη «καλύτερη λύση»
4. Εάν η «προσωρινή καλύτερη λύση» είναι καλύτερη τότε αντικατέστησε την «καλύτερη λύση» με αυτήν, αλλιώς η καλύτερη λύση παραμένει ίδια
5. Εφάρμοσε το Βήμα 2

Ο αλγόριθμος εφαρμόζεται n φορές και πραγματοποιεί έξοδο με την επίτευξη του στόχου ή μετά την ολοκλήρωση των επαναλήψεων.

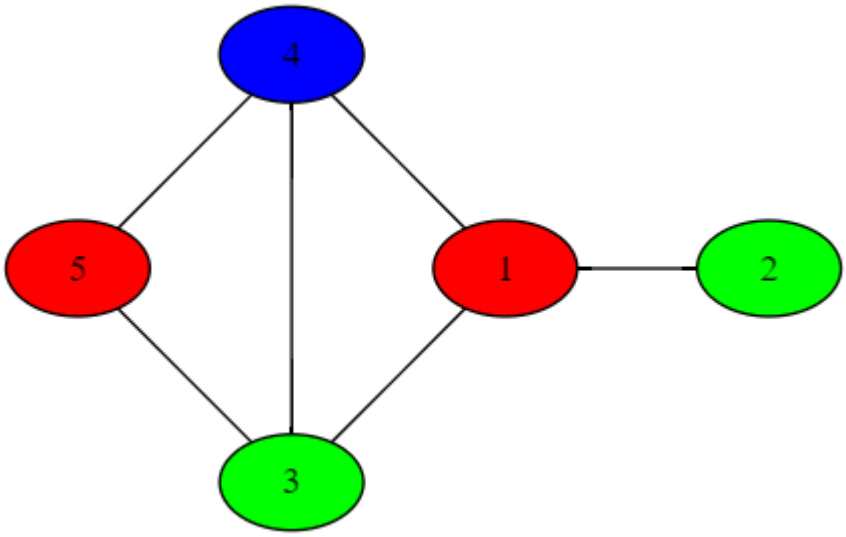
Για την υλοποίηση του αλγορίθμου δημιουργήθηκε η κλάση HillClimbing όπου εφαρμόζει τη λογική του μέσω της μεθόδου bestSolution. Η μέθοδος αυτή επιστρέφει ένα αντικείμενο της κλάσης Solution όπου περιέχει την καλύτερη λύση μετά την εφαρμογή του Hill Climbing και έχει ως παραμέτρους το αρχικό timetable και solution που έχουν δημιουργηθεί με τη χρήση μιας μεθόδου κατασκευής χρονοδιαγράμματος. Σε κάθε επανάληψη του while loop, όπου έχουμε ορίσει τις n επαναλήψεις, αρχικοποιείται ένα αντίγραφο του timetable της καλύτερης λύσης ώστε να εφαρμόζεται ο τελεστής γειτνίασης σε αυτό. Αντίστοιχα δημιουργείται ένα αντίγραφο αντικείμενο solution στο οποίο υπολογίζεται το objective και το infeasibility του αντίγραφου timetable όπου έχει πραγματοποιηθεί μία από τις 4 κινήσεις. Ο τελεστής γειτνίασης επιλέγεται τυχαία μέσω του αντικειμένου Random όπου σε κάθε επανάληψη μας δίνει μία τυχαία επιλογή από το 0 ως το 3. Τέλος πραγματοποιείται ένας έλεγχος όπου εάν το infeasibility του tempSolution είναι μικρότερο από το infeasibility του solution ή το infeasibility του tempSolution είναι ίσο με το infeasibility του solution και το objective του tempSolution είναι μικρότερο από το objective του solution τότε το tempSolution αντικαταστεί το solution και το tempTimetable το timetable. Έτσι ως αποτέλεσμα στο τέλος κάθε επανάληψης είναι να έχει κρατηθεί η καλύτερη λύση.

