



ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ

**ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟ-
ΛΟΓΙΣΤΩΝ**

**Πολυπρακτορική Συνεργατική Επίλυση Προβλημάτων:
Μελέτη περίπτωσης σε ναυτικές επιχειρήσεις διάσωσης**

Διπλωματική Εργασία

Βλαχούλης Αριστείδης

Επιβλέπουσα: Δασκαλοπούλου Ασπασία, Επίκουρος Καθηγήτρια Π.Θ

Βόλος 2020



ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ

**ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟ-
ΛΟΓΙΣΤΩΝ**

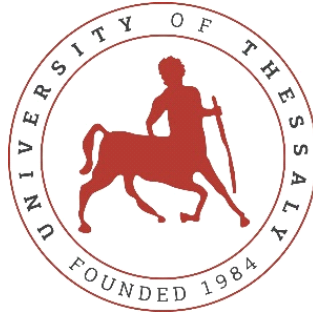
**Πολυπρακτορική Συνεργατική Επίλυση Προβλημάτων:
Μελέτη περίπτωσης σε ναυτικές επιχειρήσεις διάσωσης**

Διπλωματική Εργασία

Βλαχούλης Αριστείδης

Επιβλέπουσα: Δασκαλοπούλου Ασπασία, Επίκουρος Καθηγήτρια Π.Θ.

Βόλος 2020



UNIVERSITY OF THESSALY

SCHOOL OF ENGINEERING

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

**Multi-agent Collaborative Problem Solving: A case study
in naval search & rescue operations**

Diploma Thesis

Vlachoulis Aristeidis

Supervisor: Daskalopoulou Aspasia, Assistant Professor U.TH.

Volos 2020

ΕΥΧΑΡΙΣΤΙΕΣ

Στο σημείο αυτό θα ήθελα να ευχαριστήσω θερμά την καθηγήτρια κυρία Δασκαλοπούλου Ασπασία όπως και τους συνεπιβλέποντες καθηγητές Μπαργιώτα Δημήτριο και Τσουκαλά Ελευθέριο τόσο για τις καίριες συμβουλές τους κατά τη διάρκεια εκπόνησης της παρούσας διπλωματικής εργασίας, όσο και για την καθοδήγηση που μου παρείχαν σε όλα τα στάδια της μελέτης και της υλοποίησης. Επίσης θα ήθελα να ευχαριστήσω θερμά την οικογένειά μου για την συνεχή υποστήριξη που μου προσέφερε καθ' όλη τη διάρκεια των σπουδών μου.

ΥΠΕΥΘΥΝΗ ΔΗΛΩΣΗ ΠΕΡΙ ΑΚΑΔΗΜΑΪΚΗΣ ΔΕΟΝΤΟΛΟΓΙΑΣ ΚΑΙ ΠΝΕΥΜΑΤΙΚΩΝ ΔΙΚΑΙΩΜΑΤΩΝ

«Με πλήρη επίγνωση των συνεπειών του νόμου περί πνευματικών δικαιωμάτων, δηλώνω ρητά ότι η παρούσα διπλωματική εργασία, καθώς και τα ηλεκτρονικά αρχεία και πηγαίοι κώδικες που αναπτύχθηκαν ή τροποποιήθηκαν στα πλαίσια αυτής της εργασίας, αποτελεί αποκλειστικά προϊόν προσωπικής μου εργασίας, δεν προσβάλλει κάθε μορφής δικαιώματα διανοητικής ιδιοκτησίας, προσωπικότητας και προσωπικών δεδομένων τρίτων, δεν περιέχει έργα/εισφορές τρίτων για τα οποία απαιτείται άδεια των δημιουργών/δικαιούχων και δεν είναι προϊόν μερικής ή ολικής αντιγραφής, οι πηγές δε που χρησιμοποιήθηκαν περιορίζονται στις βιβλιογραφικές αναφορές και μόνον και πληρούν τους κανόνες της επιστημονικής παράθεσης. Τα σημεία όπου έχω χρησιμοποιήσει ιδέες, κείμενο, αρχεία ή/και πηγές άλλων συγγραφέων, αναφέρονται ευδιάκριτα στο κείμενο με την κατάλληλη παραπομπή και η σχετική αναφορά περιλαμβάνεται στο τμήμα των βιβλιογραφικών αναφορών με πλήρη περιγραφή. Αναλαμβάνω πλήρως, ατομικά και προσωπικά, όλες τις νομικές και διοικητικές συνέπειες που δύναται να προκύψουν στην περίπτωση κατά την οποία αποδειχθεί, διαχρονικά, ότι η εργασία αυτή ή τμήμα της δεν μου ανήκει διότι είναι προϊόν λογοκλοπής».

