ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ
Σχολή Μηχανικών Παραγωγής και Διοίκησης

Χρήση Εξελικτικών Αλγορίθμων για την Επίλυση του Διλήμματος του Φυλακισμένου

Μεταπτυχιακή Διατριβή

Μανούσος Ρηγάκης

Επιβλέπων
Ιωάννης Μαρινάκης

Χανιά, Απρίλιος 2016

ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ
Σχολή Μηχανικών Παραγωγής και Διοίκησης

Χρήση Εξελικτικών Αλγορίθμων για την Επίλυση του Διλήμματος του Φυλακισμένου

Μεταπτυχιακή Διατριβή

Μανούσος Ρηγάκης

Εξεταστική Επιτροπή
Ιωάννης Μαρινάκης, Επίκουρος Καθηγητής (Επιβλέπων)
Αριστομένης Αντωνιάδης, Καθηγητής
Γεώργιος Σταυρουλάκης, Καθηγητής

Χανιά, Απρίλιος 2016

TECHNICAL UNIVERSITY OF CRETE
School of Production Engineering and Management

Evolutionary Algorithms for Solving the Prisoner's Dilemma

Master Thesis

Manousos Rigakis

Supervisor
Yannis Marinakis

Chania, April 2016

Στην γιαγιά μου
Ευτυχία

Περίληψη

Η συγκεκριμένη μεταπτυχιακή διατριβή προτείνει δύο δυαδικούς αλγορίθμους για την ανάπτυξη στρατηγικών για το επαναληπτικό δίλημμα του φυλακισμένου (IPD). Για να καθορίσουμε την ποιότητα των στρατηγικών πραγματοποιείται μια σύγκριση ανάμεσα στον δυαδικό αλγόριθμο της τεχνητής αποικίας μελισσών (ABC), του δυαδικού αλγορίθμου της διαφορικής εξέλιξης (DE) και σε αρκετές βιβλιογραφικές στρατηγικές. Ενώ για την καλύτερη διερεύνηση των αποτελεσμάτων μας και για την επιλογή του καταλληλότερου από αυτούς τους δύο αλγορίθμους για το συγκεκριμένο πρόβλημα, οι δύο αλγόριθμοι που υλοποιήσαμε αντιμετωπίζουν τον αλγόριθμο βελτιστοποίησης σμήνους σωματιδίων (PSO). Τέλος, οι αλγόριθμοι DE και ABC συγκρίνονται μεταξύ τους παίζοντας ο ένας ενάντια στον άλλον. Στην συγκεκριμένη διατριβή εξετάζουμε λοιπόν την καταλληλότητα των δύο αυτών αλγορίθμων (ABC, DE) να παράγουν στρατηγικές για το IPD, το όποιο δεν έχει μελετηθεί στο παρελθόν.

Λέξεις κλειδιά

Αλγόριθμος Τεχνητής Αποικίας Μελισσών, Αλγόριθμος Διαφορικής Εξέλιξης, Αλγόριθμος Βελτιστοποίησης Σμήνους Σωματιδίων, Επαναληπτικό Δίλημμα του Φυλακισμένου.

Abstract

This master thesis proposes two binary algorithms to develop game strategies for Iterated Prisoner's Dilemma (IPD). To determine the quality of the evolved strategies, a comparison is made between the binary ABC approach, the binary DE approach and several benchmark strategies. For better investigation of our results and the selection of the most suitable algorithm for the specific problem, both algorithms that we produced, counter the Particle Swarm Optimization algorithm (PSO). At last both algorithms DE and ABC are playing against each other. In this master thesis we examine the suitability of DE and ABC to generate strategies for the IPD which has not been investigated before.

Keywords

Artificial Bee Colony, Differential Evolution, Particle Swarm Optimization, Iterated Prisoner's Dilemma

Περιεχόμενα

Κατάλογος Σχημάτων

Ευχαριστίες

Θα ήθελα να ευχαριστήσω όλους όσοι με στήριξαν κατά την περίοδο των σπουδών μου, σε προπτυχιακό και μεταπτυχιακό επίπεδο, φίλους, καθηγητές καθώς και την οικογένεια μου. Ευχαριστώ ιδιαίτερα τον καθηγητή μου Ιωάννη Μαρινάκη ο οποίος υπήρξε για μένα ο άνθρωπος που μου έδειξε τους ορίζοντες της θεωρίας παιγνίων και στάθηκε δίπλα μου με υπομονή και κατανόηση κατά την εκπόνηση της μεταπτυχιακής μου διατριβής. Θέλω επίσης να ευχαριστήσω θερμά την Δρ. Μαγδαληνή Μαρινάκη πού ήταν πάντα δίπλα μου πρόθυμη να με βοηθήσει. Ευχαριστώ επίσης όλους τους φίλους μου, καθώς και την σύντροφο μου Δήμητρα που όλα αυτά τα χρόνια υπήρξαν στήριγμα για εμένα και με βοήθησαν να αντιμετωπίσω οποιοδήποτε πρόβλημα εμφανιζόταν, τόσο σε επιστημονικό αλλά κυρίως σε προβλήματα της καθημερινής ζωής. Τέλος, τις μεγαλύτερες ευχαριστίες τις οφείλω στους γονείς μου Μανώλη και Χρυσούλα και στην αδερφή μου Έφη για την αμέριστη συμπαράσταση τους όλα αυτά τα χρόνια καθώς και τα χρόνια που θα ακολουθήσουν.

Κεφάλαιο 1

Εισαγωγή

Το δίλημμα του φυλακισμένου είναι το πιο δημοφιλή παίγνιο στρατηγικής το οποίο εφαρμόζεται σε πολλές επιστήμες όπως την οικονομία, βιολογία [4], θεωρία παιγνίων, καθώς και στις πολίτικές επιστήμες. Αρχικά προτάθηκε από τους Merril Flood και Melvin Dresher το 1950 [5] ως ένα μη-συνεργατικό παίγνιο και αργότερα ο Albert W. Tucker [17] το ονόμασε σαν δίλημμα του φυλακισμένου. Αυτό το παίγνιο είναι χρήσιμο για να παρουσιαστεί η εξέλιξη της συνεργατικής συμπεριφοράς. Η πρώτη μελέτη του συγκεκριμένου προβλήματος προτάθηκε από τον Axelrod [2] το 1987 ο οποίος χρησιμοποίησε γενετικούς αλγορίθμους.

Στην συγκεκριμένη μεταπτυχιακή διατριβή, εξετάζουμε την ικανότητα δύο αλγορίθμων εμπνευσμένων από την φύση (αλγόριθμος τεχνητής αποικίας μελισσών [8], αλγόριθμος βελτιστοποίησης σμήνους σωματιδίων [9]) καθώς και τον εξελικτικό αλγόριθμο της διαφορικής εξέλιξης [15] για να δημιουργήσουμε ανταγωνιστικές στρατηγικές για το επαναληπτικό δίλημμα του φυλακισμένου.

Αυτή η μεταπτυχιακή διατριβή αποτελείτε από τα ακόλουθα κεφάλαια: Στο Κεφάλαιο 2 αναφέρουμε τι ακριβώς είναι η θεωρία παιγνίων, που εφαρμόζεται ενώ γίνεται και μία μικρή ιστορική αναδρομή. Στο κεφάλαιο 3 ορίζουμε τυπικά τις έννοιες των παίγνιων μη σταθερού αθροίσματος, των συνεργατικών και μη συνεργατικών παίγνιων καθώς και της ισορροπίας Nash [11]. Το μεγαλύτερο μέρος του Κεφαλαίου αυτού έχει χρησιμοποιηθεί για να αναλυθεί εκτενέστερα το δίλημμα του φυλακισμένου και οι βιβλιογραφικές στρατηγικές που χρησιμοποιήσαμε. Στο Κεφάλαιο 4 και 5 παρουσιάζονται αναλυτικότερα οι αλγόριθμοι που χρησιμοποιήσαμε καθώς και το πώς τους τροποποιήσαμε για να προσεγγίσουμε το επαναληπτικό δίλημμα του φυλακισμένου. Τέλος στο κεφάλαιο 6 παρουσιάζουμε τα αποτελέσματα και αναφέρουμε τα συμπεράσματα μας ενώ παράλληλα επιγραμματικά αναφέρουμε κάποιους μελλοντικούς στόχους.

Κεφάλαιο 2

Βασικές Έννοιες και Ιστορική Αναδρομή

2.1 Τι είναι η θεωρία παιγνίων και που εφαρμόζεται

Η θεωρία παιγνίων παρέχει τη μαθηματική βάση για την ανάλυση διαδραστικών διαδικασιών λήψης αποφάσεων ανάμεσα σε ορθολογικά άτομα τα οποία είναι αμοιβαία αλληλεξαρτώμενα και μάλιστα με αντικρουόμενα συμφέροντα.

Ένα παίγνιο περιλαμβάνει τρία σύνολα: α) το σύνολο των παικτών, β) το σύνολο των δράσεων, γ) το σύνολο των κερδών. Οι παίκτες είναι αυτοί που λαμβάνουν τις αποφάσεις στο εκάστοτε υπό μελέτη πρόβλημα. Οι παίκτες μπορεί να είναι ένα πρόσωπο, μία οργάνωση, ένα κράτος, ή ένας συνασπισμός. Η θεωρία παιγνίων μπορεί να εφαρμοστεί σε διάφορα προβλήματα πολιτικής, ψυχολογικής, κοινωνικής και οικονομικής μορφής. Γενικά, όταν μελετάμε ένα παίγνιο, υποθέτουμε ότι κάθε παίκτης δεν επιλέγει απαραίτητα μία μόνο από τις διαθέσιμες ενέργειές του, αλλά είναι ελεύθερος να επιλέξει οποιαδήποτε κατανομή πιθανότητας στο σύνολο των ενεργειών του. Μία τέτοια κατανομή πιθανότητας ονομάζεται στρατηγική. Η ειδική περίπτωση στρατηγικής που θέτει πιθανότητα 1 σε μία μόνο ενέργεια ονομάζεται αγνή στρατηγική. Η σπουδαιότερη και περισσότερο αναγνωρισμένη έννοια λύσης στην θεωρία παιγνίων είναι η ισορροπία Nash [10]. Πρόκειται για έναν συνδυασμό στρατηγικών, μία για κάθε παίκτη, όπου δεν υπάρχει παίκτης που να μπορεί να αυξήσει την αναμενόμενη ωφέλεια του αν αλλάξει την στρατηγική του. Με άλλα λόγια σε μία ισορροπία Nash [11] κάθε παίκτης επιλέγει την στρατηγική που μεγιστοποιεί την αναμενόμενη ωφέλεια του, δοθέντος των στρατηγικών των άλλων παικτών. Αυτή η σημαντική έννοια ισορροπίας προτάθηκε από τον Nash [11] το 1951, ο οποίος μάλιστα απέδειξε ότι κάθε παίγνιο (με πεπερασμένο πλήθος παικτών) έχει μία τέτοια ισορροπία. Ένα από τα πιο διάσημα προβλήματα της θεωρίας παιγνίων είναι το Δίλημμα του Φυλακισμένου όπου αναλύεται εκτενέστερα παρακάτω.

2.2 Ιστορική Αναδρομή

Η πρώτη γνωστή αναφορά στην θεωρία παιγνίων έγινε τον 18ο αιώνα από τον Γάλλο οικονομολόγο Augustin Cournot ο οποίος κατάφερε να αναλύσει ολιγοπωλιακές καταστάσεις με τρόπο παρόμοιο με τις σύγχρονες μεθόδους της θεωρίας παιγνίων. Το όνομα του πήρε και το αντίστοιχο μοντέλο (Cournot).

Η ουσιαστική ανάπτυξη της θεωρίας παιγνίων ξεκίνησε κατά την δεκαετία του 1920 από τους μαθηματικούς Èmile Borel (1871-1956) και John von Neumann (1903-1957) ο οποίος το 1928 απέδειξε ότι τα παίγνια μηδενικού αθροίσματος έχουν πάντα λύση και ότι η απώλεια ενός παίκτη είναι ίση με το κέρδος του δεύτερου.

Το 1944 οι John von Neumann και Oscar Morgestern (1902-1977) εξέδωσαν το βιβλίο *Theory of Games and Economic Behavior*. Μέσα από αυτό όρισαν αξιωματικά την θεωρία της χρησιμότητας (Utility Theory), ανέλυσαν διεξοδικά τις βέλτιστες λύσεις στα παίγνια μηδενικού αθροίσματος και εισήγαγαν μια νέα κατηγορία παιγνίων, τα συνεργατικά παίγνια (Cooperative Games).

Στις αρχές της δεκαετίας του 1950 ο μαθηματικός John F. Nash (1928-2015) έδωσε μια νέα διάσταση στη μελέτη των παιγνίων, εισάγοντας μια σπουδαία έννοια ισορροπίας, γνωστή πλέον ως *ισορροπία Nash*. Η έννοια της *ισορροπίας Nash* εφαρμόζεται και στα παίγνια μη-μηδενικού αθροίσματος. Η εργασία του Nash μπορεί να θεωρηθεί ότι αποτελεί επέκταση της εργασίας του Cournot. Ουσιαστικά, *ισορροπίας Nash* σε ένα παίγνιο είναι μία κατάσταση από την οποία δεν συμφέρει κανέναν παίκτη να ξεφύγει μεμονωμένα.

Από εκείνο το σημείο και μετά η θεωρία παιγνίων είχε αλματώδη ανάπτυξη και άρχισε να εφαρμόζεται σε όλους τους τομείς και τις πολιτικές επιστήμες, ενώ πληθώρα ερευνητικών πειραμάτων ξεκίνησαν προσπαθώντας να βρουν λύση σε όλο και περισσότερα προβλήματα.

Το 1965 ο Reinhard Selten (1930-) μελέτησε τα δυναμικά παίγνια, δηλαδή τα παίγνια που εξελίσσονται στο χρόνο. Το 1975 ο John Harsanyi (1920-2000) γενίκευσε τις ιδέες του Nash σε παίγνια μη-πλήρους πληροφόρησης σχετικά με τις προτιμήσεις και αποφάσεις των άλλων παικτών. Για τις εργασίες τους, οι τρεις αυτοί άνθρωποι τιμήθηκαν αργότερα, το 1994, με το βραβείο Νόμπελ της Σουηδικής Ακαδημίας Επιστημών.

Κατά την δεκαετία του 1970 άρχισε να εφαρμόζεται στον κλάδο της βιολογίας σαν αποτέλεσμα της εργασίας του John Maynard Smith (1920-2004) που εισήγαγε την έννοια της *εξελικτικά σταθερής στρατηγικής*.

Το 2005 ο Αμερικανός επιστήμονας Tomas Schelling (1921-) και ο Γερμανός

θεωρητικός παιγνίων Robert Aumann (1930-) ανέπτυξαν περαιτέρω τις έννοιες της συνεργασίας και του ανταγωνισμού. Για αυτήν τους την συνεισφορά βραβεύθηκαν με το βραβείο Νόμπελ στις οικονομικές επιστήμες.

Το 2007 οι Roger Myerson (1951-), Leionid Hurwicz (1917-2008) και Eric Maskin (1950-) κέρδισαν το βραβείο Νόμπελ για τις οικονομικές επιστήμες για *τη θεμελίωση της θεωρίας σχεδιασμού μηχανισμών*.

Κεφάλαιο 3

Παίγνια και το Δίλημμα του Φυλακισμένου

3.1 Παίγνια μη-σταθερού αθροίσματος

Τα περισσότερα πρότυπα παιγνίου που εμφανίζονται σε οικονομικές αλλά και άλλες κοινωνικές περιπτώσεις δεν είναι σταθερού ή μηδενικού αθροίσματος, διότι είναι σπάνιο φαινόμενο να υπάρχουν π.χ. οικονομικοί ή εμπορικοί ανταγωνιστές που έχουν ολική σύγκρουση συμφερόντων. Στα παίγνια αυτά παρατηρείται δηλαδή το φαινόμενο να ποικίλει το άθροισμα των αμοιβών ή απωλειών των παικτών. Έτσι εάν a_{ij} είναι η αμοιβή του παίκτη I και b_{ij} η αμοιβή του παίκτη II για την επιλογή στρατηγικής i από τον πρώτο και j από τον δεύτερο, έχουμε ότι η ποσότητα $a_{ij} + b_{ij}$ είναι διαφορετική για διαφορετικά ζεύγη στρατηγικών (i, j), σε αντίθεση προς τα παίγνια μηδενικού αθροίσματος, όπου $b_{ij} = -a_{ij}, \forall(i, j)$, και τα παίγνια σταθερού αθροίσματος, όπου $b_{ij} = c - a_{ij}, \forall(i, j)$. Για τον λόγο αυτό, τα παίγνια αυτά ονομάζονται **παίγνια μη-σταθερού αθροίσματος** [20] (non-costant sum games). Στα παίγνια αυτά είναι αναγκαστικό να αναφέρονται ρητώς οι αμοιβές αμφοτέρων των παικτών αφού δεν είναι δυνατόν να συμπεράνουμε τις αμοιβές του ενός από τις αμοιβές του άλλου.

Γενικά, ένα παίγνιο με μη-σταθερό άθροισμα διατυπώνεται ως ένα ζεύγος μητρών αμοιβής ή απώλειας, A και B, ή ως μία διπλή μήτρα $[A, B]$, όπου κάθε στοιχείο της είναι ένα διατεταγμένο ζεύγος (a_{ij}, b_{ij}) στοιχείων της A και B αντίστοιχα. Τα στοιχεία a_{ij} και b_{ij} είναι οι αμοιβές ή απώλειες του παίκτη I και II, εφόσον επιλέξουν την στρατηγική i και j αντίστοιχα. Παίγνια αυτής της μορφής ονομάζονται **δι-μητρικά παίγνια** ή **δι-πινακοπαίγνια** (bi-matrix games).

3.2 Συνεργατικά και μη συνεργατικά παίγνια

Στα παίγνια μη-σταθερού αθροίσματος, ενώ η συνεργασία μεταξύ των παικτών μπορεί να αποβεί σε κοινό όφελος συχνά παρατηρείται το φαινόμενο της απαγόρευσης μιας τέτοιας συνεργασίας. Άλλες πάλι φορές μία δεσμευτική συμφωνία μεταξύ των παικτών αποτρέπεται λόγω της έλλειψης αμοιβαίας εμπιστοσύνης ή και επικοινωνίας. Συνεπώς διακρίνουμε δύο ακραίες περιπτώσεις δι-μητρικών παιγνίων.

- **Συνεργατικά παίγνια** (cooperative games [11]) Σε αυτού του είδους παίγνια οι παίκτες έχουν το δικαίωμα επικοινωνίας πριν την έναρξη της παρτίδας και συνεπώς επιτρέπεται να προβούν σε όλες τις μορφές συνεργασίας που είναι δεσμευτικές, και
- **Μη συνεργατικά παίγνια** (non-cooperative games [11]) όπου η επικοινωνία μεταξύ των παικτών πριν την έναρξη της παρτίδας δεν επιτρέπεται και συνεπώς οποιαδήποτε συμφωνία είναι αδύνατη.