8. Χρωματισμός Γραφήματος

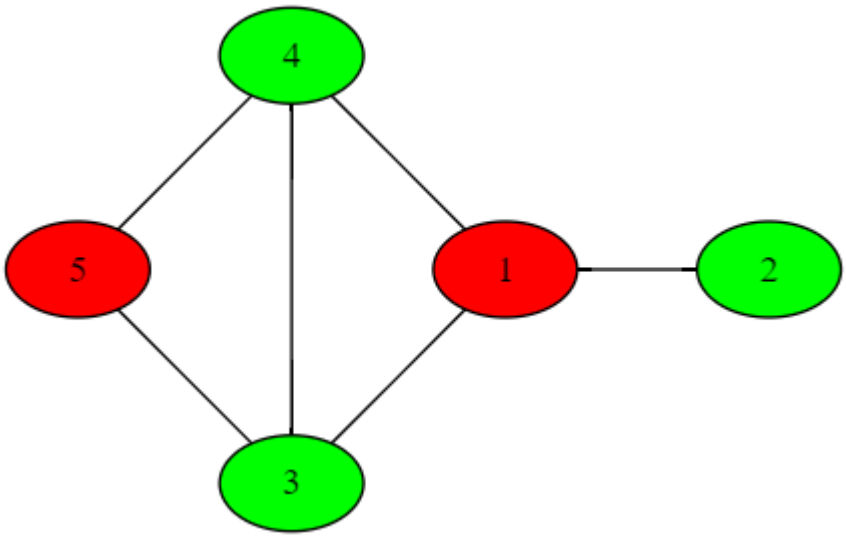
Το πρόβλημα χρονοπρογραμματισμού αθλητικών διοργανώσεων μπορεί να αναπαρασταθεί σε ένα γράφημα ενώ η κατασκευή του χρονοδιαγράμματος πραγματοποιείται με χρωματισμό του γραφήματος.

6.1.Χρωματισμός Κορυφών

Ο χρωματισμός κορυφών είναι από τα πιο δημοφιλή προβλήματα της κατηγορίας NP-Hard προβλημάτων. Σκοπός του είναι η ανάθεση μη κοινού χρώματος για κάθε κορυφή σε σύγκριση με τις γειτονικές κορυφές της. Ένα γράφημα με χρωματισμένες κορυφές θεωρείται βέλτιστο όταν χρησιμοποιεί το μικρότερο δυνατό πλήθος χρωμάτων.