Ο Δηλών

(Υπογραφή)

Βλαχούλης Αριστείδης

Ιούνιος 2020

ΠΕΡΙΛΗΨΗ

Ζούμε σε μια εποχή όπου η τεχνητή νοημοσύνη εμφανίζει συνεχή και σταθερή άνθηση και παρατηρείται η τάση ευφυή συστήματα να χρησιμοποιούνται ολοένα και συχνότερα για την επίλυση προβλημάτων καθημερινότητας. Παρόλα αυτά πολλά ερωτήματα και αμφιβολίες εγείρονται περί του ορθού συντονισμού, επικοινωνίας και εν τέλει την λήψη απόφασης από τα συστήματα αυτά. Είναι ζωτικής σημασίας ζήτημα, τα ευφυή συστήματα αυτά να μπορούν να επικοινωνούν και να συναποφασίζουν για τις πράξεις τους (οι οποίες μεταβάλλουν το περιβάλλον τους) με τρόπο ορθό και βέλτιστο, καθώς τυχόν αστοχίες ακόμη και καθυστερήσεις μπορούν να αποβούν μοιραίες ειδικά σε καταστάσεις άμεσης δράσης. Στην παρούσα διπλωματική εργασία μελετήθηκε η πολυπρακτορική συνεργατική επίλυση προβλημάτων για την περίπτωση ναυτικών επιχειρήσεων διάσωσης, που αφορά μια ομάδα ευφυών πρακτόρων οι οποίοι συνεργάζονται με σκοπό τη βέλτιστη κάλυψη θαλάσσιων ζωνών-περιοχών καθώς και την άμεση ανταπόκριση σε περιπτώσεις άμεσης δράσης (διάσωση). Τέλος δημιουργήθηκε ένα πρόγραμμα προσομοίωσης για ενδεικτική απεικόνιση του περιβάλλοντος και των πρακτόρων.

ABSTRACT

We live in an era in which artificial intelligence is constantly blooming and that's why we tend to see more and more the use of intelligent systems as the solution to everyday problems. Nevertheless, there are major issues for these systems such as: right coordination, optimal communication and finally the decision-making process. It is a matter of vital importance for these intelligent systems to be capable to communicate and to co-decide before they act, as by any chance misfires could be proven catastrophic or even fatal especially in cases of emergency. In this diploma thesis we studied the multi-agent collaborative problem solving in the case of naval search and rescue operations for a group of intelligent agents whose purpose is to collaborate in order to optimally react in cases of emergency. Finally, we created a simulation program for indicative display of the environment and the agents.

ΠΡΟΛΟΓΟΣ

Η παρούσα διπλωματική εργασία εκπονήθηκε ως το τελευταίο βήμα για την απόκτηση διπλώματος και την ολοκλήρωση των προπτυχιακών μου σπουδών στο τμήμα Ηλεκτρολόγων Μηχανικών & Μηχανικών Υπολογιστών του Πανεπιστημίου Θεσσαλίας, στην πόλη του Βόλου υπό την επίβλεψη της Επικούρου Καθηγήτριας Δασκαλοπούλου Ασπασίας.

ΠΕΡΙΕΧΟΜΕΝΑ

ΕΥΧΑΡΙΣΤΙΕΣ.....	IV
ΠΕΡΙΛΗΨΗ	VI
ABSTRACT	VII
ΠΡΟΛΟΓΟΣ	VIII
ΠΕΡΙΕΧΟΜΕΝΑ	X
ΚΑΤΑΛΟΓΟΣ ΣΧΗΜΑΤΩΝ	XIII
ΚΕΦΑΛΑΙΟ 1.....	1
Εισαγωγή.....	1
1.1 Αντικείμενο της διπλωματικής	1
1.1.1 Συνεισφορά.....	2
1.2 Οργάνωση του τόμου	2
ΚΕΦΑΛΑΙΟ 2.....	5
Θεωρητικό Υπόβαθρο	5
2.1 Έννοιες από το πεδίο της Τεχνητής Νοημοσύνης.....	5
2.1.1 Πράκτορας - Agent.....	5
2.1.2 Περιβάλλον - Environment	11
2.2 Βασικές έννοιες κατανεμημένων συστημάτων.....	14
2.2.1 Βασικές λειτουργίες των κατανεμημένων συστημάτων.....	14
2.3 Βασικές έννοιες από το πεδίο των δικτύων.....	15
2.3.1 Clustering method	18
2.3.2 Clustering και κατανεμημένα συστήματα	19
ΚΕΦΑΛΑΙΟ 3.....	21
Περιβάλλον διεκπεραίωσης των ναυτικών επιχειρήσεων	21
3.1 Η φύση του περιβάλλοντος.....	21
3.2 Προσβασιμότητα	21