3.3 Μαθηματική Διατύπωση Στρατηγικών Παιγνίων

Ένα στρατηγικό παίγνιο αποτελείται απο n παίκτες, από τους οποίους ο καθένας μπορεί να επιλέξει από ένα διαφορετικό σύνολο στρατηγικών που αντιστοιχεί στον εν λόγω παίκτη. Ανάλογα με τις στρατηγικές που επιλέγουν οι παίκτες καθορίζεται για καθέναν από αυτούς το όφελος του (payoff).

Θεώρημα 3.1 *Ένα στρατηγικό παίγνιο ορίζεται από [11], [18]:*

- *ένα μη κενό σύνολο N (το σύνολο των παικτών)*
- *για κάθε παίκτη $i \in N$*
  - *ένα μη κενό σύνολο S_i (το σύνολο των στρατηγικών του παίκτη i)*
  - *μία συνάρτηση $u_i : \Pi_{i \in N} S_i \rightarrow \Re$ (η συνάρτηση που εκφράζει το όφελος (payoff) του παίκτη i)*

3.4 Ισορροπία Nash

Η ισορροπία Nash [10] προσπαθεί να μοντελοποιήσει την ισορροπία ενός παίγνιου στο οποίο οι παίκτες δεν συνεργάζονται μεταξύ τους και στο οποίο κάθε

παίκτης δρα ορθολογικά (rationally) και έχει πλήρη γνώση των στρατηγικών των υπολοίπων παικτών. Είναι λογικό να θεωρήσουμε ότι ένα τέτοιο παίγνιο ισορροπεί όταν κανείς παίκτης δεν έχει κίνητρο να αλλάξει να αποκλίνει μονομερώς από την στρατηγική του. Τυπικότερα η ισορροπία Nash [10] ορίζεται ως ακολούθως.

Θεώρημα 3.2 *Ισορροπία Nash*

Ένα στρατηγικό προφίλ $s \in \Pi_{i \in N} S_i$ ονομάζεται γνήσια ισορροπία Nash [10], [18] ή απλά ισορροπία Nash ενός στρατηγικού παίγνιου G αν για κάθε παίκτη $i \in N$ ισχύει:

$$u_i(s_i, s_{-i}) \geq u_i(x, s_{-i}), \forall x \in S_i \tag{3.1}$$

3.5 Το Δίλημμα του Φυλακισμένου (Prisoner's Dilemma (PD))

Το πιο γνωστό και σημαντικό παίγνιο στην ιστορία της θεωρίας παιγνίων είναι το παίγνιο του διλήμματος του φυλακισμένου (Prisoner's dilemma). Τον Ιανουάριο του 1950 οι Melvin Dresher και Merrill Flood [5] επινόησαν το συγκεκριμένο παίγνιο και το χρησιμοποίησαν σαν παράδειγμα στο RAND Corporation [1]. Αργότερα όταν παρουσιάστηκε αυτό το παράδειγμα σε ένα σεμινάριο στο Stanford University, ο Albert W. Tucker σκαρφίστηκε μία ιστορία πάνω στην οποία βάσισε όλη του την διάλεξη. Το παίγνιο αυτό έμεινε από τότε στην ιστορία κάνοντας τη θεωρία παιγνίων γνωστή σε όλες τις κοινωνικές επιστήμες, ενώ και πάρα πολλοί μελετητές έχουν ασχοληθεί με αυτό γράφοντας διάφορα βιβλία. Η ιστορία του Tucker [17] έχει ως εξής: δύο ύποπτοι για ένα έγκλημα συλλαμβάνονται από την αστυνομία και κρατούνται σε διαφορετικά κελιά ώστε να μην έχουν μεταξύ τους επικοινωνία. Οι αστυνομικοί είναι σίγουροι για την ενοχή τους αλλά ελλείψει αποδεικτικών στοιχείων τους προσφέρουν μια συμφωνία. Αν και οι δύο καταδώσουν ο ένας τον άλλο δηλαδή οι ύποπτοι δεν συνεργαστούν μεταξύ τους (Προδοσία), θα καταδικαστούν μόνο σε τρία χρόνια φυλάκισης. Αν μόνο ο ένας προδώσει τον άλλον θα αφεθεί ελεύθερος ενώ ο άλλος που θα παραμείνει σιωπηλός θα φυλακιστεί για πέντε χρόνια. Τέλος, αν κανένας δεν προδώσει το συγκρατούμενο του δηλαδή κανείς τους δεν συνεργαστεί με την αστυνομία και οι δύο θα περάσουν έναν χρόνο στη φυλακή. Το παραπάνω πρόβλημα μπορεί να παρουσιαστεί στον επόμενο Πίνακα:

[1] Η RAND Corporation, βρίσκεται στη Σάντα Μόνικα της Καλιφόρνια. Ιδρύθηκε μετά τον Δεύτερο Παγκόσμιο Πόλεμο για να είναι το κέντρο της εθνικής ασφάλειας και ανάλυσης.

Πίνακας 3.1: Το δίλημμα του φυλακισμένου (Χρόνια Φυλάκισης)

		Παίκτης Β	
		Συνεργασία	Προδοσία
Παίκτης Α	Συνεργασία	1 χρόνο φυλακή	5 χρόνια, ελευθερία
	Προδοσία	ελευθερία, 5 χρόνια	3 χρόνια φυλακή

Το δίλημμα αυτό παίρνει τη μορφή του παρακάτω παιγνίου, όπου τα νούμερα είναι η ωφέλεια που αποκομίζει ο παίκτης. Στον Πίνακα (3.2) φαίνεται πως λειτουργεί το παιχνίδι. Ο ένας παίκτης διαλέγει από την οριζόντια στήλη, ανάλογα με το εάν θέλει να συνεργαστεί ή όχι, κι ο άλλος διαλέγει ταυτόχρονα από την κάθετη. Οι δύο επιλογές μαζί βγάζουν τα εξής τέσσερα πιθανά αποτελέσματα του Πίνακα (3.2). Αν και οι δύο συνεργαστούν, και οι δύο τα πάνε πολύ καλά. Παίρνουν ένα Σ, δηλαδή κέρδος για αμοιβαία Συνεργασία. Στον Πίνακα (3.2) αυτή η αμοιβή είναι 3 πόντοι και θα μπορούσαν να αντιστοιχούν σε χρήματα που κερδίζει ο κάθε παίκτης ανάλογα με το αποτέλεσμα. Εάν ο ένας παίκτης συνεργαστεί (παραμείνει σιωπηλός) και ο άλλος προδώσει, ο προδότης παίρνει ένα Π, τον Πειρασμό της προδοσίας, και αυτός που περέμεινε σιωπηλός ένα Κ, την αμοιβή του Κορόιδου. Στο παράδειγμα παίρνουν 5 και 0 πόντους, αντίστοιχα. Αν τώρα προδώσουν και οι δύο, παίρνουν και οι δύο ένα Τ και 1 πόντο, δηλαδή την Τιμωρία για αμοιβαία προδοσία.

Υποθέτοντας ότι ο πρώτος παίκτης σκέφτεται πως ο αντίπαλος του θα συνεργαστεί. Αυτό σημαίνει ότι ο πρώτος παίκτης θα πετύχει ένα από τα εξής δύο αποτελέσματα της πρώτης στήλης του Πίνακα (3.2): ή θα συνεργαστεί και αυτός, παίρνοντας 3 πόντους για αμοιβαία συνεργασία η θα προδώσει παίρνοντας τους 5 πόντους του πειρασμού. Όπως φαίνεται από τα παραπάνω συμφέρει τον πρώτο παίκτη να προδώσει εάν θεωρήσει ότι ο άλλος παίκτης θα συνεργαστεί. Τώρα εάν ο πρώτος παίκτης υποθέσει ότι ο αντίπαλος του θα προδώσει, οι επιλογές του βρίσκονται στην δεύτερη στήλη του Πίνακα (3.2) πράγμα που του δίνει τη δυνατότητα ή να συνεργαστεί και να πάρει τους 0 πόντους του κορόιδου ή να προδώσει και να πάρει τον 1 πόντο της τιμωρίας για αμοιβαία προδοσία. Επομένως, συμφέρει τον πρώτο παίκτη να προδώσει εάν πιστεύει ότι ο άλλος παίκτης θα προδώσει και αυτός. Συμπερασματικά, συμφέρει τον πρώτο παίκτη να προδώσει ότι και να κάνει ο αντίπαλος του. Αλλά την ίδια λογική θα ακολουθήσει και ο δεύτερος παίκτης με αποτέλεσμα και οι δύο παίκτες να παίρνουν από ένα πόντο που είναι χειρότερος από τους 3 πόντους που θα μπορούσαν να πάρουν και οι δύο εάν συνεργάζονταν. Η ατομική λογική οδηγεί σε ένα αποτέλεσμα που είναι χειρότερο από το καλύτερο δυνατό. Εξ' αυτού και το δίλημμα.

Το δίλημμα του φυλακισμένου είναι μια αφαιρετική διατύπωση κάποιων πολύ συνηθισμένων αλλά και με ιδιαίτερο ενδιαφέρον καταστάσεων στις οποίες το ατομικό συμφέρον οδηγεί στην αμοιβαία προδοσία ενώ προτιμότερη επιλογή και για τις δύο πλευρές θα ήταν η μεταξύ τους συνεργασία. Ο ορισμός του Διλήμματος του Φυλακισμένου προϋποθέτει την ύπαρξη ορισμένων σταθερών σχέσεων ανάμεσα στα τέσσερα πιθανά αποτελέσματα. Η πρώτη σχέση καθορίζει την τάξη των τεσσάρων αμοιβών. Η καλύτερη αμοιβή που μπορεί να πετύχει ένας παίκτης είναι το Π, ο πειρασμός της προδοσίας όταν ο άλλος παίκτης επιλέγει να συνεργαστεί. Η χειρότερη είναι το Κ, η αμοιβή του κορόιδου για συνεργασία τη στιγμή που ο άλλος διαλέγει προδοσία. Θεωρώντας για τα άλλα δύο αποτελέσματα ότι το Σ, για αμοιβαία συνεργασία, είναι καλύτερο από το Τ, δηλαδή την τιμωρία για αμοιβαία προδοσία. Αυτό οδηγεί σε μία κατάταξη από το καλύτερο προς το χειρότερο, Π (T), Σ (R), Τ (P) και Κ (S), ή 5, 3, 1 και 0 αντίστοιχα, στο παράδειγμα του Πίνακα (3.2).

$$T > R > P > S \tag{3.2}$$

Το δεύτερο σκέλος του ορισμού του Διλήμματος του Φυλακισμένου είναι ότι οι παίκτες δεν μπορούν να απαλλαγούν από το δίλημμα τους εκμεταλλευόμενοι εναλλάξ ο ένας τον άλλο. Επομένως η ίση πιθανότητα εκμετάλλευσης του άλλου και εκμετάλλευσης από τον άλλο δεν είναι τόσο καλό αποτέλεσμα όσο η αμοιβαία συνεργασία, κατά συνέπεια, το κέρδος από την αμοιβαία συνεργασία θα είναι μεγαλύτερο από τον μέσο όρο που μας δίνει ο πειρασμός της προδοσίας και η μηδενική αμοιβή του κορόιδου.

$$2R > S + T \tag{3.3}$$

Αυτό το αξίωμα, μαζί με την ιεραρχική διαβάθμιση των αμοιβών, ορίζει το Δίλημμα του Φυλακισμένου.

Πίνακας 3.2: Το δίλημμα του φυλακισμένου (Πίνακας Κέρδους [2])

		Παίκτης Β	
		Συνεργασία	Προδοσία
Παίκτης Α	Συνεργασία	3,3	0,5
	Προδοσία	5,0	1,1

3.6 Επαναληπτικό Δίλημμα του Φυλακισμένου (Iterated Prisoner's Dilemma (IPD))

Δύο εγωιστές που παίζουν το παιχνίδι μία μόνο φορά (one-shot game) θα οδηγηθούν και οι δύο στην επικρατέστερη επιλογή, την προδοσία, και ο κάθε ένας τους θα πάρει λιγότερα από όσα θα μπορούσαν να πάρουν και οι δύο αν συνεργάζονταν. Όταν το παιχνίδι παίζεται σε γύρους που ο πεπερασμένος αριθμός τους είναι προκαταβολικά γνωστός, οι δύο παίκτες πάλι δεν έχουν κανένα κίνητρο για να συνεργαστούν. Ολοφάνερο είναι αυτό στην τελευταία κίνηση, στην οποία κανείς από τους δύο δεν σκέφτεται πώς θα μπορούσε να διαμορφωθεί η κατάσταση στο μέλλον. Αλλά το ίδιο ισχύει και για την προτελευταία κίνηση. Λείπει και αυτή τη φορά το κίνητρο της συνεργασίας, καθώς και οι δύο παίκτες προβλέπουν ότι στην επόμενη και τελευταία κίνηση η επιλογή και των δύο θα είναι η προδοσία. Η λογική αυτή της αμοιβαίας προδοσίας διατρέχει αναδρομικά όλο το παιχνίδι. Και εύκολα έτσι παρατηρείται να υπερισχύει από την πρώτη κίνηση μιας οποιασδήποτε αναμέτρησης που εκτυλίσσεται σε διαδοχικούς γύρους και της οποίας η συνολική διάρκεια είναι γνωστή. Η ίδια ωστόσο λογική δεν ευσταθεί όταν οι παίκτες αλληλοεπιδρούν σε έναν άγνωστο αριθμό γύρων. Και στις περισσότερες πραγματικές καταστάσεις, οι παίκτες δεν είναι ποτέ σίγουροι για το πότε θα τους δοθεί η τελευταία ευκαιρία αλληλεπίδρασης.

Το 1979 ο Robert Axelrod [1] οργάνωσε το πρώτο σημαντικό τουρνουά με επαναληπτικό δίλημμα του φυλακισμένου προσελκύοντας στρατηγικές που είχαν δημοσιευθεί μέχρι τότε στον τομέα. Η κάθε στρατηγική έπαιξε με κάθε άλλη για πάνω από 200 κινήσεις. Νικητής του τουρνουά αναδείχθηκε ο Anatol Rapoport [13] γεννημένος στην Ρωσία, μαθηματικός και ψυχολόγος. Η στρατηγική του ονομαζόταν οφθαλμός αντί οφθαλμού (Tit-For-Tat [13]), η στρατηγική ξεκινούσε πάντα με συνεργασία και μετά από αυτό μίμηση της προηγούμενης κίνησης του αντιπάλου οποιαδήποτε και αν ήταν αυτή.

Το επαναληπτικό δίλημμα του φυλακισμένου μελετήθηκε έντονα στην κοινότητα των μελετητών των εξελικτικών αλγορίθμων. Συγκεκριμένα ο ίδιος ο Axelrod [1] ανέπτυξε έναν γενετικό αλγόριθμο και αναπαριστώντας τις στρατηγικές σαν Πίνακα από bit.

3.7 Δημοφιλείς στρατηγικές (Benchmark Strategies)

Οι πιο δημοφιλείς 'γνωστές στρατηγικές' είναι οι ακόλουθες:

- **AC** (Always Cooperate): Πρόκειται για την πιο αθώα στρατηγική αφού

αυτός ο παίκτης συνεργάζεται σε κάθε κίνηση.

- **Random**: Αυτός ο παίκτης επιλέγει τυχαία την στρατηγική του σε κάθε κίνηση, αυτό τον καθιστά σαν έναν από τους πιο απρόβλεπτους παίκτες.
- **Pavlov**: Ξεκινάει με συνεργασία, μετά επαναλαμβάνει την προηγούμενη κίνηση του αν ήταν κερδοφόρα, αλλιώς την αλλάζει.
- **TFT** (Tit-For-Tat): Ξεκινάει με συνεργασία και στην συνέχεια κάνει ότι έκανε ο αντίπαλος του.
- **ETFT** (Evil-Tit-For-Tat): Ξεκινάει με προδοσία και στην συνέχεια κάνει ότι έκανε ο αντίπαλος του.

Κεφάλαιο 4

Μεθευρετικοί Αλγόριθμοι

4.1 Εισαγωγή

Η διαρκής αναζήτηση μεθόδων βελτιστοποίησης των λύσεων σε διάφορα προβλήματα της καθημερινότητας οδήγησε τους επιστήμονες να παρατηρήσουν την λειτουργία της φύσης και τα όντα που την αποτελούν, ανακαλύπτοντας αποδοτικές διαδικασίες που επιτελεί η φύση και τα όντα της για να επιτύχουν την επιβίωση τους, παραδείγματος χάριν στην αναζήτηση τροφής ή στον πολλαπλασιασμό των ειδών. Οι αλγόριθμοι που έχουν εμπνευστεί με βάση τις διαδικασίες της φύσης ονομάζονται εμπνευσμένοι από την φύση και είναι κατηγορία των μεθευρετικών αλγορίθμων. Σημαντική συνεισφορά στους μεθευρετικούς αλγορίθμους αποτέλεσε η δημοσίευση του Dorigo [3] με τον αλγόριθμο βελτιστοποίησης αποικίας μυρμηγκιών (ant colony optimization).

Οι εμπνευσμένοι απο την φύση αλγόριθμοι ουσιαστικά είναι διαδικασίες που μοντελοποιούν συμπεριφορές που λαμβάνουν χώρα στην φύση και τείνουν να έχουν βέλτιστη συμπεριφορά. Για παράδειγμα, η συμπεριφορά των σμηνών, όπως των μελισσών και των μυρμηγκιών καθώς συλλέγουν την τροφή τους. Σημαντικοί μεθευρετικοί αλγόριθμοι είναι ο γενετικός αλγόριθμος (genetic algorithm [7]), ο αλγόριθμος τεχνητής αποικίας μελισσών (artificial bee colony [8]) και ο αλγόριθμος βελτιστοποίησης σμήνους σωματιδίων (particle swarm optimization [9]). Οι μεθευρετικοί αλγόριθμοι χωρίζονται σε τρείς βασικές κατηγορίες, στούς μεθευρετικούς μίας λύσης, στους εξελικτικούς (evolutionary algorithms) και στους αλγορίθμους νοημοσύνης σμήνους (swarm intelligence-based algorithms).

Οι εξελικτικοί αλγόριθμοι (evolutionary algorithms) σαν τεχνικές βελτιστοποίησης είναι εμπνευσμένοι απο την βιολογία. Συγκεκριμένα, μιμούνται τις φυσι-

κές διαδικασίες της επιλογής (selection) ή της αναπαραγωγής (re-production), της μετάλλαξης (mutation) και της διασταύρωσης (crossover) και τους χρησιμοποιούν ως μηχανισμό για την εύρεση καλύτερων λύσεων σε προβλήματα βελτιστοποίησης. Στην κατηγορία των εξελικτικών αλγορίθμων ανήκουν οι γενετικοί αλγόριθμοι (genetic algorithms), ο αλγόριθμος διαφορικής εξέλιξης (differential evolution [15]) καθώς και άλλοι.

Οι αλγόριθμοι νοημοσύνης σμήνους (swarm intelligence algorithms) προσωμειώνουν την συμπεριφορά όντων που δρουν σαν σμήνος και συνεργάζονται μεταξύ τους για να ολοκληρώσουν διάφορες λειτουργίες τους, όπως η ανεύρεση τροφής. Σαν σμήνη στην φύση λειτουργούν τα μυρμήγκια, οι μέλισσες, τα ψάρια, τα πουλιά και είναι αποδεδειγμένο ότι πετυχαίνουν εξαιρετικά αποτελέσματα συνεργαζόμενα μεταξύ τους και πολλές φορές τα θαυμάζουμε για τα αποτελέσματα αυτά, έτσι επιλέξαμε να τα μιμηθούμε με αλγορίθμους, ώστε να επιτύχουμε βέλτιστα αποτελέσματα σε προβλήματα.