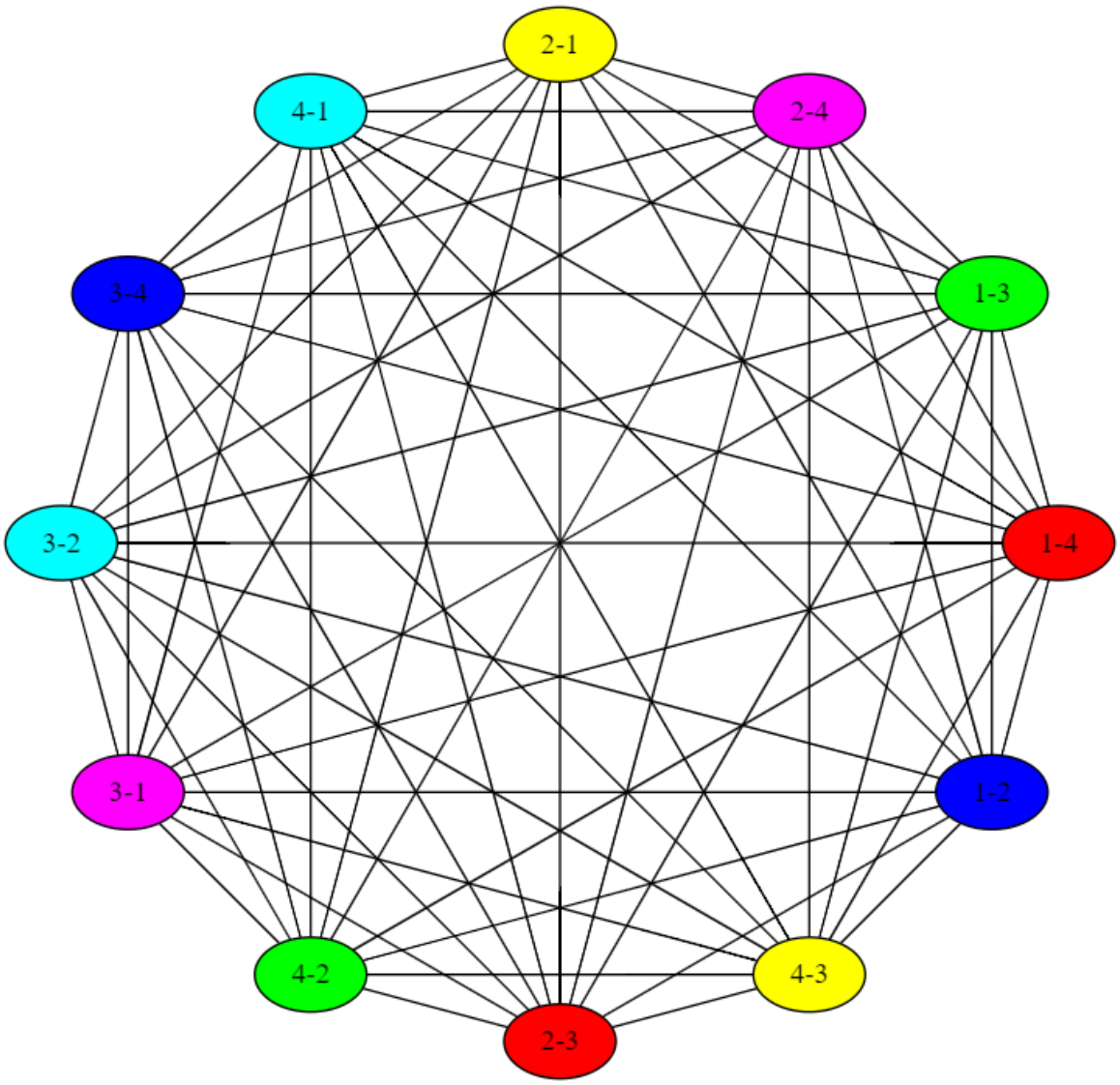
Εικόνα 8.1 Σωστός χρωματισμός κορυφών γραφήματος



Εικόνα 8.2 Λάθος χρωματισμός κορυφών γραφήματος

Για την αναπαράσταση του χρονοδιαγράμματος για το πρόβλημα χρωματισμού κορυφών του γραφήματος, κάθε κορυφή αναπαριστά έναν αγώνα, οι ακμές υπάρχουν ανάμεσα σε κορυφές-αγώνες που δεν διεξάγονται στην ίδια αγωνιστική (slot) και το χρώμα της κορυφής αναπαριστά το slot που διεξάγεται ο αγώνας της κορυφής.

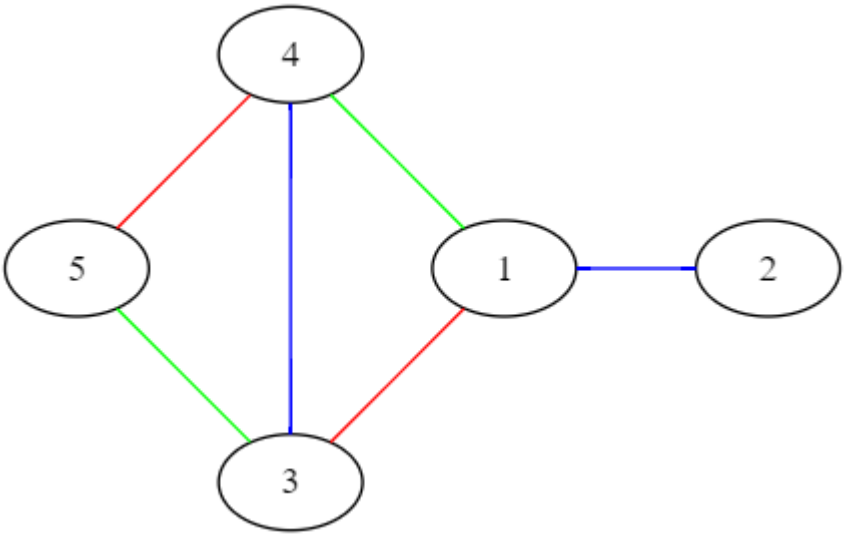
Στο παράδειγμα στην **Εικόνα 8.3** έχει κατασκευαστεί το χρονοδιάγραμμα με πλήθος $2(n-1) = 6$ χρωμάτων-slot για n = 4 ομάδες. Το πλήθος των κορυφών ισούται με $n(n-1) = 12$ κορυφές, όλοι οι υπολογισμοί ισχύουν μόνο για Double Round Robin.