3.3 Γεωγραφική Έκταση.....	22
3.4 Καιρικές Συνθήκες.....	22
ΚΕΦΑΛΑΙΟ 4.....	25
Σχεδιασμός πράκτορα για τις ναυτικές επιχειρήσεις διάσωσης	25
4.1 Η συνεισφορά της Τεχνητής Νοημοσύνης	25
4.2 Χαρακτηριστικά Πράκτορα	26
4.2.1 Περιγραφή πράκτορα διάσωσης.....	26
4.2.2 Αντίληψη του χώρου και των πράξεων των πρακτόρων (Perception of Region & Action in Agents)	28
4.2.3 Ρεπερτόριο ενεργειών πράκτορα διάσωσης	29
4.3 Αισθητήρες - Sensors.....	30
ΚΕΦΑΛΑΙΟ 5.....	33
Προτεινόμενη υλοποίηση και λειτουργία του συστήματος.....	33
5.1 Περιγραφή περιβάλλοντος-κόσμου του συστήματος.....	33
5.1.1 Χαρακτηριστικά του Συστήματος	33
5.1.2 Παράμετροι του συστήματος.....	34
5.1.3 Αντικείμενα και οντότητες του συστήματος	35
5.2 Κινησιολογία Πρακτόρων.....	36
5.2.1 Ρεπερτόριο Ενεργειών Πράκτορα	37
5.3 Εξυπηρέτηση των προβλημάτων - Λήψη αποφάσεων.....	41
5.3.1 Υλοποίηση Clustering	42
5.3.2 Distributed Clustering Algorithm (DCA).....	42
5.3.3 Επικοινωνία των ClusterHead με τα μέλη του Cluster και εξυπηρέτηση των προβλημάτων.....	45
5.3.4 Περιπτώσεις ναυαγίων	49
5.4 Πρόγραμμα Πράκτορα	49
5.4.1 Βάση γνώσης πρακτόρων	51
ΚΕΦΑΛΑΙΟ 6.....	53
Πλατφόρμα υλοποίησης του συστήματος	53

6.1 Μοντέλα συστημάτων βασισμένα σε πράκτορες.....	54
6.2 Γλώσσα προγραμματισμού και εργαλεία υλοποίησης πλατφόρμας.....	55
6.3 Οδηγίες πλατφόρμας προσομοίωσης.....	56
6.4 Αποτελέσματα υλοποίησης.....	67
ΚΕΦΑΛΑΙΟ 7.....	69
Επίλογος.....	69
7.1 Σύνοψη και συμπεράσματα	69
7.2 Μελλοντικές επεκτάσεις.....	70
ΒΙΒΛΙΟΓΡΑΦΙΑ	71
ΣΥΝΤΟΜΟΓΡΑΦΙΕΣ	73
ΟΡΟΛΟΓΙΑ - ΓΛΩΣΣΑΡΙ	75

ΚΑΤΑΛΟΓΟΣ ΣΧΗΜΑΤΩΝ

Σχήμα 2.1 Σχηματική απεικόνιση πράκτορα [1]	6
Σχήμα 2.2 Σχηματικό διάγραμμα απλού αντανakλαστικού πράκτορα [1]	7
Σχήμα 2.3 Σχηματικό διάγραμμα πράκτορα με μοντέλο [1]	8
Σχήμα 2.4 Σχηματικό διάγραμμα πράκτορα βασισμένου σε στόχο [1]	9
Σχήμα 2.5 Σχηματική απεικόνιση πράκτορα βασισμένου στη χρησιμότητα [1]	10
Σχήμα 2.6 Σχηματική απεικόνιση πράκτορα που μαθαίνει [1]	11
Σχήμα 2.7 Σχηματική απεικόνιση των peer-to-peer δικτύων [5]	17
Σχήμα 2.8 Σχηματική απεικόνιση των Server-Based δικτύων [5]	18
Σχήμα 4.1 Εικόνα ενός αυτόνομου πλοίου με χρήση ηλιακής ενέργειας για κινητήρια δύναμη [9]	27
Σχήμα 4.2 Σχηματική απεικόνιση των ερωτημάτων που καλείται να απαντήσει ένας πράκτορας καθώς λάβει τα εξωτερικά ερεθίσματα [11]	29
Σχήμα 4.3 Σχηματική απεικόνιση λειτουργίας SO.N.A.R [12]	31
Σχήμα 5.1 Ενδεικτικό-τυχαίο στιγμιότυπο του περιβάλλοντος υλοποίησης	36
Σχήμα 5.2 Σχηματική απεικόνιση των επιτρεπτών κινήσεων ενός πράκτορα καθώς και σκιαγράφηση του πεδίου το οποίο μπορεί να ανιχνεύσει μέσω αισθητήρων	37
Σχήμα 5.3 Σχηματική απεικόνιση των βέλτιστων διαδρομών που μπορεί να κάνει ένας πράκτορας για την μετάβαση του στον επιθυμητό κόμβο(X,Y) χωρίς ύπαρξη εμποδίων	39
Σχήμα 5.4 Σχηματική απεικόνιση των βέλτιστων διαδρομών που θα λάμβανε ο πράκτορας στην περίπτωση άνευ εμποδίων	40
Σχήμα 5.5 Σχηματική υλοποίηση του DCA για τη δημιουργία cluster σε ένα σύστημα 8 κόμβων.....	45