4.2 Ο Αλγόριθμος της Τεχνητής Αποικίας Μελισσών (Artificial Bee Colony (ABC))

Ο αλγόριθμος της τεχνητής αποικίας μελισσών (artificial bee colony [8] [19]) είναι ένας αλγόριθμος που έχει σχεδιαστεί με βάση τη συμπεριφορά ενός σμήνους μελισσών κατά την διάρκεια αναζήτησης τροφής. Στον αλγόριθμο υπάρχουν τρία υποσύνολα από μέλισσες, οι εξερευνήτριες (employed bees), οι οποίες είναι οι μέλισσες που πρώτες φτάνουν σε μία πηγή τροφής (πιθανή λύση του προβλήματος) από ένα προκαθορισμένο σύνολο από πιθανές πηγές τροφής και μοιράζονται αυτή την πληροφορία κάνοντας τον χορό τον μελισσών (waggle dance) με τις υπόλοιπες μέλισσες στην κυψέλη, οι θεατές μέλισσες (onlookers bees) οι οποίες είναι μέλισσες που περιμένουν στην κυψέλη και με βάση την πληροφορία που παίρνουν από τις εργάτριες αναζητούν μία καλύτερη πηγή τροφής στη γειτονιά που βρίσκεται η πηγή τροφής που τους υπέδειξαν οι εργάτριες, και τέλος, οι ανιχνεύτριες (scout bees) οι οποίες στην ουσία είναι εξερευνήτριες που στην ουσία η πηγή τροφή τους έχει εξαντληθεί και αναζητούν τυχαία στον χώρο για καινούργια πηγή τροφής.

Αναλυτικότερα ο αλγόριθμος λειτουργεί ως εξής: αρχικά ένα σύνολο από πηγές τροφής επιλέγονται τυχαία από τις εξερευνήτριες μέλισσες και η διαθέσιμη ποσότητα νέκταρ τους υπολογίζεται (τιμή της αντικειμενικής συνάρτησης). Στην συνέχεια, και αφού έχει υπολογιστεί η τιμή της αντικειμενικής συνάρτησης σε κάθε μια πηγή τροφής αντιστοιχίζεται μία εξερευνήτρια μέλισσα. Οι εξερευνήτριες μέλισσες επιστρέφουν στην κυψέλη και ενημερώνουν τις θεατές μέλισσες

σε ποια σημεία βρίσκονται οι πηγές τροφής. Στην συνέχεια, οι θεατές μέλισσες επιλέγουν την πηγή τροφής που θα επισκεφτούν βασιζόμενες στην πληροφορία που πήραν για το νέκταρ κάθε πηγής. Η πιθανότητα επιλογής μίας πηγής τροφής δίνεται από την εξίσωση:

$$P_i = \frac{f_i}{\sum_{k=1}^{K} f_k}, i = 1, 2, ..., K \tag{4.1}$$

όπου το f_i είναι η τιμή της αντικειμενικής συνάρτησης για κάθε πηγής τροφής και K το σύνολο των πηγών τροφής. Στην συνέχεια, οι εξερευνήτριες και οι θεατές μέλισσες τοποθετούνται στις επιλεγμένες πηγές. Για να παραχθεί μία καινούργια πηγή τροφής χρησιμοποιείται η ακόλουθη εξίσωση:

$$y_{ij}(t+1) = y_{ij}(t) + \phi(y_{ij}(t) - y_{kj}(t)) \tag{4.2}$$

όπου το y_{ij} είναι μια πηγή τροφής, το ϕ είναι ένας τυχαίος αριθμός στο διάστημα $(0, 1)$, το k είναι μια διαφορετική πηγή τέτοια ώστε $k \neq i$ και $k \in K$ ενώ το t είναι η τρέχουσα επανάληψη. Αν για κάποια πηγή τροφής βρεθεί από τις θεατές μέλισσες με τις κινήσεις τοπικής αναζήτησης μία καλύτερη πηγή τροφής τότε η πηγή τροφής αντικαθίσταται στην μνήμη των μελισσών από την καλύτερη της. Θα πρέπει να τονιστεί ότι αν υπάρχουν πολλές μέλισσες σε μία πηγή τροφής τότε από τις κινήσεις τοπικής αναζήτησης που πραγματοποιεί η κάθε μέλισσα, η συγκεκριμένη πηγή τροφής έχει πολύ μεγαλύτερες ικανότητες αναζήτησης σε διαφορετικά σημεία του χώρου λύσεων. Όταν αναφέρουμε ότι η πηγή τροφής έχει μεγαλύτερες ικανότητες αναζήτησης σημαίνει ότι συνάρτηση ποιότητας της συγκεκριμένης πηγής τροφής έχει καλύτερη τιμή. Στη συνέχεια, όλες οι μέλισσες επιστρέφουν ξανά στην κυψέλη και η διαδικασία ξεκινάει από την αρχή με τον χορό της μέλισσας. Εάν για έναν αριθμό επαναλήψεων η λύση δεν μπορεί να βελτιωθεί, τότε θεωρείται ότι αυτή η πηγή τροφής έχει εξαντληθεί και μία ανιχνεύτρια μέλισσα τοποθετείται σε μία καινούργια τυχαία θέση μέσα στον χώρο λύσεων. Για να μπορεί η καινούργια τυχαία λύση του προβλήματος (πηγή τροφής) να συγκλίνει γρήγορα σε μία καλή λύση η εξερευνήτρια μέλισσα τοποθετείται στην νέα αυτή θέση με βάση την παρακάτω εξίσωση:

$$y_{ij}(t+1) = y_{min,j}(t) + \phi(y_{max,j}(t) - y_{min,j}(t)) \tag{4.3}$$

όπου τα $y_{min,j}$ και $y_{max,j}$ είναι η μέγιστη και η ελάχιστη τιμή που μπορεί να πάρει μία μεταβλητή στο πεδίο τιμών και ϕ ένας τυχαίος αριθμός στο διάστημα $(0, 1)$ ενώ το t είναι η τρέχουσα επανάληψη.

Αυτό που πρέπει να τονιστεί είναι ότι αν σε κάποια πηγή τροφής εμφανιστεί ότι θα ακολουθήσουν 10 μέλισσες την εξερευνήτρια που βρήκε την πηγή τροφής τότε σημαίνει ότι θα εφαρμοστούν 10 προσπάθειες για εύρεση νέας πηγής τροφής με χρήση της Εξίσωσης (4.2). Έτσι, ένα βασικό σημείο που πρέπει να προσεχθεί στον αλγόριθμο είναι το πλήθος των πηγών τροφής να είναι μεγαλύτερο από το σύνολο των εξερευνητριών ώστε να μπορούν να τοποθετηθούν πάνω στις πηγές τροφής χωρίς να περισσέψει κάποια μέλισσα.

4.3 Ο Αλγόριθμος της Διαφορικής Εξέλιξης (Differential Evolution (DE))

Ο αλγόριθμος της διαφορικής εξέλιξης [15] είναι ένας στοχαστικός, βασιζόμενος στον πληθυσμό αλγόριθμος που προτάθηκε από τους Storn και Price [15], [19]. Πρόκειται για έναν εξελικτικό αλγόριθμο που όμως έχει ένα αριθμό από διαφορές από τους κλασσικούς γενετικούς αλγορίθμους όπως το γεγονός ότι αυτή η μέθοδος εστιάζει στην απόσταση μεταξύ των μελών του πληθυσμού και στις διαφορετικές κατευθύνσεις που μπορεί να κινηθεί κάποιο μέλος του πληθυσμού. Για να εφαρμοστεί η διαφορική εξέλιξη θα πρέπει να έχουμε κωδικοποιήσει τις λύσεις με αναπαράσταση πραγματικού αριθμού ώστε να μπορούν να εφαρμοστούν οι τελεστές μετάλλαξης που θα περιγράψουμε παρακάτω.

Αρχικά, από την στιγμή που η επιτυχία του αλγορίθμου τη διαφορικής εξέλιξης βασίζεται στις αποστάσεις μεταξύ των ατόμων της ομάδας πρέπει να υπάρχει σωστή κατανομή των ατόμων στο χώρο λύσεων του προβλήματος με σχετική απόσταση μεταξύ δύο ατόμων. Καθώς θα εξελίσσονται οι επαναλήψεις οι αποστάσεις ανάμεσα στα άτομα γίνονται μικρότερες μέχρι που στο τέλος των επαναλήψεων θεωρητικά θα πρέπει να συγκλίνουν σε μία λύση ή λύσεις γύρω από ένα βέλτιστο σημείο. Οι αποστάσεις μεταξύ των ατόμων του πληθυσμού είναι ένας πολύ καλός δείκτης της διασποράς των λύσεων του πληθυσμού. Εάν οι αποστάσεις ανάμεσα στα άτομα είναι μεγάλες τότε ο αλγόριθμος θα πραγματοποιεί μεγάλα βήματα κατά την διάρκεια των επαναλήψεων και έτσι θα μπορεί να εξερευνήσει όσο το δυνατόν μεγαλύτερα σημεία του χώρου αναζήτησης. Από την άλλη αν οι αποστάσεις ανάμεσα στα άτομα είναι μικρές τότε τα βήματα που θα πραγματοποιούνται θα είναι επίσης μικρά γεγονός που θα αυξάνει την ικανότητα αναζήτησης σε περιορισμένα σημεία του χώρου λύσεων, για παράδειγμα γύρω από ένα τοπικό ελάχιστο.

Ο τελεστής μετάλλαξης χρησιμοποιείται αρχικά για να παραχθεί ένα δοκιμαστικό διάνυσμα για κάθε μέλος του πληθυσμού με μετάλλαξη ενός διανύσματος στόχου στο οποίο προστίθεται η διαφορά ανάμεσα στις τιμές των γονιδίων δύο ή

περισσότερων ατόμων του πληθυσμού πολλαπλασιασμένων με κάποιο βάρος. Το δοκιμαστικό διάνυσμα τότε θα χρησιμοποιηθεί από τον τελεστή διασταύρωσης για να δημιουργηθεί ο απόγονος.

Για κάθε γονέα $x_i(t)$, το δοκιμαστικό διάνυσμα, το $u_i(t)$, δημιουργείται ως εξής: ένα διάνυσμα στόχου (κάποιο άλλο μέλος του πληθυσμού), $x_{i_1}(t)$, επιλέγεται από τον πληθυσμό, έτσι ώστε $i \neq i_1$. Επίσης, δύο άλλα μέλη του πληθυσμού, $x_{i_2}(t)$ και $x_{i_3}(t)$, επιλέγονται τυχαία από τον πληθυσμό τέτοια ώστε $i \neq i_1 \neq i_2 \neq i_3$. Χρησιμοποιώντας τα συγκεκριμένα διανύσματα, το δοκιμαστικό διάνυσμα υπολογίζεται με αντιμετάθεση των στοιχείων του διανύσματος στόχου ως ακολούθως:

$$u_i(t) = x_{i_1}(t) + \beta(x_{i_2}(t) - x_{i_3}(t)) \tag{4.4}$$

όπου το $\beta \in (0, \infty)$ είναι ένας παράγοντας κανονικοποίησης. Το πάνω όριο του β είναι συνήθως η τιμή 1 γιατί πειραματικά έχει αποδειχθεί ότι εάν το $\beta > 1$ δεν υπάρχει μεγάλη έως καθόλου βελτίωση των λύσεων. Γενικά όσο μικρότερο είναι το β, τόσο μικρότερο είναι και το βήμα κατά τη διάρκεια της μετάλλαξης και τόσο περισσότερο χρόνο θα πάρει στον αλγόριθμο να συγκλίνει. Από την άλλη μεριά αν το β είναι πολύ μεγάλο τότε το βήμα θα είναι μεγάλο, γεγονός που μπορεί να οδηγήσει σε υπερπήδηση κάποιου τοπικού ελάχιστου. Έτσι, η πιο συνηθισμένη τιμή που χρησιμοποιείται είναι $\beta = 0.5$.

Εκτός από τον κλασικό τελεστή μετάλλαξης (Εξίσωση 4.4), έχουν προταθεί αρκετές παραλλαγές του. Στην συγκεκριμένη διατριβή χρησιμοποιήθηκε εκτός από τον κλασικό τελεστή μετάλλαξης (Εξίσωση 4.4) και ο ακόλουθος τρόπος υπολογισμού του τελεστή μετάλλαξης όπου στην θέση του τυχαίου διανύσματος στόχου χρησιμοποιείται το καλύτερο μέλος του πληθυσμού ως διάνυσμα στόχος. Έτσι η Εξίσωση (4.4) μετατρέπεται όπως παρακάτω:

$$u_i(t) = x_{opt}(t) + \beta(x_{i_2}(t) - x_{i_3}(t)) \tag{4.5}$$

Μετά την ολοκλήρωση της φάσης της μετάλλαξης εφαρμόζεται ένας τελεστής διασταύρωσης. Στις πρώτες εφαρμογές που εμφανίστηκαν με την μέθοδο της διαφορικής εξέλιξης, η συνάρτηση ποιότητας του δοκιμαστικού διανύσματος και του γονέα συγκρίνονταν και αυτό που είχε καλύτερη τιμή επιλέγονταν για την επόμενη γενιά. Στη συνέχεια για να βελτιωθεί η αποδοτικότητα της μεθόδου χρησιμοποιείται ένας τελεστής διασταύρωσης που ονομάζεται **δυωνυμικός τελεστής διασταύρωσης** (binomial crossover) ή **ομοιόμορφος τελεστής διασταύρωσης** (uniform crossover). Σε αυτόν τον τελεστή διασταύρωσης τα γονίδια επιλέγονται τυχαία από το δοκιμαστικό διάνυσμα και από τον γονέα.

Αρχικά επιλέγεται μια παράμετρος για τον τελεστή διασταύρωσης C_r η οποία ελέγχει την αναλογία των γονιδίων που θα επιλεγούν από το δοκιμαστικό διάνυσμα. Η τιμή του C_r συγκρίνεται με έναν τυχαίο αριθμό ϕ στο διάστημα $(0, 1)$. Εάν ο τυχαίος αριθμός είναι μικρότερος ή ίσος από το C_r, η τιμή του γονιδίου του απογόνου κληρονομείται απο το δοκιμαστικό διάνυσμα, αλλιώς επιλέγεται από το γονέα:

$$x_i(t+1) = \begin{cases} u_i(t), & \text{εάν } \phi \leq C_r \\ x_i(t), & \text{αλλιώς} \end{cases} \tag{4.6}$$

Μετά τον τελεστή διασταύρωσης, η συνάρτηση καταλληλόλητας του απογόνου $x_i(t+1)$ υπολογίζεται και αν είναι καλύτερο από την συνάρτηση καταλληλόλητας του γονέα τότε επιλέγεται για την επόμενη γενιά αλλιώς ο γονέας επιβιώνει για μία ακόμα γενιά.

4.4 Ο Αλγόριθμος Βελτιστοποίησης Σμήνους Σωματιδίων (Particle Swarm Optimization (PSO))

Ο αλγόριθμος βελτιστοποίησης σμήνους σωματιδίων (Particle Swarm Optimization(PSO) [9], [19]) προτάθηκε από τους Kennedy και Eberhart [9] για να προσομοιώσει την κοινωνική συμπεριφορά κάποιων οργανισμών όπως το πέταγμα σε μορφή σμήνους των πουλιών και την ομαδική κίνηση των ψαριών. Ο αλγόριθμος βελτιστοποίησης σμήνους σωματιδίων χρησιμοποιεί τη φυσική κίνηση των ατόμων του πληθυσμού στο σμήνος και έχει ένα πολύ ευέλικτο και καλά ισορροπημένο μηχανισμό που προσαρμόζεται στις ολικές και τοπικές ικανότητες εξερεύνησης του σμήνους. Οι αλλαγές ενός σωματιδίου μέσα στο σμήνος επηρεάζονται από την εμπειρία και τις γνώσεις των γειτονικών του σωματιδίων και ως εκ τούτου ο αλγόριθμος σμήνους σωματιδίων μπορεί να θεωρηθεί ως ένας συμβατικός συνεργατικός αλγόριθμος. Η συγκεκριμένη μέθοδος εφαρμόζεται κυρίως σε προβλήματα που έχουν συνεχείς μεταβλητές ενώ τα τελευταία χρόνια έχει εφαρμοστεί και σε προβλήματα συνδυαστικής βελτιστοποίησης. Ο αλγόριθμος βελτιστοποίησης σμήνους σωματιδίων έχει μερικά πολύ ενδιαφέροντα χαρακτηριστικά. Αρχικά έχει μνήμη, πράγμα που σημαίνει ότι η γνώση από προηγούμενες καλές λύσεις κληρονομείται στις επόμενες γενιές και δεν χάνεται από επανάληψη σε επανάληψη. Επίσης, παρατηρείται πολύ μεγάλη συνεργασία ανάμεσα στα σωματίδια του σμήνους αφού τα μέλη της ομάδας συνεργάζονται μεταξύ τους στην κατασκευή των λύσεων.

Αρχικά δημιουργείται ένας αριθμός από σωματίδια (το κάθε σωματίδιο είναι

μία ενδεχόμενη λύση του προβλήματος), όπου το κάθε ένα από αυτά έχει μία συγκεκριμένη θέση στον χώρο λύσεων και κινείται με μια συγκεκριμένη ταχύτητα.

Η θέση που έχει το κάθε σωματίδιο αντιπροσωπεύει μια συγκεκριμένη λύση στο πρόβλημα και αναπαρίσταται με ένα n-διαστάσεων διάνυσμα στον χώρο των λύσεων x_{ij}, $i = 1, 2, ..., N$, $j = 1, 2, ..., n$ (N είναι το μέγεθος του πληθυσμού, n είναι ο αριθμός των διαστάσεων), ενώ η απόδοσή της εκτιμάται από μία προκαθορισμένη συνάρτηση ποιότητας (fitness function ($f(x_{ij})$). Επίσης η ταχύτητα u_{ij} αντιπροσωπεύει τις αλλαγές που θα γίνουν έτσι ώστε να κινηθεί το σωματίδιο από μία θέση σε μία άλλη. Η κατεύθυνση που θα κινηθεί το σωματίδιο υπολογίζεται από την δυναμική αλληλεπίδραση της δικής του εμπειρίας αλλά και της εμπειρίας ολόκληρου του σμήνους. Κάθε σωματίδιο έχει την δυνατότητα να ακολουθήσει τρεις διαφορετικές διαδρομές, είτε να ακολουθήσει μια δική του, είτε να κινηθεί προς την βέλτιστη θέση που είχε κατά την διάρκεια των επαναλήψεων $p_{best_{ij}}$ ή τέλος, να κινηθεί προς την θέση που έχει το βέλτιστο σωματίδιο στον πληθυσμό g_{best_j}.