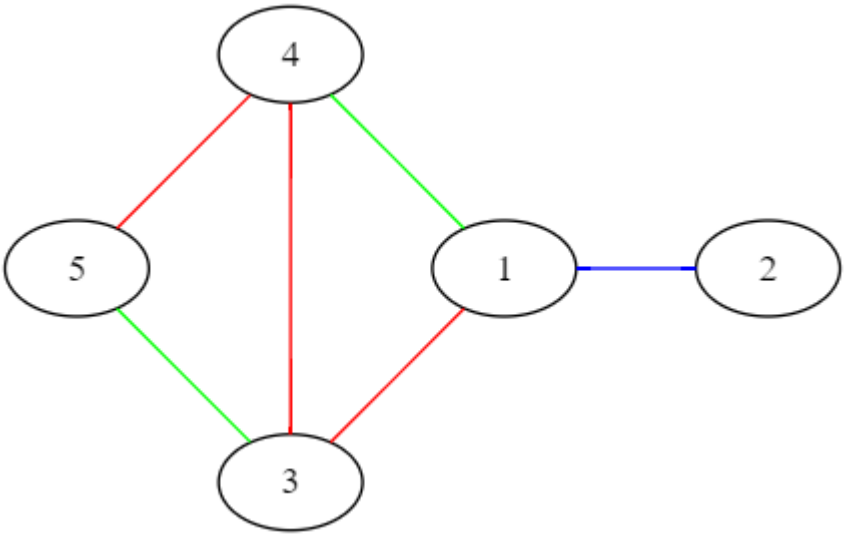
Εικόνα 8.3 Αναπαράσταση Double Round Robin κατασκευασμένου χρονοδιαγράμματος με τη κυκλική μέθοδο σε γράφημα με χρωματισμένες κορυφές για 4 ομάδες

6.2.Χρωματισμός Ακμών

Ο χρωματισμός ακμών του γραφήματος είναι ακόμα ένα πρόβλημα της κατηγορίας NP-Hard προβλημάτων. Σκοπός είναι ο χρωματισμός των ακμών του γραφήματος με διαφορετικό χρώμα ώστε κάθε κορυφή να έχει μοναδικά χρωματισμένες ακμές.