Σχήμα 5.6 Σχηματική απεικόνιση ενός cluster το οποίο έχει δημιουργηθεί με τον DCA αλγόριθμο και των συνδέσεων δικτύου επικοινωνίας μεταξύ των κόμβων του	47
Σχήμα 5.7 Σχηματική απεικόνιση του περιβάλλοντος του cluster μετά την επίλυση του προβλήματος.....	48
Σχήμα 5.8 Ο πράκτορας υλοποιημένος ως συνάρτηση	50
Σχήμα 6.1 Στιγμιότυπο προσομοίωσης του συστήματος	53
Σχήμα 6.2 Απεικόνιση της εισόδου για την προσομοίωση ενός στιγμιότυπου του συστήματος	54
Σχήμα 6.3 Παράθυρο διεπαφής εισόδου του χρήστη στην πλατφόρμα	56
Σχήμα 6.4 Στιγμιότυπο του περιβάλλοντος μετά τη μετακίνηση του πράκτορα στο ναυάγιο ..	58
Σχήμα 6.5 Αρχική παρουσίαση θέσης πράκτορα μέσω τερματικού	59
Σχήμα 6.6 Σχηματική απεικόνιση του πράκτορα στο σημείο που κατονομάζει το τερματικό .	59
Σχήμα 6.7 Παρουσίαση θέσης πράκτορα μετά από 1 μετακίνηση μέσω τερματικού.....	60
Σχήμα 6.8 Σχηματική απεικόνιση πράκτορα μετά την 1η μετακίνηση.....	60
Σχήμα 6.9 Παρουσίαση θέσης πράκτορα μετά από 2 μετακινήσεις μέσω τερματικού	61
Σχήμα 6.10 Σχηματική απεικόνιση πράκτορα μετά τη 2η μετακίνηση.....	61
Σχήμα 6.11 Παρουσίαση θέσης πράκτορα μετά από 3 μετακινήσεις μέσω τερματικού	62
Σχήμα 6.12 Σχηματική απεικόνιση πράκτορα μετά την 3η μετακίνηση.....	62
Σχήμα 6.13 Παρουσίαση θέσης πράκτορα μετά από 4 μετακινήσεις μέσω τερματικού	63
Σχήμα 6.14 Σχηματική απεικόνιση πράκτορα μετά την 4η μετακίνηση.....	63
Σχήμα 6.15 Παρουσίαση θέσης πράκτορα μετά από 5 μετακινήσεις μέσω τερματικού	64
Σχήμα 6.16 Σχηματική απεικόνιση πράκτορα μετά την 5η μετακίνηση.....	64
Σχήμα 6.17 Σχηματική απεικόνιση του τερματικού του συστήματος κατά την λειτουργία της πλατφόρμας	65
Σχήμα 6.18 Σχηματική απεικόνιση πράκτορα μετά την 6η μετακίνηση.....	66
Σχήμα 6.19 Παράθυρο διεπαφής εξόδου του χρήστη από την πλατφόρμα.....	67

ΚΕΦΑΛΑΙΟ 1

Εισαγωγή

1.1 Αντικείμενο της διπλωματικής

Αντικείμενο της παρούσας διπλωματικής εργασίας αποτελεί η μελέτη για τη συνεργατική επίλυση προβλημάτων σε πολυπρακτορικά συστήματα. Η μελέτη αυτή επικεντρώνεται στη φύση των ευφών πρακτόρων καθώς και στις δυνατότητες που αυτοί παρέχουν. Επιπροσθέτως, δίνεται έμφαση στον τρόπο επικοινωνίας των πρακτόρων, στο συντονισμό καθώς και στον τρόπο με τον οποίο το σύστημα εν τέλει θα αποφασίσει να επιλύσει τα προβλήματα που εμφανίζονται. Η αξία της μελέτης αυτής διαφαίνεται περισσότερο σε περιπτώσεις άμεσης δράσης όπως η αναζήτηση και η διάσωση ανθρώπων όπου τόσο ο τρόπος όσο και ο χρόνος διεκπεραίωσης του εκάστοτε ζητήματος είναι εξίσου σημαντικός, καθώς όπως είναι προφανές η παραμικρή έλλειψη συντονισμού και γενικά χρόνου απόκρισης μπορεί να κοστίσει ανθρώπινες ζωές. Λαμβάνοντας υπ' όψη τη λεπτότητα του παραπάνω ζητήματος εμφανίζονται συνεχώς νέοι τρόποι για την αποτελεσματικότερη και βέλτιστη επίλυσή του. Σύνηθες φαινόμενο για επίλυση τέτοιων προβλημάτων έως και σήμερα είναι η χρήση ανθρώπινου δυναμικού, παρόλα αυτά παρατηρείται η τάση τέτοιου είδους ζητήματα να εμπλουτίζονται με τη χρήση τεχνητής νοημοσύνης καθώς παρατηρείται πως ομάδες ευφών πρακτόρων λαμβάνουν ίδιες ή και αποτελεσματικότερες αποφάσεις απ' ό,τι μια ομάδα ανθρώπων.