Οι εξισώσεις υπολογισμού των ταχυτήτων και των θέσεων των σωματιδίων, αντίστοιχα, δίνονται παρακάτω:

$$v_{ij}(t+1) = v_{ij}(t) + c_1 * \phi_1(p_{best_{ij}} - x_{ij}(t)) + c_2 * \phi_2(g_{best_j} - x_{ij}(t)) \tag{4.7}$$

$$x_{ij}(t+1) = x_{ij}(t) + v_{ij}(t+1) \tag{4.8}$$

οπου t είναι η τρέχουσα επανάληψη, c_1 και c_2 είναι οι μεταβλητές επιτάχυνσης, ϕ_1 και ϕ_2 δύο τυχαίοι αριθμοί στο διάστημα $(0, 1)$. Στην συγκεκριμένη μεταπτυχιακή διατριβή οι μεταβλητές επιτάχυνσης c_1, c_2 είναι ίσες με την τιμή 2. Η βέλτιστη θέση $p_{best_{ij}}$ ενός σωματιδίου στο σμήνος υπολογίζεται με δυο διαφορετικές εξισώσεις ανάλογα με το εάν πρόκειται για πρόβλημα ελαχιστοποίησης ή πρόβλημα μεγιστοποίησης.

Σε περίπτωση προβλήματος ελαχιστοποίησης:

$$p_{best_{ij}}(t+1) = \begin{cases} x_{ij}(t+1), & \text{εάν } f(x_{ij}(t+1)) < f(x_{ij}(t)) \\ p_{best_{ij}}(t), & \text{αλλιώς} \end{cases} \tag{4.9}$$

Σε περίπτωση προβλήματος μεγιστοποίησης:

$$p_{best_{ij}}(t+1) = \begin{cases} x_{ij}(t+1), & \text{εάν } f(x_{ij}(t+1)) > f(x_{ij}(t)) \\ p_{best_{ij}}(t), & \text{αλλιώς} \end{cases} \tag{4.10}$$

Η βέλτιστη θέση όλου του σμήνους τη χρονική στιγμή t υπολογίζεται από την εξίσωση:

$$\begin{aligned} g_{best_j} \in (g_{best_{1j}}, g_{best_{2j}}, ..., g_{best_{Nj}} \mid f(g_{best_j})) \\ = min(f(g_{best_{1j}}), f(g_{best_{2j}}), ..., f(g_{best_{Nj}})) \end{aligned} \tag{4.11}$$

Κεφάλαιο 5

Χρήση Μεθευρετικών Αλγορίθμων για την Εξέλιξη Στρατηγικών στο Δίλημμα του Φυλακισμένου

5.1 Εισαγωγή

Σε αυτό το κεφάλαιο αναλύουμε τους τρεις αλγορίθμους που προτείνουμε για την διερεύνηση του Επαναληπτικού Διλήμματος του Φυλακισμένου. Συγκεκριμένα, ο αλγόριθμος της τεχνητής αποικίας μελισσών και ο αλγόριθμος της διαφορικής εξέλιξης δεν έχουν ξαναχρησιμοποιηθεί για την επίλυση του συγκεκριμένου προβλήματος ενώ ο αλγόριθμος βελτιστοποίησης σμήνους σωματιδίων έχει αναλυθεί και στο παρελθόν [6]. Σκοπός λοιπόν είναι σε αυτήν την διατριβή να παρατηρήσουμε την ποιότητα λύσεων σε σχέση με τον αλγόριθμο βελτιστοποίησης σμήνους σωματιδίων καθώς και τις δημοφιλείς στρατηγικές (υποκεφάλαιο 3.7) που έχουν δημοσιευθεί στο παρελθόν και προγραμματίσαμε. Στην συνέχεια παρουσιάζεται η προσαρμογή που έγινε στους τρείς αλγορίθμους για την επίλυση του Επαναληπτικού Διλήμματος του Φυλακισμένου καθώς και το πώς μοντελοποιήσαμε το συγκεκριμένο πρόβλημα. Αρχικά οι μεθευρετικοί αλγόριθμοι που αναφέραμε στο προηγούμενο κεφάλαιο χρησιμοποιούνται με μεγάλη επιτυχία σε ποικίλα προβλήματα αλλά κυρίως σε προβλήματα με συνεχείς μεταβλητές έτσι για την επίλυση του δικού μας προβλήματος έπρεπε να τους μετατρέψουμε σε δυαδικούς αλγορίθμους. Ο τρόπος που έγινε αυτό παρουσιάζεται παρακάτω.

5.2 Μοντελοποίηση Προβλήματος

Όπως αναφέραμε και στο Κεφάλαιο (3.5) ο κάθε παίκτης μπορεί να πάρει δύο αποφάσεις στο Δίλημμα του Φυλακισμένου (συνεργασία ή προδοσία) και ανάλογα με το τι θα επιλέξει και ο αντίπαλος του, θα πάρουν και την ανάλογη αμοιβή. Επειδή δεν θα είχε νόημα να μελετήσουμε το συγκεκριμένο πρόβλημα σε έναν γύρο ανταλλαγής αποφάσεων, χρησιμοποιήσαμε το Επαναληπτικό Δίλημμα του Φυλακισμένου όπου οι δύο παίκτες έρχονται αντιμέτωποι για πολλούς γύρους βγάζοντας αθροιστικά την αμοιβή που παίρνει ο καθένας.

Στην συγκεκριμένη μοντελοποίηση θεωρήσαμε πως ο κάθε παίκτης j περιγράφεται από ένα διάνυσμα (decision(i)), $i = 1, 2, ..., M$ όσες δηλαδή και οι αποφάσεις που θα ανταλλάξει με τον αντίπαλο του. Κάθε θέση σε αυτό το διάνυσμα μπορεί να πάρει μόνο δύο διακριτές τιμές 1 και 0. Το 1 αντιστοιχεί σε συνεργασία και το 0 σε προδοσία. Συνεργασία όπως αναφέραμε στο Κεφάλαιο (3.5) σημαίνει πως ο ύποπτος συνεργάστηκε με τον αντίπαλο του και έμεινε σιωπηλός ενώ αντίθετα προδοσία σημαίνει ότι ο ύποπτος πρόδωσε τον αντίπαλο του στην αστυνομία.

Διάνυσμα 5 αποφάσεων κάθε παίκτη					
Παίκτης 1	1	0	1	1	0
Παίκτης 2	1	1	1	0	0
⋮			⋱		
Παίκτης N	0	0	0	1	1

5.3 Μετατροπή Αλγορίθμων για την Βελτιστοποίηση Προβλημάτων με Διακριτές Μεταβλητές

Για την υλοποίηση του αλγορίθμου της διαφορικής εξέλιξης, του αλγόριθμου της τεχνητής αποικίας μελισσών και του αλγόριθμου βελτιστοποίησης σμήνους σωματιδίων, όπου όπως έχουμε αναφέρει και παραπάνω λειτουργούν καλά για συνεχείς μεταβλητές, έπρεπε να μετατρέψουμε το διάνυσμα αποφάσεων κάθε παίκτη από δυαδικό σε συνεχές για να μπορέσουμε να εφαρμόσουμε τις εξισώσεις (4.4), (4.1), (4.2), (4.7), (4.8) που χρησιμοποιεί κάθε αλγόριθμος αντίστοιχα. Ένας τρόπος που έχει προταθεί για την μετατροπή από διακριτές τιμές σε συνεχείς και αντίστροφα και εφαρμόζεται και στους τρεις αλγορίθμους που υλοποιήσαμε είναι με την χρήση της σιγμοειδής συνάρτησης.

$$sig(x_{ij}) = \frac{1}{1 + exp(-x_{ij})} \tag{5.1}$$

Αφού γίνει χρήση των Εξισώσεων (4.4), (4.1), (4.2), (4.7), (4.8) διακριτοποιούνται ξανά σύμφωνα με την ακόλουθη Εξίσωση:

$$x_{ij}(t+1) = \begin{cases} 1, & \text{εάν } \phi < y_{ij}(t+1) \\ 0, & \text{εάν } \phi \geq y_{ij}(t+1) \end{cases} \tag{5.2}$$

όπου $y_{ij}(t+1)$ είναι η καινούργια λύση που παίρνουμε κάνοντας χρήση των αλγορίθμων, t η τρέχουσα επανάληψη και ϕ ένας τυχαίος αριθμός στο $(0, 1)$.

5.4 Ανάλυση Αλγορίθμων

5.4.1 Ο Αλγόριθμος της Τεχνητής Αποικίας Μελισσών Παίζει το Επαναληπτικό Δίλημμα του Φυλακισμένου

Στην συγκεκριμένη υλοποίηση [14] αρχικά δημιουργήσαμε N παίκτες όσες και οι εξερευνήτριες μέλισσες, που ο κάθε ένας από αυτούς έχει ένα τυχαίο διάνυσμα M αποφάσεων (decision vector) αποτελούμενο από 0 και 1. Στην συνέχεια κάθε παίκτης παίζει ενάντια σε όλους τους άλλους, διαμορφώνοντας έτσι τον πίνακα με το κέρδος (payoff matrix) που αποκόμισε ενάντια στους υπολοίπους. Αφού λοιπόν έχουν διαμορφώσει τον πίνακα με το κέρδος τους, οι εξερευνήτριες μέλισσες (παίκτες) επιστρέφουν στην κυψέλη. Για να μεταφέρουν την πληροφορία στις θεατές μέλισσες θα πρέπει πρώτα να μετατρέψουμε τα δυαδικά διανύσματα απόφασης κάθε παίκτη σε συνεχή. Έτσι αφού κάνουμε χρήση της Εξίσωσης (5.1) χρησιμοποιούμε την Εξίσωση (4.1) για υπολογίσουμε την πιθανότητα που οι θεατές μέλισσες θα επισκεφτούν την κάθε λύση. Το ενδιαφέρον είναι ότι όσο μεγαλύτερη αμοιβή έχει ένας παίκτης (καλύτερη τιμή αντικειμενικής συνάρτησης), η πιθανότητα να τοποθετηθούν περισσότερες θεατές μέλισσες σε εκείνον τον παίκτη θα είναι μεγαλύτερη. Στην συνέχεια και αφού τοποθετηθούν οι θεατές μέλισσες δημιουργούμε καινούργια διανύσματα απόφασης για κάθε παίκτη όσες και οι θεατές μέλισσες που τοποθετήθηκαν σε αυτόν. Τέλος, επαναφέρουμε τα διανύσματα απόφασης σε δυαδικά με την Εξίσωση (5.2) έτσι ώστε να παίζουν πάλι όλοι με όλους και να δημιουργήσουμε τον καινούργιο πίνακα αμοιβών του κάθε παίκτη. Ο καλύτερος παίκτης από όλους μετά το τέλος των επαναλήψεων θα αντιμετωπίσει τις γνωστές βιβλιογραφικές στρατηγικές καθώς και άλλους αλγορίθμους που υλοποιήσαμε. Παρακάτω παρουσιάζεται ένας ψευδοκώδικας της μεθόδου.

Πίνακας 5.1: **Αλγόριθμος Τεχνητής Αποικίας Μελισσών**

1: Καθορισμός αριθμού εξερευνητριών μελισσών (παίκτες) (N)
2: Καθορισμός αριθμού θεατών μελισσών (T) και αριθμού ανιχνευτριών μελισσών (S)
3: Καθορισμός αριθμού εκτελέσεων του αλγορίθμου (W), επαναλήψεων (L) και αριθμό αποφάσεων (M)
4: ***Αρχικοποίηση***
5: Δημιουργία τυχαίων διανυσμάτων (x)
6: Αντιστοιχία κάθε παίκτη με μία εξερευνήτρια μέλισσα
7: Υπολογισμός του κέρδους κάθε παίκτη σύμφωνα με τον Πίνακα (3.2) παίζοντας όλοι με όλους.
8: ***Κύρια Φάση***
9: **while** ο μέγιστος αριθμός επαναλήψεων δεν έχει επιτευχθεί
10: Επιστροφή εξερευνητριών μελισσών στην κυψέλη
11: Μετατροπή των διανυσμάτων απόφασης σε συνεχής τιμές με την Εξίσωση (5.1)
12: Υπολογισμός πιθανότητας P_i και τοποθέτηση θεατών μελισσών με χρήση της Εξίσωσης (4.1)
13: Δημιουργία νέων διανυσμάτων απόφασης με την Εξίσωση (4.2)
14: Παίζουν όλοι με όλους
15: Συγκρίνουμε τις αμοιβές κάθε παίκτη και κρατάμε αυτόν με τις μεγαλύτερες
16: **endwhile**
17: Αποθηκεύουμε το διάνυσμα απόφασης από όλους τους παίκτες που δίνει την μεγαλύτερη αμοιβή
18: Ο καλύτερος παίκτης ανταλλάσσει M στρατηγικές ενάντια στις γνωστές στρατηγικές (υποκεφάλαιο 3.7)
19: Επιστροφή στο βήμα 4 μέχρι W εκτελέσεις ολοκληρωθούν
20: Όταν όλες οι εκτελέσεις ολοκληρωθούν, W παίκτες έχουν δημιουργηθεί με τον αλγόριθμο Τεχνητής Αποικίας Μελισσών.

5.4.2 Ο Αλγόριθμος της Διαφορικής Εξέλιξης Παίζει το Επαναληπτικό Δίλημμα του Φυλακισμένου

Στην συγκεκριμένη υλοποίηση δημιουργήσαμε N παίκτες (αρχικός πληθυσμός (γονείς)) που ο κάθε ένας από αυτούς έχει ένα τυχαίο διάνυσμα M αποφάσεων (decision vector) αποτελούμενο από 0 και 1, όπως δηλαδή και στον αλγόριθμο της τεχνητής αποικίας μελισσών, έπειτα για να μπορέσουμε να υπολογίσουμε τον πίνακα κέρδους κάθε παίκτη τους βάλαμε να παίξουν όλοι με όλους. Αφού λοιπόν έχουμε υπολογίσει το κέρδος του κάθε παίκτη- γονέα υπολογίζουμε το δοκιμαστικό του διάνυσμα εφαρμόζοντας τον τελεστή μετάλλαξης, (Εξίσωση (4.4)). Στην παρούσα μεταπτυχιακή διατριβή εκτός από την Εξίσωση (4.4) για τον υπολογισμό του δοκιμαστικού διανύσματος χρησιμοποιήσαμε και την Εξίσωση (4.5) όπου στην θέση του τυχαίου διανύσματος στόχου, χρησιμοποιείται το καλύτερο μέλος του πληθυσμού (παίκτης) ως διάνυσμα στόχος. Αφού δημιουργήσουμε τα δοκιμαστικά διανύσματα χρησιμοποιούμε τον τελεστή μετάλλαξης (Εξίσωση (4.6)) για να δημιουργηθούν οι απόγονοι (καινούργιοι παίκτες). Για να μπορέσουμε να υπολογίσουμε τους καινούριους πίνακες αμοιβών για κάθε παίκτη θα πρέπει να μετατρέψουμε τα διανύσματα απόφασης πάλι σε δυαδικές τιμές χρησιμοποιώντας την Εξίσωση (4.2). Έτσι αφού κάθε παίκτης αντιμετωπίσει όλους τους υπόλοιπους και διαμορφωθεί ο καινούργιος πίνακας αμοιβών συγκρίνουμε τους γονείς με τους απογόνους και κρατάμε αυτόν με την μεγαλύτερη αμοιβή. Παρακάτω παρουσιάζεται ένας ψευδοκώδικας της μεθόδου.

Πίνακας 5.2: **Αλγόριθμος Διαφορικής Εξέλιξης**

1: Καθορισμός αριθμού εξερευνητριών μελισσών (παίκτες) (N)
2: Καθορισμός παραμέτρων ελέγχου (β) και (C_r)
3: Καθορισμός αριθμού εκτελέσεων του αλγορίθμου (W), επαναλήψεων (L) και αριθμό αποφάσεων (M)
4: Καθορισμός του τελεστή μετάλλαξης
5: ***Αρχικοποίηση***
6: Δημιουργία τυχαίων διανυσμάτων (x)
7: Αντιστοιχία κάθε παίκτη με ένα γονέα
8: Υπολογισμός του κέρδους κάθε παίκτη σύμφωνα με τον Πίνακα (3.2) παίζοντας όλοι με όλους.
9: ***Κύρια Φάση***
10: **while** ο μέγιστος αριθμός επαναλήψεων δεν έχει επιτευχθεί
11: Μετατροπή τον διανυσμάτων απόφασης σε συνεχή τιμές με την Εξίσωση (5.1)
12: Επιλογή το διάνυσμα του γονέα $x_i(t)$
13: Δημιουργία δοκιμαστικών διανυσμάτων $u_i(t)$ απόφασης με την Εξίσωση (4.4), (4.5)
14: Δημιουργία απογόνου $x_i'(t)$ εφαρμόζοντας τον τελεστή διασταύρωσης
15: Παίζουν όλοι με όλους
16: Συγκρίνουμε τις αμοιβές κάθε παίκτη (γονέα ή απόγονο) και κρατάμε αυτόν με την μεγαλύτερη αμοιβή
17: **endwhile**
18: Αποθηκεύουμε το διάνυσμα απόφασης από όλους τους παίκτες που δίνει την μεγαλύτερη αμοιβή
19: Ο καλύτερος παίκτης ανταλλάσσει M στρατηγικές ενάντια στις γνωστές στρατηγικές (υποκεφάλαιο 3.7)
20: Επιστροφή στο βήμα 4 μέχρι W εκτελέσεις ολοκληρωθούν
21: Όταν όλες οι εκτελέσεις ολοκληρωθούν, W παίκτες έχουν δημιουργηθεί με τον αλγόριθμο Διαφορικής Εξέλιξης.

5.4.3 Ο Αλγόριθμος Βελτιστοποίησης Σμήνους Σωματιδίων Παίζει το Επαναληπτικό Δίλημμα του Φυλακισμένου

Και εδώ οι παίκτες (σμήνος) αποτελούνται από ένα διάνυσμα απόφασης (decision vector) όπου σε κάθε θέση του εμπεριέχεται η απόφαση κάθε παίκτη για την συγκεκριμένη επανάληψη. Τα διανύσματα-στρατηγικές ξεκινούν να παίζουν μεταξύ τους, όπου από τα μεταξύ τους παιχνίδια διαμορφώνεται ο πίνακας αμοιβών τους (payoff matrix). Τα payoff συγκρίνονται μεταξύ τους διαμορφώνοντας τα διανύσματα p_{best} και g_{best}. Το p_{best} περιέχει την καλύτερη στρατηγική του παίκτη, με το αντίστοιχο (payoff) σε όλη την διάρκεια των επαναλήψεων ενώ το g_{best} περιέχει το καλύτερο p_{best} όλου του πλήθους. Στην συνέχεια, στα διανύσματα decision, p_{best} και g_{best} εφαρμόζουμε την Εξίσωση (5.1) και μετατρέπουμε τις τιμές από δυαδικές σε συνεχείς. Έτσι μας δίνεται η δυνατότητα να εφαρμόσουμε τις Εξισώσεις (4.7) και (4.8). Μόλις πραγματοποιηθεί ο υπολογισμός των μεγεθών της θέσης και της ταχύτητας, επαναφέρουμε το διάνυσμα decision σε δυαδικές τιμές και επαναλαμβάνουμε τα παιχνίδια με τους υπόλοιπους παίκτες. Μόλις ολοκληρωθεί η παραπάνω διαδικασία, ο καλύτερος παίκτης (διάνυσμα g_{best}) ξεκινάει να αντιμετωπίσει αντιπάλους που ακολουθούν τις γνωστές στρατηγικές καθώς και τους αλγορίθμους που έχουμε αναλύσει παραπάνω.