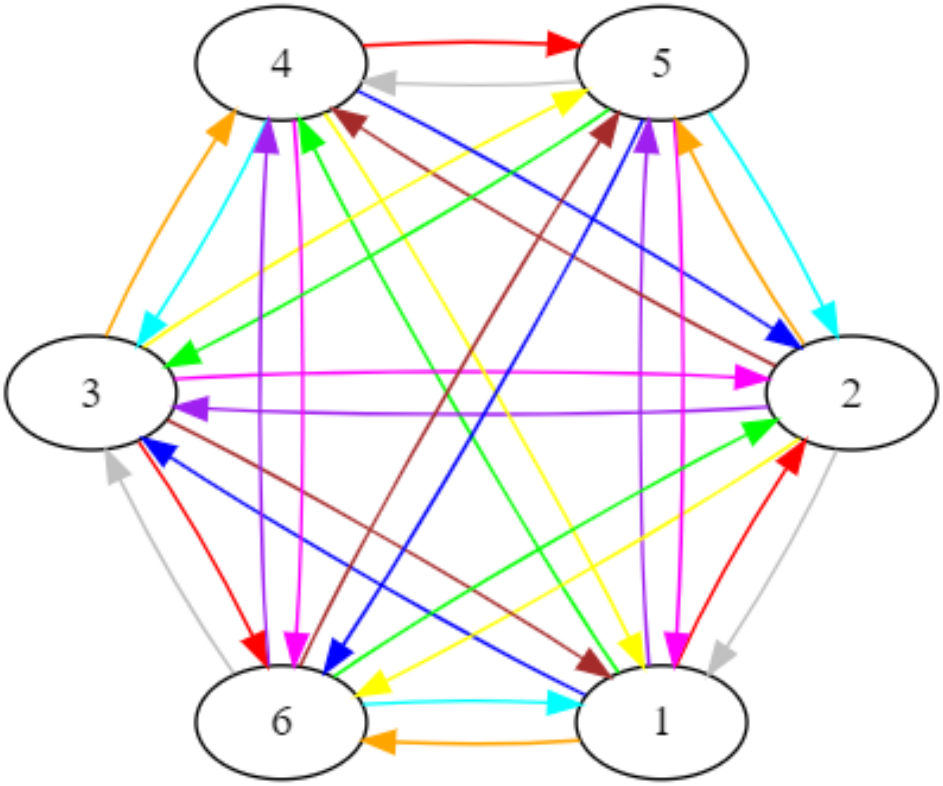
Εικόνα 8.4 Σωστός χρωματισμός ακμών του γραφήματος



Εικόνα 8.5 Λάθος χρωματισμός ακμών του γραφήματος

Μία προσέγγιση λύσης είναι ο χρωματισμός του κατευθυνόμενου γραφήματος που αναπαριστά και αντικαθιστά το χρονοδιάγραμμα. Κάθε ομάδα απεικονίζεται από μία κορυφή στο γράφημα, οι αγώνες απεικονίζονται ως κατευθυνόμενες ακμές και τα slot απεικονίζονται ως χρώμα ακμής.

Η **Εικόνα 8.6** απεικονίζει τη λύση σε κατευθυνόμενο γράφημα με χρωματισμένες ακμές για μία περίπτωση προβλήματος με $n = 6$ ομάδες-κορυφές, $n(n-1) = 30$ αγώνες-ακμές και με πλήθος $2(n-1) = 10$ χρωμάτων-slot.



Εικόνα 8.6 Double Round Robin γράφημα με χρωματισμένες ακμές

9. Αποτελέσματα

Instance	Circle Method		Berger Method	
	Infeasibility	Objective	Infeasibility	Objective
Test Instance 1	0	1079	0	1087
Test Instance 2	0	183	0	182
Test Instance 3	7	1254	0	1253
Test Instance 4	10	4471	10	4431
Test Instance 5	9	195	8	218
Test Instance 6	45	4524	48	3957
Test Instance 7	114	4366	16	3836
Test Instance 8	4	4405	2	2812

Πίνακας 9.1 Αποτελέσματα των Test Instances μέσω του Hill Climbing

Στον παραπάνω **Πίνακας *9.1*** έγινε καταγραφή των κοστών παραβίασης περιορισμών της λύσης που παράχθηκε μέσω του αλγορίθμου Hill Climbing. Για κάθε μέθοδο κατασκευής χρονοδιαγράμματος και για κάθε περίπτωση προβλήματος, η εφαρμογή εκτελέστηκε πέντε φορές και έγινε καταγραφή του καλύτερου κόστους. Οι επαναλήψεις του αλγορίθμου Hill Climbing ορίστηκαν στις 10.000 για κάθε εκτέλεση της εφαρμογής. Προτεραιότητα στη σύγκριση μεταξύ των λύσεων, έχει η λύση με τη μικρότερη τιμή infeasibility. Σε περίπτωση που οι δύο λύσης έχουν ίση τιμή infeasibility, τότε καλύτερη λύση θεωρείται εκείνη που έχει τη μικρότερη τιμή objective. Τέλος, η μέθοδος κατασκευής χρονοδιαγράμματος χρησιμοποιείται μόνο για την αρχική λύση, η τελική λύση όπως αναφέρθηκε και στην αρχή παράγεται από τον αλγόριθμο του Hill Climbing.