Στα πλαίσια της εργασίας αυτής επιλέχθηκε να μελετηθεί η περίπτωση των ναυτικών επιχειρήσεων διάσωσης για μια ομάδα ευφών πρακτόρων. Επίσης, για τους σκοπούς της συγκεκριμένης διπλωματικής δημιουργήθηκε ένα πρόγραμμα προσομοίωσης το οποίο θα βοηθήσει

στην κατανόηση τόσο του προβλήματος , όσο και στην προτεινόμενη υλοποίηση για την επίλυσή του.

1.1.1 Συνεισφορά

Η συνεισφορά της διπλωματικής συνοψίζεται ως εξής:

- Σχεδίαση αλγορίθμων και μεθόδων για βέλτιστη επικοινωνία καθώς και το συντονισμό μεταξύ των πρακτόρων.
- Εισαγωγή εννοιών του τομέα της τεχνητής νοημοσύνης, του τομέα των καταναμημένων συστημάτων καθώς και της θεωρίας δικτύων για ρεαλιστικά περιβάλλοντα άμεσης δράσης.
- Υλοποίηση βασικού λογισμικού για καλύτερη και αποτελεσματικότερη κατανόηση του προβλήματος και της προτεινόμενης λύσης του.
- Αναγωγή της υλοποίησης σε εξυπηρέτηση πολλαπλών ταυτόχρονων προβλημάτων.

1.2 Οργάνωση του τόμου

Στο κεφάλαιο 2 παρουσιάζεται το θεωρητικό υπόβαθρο το οποίο είναι απαραίτητο για την κατανόηση της παρούσας διπλωματικής εργασίας. Συγκεκριμένα γίνονται αναφορές στον τομέα της τεχνητής νοημοσύνης, των καταναμημένων συστημάτων και των δικτύων με αναλυτική παρουσίαση βασικών εννοιών και αρχών. Στο κεφάλαιο 3 επισημαίνονται στοιχεία καθώς και παράμετροι για το περιβάλλον των ναυτικών επιχειρήσεων διάσωσης που μελετάμε, ενώ στο

κεφάλαιο 4 αντίστοιχα αναφέρονται λεπτομερώς τα χαρακτηριστικά και οι προϋποθέσεις που πρέπει να πληρεί ένας πράκτορας σχεδιαστικά και αρχιτεκτονικά για να είναι μέρος του περιβάλλοντος αυτού. Η υλοποίηση του πράκτορα και του συστήματος συνολικά πραγματοποιείται στο κεφάλαιο 5, στο οποίο γίνεται λεπτομερής αναφορά στον τρόπο δόμησης του πράκτορα και του συστήματος, με την προτεινόμενη υλοποίηση να εμπίπτει σε ιεραρχική δομή. Στο κεφάλαιο 6 δίδεται μια πλατφόρμα προσομοίωσης ενός απλοϊκού μοντέλου του συστήματος με τη χρήση της Python, με σκοπό την περαιτέρω κατανόηση συμπεριφοράς του συστήματος για τη συγκεκριμένη υλοποίηση. Τέλος στο κεφάλαιο 7 έχουμε τη σύνοψη της παρούσας μελέτης, συμπεράσματα τα οποία προκύπτουν καθώς επίσης αναφέρονται ορισμένες μελλοντικές επεκτάσεις.