Πίνακας 5.3: **Αλγόριθμος Βελτιστοποίησης Σμήνους Σωματιδίων**

1: Καθορισμός αριθμού σωματιδίων (παίκτες) (N)
2: Καθορισμός αριθμού εκτελέσεων του αλγορίθμου (W), επαναλήψεων (L) και αριθμό αποφάσεων (M)
3: ***Αρχικοποίηση***
4: Δημιουργία τυχαίων διανυσμάτων (x)
5: Αντιστοιχία κάθε παίκτη με ένα σωματίδιο
6: Υπολογισμός του κέρδους κάθε παίκτη σύμφωνα με τον Πίνακα (3.2) παίζοντας όλοι με όλους.
7: ***Κύρια Φάση***
8: **while** ο μέγιστος αριθμός επαναλήψεων δεν έχει επιτευχθεί
9: Μετατροπή τον διανυσμάτων απόφασης σε συνεχή τιμές με την Εξίσωση (5.1)
10: Σύγκριση μέσω των τύπων PSO το προσωπικό καλύτερο του κάθε παίκτη.
11: Σύγκριση με το συνολικό καλύτερο όλου του χώρου λύσεων (διάνυσμα g_{best})
12: Ανανέωση της ταχύτητας με την Εξίσωση (4.8)
13: Ανανέωση της θέσης με την Εξίσωση (4.7)
14: Μετατροπή τον διανυσμάτων απόφασης σε διακριτές τιμές με την Εξίσωση (5.2)
15: Συγκρίνουμε τις αμοιβές κάθε παίκτη και κρατάμε αυτόν με την μεγαλύτερη αμοιβή
16: **endwhile**
17: Αποθηκεύουμε το διάνυσμα απόφασης απο όλους τους παίκτες που δίνει την μεγαλύτερη αμοιβή
18: Ο καλύτερος παίκτης ανταλλάσσει M στρατηγικές ενάντια στις γνωστές στρατηγικές (υποκεφάλαιο 3.7)
19: Επιστροφή στο βήμα 4 μέχρι W εκτελέσεις ολοκληρωθούν
20: Όταν όλες οι εκτελέσεις ολοκληρωθούν, W παίκτες έχουν δημιουργηθεί με τον αλγόριθμο Βελτιστοποίησης Σμήνους Σωματιδίων.

5.5 Παραλλαγή Αλγορίθμων

Στην προσπάθεια μας να βελτιώσουμε τα αποτελέσματα μας, τροποποιήσαμε τους αλγορίθμους μας έτσι ώστε να έχουν μνήμη. Στην συγκεκριμένη προσέγγιση δημιουργήσαμε 5 διαφορετικά προγράμματα (όσες και οι γνωστές στρατηγικές πού χρησιμοποιήσαμε (υποκεφάλαιο 3.7)) έτσι ώστε το κάθε ένα από αυτά να αντιμετωπίζει ξεχωριστά κάθε γνωστή στρατηγική (υποκεφάλαιο 3.7). Αρχικά, μετατρέψαμε τα προγράμματα μας έτσι ώστε σε κάθε εκτέλεση του αλγορίθμου W να δημιουργούνται P, $(P = (1, 2..., N))$ καινούργιοι παίκτες επαναληπτικά, όπου και οι P θα αντιμετωπίσουν τις γνωστές στρατηγικές. Ο παίκτης που έχει συγκεντρώσει την μεγαλύτερη αμοιβή θεωρείται και πιο ανταγωνιστικός οπότε κρατάμε το διάνυσμα αποφάσεων του και το χρησιμοποιούμε σαν αρχική λύση για την επόμενη επανάληψη του αλγορίθμου. Με αυτόν τον τρόπο καταφέραμε τα αρχικά διανύσματα που δημιουργούμε να μην είναι εντελώς τυχαία αλλά τροποποιημένα για να αντιμετωπίζουν την εκάστοτε στρατηγική. Με την δεύτερη εκδοχή των αλγορίθμων (με μνήμη) τα αποτελέσματα μας βελτιώθηκαν αρκετά με κάποιες στρατηγικές ενώ με άλλες χειροτέρεψαν. Τα αποτελέσματα και από τις δύο παραλλαγές παρουσιάζονται αναλυτικότερα στο επόμενο κεφάλαιο.

Κεφάλαιο 6

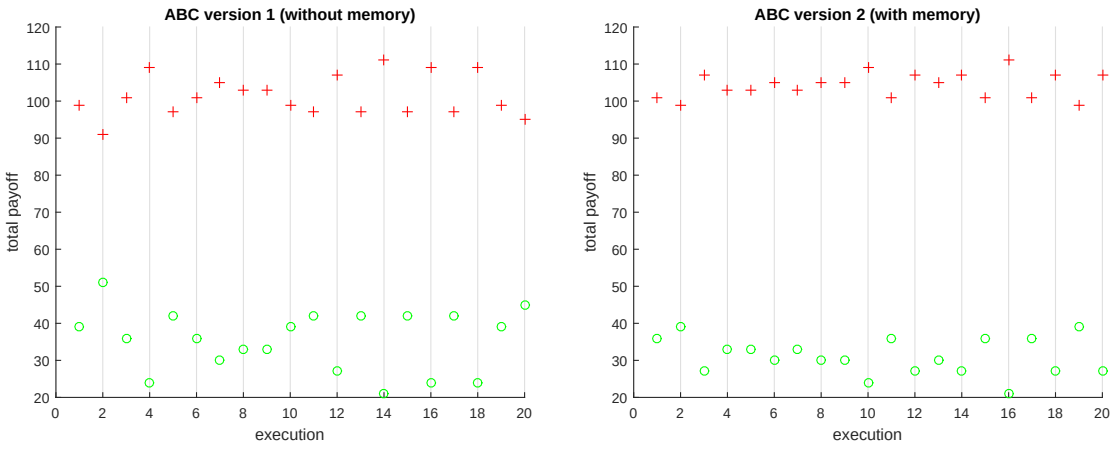
Αποτελέσματα και Συμπεράσματα

Το κεφάλαιο αυτό είναι το σημαντικότερο κεφάλαιο της εργασίας και περιέχει ερευνητικά αποτελέσματα για το Επαναληπτικό Δίλημμα του Φυλακισμένου που έχουν μεγάλο ενδιαφέρον αλλά δεν έχουν μελετηθεί στην βιβλιογραφία. Το κεφάλαιο χωρίζεται σε υποκεφάλαια όπου το κάθε ένα περιέχει τα αποτελέσματα που πήραμε με κάθε αλγόριθμο για το συγκεκριμένο πρόβλημα ενάντια στις δημοφιλείς στρατηγικές (υποκεφάλαιο 3.7), ενώ στο τέλος οι αλγόριθμοι που προτείνουμε αντιμετωπίζουν ο ένας τον άλλον και τα αποτελέσματα έχουν αρκετό ενδιαφέρον. Οι αλγόριθμοι καθώς και οι δημοφιλείς στρατηγικές που προγραμματίσαμε στα πλαίσια αυτής της μεταπτυχιακής διατριβής έγινε σε περιβάλλον Matlab.

6.1 Αποτελέσματα ABC

6.1.1 Αποτελέσματα με χαμηλό αριθμό επαναλήψεων

Αρχικά τρέξαμε τον αλγόριθμο για $N = 5$ όπου N είναι ο αριθμός των ABC παικτών, $L = 5$ όπου L ο αριθμός επαναλήψεων του αλγορίθμου, $M = 5$ όπου M είναι οι αποφάσεις που θα ανταλλάξουν μεταξύ τους οι παίκτες. Να σημειωθεί επίσης ότι το $W = 20$, όπου W είναι ο αριθμός των συνολικών εκτελέσεων του αλγορίθμου. Τα W μεταξύ τους θεωρούνται ανεξάρτητες διαδικασίες και δεν συγκρίνονται το ένα με το άλλο. Όπως έχουμε αναφέρει και παραπάνω κάθε φορά που μία εκτέλεση W του αλγορίθμου πραγματοποιείται ένας παίκτης δημιουργείται έτοιμος να αντιμετωπίσει τις γνωστές στρατηγικές (υποκεφάλαιο 3.7). Επομένως όταν $W = 20$ θα έχουν δημιουργηθεί 20 παίκτες από τον αλγόριθμο ABC που θα αντιμετωπίσουν 20 παίκτες που ακολουθούν τις γνωστές στρατηγικές. Στις γραφικές παραστάσεις που ακολουθούν, εμφανίζεται με



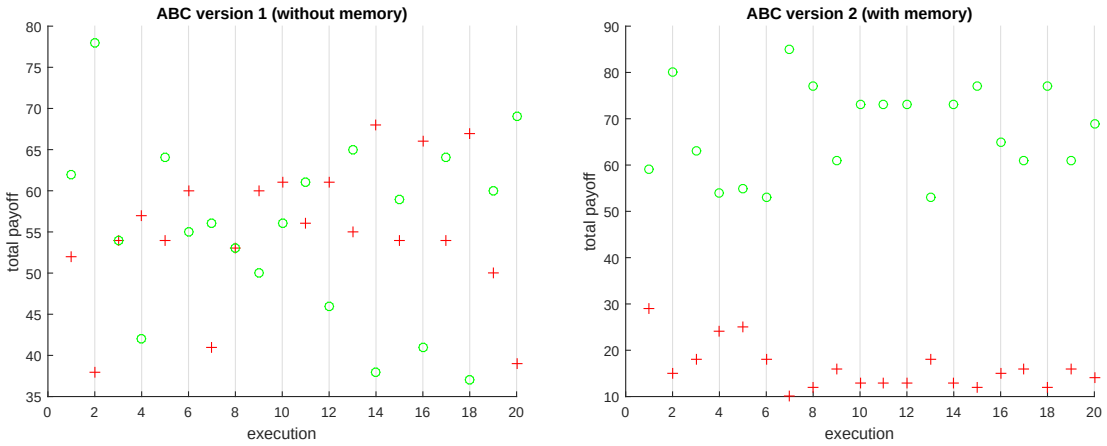
Σχήμα 6.1: ABC vs AC (ABC:red cross, AC:green circle)

κόκκινο σταυρό η επίδοση του ABC και με πράσινο κύκλο η επίδοση του εκάστοτε αντιπάλου. Στον οριζόντιο άξονα απεικονίζονται οι εκτελέσεις W (κάθε εκτέλεση και ένας παίκτης), ενώ στον κάθετο άξονα εμφανίζεται η αμοιβή του κάθε παίκτη.

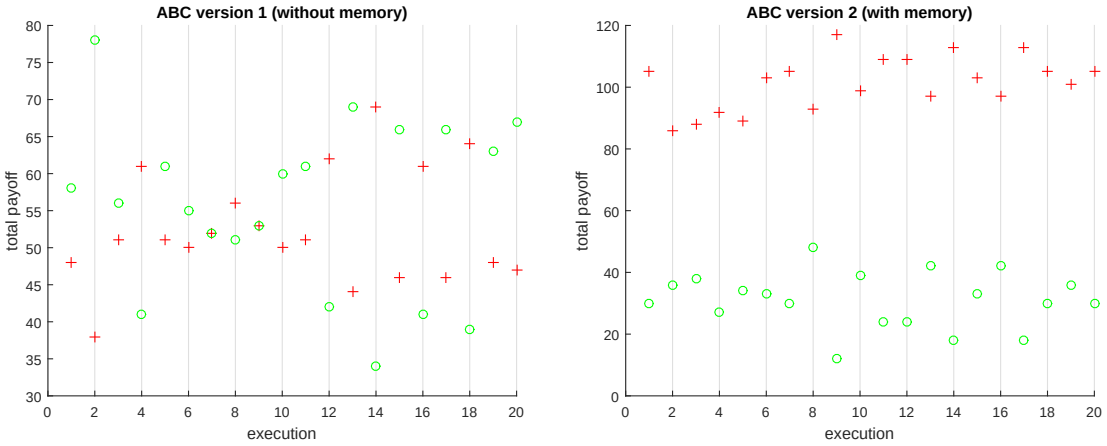
Στο Σχήμα (6.1) παρατηρούμε την συμπεριφορά των παικτών που δημιουργούνται από τις δύο εκδοχές (version 1 και version 2) του ABC ενάντια σε παίκτες που κάνουν χρήση της στρατηγικής AC (υποκεφάλαιο 3.7). Και στις δύο εκδοχές κάθε παίκτης που έχει δημιουργηθεί από τον αλγόριθμο της τεχνητής αποικίας μελισσών συγκεντρώνει υψηλότερο payoff ενάντια στον πάντα συνεργατικό αντίπαλο του.

Στο Σχήμα (6.2) παρατηρούμε τα payoff που λαμβάνει κάθε παίκτης που έχει δημιουργηθεί από τον αλγόριθμο ABC ενάντια σε τυχαίους παίκτες. Στο συγκεκριμένο σχήμα αξίζει να αναφέρουμε ότι η πρώτη έκδοση του ABC (χωρίς μνήμη) εμφανίζει καλύτερα αποτελέσματα από την δεύτερη έκδοση (με μνήμη) κάτι απολύτως λογικό αφού η μνήμη όπως αναφέραμε προηγουμένως (υποκεφάλαιο(5.5)) μας βοηθάει να κρατάμε τον καλύτερο παίκτη ενάντια στην εκάστοτε στρατηγική, αλλά όταν ο αντίπαλος είναι τυχαίος σε κάθε εκτέλεση του αλγορίθμου, η μνήμη μπορεί να λειτουργήσει εναντίον μας αφού μία καλή στρατηγική στην προηγούμενη εκτέλεση μπορεί να είναι χείριστη για την επόμενη.

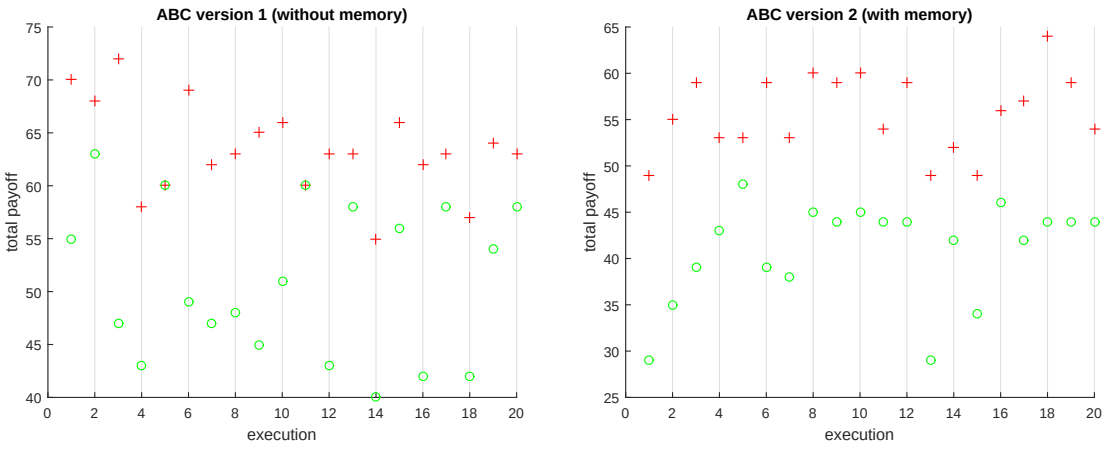
Στο Σχήμα (6.3) παρατηρούμε τα payoff του ABC ενάντια στην στρατηγική PAVLOV (υποκεφάλαιο 3.7). Ενάντια σε αυτήν την στρατηγική διαφαίνεται η εντυπωσιακή διαφορά ανάμεσα στις δύο εκδόσεις του ABC με μνήμη και



Σχήμα 6.2: ABC vs RANDOM (ABC:red cross, RANDOM:green circle)



Σχήμα 6.3: ABC vs PAVLOV (ABC:red cross, PAVLOV:green circle)



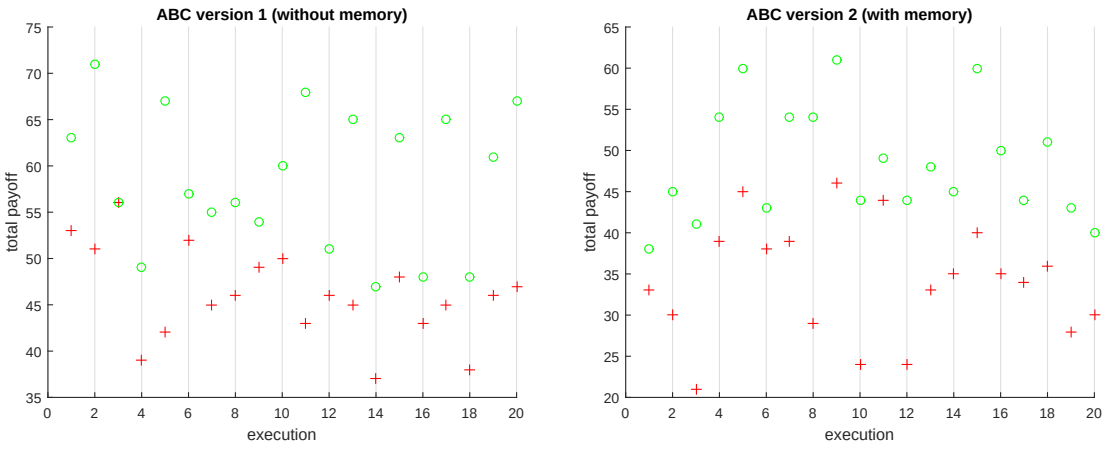
Σχήμα 6.4: ABC vs TFT (ABC:red cross, TFT:green circle)

χωρίς. Η δεύτερη έκδοση βελτιώνει εντυπωσιακά τις αμοιβές των ABC παικτών, πετυχαίνοντας μεγαλύτερο ποσοστό νίκης.

Μία από τις πιο ανταγωνιστικές στρατηγικές που υπάρχει στην βιβλιογραφία και έχει κερδίσει σε πολλούς διαγωνισμούς είναι η Tit-For-Tat (υποκεφάλαιο 3.7). Οι ABC παίκτες χωρίς μνήμη κερδίζουν την συγκεκριμένη στρατηγική με χειρότερη έκβαση την ισοπαλία. Η δεύτερη έκδοση με μνήμη παρουσιάζει ακόμα καλύτερα αποτελέσματα, κερδίζοντας απόλυτα την στρατηγική Tit-For-Tat. Όλα τα παραπάνω φαίνονται στο Σχήμα (6.4).

Στο Σχήμα (6.5) παρατηρούμε την πρώτη στρατηγική η οποία νικάει σχεδόν σε κάθε εκτέλεση τον αλγόριθμο ABC. Η καλύτερη δυνατή λύση είναι αυτή της ισοπαλίας μεταξύ των δυο παικτών που όμως παρατηρείται σπάνια. Ένας παίκτης ο οποίος ακολουθεί την στρατηγική Evil-Tit-For-Tat ξεκινάει πάντα με προδοσία και έχει πλεονέκτημα έναντι του αντιπάλου του αφού λαμβάνει payoff ανεξάρτητα από την κίνηση αυτού. Πιο συγκεκριμένα, εάν ο παίκτης που δημιουργείται από τον αλγόριθμο ABC συνεργαστεί (σιωπηλός), θα λάβει το χειρότερο δυνατό αποτέλεσμα, 0 πόντους από τον πίνακα αμοιβών. Στην περίπτωση που ο ABC παίκτης αποφασίσει να προδώσει και αυτός και οι δύο παίκτες θα λάβουν τον 1 πόντο σύμφωνα με τον Πίνακα (3.2). Ενώ και η δεύτερη έκδοση του αλγορίθμου μας αντί για να βελτιώσει τις αμοιβές του δυστυχώς τις χειροτερεύει.