Στη περίπτωση προβλήματος Test Instance 1 παρατηρούμε ένα μικρό προβάδισμα με τη χρήση της Circle Method ενώ στη περίπτωση προβλήματος Test Instance 2 με μικρή διαφορά η καλύτερη λύση δίνεται με τη χρήση της Berger Method. Το Test Instance 1 και 2 είναι οι μικρότερες περιπτώσεις προβλημάτων από το Σύνολο περιπτώσεων προβλημάτων, ως συμπέρασμα τα μικρά προβλήματα δεν επηρεάζονται σε μεγάλο βαθμό από τη χρήση διαφορετικών μεθόδων κατασκευής χρονοδιαγραμμάτων.

Στη περίπτωση προβλήματος Test Instance 3 παρατηρούμε ότι η καλύτερη λύση δίνεται με τη χρήση του Berger Method. Η διαφορά του infeasibility μεταξύ των δύο λύσεων είναι σημαντική διότι στη καλύτερη λύση η τιμή του είναι μηδενική, ενώ τη ίδια στιγμή είναι μικρότερη και η τιμή του Objective. Σε αυτή τη περίπτωση προβλήματος επηρεάστηκε το αποτέλεσμα από τη μέθοδο κατασκευής χρονοδιαγράμματος.

Στη περίπτωση προβλήματος Test Instance 4 η καλύτερη λύση δίνεται μέσω του Berger Method, όμως παρατηρούμε πως η διαφορά είναι μικρή ανάμεσα στις δύο λύσεις εφόσον το infeasibility έχει την ίδια τιμή ενώ η τιμή του objective έχει μία μικρή διαφορά που καθορίζει την καλύτερη λύση.

Στη περίπτωση προβλήματος Test Instance 5 παρατηρούμε πως η λύση μέσω της Berger Method έχει μικρότερη τιμή infeasibility και μεγαλύτερη τιμή objective σε σχέση με τη λύση μέσω της Circle Method. Όμως λόγω προτεραιότητας που αναφέραμε παραπάνω, η καλύτερη λύση είναι μέσω της Berger Method.

Στη περίπτωση προβλήματος Test Instance 6 παρατηρούμε πως η καλύτερη λύση δίνεται μέσω της Circle Method με μικρή διαφορά στο infeasibility αλλά αρκετά σημαντική ως προς το χειρότερο διαφορά στο objective. Συμπέρασμα, η τιμή objective στο αποτέλεσμα της λύσης με τη χρήση του Berger Method επηρεάζεται σημαντικά σε αυτή τη περίπτωση προβλήματος αλλά όχι τόσο η τιμή infeasibility.

Στη περίπτωση προβλήματος Test Instance 7 παρατηρούμε τεράστια διαφορά ανάμεσα στις δύο τιμές των δύο λύσεων. Η καλύτερη λύση δίνεται μέσω της Berger Method, όπου το κόστος παραβίασης σκληρών περιορισμών σε σύγκριση με το κόστος της λύσης μέσω Circle Method είναι πολύ μικρότερο. Αντίστοιχα το ίδιο παρατηρούμε και με το κόστος των μαλακών περιορισμών. Συμπέρασμα, η λύση του Test Instance 7 επηρεάζεται σε μεγάλο βαθμό από τη μέθοδο κατασκευής χρονοδιαγράμματος.

Τέλος, στη περίπτωση προβλήματος Test Instance 8 η καλύτερη λύση δίνεται μέσω της Berger Method η οποία υπερτερεί με χαμηλότερο κόστος και στις δύο τιμές της. Παρατηρούμε μία αρκετά χαμηλότερη τιμή του objective σε σχέση με τη τιμή του objective μέσω της Circle Method.