ΚΕΦΑΛΑΙΟ 2

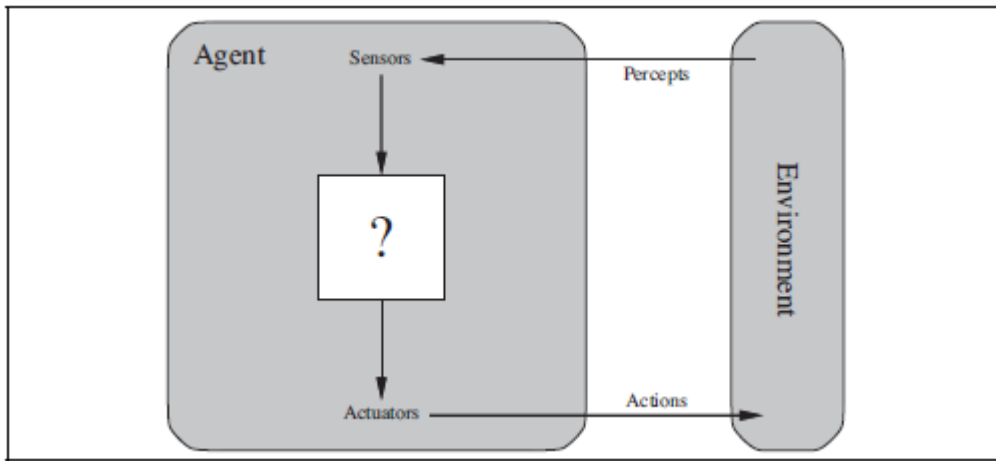
Θεωρητικό Υπόβαθρο

Στο κεφάλαιο που ακολουθεί περιγράφονται με αρκετή έμφαση βασικές έννοιες οι οποίες είναι απαραίτητες για την κατανόηση τόσο του περιβάλλοντος όσο και της φύσης του συστήματος γενικότερα. Συγκεκριμένα στο πρώτο μέρος του κεφαλαίου θα αναλυθούν ορισμοί και έννοιες Τεχνητής Νοημοσύνης και συγκεκριμένα έννοιες που αφορούν πράκτορες και περιβάλλον πρακτόρων. Εν συνεχεία δίδονται βασικές έννοιες για τη δόμηση των ευφυών πρακτόρων με αναφορές στο πεδίο των κατανεμημένων συστημάτων, ενώ τέλος θα γίνει χρήση θεωρίας δικτύων και κάλυψης χώρου μέσω συγκεκριμένης τοπολογίας. Η ανάγνωση του κεφαλαίου αυτού είναι απαραίτητη για την κατανόηση εννοιών καθώς και παραδειγμάτων που ακολουθούν στα επόμενα κεφάλαια.

2.1 Έννοιες από το πεδίο της Τεχνητής Νοημοσύνης

2.1.1 Πράκτορας - Agent

Η Τεχνητή Νοημοσύνη είναι εξ' ορισμού η επιστήμη που μελετά τους Λογικούς Πράκτορες (rational agents). Λογικός Πράκτορας (Σχήμα 2.1) είναι οτιδήποτε μπορεί να παίρνει αποφάσεις είτε ως άτομο, μηχανή ή λογισμικό. Ένας πράκτορας πρέπει να εμφανίζει τη δυνατότητα να αντιλαμβάνεται το περιβάλλον γύρω του μέσω αισθητήρων (sensors), καθώς και να επιδρά πάνω σε αυτό μέσω μηχανισμών δράσης (actuators).



Σχήμα 2.1 Σχηματική απεικόνιση πράκτορα [1]

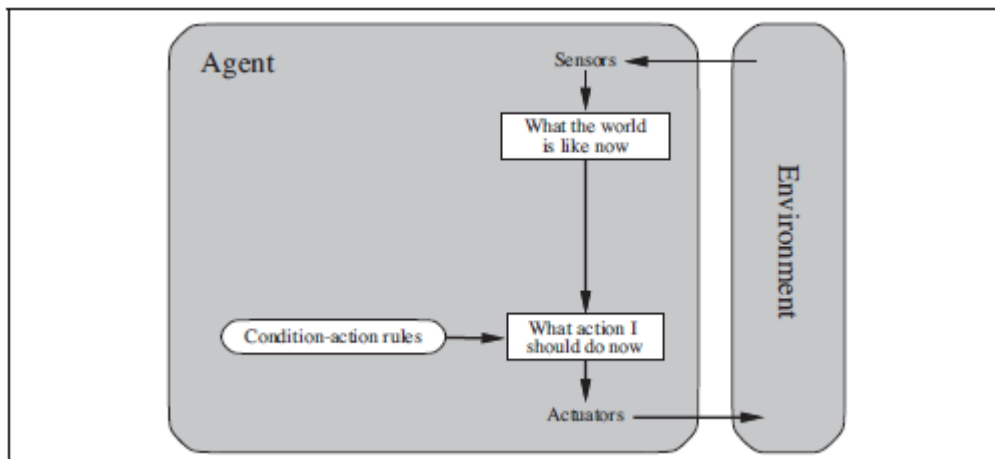
Ένας άνθρωπος μπορεί να θεωρηθεί και πράκτορας, διότι έχει αισθητήρες όπως για παράδειγμα την όραση, την ακοή την αφή και άλλους. Όμοια σκεπτόμενοι ένα ρομπότ μπορεί να έχει κάμερα ενσωματωμένη ή αισθητήρες με ενδείξεις θερμότητας ή και πολλά άλλα, οπότε μπορεί να θεωρηθεί πράκτορας. Τέλος ένα υπολογιστικό σύστημα (software program) μπορεί να θεωρηθεί πράκτορας γιατί μπορεί να λαμβάνει, είτε πληκτρολογήσεις, είτε περιεχόμενα αρχείων ή ακόμη και πακέτα μέσω επικοινωνίας (network packets) ως δεδομένα που θα έπαιρνε στις άλλες περιπτώσεις από τους αισθητήρες καθώς επίσης μπορεί να δράσει στο περιβάλλον, είτε μέσω απεικόνισης σε οθόνη, είτε γράφοντας σε ένα αρχείο ή στέλνοντας με τη σειρά του πακέτα επικοινωνίας.

Τα ερεθίσματα που λαμβάνονται από τους εκάστοτε αισθητήρες του κάθε πράκτορα ονομάζονται **αντιλήψεις-Percepts**. Οι αντιλήψεις του πράκτορα σε ορισμένα μοντέλα πρακτόρων καθορίζουν τη δράση του. Σε άλλα μοντέλα προηγείται μια διαδικασία πριν αποφασιστεί η δράση.

Μοντέλα Πρακτόρων

- Απλοί αντανεκλαστικοί πράκτορες (simple reflex agents)

Είναι οι πράκτορες οι οποίοι επιλέγουν να ενεργήσουν με βάση μόνο την τρέχουσα αντίληψη, χωρίς να ενδιαφέρονται για τυχόν ιστορικό αντιλήψεων.