Στην συνέχεια για να μπορέσουμε να συγκρίνουμε κατά πόσο ο αλγόριθμος μας (ABC) βελτιώνεται αυξάνοντας τις επαναλήψεις καθώς και τη βελτίωση ανάμεσα στις δύο εκδόσεις (με μνήμη και χωρίς) εισάγομε την ποσοστιαία



Σχήμα 6.5: ABC vs ETFT (ABC:red cross, ETFT:green circle)

διαφορά του ABC με τον εκάστοτε αντίπαλο.

$$Perc_{opponent} = \frac{payoff_{ABC} - payoff_{opponent}}{payoff_{opponent}} \tag{6.1}$$

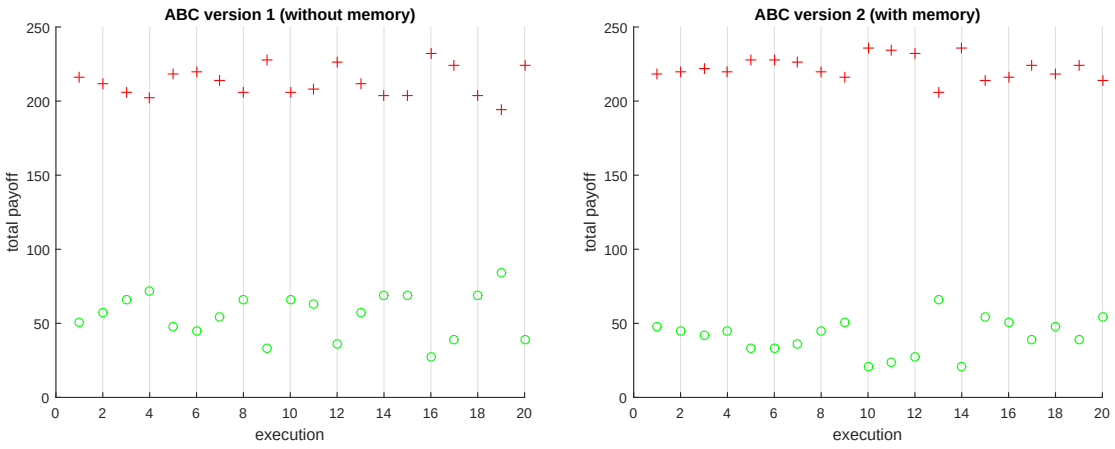
Έτσι για τις παραπάνω περιπτώσεις οι ποσοστιαίες διαφορές είναι οι ακόλουθες:

Πίνακας 6.1: Ποσοστιαίες διαφορές (%) ανάμεσα στις δύο εκδόσεις του ABC και στις δημοφιλείς στρατηγικές.

	W=20 N=5,M=5,L=5	
Strategies	ABC(Version 1)	ABC(Version 2)
AC	1,6774	2,2627
Random	-0,0647	-0,7724
Pavlov	-0,1213	1,9606
TFT	0,2255	0,3461
ETFT	-0,2215	-0,2944

6.1.2 Αποτελέσματα με μεγαλύτερο αριθμό επαναλήψεων

Αφού αυξήσαμε τον αριθμό τον παικτών σε $N = 20$, τον αριθμό τον επαναλήψεων σε $L = 60$ και τον αριθμό των αποφάσεων που ανταλλάσσουν μεταξύ



Σχήμα 6.6: ABC vs AC (ABC:red cross, AC:green circle)

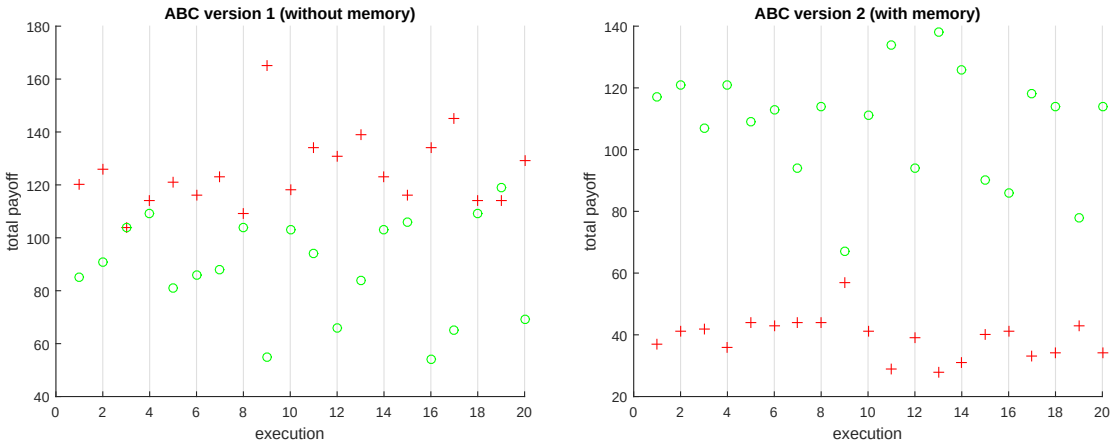
τους οι παίκτες σε $M = 10$. Τα W είναι οι ανεξάρτητες εκτελέσεις του αλγορίθμου οπότε τα κρατήσαμε σταθερά σε $W = 20$. Παρακάτω παραθέτουμε τα αντίστοιχα scatter plot για αυτές τις τιμές.

Όπως παρατηρούμε στο Σχήμα (6.6), έχει σημειωθεί αρκετή βελτίωση των επιδόσεων των ABC παικτών απέναντι στην πιο αθώα στρατηγική (AC) παρόλο που τα αποτελέσματα μας ήταν ήδη ικανοποιητικά. Μεγαλύτερο ενδιαφέρον παρουσιάζουν τα αποτελέσματα με περισσότερες επαναλήψεις απέναντι σε πιο ανταγωνιστικές στρατηγικές όπως θα δούμε παρακάτω.

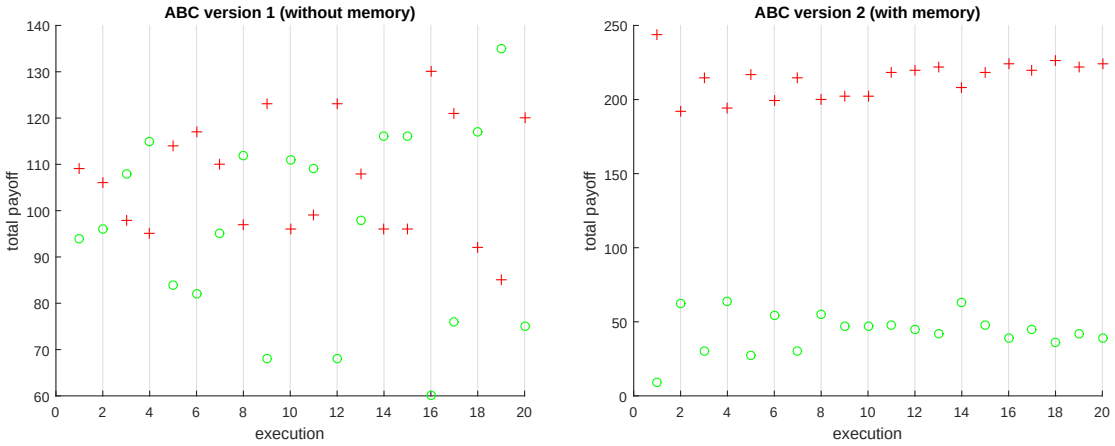
Με τυχαίο αντίπαλο διαπιστώνουμε Σχήμα (6.7) ότι αυξάνοντας τις επαναλήψεις καταφέραμε να ανατρέψουμε εξ' ολοκλήρου την αρχική επίδοση κερδίζοντας στους $18/20$ παίκτες ενώ φέρνουμε μία ισοπαλία και μια ήττα.

Με αντίπαλους παίκτες που κάνουν χρήση της στρατηγικής PAVLOV (υποκεφάλαιο 3.7) η αύξηση των επαναλήψεων προσφέρει πολύ καλύτερα αποτελέσματα στην πρώτη έκδοση του αλγορίθμου ABC αλλά ακόμα υπάρχουν παίκτες οι οποίοι δεν καταφέρνουν να συγκεντρώσουν περισσότερους πόντους από τον PAVLOV αντίπαλο τους. Στην δεύτερη έκδοση του αλγορίθμου (με μνήμη) ο αλγόριθμος ABC νικάει σε όλες τις εκτελέσεις του αλγορίθμου.

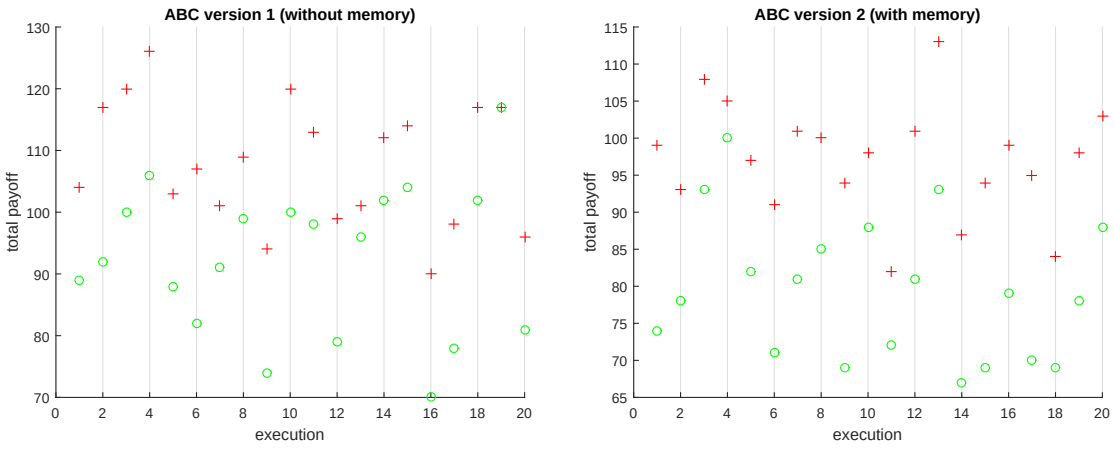
Στο Σχήμα (6.9) παρατηρούμε ότι η αύξηση των επαναλήψεων του αλγορίθμου ABC δεν προσφέρει βελτίωση ενάντια στην στρατηγική Tit-For-Tat. Αυτό συμβαίνει διότι η στρατηγική TFT βασίζεται στην ιδέα ότι ο ανταγωνιστής (αλγόριθμος ABC) επιθυμεί την μέγιστη αθροιστική αμοιβή. Συνεπώς, εάν εκείνο που μας ενδιαφέρει είναι η νίκη (μεγαλύτερη αμοιβή) και όχι η καλύτερη δυ-



Σχήμα 6.7: ABC vs RANDOM (ABC:red cross, RANDOM:green circle)



Σχήμα 6.8: ABC vs PAVLOV (ABC:red cross, PAVLOV:green circle)



Σχήμα 6.9: ABC vs TFT (ABC:red cross, TFT:green circle)

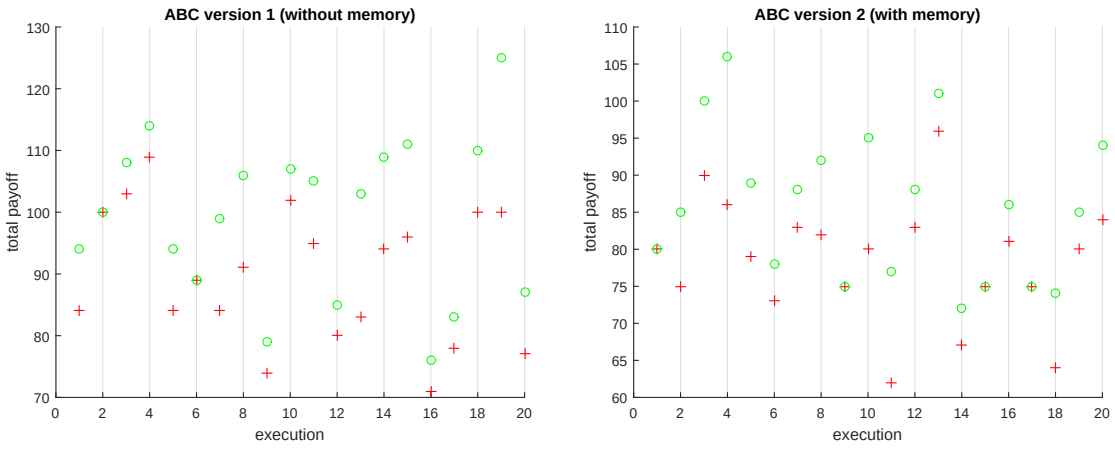
Πίνακας 6.2: Ποσοστιαίες διαφορές (%) ανάμεσα στις δύο εκδόσεις του ABC και στις δημοφιλείς στρατηγικές.

	W=20 N=20,M=10,L=60	
Strategies	ABC(Version 1)	ABC(Version 2)
AC	2,5341	3,9573
Random	0,3231	-0,6565
Pavlov	0,0334	3,4466
TFT	0,1602	0,2160
ETFT	-0,0986	-0,0874

νατή λύση και για τους δύο, ο αλγόριθμος της τεχνητής αποικίας μελισσών μπορεί να κερδίσει εύκολα ακόμα και με λίγες επαναλήψεις την στρατηγική Tit-For-Tat.

Όπως φαίνεται στο Σχήμα (6.10) η στρατηγική Evil-Tit-For-Tat καταφέρνει και κερδίζει πάλι τον αλγόριθμο μας και στις δύο εκδοχές (με μνήμη και χωρίς). Παρόλο, που αυξήσαμε τις επαναλήψεις τα αποτελέσματα βελτιώθηκαν ελάχιστα πετυχαίνοντας μερικές ισοπαλίες σε σχέση με τα αποτελέσματα με τον μικρό αριθμό επαναλήψεων ενώ ταυτόχρονα μειώσαμε την διαφορά των αμοιβών μεταξύ των παικτών.

Στην συγκεκριμένη μεταπτυχιακή διατριβή για να μπορέσουμε να παρατηρήσουμε το πως συμπεριφέρεται ο αλγόριθμος της τεχνητής αποικίας μελισσών, εάν έχει να αντιμετωπίσει έναν άλλον αλγόριθμο εμπνευσμένο από την φύση,



Σχήμα 6.10: ABC vs ETFT (ABC:red cross, ETFT:green circle)

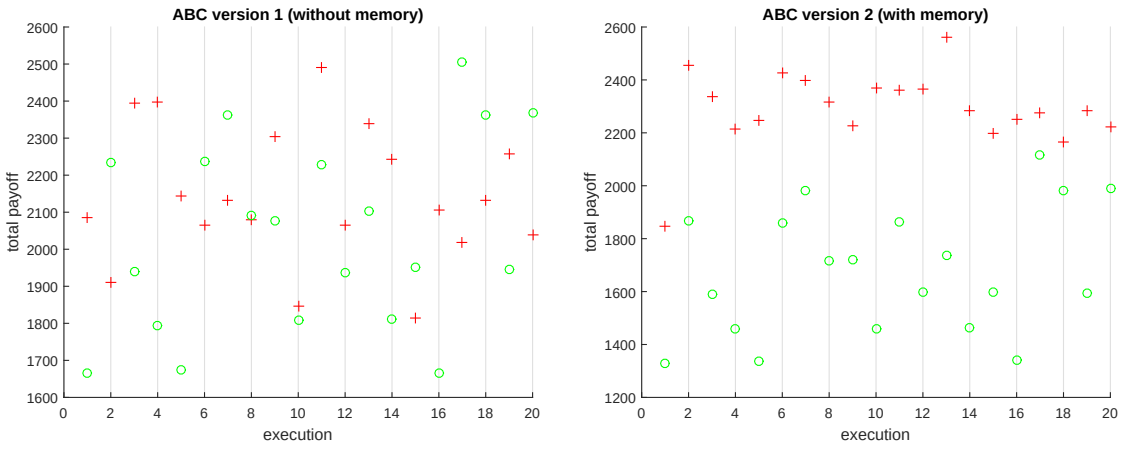
Πίνακας 6.3: Ποσοστιαίες διαφορές (%) ανάμεσα στις δύο εκδόσεις του ABC αλγόριθμου και στόν PSO αλγόριθμο.

	W=20 N=5,M=5,L=10	
Strategies	ABC(Version 1)	ABC(Version 2)
PSO	0,0620	0,3331

υλοποιήσαμε τον αλγόριθμο βελτιστοποίησης σμήνους σωματιδίων. Για να παρατηρήσουμε εάν λειτουργεί η μνήμη του αλγόριθμου μας (ABC) ενάντια σε έναν άλλον αλγόριθμο (PSO), προγραμματίσαμε τον δεύτερο σαν μία καθαρή στρατηγική. Παρακάτω, στο Σχήμα (6.11) παρουσιάζονται τα αποτελέσματα μεταξύ των δύο αλγορίθμων με την ίδια παραμετροποίηση.

Στο Σχήμα (6.11) παρατηρούμε ότι στην έκδοση του ABC χωρίς μνήμη οι δύο αλγόριθμοι ABC και PSO βρίσκονται πολύ κοντά κερδίζοντας πότε ο ένας και πότε ο άλλος. Συνολικά, έχουμε 12/20 νίκες για τον ABC αλγόριθμο και 8/20 για τον PSO. Το σκηνικό αυτό ανατρέπεται τελείως όταν συγκρίνουμε τους δύο αλγορίθμους στην έκδοση με μνήμη. Όπως παρατηρούμε στο Σχήμα (6.11) οι παίκτες που έχουν παραχθεί με τον αλγόριθμο ABC κερδίζουν απόλυτα τους παίκτες που έχουν παραχθεί από τον PSO. Αυτό μας οδηγεί στο συμπέρασμα ότι η μνήμη του αλγορίθμου ABC λειτουργεί ικανοποιητικά απέναντι στον PSO.

Στον Πίνακα (6.4) παρατηρούμε την τις ποσοστιαίες διαφορές του αλγορίθμου της τεχνητής αποικίας μελισσών και όλων των αντιπάλων που δημιουργήσαμε



Σχήμα 6.11: ABC vs PSO (ABC:red cross, PSO:green circle)

Πίνακας 6.4: Ποσοστιαίες διαφορές (%) ανάμεσα στις δύο εκδόσεις του ABC αλγόριθμου, δημοφιλείς στρατηγικές και στον αλγόριθμο PSO.