	Circle Method		Berger Method	
Instance	**Infeasibility**	**Objective**	**Infeasibility**	**Objective**
Early 1	26	1764	2	1050
Early 2	40	848	2	529
Early 3	11	2523	11	2384
Early 4	73	1719	40	617
Early 5	127	3725	71	2848
Early 6	62	6283	55	4410
Early 7	48	20290	44	12652
Early 8	1	3139	0	2975
Early 9	1	2628	0	2068
Early 10	120	4772	54	3487
Early 11	47	19498	41	10586
Early 12	38	2010	24	810
Early 13	3	1615	4	1057
Early 14	2	1904	2	1459
Early 15	33	7002	32	6489

Πίνακας 9.2 Αποτελέσματα των Early Instances μέσω του Hill Climbing

	Circle Method		Berger Method	
Instance	**Infeasibility**	**Objective**	**Infeasibility**	**Objective**
Middle 1	90	6268	75	5551
Middle 2	112	7193	81	6350
Middle 3	83	13730	78	10045
Middle 4	11	247	8	224
Middle 5	6	2097	3	1833
Middle 6	37	2865	14	2126
Middle 7	38	9254	46	10110
Middle 8	4	1462	0	1048
Middle 9	4	3230	2	2495
Middle 10	69	2376	16	2194
Middle 11	43	4682	22	4693
Middle 12	6	2855	0	1375
Middle 13	38	12141	36	11480
Middle 14	4	3527	0	2943
Middle 15	4	2333	1	2438

Πίνακας 9.3 Αποτελέσματα των Middle Instances μέσω του Hill Climbing

	Circle Method		Berger Method	
Instance	**Infeasibility**	**Objective**	**Infeasibility**	**Objective**
Late 1	50	3343	46	3018
Late 2	74	6564	61	6170
Late 3	5	12404	5	6169
Late 4	12	279	7	8
Late 5	111	2815	59	2619
Late 6	35	2397	39	2603
Late 7	6	13826	4	9275
Late 8	8	2184	7	1249
Late 9	4	3429	2	2138
Late 10	137	3166	39	2212
Late 11	45	1286	0	931
Late 12	62	16410	31	10931
Late 13	33	14853	30	11532
Late 14	3	3831	1	3353
Late 15	2	1540	1	1160

Πίνακας 9.4 Αποτελέσματα των Late Instances μέσω του Hill Climbing

8. Συμπεράσματα

Από τις λύσεις που δίνονται παρατηρούμε ότι στις περισσότερες περιπτώσεις δίνεται η καλύτερη λύση μέσω της Berger Method. Στο Test Instance 1 και 2 οι διαφορές είναι μικρότερες αλλά όσο μεγαλώνει το πρόβλημα φαίνεται ξεκάθαρα η διαφορά, όπως σε πιο μέτρια προβλήματα σαν τα Test Instance 3, 4, 5 και 6. Στα μεγαλύτερα προβλήματα όπως το Test Instance 7 και 8 η διαφορά των τιμών κόστους της λύσης είναι αισθητή. Από τις λύσεις που καταγράφηκαν μπορούμε να συμπεράνουμε πως εν τέλη η μέθοδος κατασκευής χρονοδιαγράμματος επηρεάζει τη λύση που παράγεται από τον αλγόριθμο Hill Climbing. Υπάρχει όμως και ο παράγοντας της τυχαίας επιλογής τελεστή γειτνίασης στο χρονοδιάγραμμα του αλγορίθμου Hill Climbing που θα μπορούσε να μας αποτρέψει από το τελικό συμπέρασμα.

Βιβλιογραφία

Bulck, D.V., Goossens, D., Beliën, J., Davari, M., n.d. ITC2021 – Sports Timetabling Problem Description and File Format 8.

Lambrechts, E., Ficker, A.M.C., Goossens, D.R., Spieksma, F.C.R., 2018. Round-robin tournaments generated by the Circle Method have maximum carry-over. Math. Program. 172, 277–302. https://doi.org/10.1007/s10107-017-1115-x

Lewis, R., Thompson, J., 2011. On the application of graph colouring techniques in round-robin sports scheduling. Comput. Oper. Res., Project Management and Scheduling 38, 190–204. https://doi.org/10.1016/j.cor.2010.04.012

Van Bulck, D., Goossens, D., Schönberger, J., Guajardo, M., 2020. RobinX: A three-field classification and unified data format for round-robin sports timetabling. Eur. J. Oper. Res. 280, 568–580. https://doi.org/10.1016/j.ejor.2019.07.023