Σχήμα 2.2 Σχηματικό διάγραμμα απλού αντανεκλαστικού πράκτορα [1]

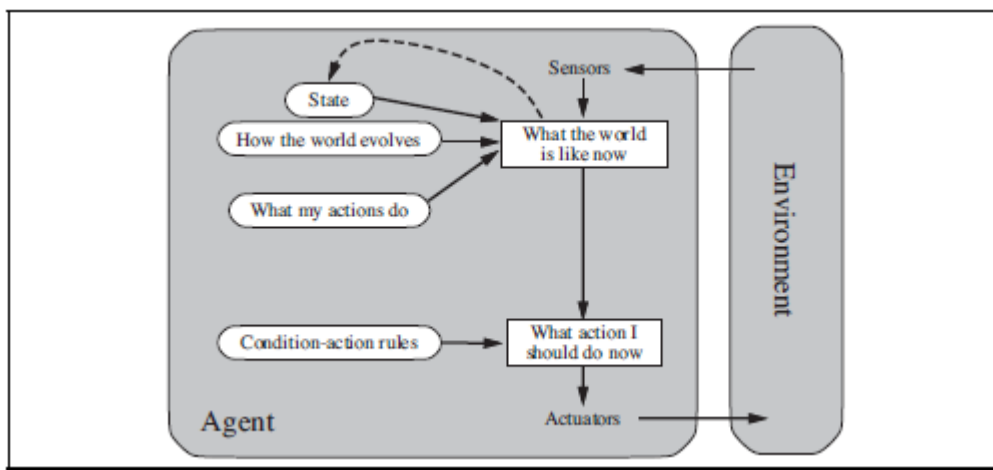
Οι απλοί αντανεκλαστικοί πράκτορες (Σχήμα 2.2) παρά την απλή δομή τους πολλές φορές είναι η καλύτερη επιλογή ακριβώς γιατί ενδιαφερόμαστε μόνο για την τρέχουσα αντίληψη. (χαρακτηριστικό παράδειγμα είναι ο θερμοστάτης)

- Απλοί αντανεκλαστικοί πράκτορες με μοντέλο (model based agents)

Ο πιο αποτελεσματικός τρόπος για να χειριστούμε τη μερική παρατηρησιμότητα του περιβάλλοντος για κάθε πράκτορα είναι να κρατά γνώση-ιστορικό για το μέρος του κόσμου που δεν μπορεί να παρατηρήσει άμεσα. Σ' αυτή την περίπτωση οι πράκτορες πρέπει να

διατηρούν μια εσωτερική κατάσταση η οποία εξαρτάται από το ιστορικό αντιλήψεων και αντικατοπτρίζει τουλάχιστον κάποιες από τις μη παρατηρήσιμες πτυχές της τρέχουσας κατάστασης. Ο πράκτορας που έχει αυτή τη δυνατότητα μπορεί να εξάγει συμπεράσματα και να διαμορφώσει ένα μοντέλο για το πως λειτουργεί ο κόσμος (Σχήμα 2.3).

Αυτό το μοντέλο βοηθά τους πράκτορες διότι μπορούν να καταλάβουν το τρόπο που μεταβάλλεται το περιβάλλον, είτε μέσω των ενεργειών τους, είτε ανεξαρτήτως αυτών.

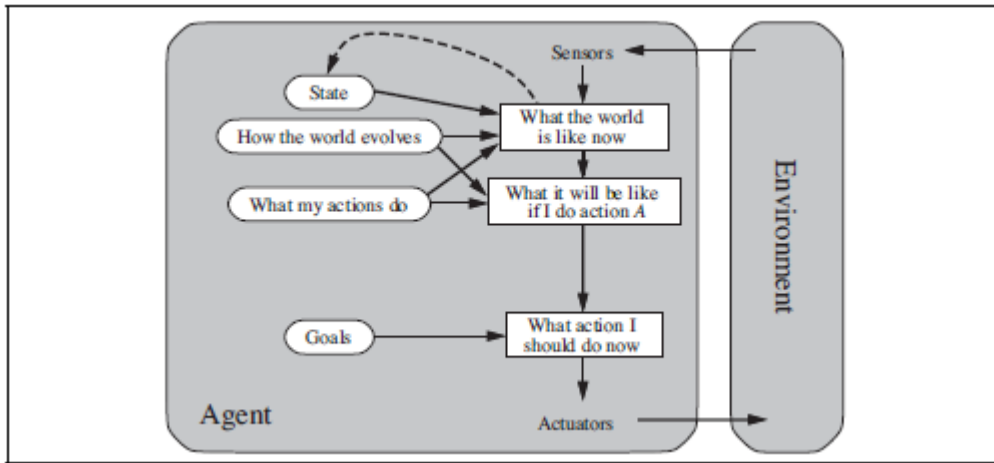


Σχήμα 2.3 Σχηματικό διάγραμμα πράκτορα με μοντέλο [1]