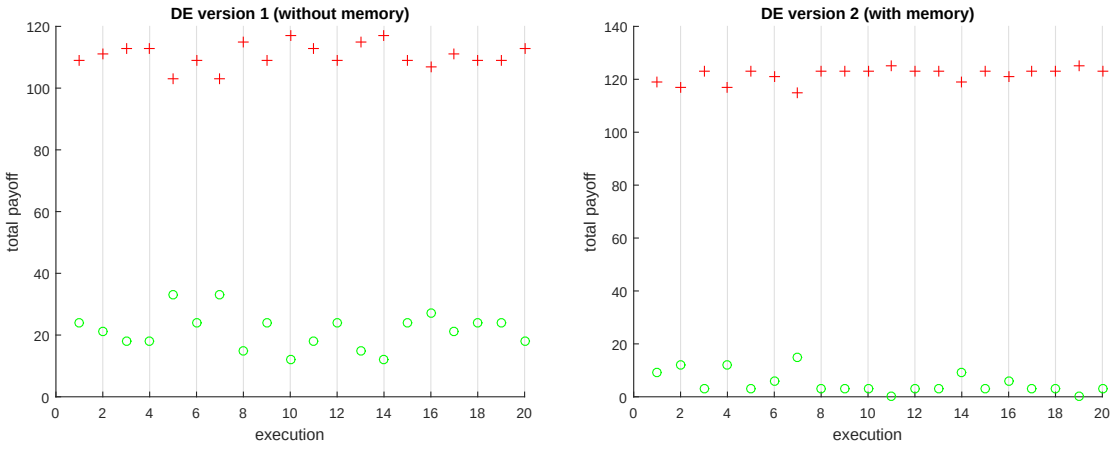
	W=20 N=5,M=5,L=10		W=20 N=20,M=5,L=50	
Strategies	ABC(Version 1)	ABC(Version 2)	ABC(Version 1)	ABC(Version 2)
AC	1,9863± 0,18	2,5634± 0,15	2,6674± 0,15	3,9222± 0,17
Random	0,1756± 0,17	-0,6832± 0,23	0,3660± 0,17	-0,6077± 0,16
Pavlov	0,1310± 0,17	2,2174± 0,12	0,1683± 0,17	2,8255± 0,18
TFT	0,2818± 0,11	0,3940± 0,17	0,4293± 0,16	0,4300± 0,13
ETFT	-0,1897± 0,16	-0,2841± 0,16	-0,1663± 0,19	-0,2201± 0,17
PSO	0,0620± 0,19	0.3331± 0,22	0,1443± 0,13	0,4886± 0,16

για δύο διαφορετικές παραμετροποιήσεις. Επίσης, στον ίδιο πίνακα βλέπουμε την ευστάθεια του αλγορίθμου και το πόσο αποκλίνει από εκτέλεση σε εκτέλεση. Για 10 διαφορετικές εκτελέσεις καταλήξαμε στο συμπέρασμα ότι ο αλγόριθμός μας προσφέρει σχεδόν πάντα τα ίδια αποτελέσματα με πολύ μικρή μεταξύ τους απόκλιση.

6.2 Αποτελέσματα DE

6.2.1 Αποτελέσματα με χαμηλό αριθμό επαναλήψεων

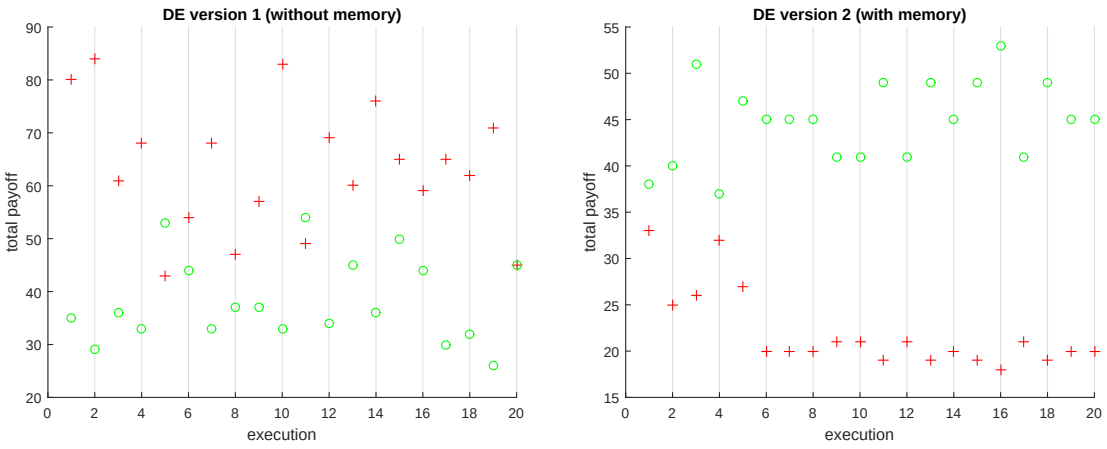
Σε αυτό το κεφάλαιο παρουσιάζονται τα αποτελέσματα του αλγορίθμου της διαφορικής εξέλιξης ενάντια στις δημοφιλείς στρατηγικές (υποκεφάλαιο 3.7) καθώς και ενάντια στον αλγόριθμο βελτιστοποίησης σμήνους σωματιδίων. Η παραμετροποίηση που χρησιμοποιήθηκε είναι η ίδια με το προηγούμενο κεφάλαιο έτσι ώστε να μπορέσουμε να συγκρίνουμε τα αποτελέσματα που μας προσφέρουν οι δύο αλγόριθμοι (ABC και DE) και να καταλήξουμε στο ποιος από τους δύο είναι καταλληλότερος για το συγκεκριμένο πρόβλημα της θεωρίας παιγνίων. Αρχικά, λοιπόν χρησιμοποιήσαμε 20 διαφορετικές εκτελέσεις του αλγορίθμου $W = 20$, $N = 5$ όπου N είναι ο αριθμός των DE παικτών, $L = 5$ όπου L ο αριθμός επαναλήψεων του αλγορίθμου και $M = 5$ όπου M είναι οι αποφάσεις που θα ανταλλάξουν μεταξύ τους οι παίκτες. Τα αποτελέσματα για την συγκεκριμένη παραμετροποίηση παρουσιάζονται παρακάτω.



Σχήμα 6.12: DE vs AC (DE:red cross, AC:green circle)

Όπως παρατηρούμε στο Σχήμα (6.12) ο αλγόριθμος DE (χωρίς μνήμη) κερδίζει σε όλες τις εκτελέσεις (διαφορετικοί παίκτες) τους πάντα συνεργατικούς αντιπάλους του. Η δεύτερη έκδοση του αλγορίθμου με μνήμη προσφέρει ακόμα καλύτερα αποτελέσματα αυξάνοντας ακόμα περισσότερο τις αμοιβές κάθε παίκτη του αλγορίθμου μας.

Με τυχαίο αντίπαλο (στρατηγική RANDOM) όπως έχουμε αναφέρει και παραπάνω στα αντίστοιχα αποτελέσματα του αλγορίθμου ABC ενάντια σε αυτήν την

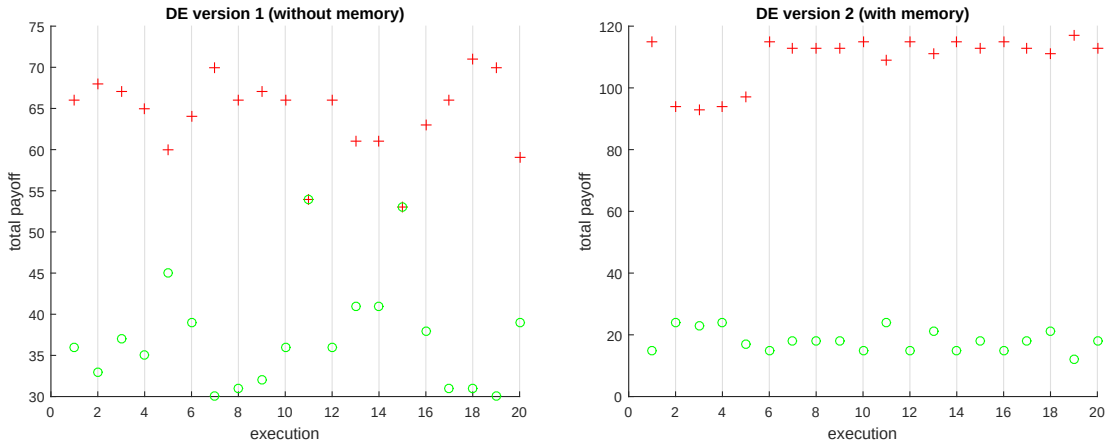


Σχήμα 6.13: DE vs RANDOM (DE:red cross, RANDOM:green circle)

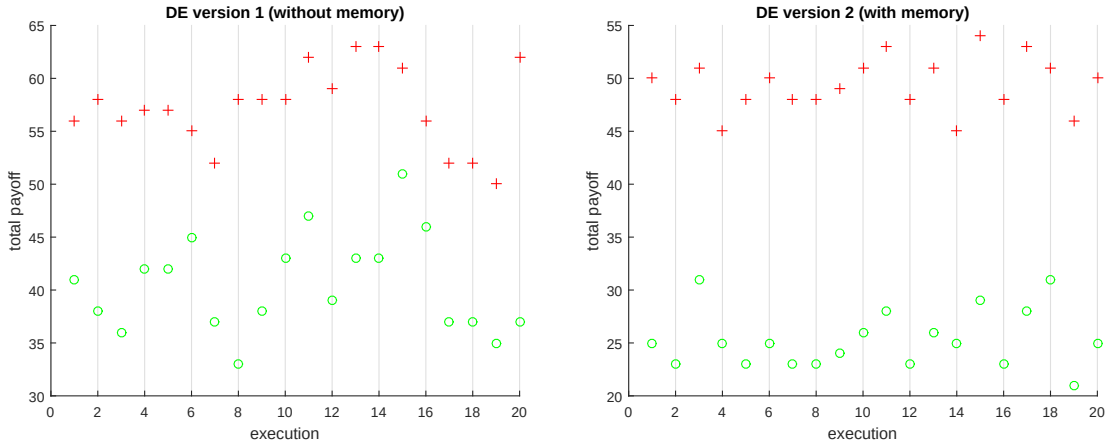
στρατηγική, ένας αλγόριθμος αντιμετωπίζει καλύτερα τυχαίους παίκτες όταν λειτουργεί χωρίς μνήμη. Έτσι και εδώ (Σχήμα 6.13) ο αλγόριθμος DE (χωρίς μνήμη) συμπεριφέρεται αρκετά ανταγωνιστικά ενάντια στους τυχαίους αντιπάλους του αλλά όταν προσαρμόσουμε στον αλγόριθμο μνήμη τα αποτελέσματα δεν είναι ικανοποιητικά.

Συνεχίζοντας, την μελέτη προσαρμοστικότητας του αλγορίθμου DE ενάντια στις βιβλιογραφικές στρατηγικές (υποκεφάλαιο 3.7), ο αλγόριθμος DE αντιμετωπίζει την στρατηγική PAVLOV, τα αποτελέσματα φαίνονται στο Σχήμα (6.14). Εδώ παρατηρούμε ότι ακόμα και στην πρώτη έκδοση του αλγορίθμου (χωρίς μνήμη) κερδίζει την στρατηγική PAVLOV καταφέρνοντας 18/20 νίκες και 2/20 ισοπαλία, ενώ με την δεύτερη έκδοση (με μνήμη) οι αμοιβές κάθε παίκτη βελτιώνονται πολύ καταφέρνοντας 20/20 νίκες. Σε σύγκριση με το πώς ο αλγόριθμος ABC αντιμετώπισε την στρατηγική PAVLOV (Σχήμα 6.3) παρατηρούμε ότι ο αλγόριθμος DE προσφέρει πολύ καλύτερα αποτελέσματα με την ίδια παραμετροποίηση. Αυτό οφείλετε στο ότι ο αλγόριθμος ABC χρειάζεται περισσότερες από $L = 5$ επαναλήψεις για να προσαρμοστεί.

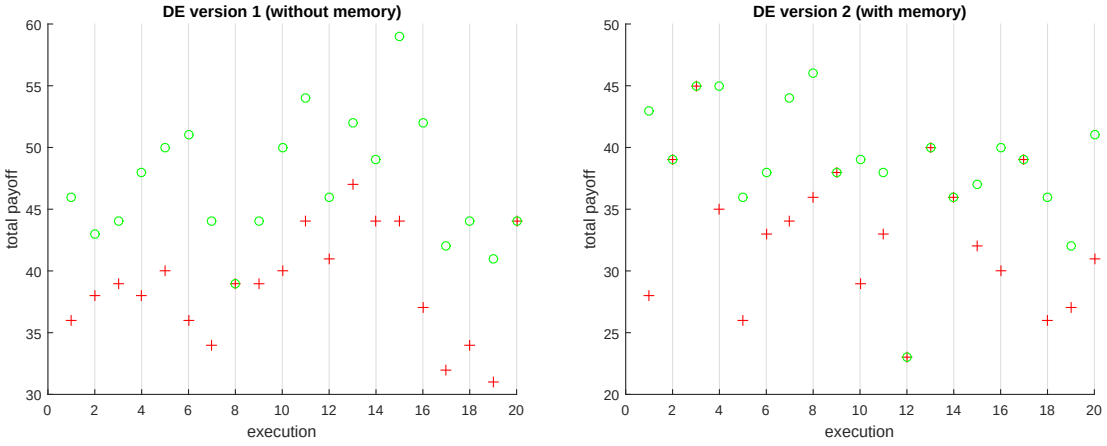
Στο Σχήμα (6.15) παρατηρούμε τα αποτελέσματα του DE με αντίπαλο την πιο γνωστή βιβλιογραφική στρατηγική (Tit-For-Tat). Ακόμα και με μικρό αριθμό επαναλήψεων τα αποτελέσματα είναι ικανοποιητικά και στις δύο εκδόσεις (με μνήμη και χωρίς μνήμη), κερδίζοντας απόλυτα σε όλες τις εκτελέσεις. Και σε αυτήν την στρατηγική ο αλγόριθμος της διαφορικής εξέλιξης προσφέρει καλύτερα αποτελέσματα από τα αντίστοιχα που προσφέρει ο αλγόριθμος τεχνητής αποικίας μελισσών (Σχήμα 6.4).



Σχήμα 6.14: DE vs PAVLOV (DE:red cross, PAVLOV:green circle)



Σχήμα 6.15: DE vs TFT (DE:red cross, TFT:green circle)



Σχήμα 6.16: DE vs ETFT (DE:red cross, ETFT:green circle)

Μια από τις δημοφιλής στρατηγικές η οποία καταφέρνει να ανταγωνίζεται πλήρως τον αλγόριθμο DE και μάλιστα στις περισσότερες εκτελέσεις να τον κερδίζει, είναι η Evil-Tit-For-Tat. Στο Σχήμα (6.16) παρατηρούμε τα σχετικά αποτελέσματα για τον μικρό αριθμό επαναλήψεων που τρέξαμε τον αλγόριθμο της διαφορικής εξέλιξης. Συγκεκριμένα βλέπουμε ότι στην έκδοση με μνήμη βελτιώνονται αρκετά τα αποτελέσματα σε σχέση με την έκδοση χωρίς μνήμη αλλά παρόλα αυτά οι $L = 5$ επαναλήψεις φαίνονται να μην είναι αρκετές για να μάθει πλήρως την εγωιστική συμπεριφορά του αντιπάλου του. Εδώ αξίζει να θυμίσουμε ότι ο παίκτης που ακολουθεί την συμπεριφορά Evil-Tit-For-Tat δρα εγωιστικά ξεκινώντας πάντα με προδοσία (υποκεφάλαιο 3.7). Καλύτερο λοιπόν αποτέλεσμα για εμάς είναι η ισοπαλία όπου εμφανίζετε 2/20 φορές στην έκδοση χωρίς μνήμη και 7/20 στην έκδοση με μνήμη. Τα αντίστοιχα αποτελέσματα του αλγορίθμου ABC είναι πολύ χειρότερα (Σχήμα 6.5).

Πίνακας 6.5: Ποσοστιαίες διαφορές (%) ανάμεσα στις δύο εκδόσεις του DE και στις δημοφιλείς στρατηγικές.

	W=20 N=5,M=5,L=5	
Strategies	DE(Version 1)	DE(Version 2)
AC	3,7747	13,4043
Random	0,5455	-0,5173
Pavlov	0,6408	4,7582
TFT	0,4013	0,9320
ETFT	-0,1786	-0,1519

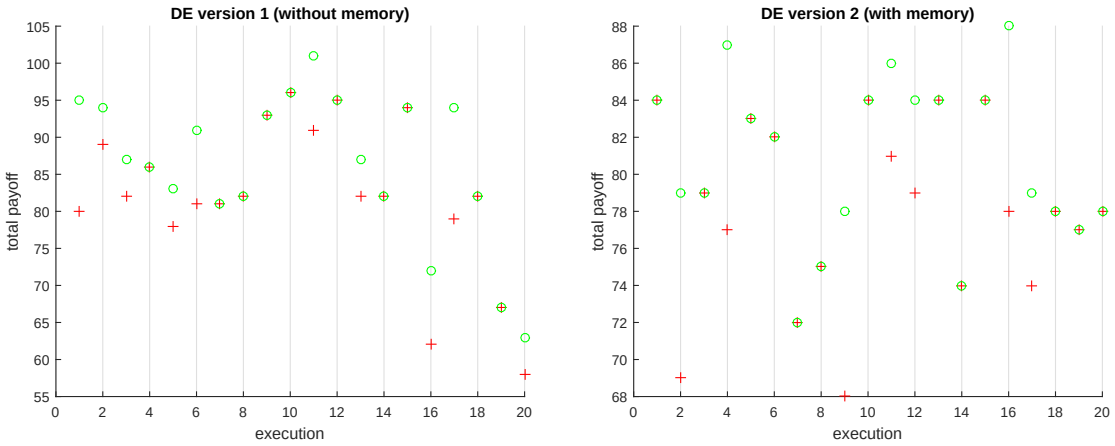
6.2.2 Αποτελέσματα με μεγαλύτερο αριθμό επαναλήψεων

Στην προσπάθεια να βελτιώσουμε τα αποτελέσματα μας καθώς και να κατανοήσουμε τις ανάγκες του αλγορίθμου της διαφορικής εξέλιξης αυξήσαμε τον αριθμό τον επαναλήψεων σε $L = 60$, τον αριθμό τον παικτών σε $N = 20$ (αρχικές λύσεις), και τον αριθμό των αποφάσεων που ανταλλάσσουν μεταξύ τους οι παίκτες σε $M = 10$. Τα W είναι οι ανεξάρτητες εκτελέσεις του αλγορίθμου οπότε τα κρατήσαμε σταθερά σε $W = 20$. Σε αυτό το υποκεφάλαιο προτιμήσαμε να μην αναλύσουμε όλα τα αποτελέσματα καθώς κάποια από αυτά δεν έχουν ιδιαίτερο ενδιαφέρον αφού ήδη ο αλγόριθμος της διαφορικής εξέλιξης προσφέρει πολύ καλά αποτελέσματα. Προτιμήσαμε λοιπόν να κάνουμε μία εκτενέστερη μελέτη του αλγόριθμου DE ενάντια στην στρατηγική Evil-Tit-For-Tat η οποία είναι ενδιαφέρουσα. Τα αποτελέσματα και η ανάλυση τους παρατίθενται παρακάτω.

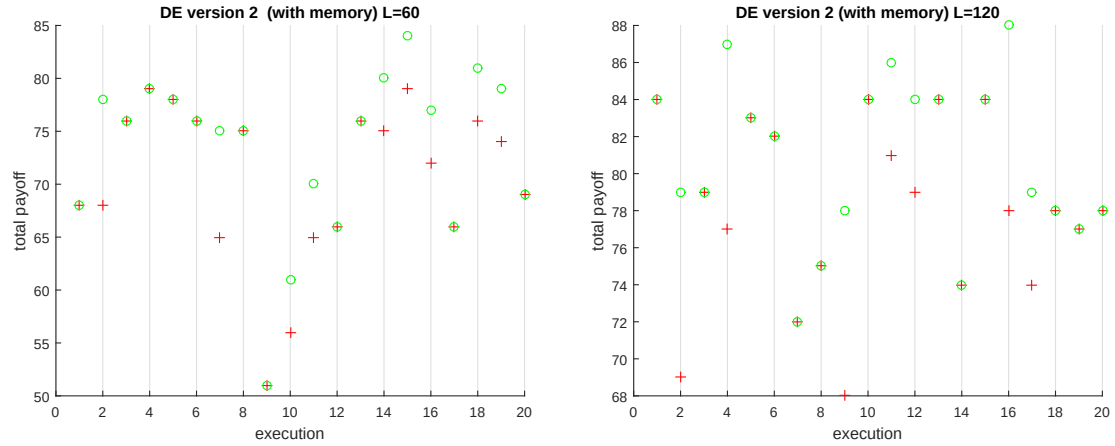
Στο Σχήμα (6.17) παρατηρούμε ότι η αύξηση των επαναλήψεων προσφέρει καλύτερα αποτελέσματα ενάντια σε αυτήν την στρατηγική πετυχαίνοντας περισσότερες ισοπαλίες.

Από εδώ και πέρα θα παρατηρήσουμε την δεύτερη έκδοση (με μνήμη) του αλγορίθμου της διαφορικής εξέλιξης ενάντια στην στρατηγική Evil-Tit-For-Tat καθώς κρατάμε σταθερές όλες τις μεταβλητές ($N = 20$, $M = 10$, $W = 20$) και αυξάνουμε μόνο τις επαναλήψεις. Για να μπορέσουμε να συγκρίνουμε τα αποτελέσματα στο τέλος θα παρουσιάσουμε σε ένα πίνακα τις ποσοστιαίες διαφορές και το πως αυτές μειώνονται με την αύξηση των επαναλήψεων.

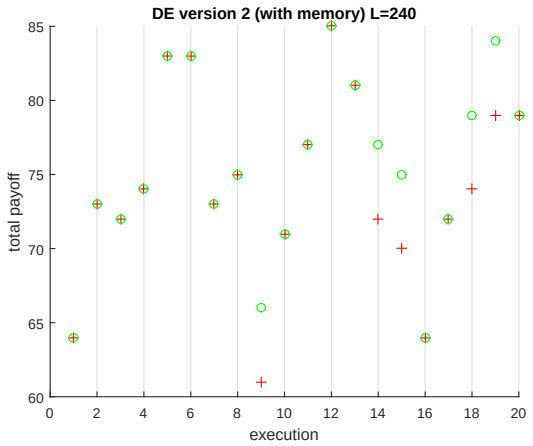
Όπως διαφαίνεται από τα Σχήματα (6.18), (6.19) καθώς και από τον Πίνακα (6.6), όσο αυξάνουμε τις επαναλήψεις βελτιώνονται τα αποτελέσματα μας ενάντια στην στρατηγική $ETFT$. Μάλιστα, μετά από πολλές επαναλήψεις ο αλγόριθμος μας μαθαίνει την συμπεριφορά του αντιπάλου του και έτσι επέρ-



Σχήμα 6.17: DE vs ETFT (DE:red cross, ETFT:green circle)



Σχήμα 6.18: DE vs ETFT (DE:red cross, ETFT:green circle)



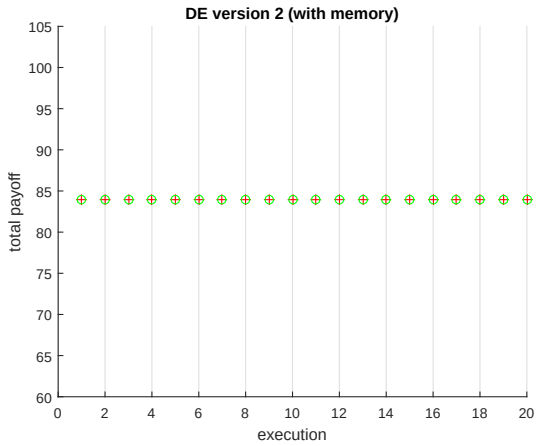
Σχήμα 6.19: DE vs ETFT (DE:red cross, ETFT:green circle)

Πίνακας 6.6: Ποσοστιαίες διαφορές (%) ανάμεσα έκδοση του DE και στην στρατηγική Evil-Tit-For-Tat.

Evil-Tit-For-Tat versus DE (Version 2)	
W=20 N=20,M=10,L=60	-0,0386
W=20 N=20,M=10,L=120	-0,0349
W=20 N=20,M=10,L=240	-0,0167
W=20 N=20,M=10,L=600	0,0

χεται η ισοπαλία ανάμεσα στους δύο παίκτες. Συγκεκριμένα στο Σχήμα (6.18) βλέπουμε ότι για $L = 60$ έχουμε 11/20 ισοπαλίες ενώ αυξάνοντας τις επαναλήψεις σε $L = 120$ πετύχαμε 13/20 ισοπαλίες, ενώ για $L = 240$ εμφανίζονται 15/20 ισοπαλίες.

Μία παραμετροποίηση που μας έδωσε πολύ καλά αποτελέσματα ενάντια στην στρατηγική Evil-Tit-For-Tat είναι η ακόλουθη: W=20, N=10, M=10, L=600. Αυξάνοντας κατά πολύ τις επαναλήψεις ο αλγόριθμος της διαφορικής εξέλιξης κατάφερε να έρθει σε ισοπαλία με όλους τους αντιπάλους του. Στο Σχήμα (6.20) παρουσιάζονται τα συγκεκριμένα αποτελέσματα.



Σχήμα 6.20: DE vs ETFT (DE:red cross, ETFT:green circle)

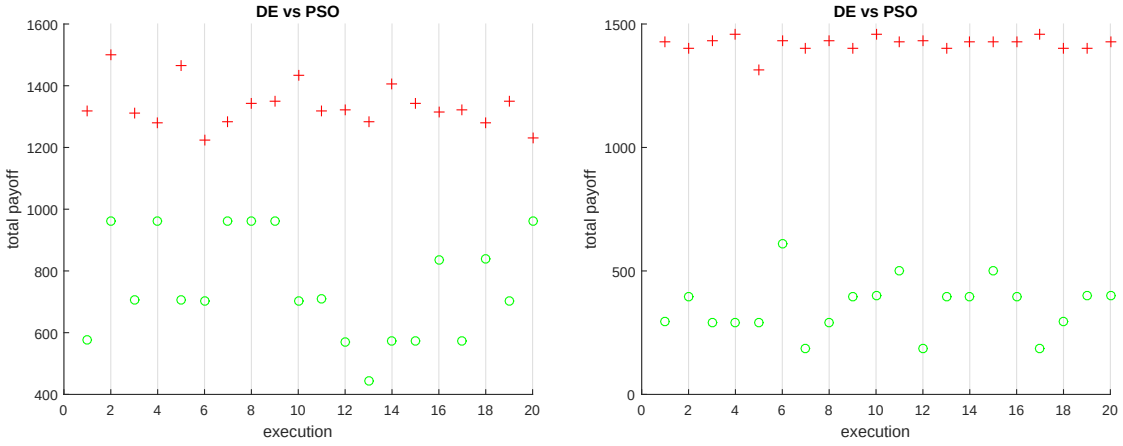
Πίνακας 6.7: Ποσοστιαίες διαφορές (%) ανάμεσα στις δύο εκδόσεις του DE αλγόριθμου και στον PSO αλγόριθμο.

	W=20 N=5,M=5,L=10	
Strategies	DE(Version 1)	DE(Version 2)
PSO	1,4720	2,3341

Στην συνέχεια, ο αλγόριθμος της διαφορικής εξέλιξης αντιμετωπίζει τον αλγόριθμο PSO όπου, όπως αναφέραμε και προηγουμένως έχει υλοποιηθεί σαν μία στρατηγική επομένως δεν του έχουμε προσαρμόσει μνήμη. Η παραμετροποίηση που χρησιμοποιήσαμε είναι η ίδια που εφαρμόστηκε και στο αντίστοιχο παράδειγμα του ABC ($L = 10$, $M = 5$, $N = 5$) (Πίνακας 6.3).

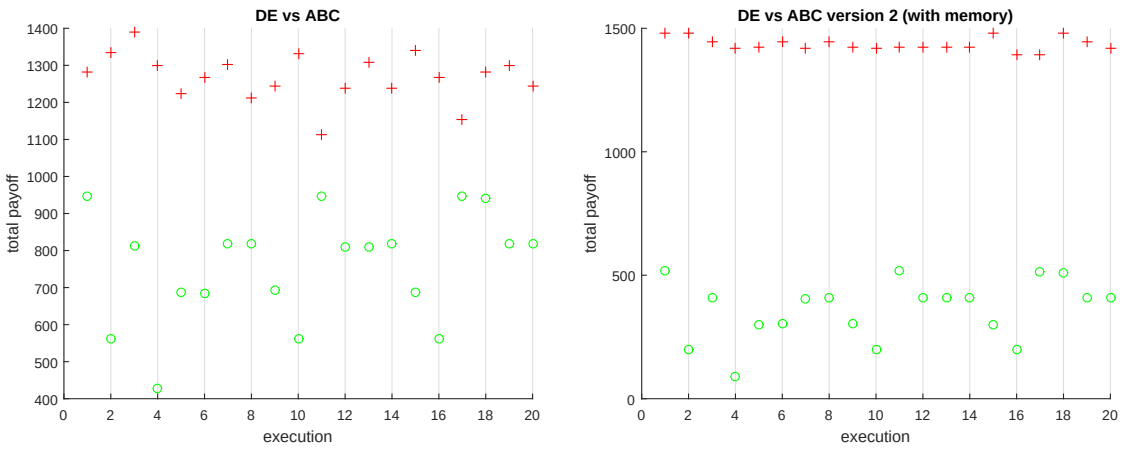
Στο Σχήμα (6.21) παρατηρούμε την υπεροχή του αλγορίθμου DE ενάντια στον αλγόριθμο PSO. Και οι δύο εκδόσεις του αλγορίθμου DE προσφέρουν πολύ καλύτερα αποτελέσματα. Οι ποσοστιαίες διαφορές για την συγκεκριμένη παραμετροποίηση φαίνονται στον Πίνακα (6.7).

Επομένως ο αλγόριθμος της διαφορικής εξέλιξης παρουσιάζει πολύ καλύτερα αποτελέσματα ενάντια στην στρατηγική PSO από ότι ο αλγόριθμος της τεχνητής αποικίας μελισσών.



Σχήμα 6.21: DE vs PSO (DE:red cross, PSO:green circle)

Τέλος, θεωρήσαμε αρκετά ενδιαφέρον να συγκρίνουμε τους δύο αλγορίθμους μεταξύ τους βάζοντας τους να παίξουν ο ένας ενάντια στον άλλον. Για την παραμετροποίηση ($W = 20$ $L = 10$, $M = 5$, $N = 5$) και για τους δύο αλγορίθμους (DE, ABC) τα αντίστοιχα σχήματα φαίνονται παρακάτω. Σε αυτό το σημείο θα πρέπει να τονιστεί ότι στο Σχήμα (6.22) ο αλγόριθμος της τεχνητής αποικίας μελισσών έχει υλοποιηθεί σαν μία στρατηγική οπότε δεν έχει προσαρμοστεί μνήμη πάνω σε αυτόν. Παρόλα αυτά από το πρώτο figure του Σχήματος (6.22) όπου και οι δύο αλγόριθμοι είναι χωρίς μνήμη παρατηρούμε ότι οι στρατηγικές του DE είναι πολύ πιο ανταγωνιστικές από αυτές του ABC.



Σχήμα 6.22: DE vs ABC (DE:red cross, ABC:green circle)

6.3 Συμπεράσματα και μελλοντική έρευνα

6.3.1 Συμπεράσματα

Τελειώνοντας τα πειράματα που παρουσιάστηκαν παραπάνω καταλήξαμε σε κάποια πολύ ενδιαφέροντα συμπεράσματα τα οποία παρουσιάζονται παρακάτω.

- Η πιο αθώα στρατηγική είναι αυτή της πάντα συνεργασίας (AC) (υποκεφάλαιο 3.7). Και οι δύο αλγόριθμοι που υλοποιήσαμε κέρδιζαν πάντα και στις δύο εκδόσεις (με μνήμη και χωρίς μνήμη).
- Ενάντια στην πιο απρόσμενη στρατηγική ($RANDOM$) και οι δύο αλγόριθμοι στην πρώτη έκδοση χωρίς μνήμη έδωσαν ανταγωνιστικές στρατηγικές ενώ όσο αυξάναμε τις επαναλήψεις βελτιώνονταν και άλλο. Αντίθετα στην έκδοση με μνήμη και οι δύο αλγόριθμοι στην προσπάθεια να μάθουν τον αντίπαλό τους κατέστρεψαν τα αποτελέσματα αφού δεν μπορούσαν να προβλέψουν την τυχαία συμπεριφορά του αντιπάλου τους στην επόμενη επανάληψη.
- Ενάντια σε παίκτες που κάνουν χρήση της στρατηγικής $PAVLOV$ και TFT και οι δύο αλγόριθμοι έδωσαν ικανοποιητικά αποτελέσματα στην πρώτη έκδοση, ενώ η μνήμη της δεύτερης έκδοσης λειτούργησε ικανοποιητικά με αποτέλεσμα και οι δυο αλγόριθμοι να προσφέρουν ακόμα καλύτερα αποτελέσματα.
- Η πιο ανταγωνιστική στρατηγική που αντιμετώπισαν οι αλγόριθμοι μας είναι η $ETFT$ όπου το καλύτερο δυνατό αποτέλεσμα που λάβαμε είναι αυτό της ισοπαλίας. Εδώ θα πρέπει να τονίσουμε ότι ο αλγόριθμος της διαφορικής εξέλιξης λόγω της μικρότερης πολυπλοκότητας του από τον αλγόριθμο τεχνητής αποικίας μελισσών μπόρεσε να πλησιάσει περισσότερο την στρατηγική $ETFT$ αφού ήταν εφικτός ο χρόνος για περισσότερες επαναλήψεις.
- Πειραματικά καταλήξαμε ότι ο αλγόριθμος ABC λειτουργούσε καλύτερα όσο αυξάναμε τον αρχικό χώρο λύσεων (παίκτες) ενώ αντίθετα ο αλγόριθμος DE λειτουργούσε καλύτερα όσο αυξάναμε τις επαναλήψεις.
- Γενικότερα, ο αλγόριθμος DE πρόσφερε καλύτερα αποτελέσματα από ότι ο ABC ενάντια σε όλες τις στρατηγικές ενώ κέρδισε και με διαφορά στο μεταξύ τους παιχνίδι (Σχήμα 6.22).

6.3.2 Μελλοντική Έρευνα

Στην μελλοντική μας έρευνα περιλαμβάνονται τα ακόλουθα:

- Εφαρμογή άλλων αλγορίθμων για την ανάπτυξη στρατηγικών για το επαναληπτικό δίλημμα του φυλακισμένου.
- Μελέτη επαναληπτικού διλήμματος του φυλακισμένου με περισσότερους από δύο παίκτες ($N - personIPD$).
- Προσπάθεια εξέλιξης της συνεργασίας, δηλαδή το αποτέλεσμα της στρατηγικής να μην είναι ως προς το όφελος του ενός αλλά ως προς το όφελος και των δύο.
- Μελέτη άλλων παίγνιων με χρήση αλγόριθμων.

Βιβλιογραφία

[1] R. Axelrod. The Evolution of Strategies in the Iterated Prisoner's Dilemma. In (L.Davis, editor) *Genetic Algorithms and Simulated Annealing*, 32-41, Pitman, London, 1987.

[2] R. Axelrod and W.D. Hamilton. The evolution of Cooperation. *Science*, 211:1390-1396, 1981.

[3] M. Dorigo, V. Maniezzo and A.Colorni. The Ant System: Optimization by a Colony of Cooperative Agents. *IEEE Transactions on Systems Mans and Cybernetics*, Part B, 26(1):1-13, 1996.

[4] L.A. Dugatkin. Animal Cooperation Among Unrelated Individuals. *Naturwissenschaften*, 89:533-541, 2002.

[5] M.M. Flood. On game-learning theory and some decision-making experiments. Technical Report *DTIC Document*, 1952.

[6] N. Franken and A.P. Engelbrecht. Particle swarm optimization approaches to coevolve strategies for the iterated prisoner's dilemma. *IEEE Transactions on Evolutionary Computation*, 9.6:562–579, 2005.

[7] Holland, J. Adaptation in natural and artificial systems. University of Michigan Press, 1975

[8] D. Karaboga and B. Basturk. On the performance of artificial bee colony (ABC) algorithm. *Applied soft computing*, 8(1):687-697, 2008.

[9] J. Kennedy and R. Eberhart. Particle Swarm Optimization. Proceedings of the *IEEE International Conference on Neural Networks*, 4:1942-1948, 1995.

[10] J.F. Nash. Equilibrium Points in n-Person Games. Proceedings of the *National Academy of Sciences*, 36:48-49, 1950.

[11] J.F. Nash. Non-Cooperative Games *Annals of Mathematics*, 54:286-295, 1951.

[12] J. von Neumann and O. Morgenstern. Theory of Games and Economic Behavior. *NJ: Princeton Univ. Press*. 1944.

[13] A. Rapoport and A.M. Chammah. Prisoner's dilemma: A study in conflict and cooperation. *University of Michigan press*, 1965.

[14] M. Rigakis, D. Trachanatzi, M. Marinaki and Y. Marinakis. Artificial Bee Colony Optimization Approach to Develop Strategies for the Iterated Prisoner's Dilemma. *Submitted paper*. 2016.

[15] R. Storn and K. Price. Differential Evolution - A simple and Efficient Heuristic for Global Optimization over Continuous Spaces. *Journal of Global Optimization*, 11(4):341-359, 1997.

[16] Y. Tekol and A. Acan. Ants Can Play Prisoner's Dilemma. *IEEE Evolutionary Computation*, 2:1348-1354, 2003.

[17] A.W. Tucker. The mathematics of Tucker: A Sampler. *The Two-Year College Mathematics Journal*, 14:228-232, 1983.

[18] Κ. Δασκαλάκης. Διπλωματική εργασία: Το πρόβλημα ύπαρξης Γνήσιας Ισορροπίας Nash σε Γραφηματικά Παίγνια. 2004.

[19] Ι. Μαρινάκης και Μ. Μαρινάκη. Σημειώσεις Μεταπτυχιακού Μαθήματος: Εξελικτικοί Αλγόριθμοι και Βελτιστοποίηση Συστημάτων Μεγάλης Κλίμακας. 2010

[20] Α. Μυγδαλάς. Θεωρία παιγνίων και προγραμματισμός Ισσοροπίας. 2007.

[21] Π. Παναγοπούλου. Διδακτορική Διατριβη: Αλγοριθμική και Εξελικτική Θεωρία Παιγνίων. 2008.