- Πράκτορες βασισμένοι στο στόχο (goal based agents)

Υπάρχουν φορές που το να γνωρίζει ένας πράκτορας την τρέχουσα κατάσταση του περιβάλλοντος ή ακόμα και να διαθέτει ιστορικό, δεν είναι αρκετό για να λάβει τη σωστή απόφαση. Ένα χαρακτηριστικό παράδειγμα είναι μια διασταύρωση στην οποία ένας οδηγός μπορεί να στρίψει είτε αριστερά, είτε δεξιά ή να συνεχίσει ευθεία. Η σωστή απόφαση για τον οδηγό καθορίζεται από τον προορισμό του. Με άλλα λόγια ένας πράκτορας όμοια θα χρειαστεί κάποιο είδος πληροφορίας-στόχου που θα του περιγράφει τα επιθυμητά σενάρια για την κατάσταση του κόσμου εφόσον πράξει.

Οι πράκτορες αυτοί συνδυάζουν ότι λογισμικό είδαμε παραπάνω στους απλούς αντανακλαστικούς πράκτορες με μοντέλο, επιλέγοντας το μοντέλο να είναι η επίτευξη ενός στόχου (goal) (Σχήμα 2.4).



Σχήμα 2.4 Σχηματικό διάγραμμα πράκτορα βασισμένου σε στόχο [1]

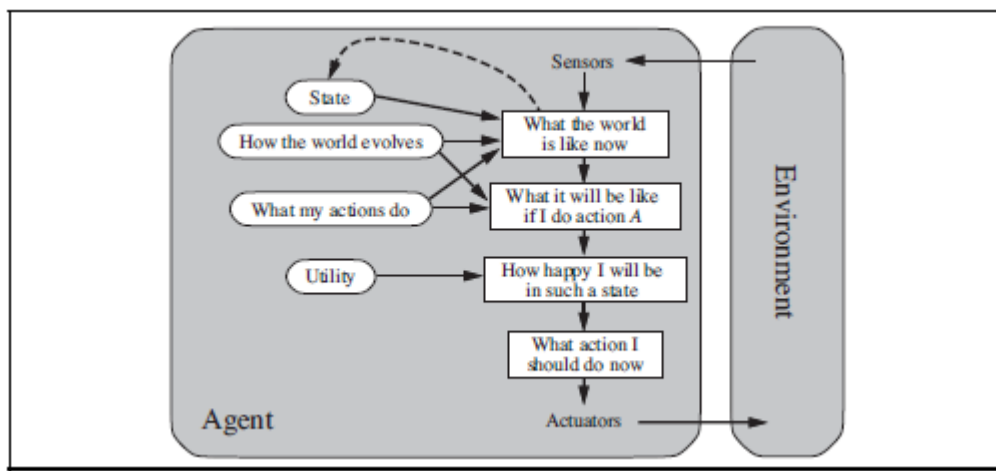
Οι πράκτορες βασισμένοι στο στόχο είναι εφοδιασμένοι με τεχνικές αναζήτησης και σχεδιασμού έτσι ώστε σε κάθε περίπτωση να μπορούν να βρουν ένα ρεπερτόριο ενεργειών που θα επιτυγχάνει το στόχο τους.

- Πράκτορες βασισμένοι στη χρησιμότητα (utility based agents)

Θέτοντας στόχους στους πράκτορες, δεν είναι αρκετό για να εξάγουν απόφαση υψηλής ποιότητας (high quality behavior) στα περισσότερα περιβάλλοντα. Όμοια με το παραπάνω παράδειγμα της διασταύρωσης και του οδηγού, το να θέσουμε απλά τον προορισμό δεν είναι τις περισσότερες φορές αρκετό. Εξασφαλίζουμε μεν την επιτυχία του στόχου, αλλά δεν έχουμε γνώση για το αν υπάρχουν πιο γρήγορες διαδρομές και συνεπώς λιγότερη απόσταση για την επίτευξη του στόχου ή αν υπάρχουν διαδρομές πιο ασφαλείς και πιο φθηνές από άλλες. Συνεπώς, η αλλαγή με το προηγούμενο μοντέλο πράκτορα είναι η προσθήκη

μιας συνάρτησης χρησιμότητας στον πράκτορα (utility function) η οποία έχει ως σκοπό να βοηθά τον πράκτορα να αποφασίζει ποια είναι η καλύτερη επιλογή με βάση τη χρησιμότητα που έχουμε αναδείξει για κάθε πιθανή κατάσταση του περιβάλλοντος.

Οι πράκτορες βασισμένοι στο στόχο παρέχουν μια σκληρά δυαδική μορφή μεταξύ επιθυμητής ή ανεπιθυμητής κατάστασης. Στην περίπτωση των πρακτόρων βασισμένων στη χρησιμότητα έχουμε τη δυνατότητα να ξέρουμε πόσο επιθυμητή είναι μια κατάσταση ή όχι (Σχήμα 2.5).



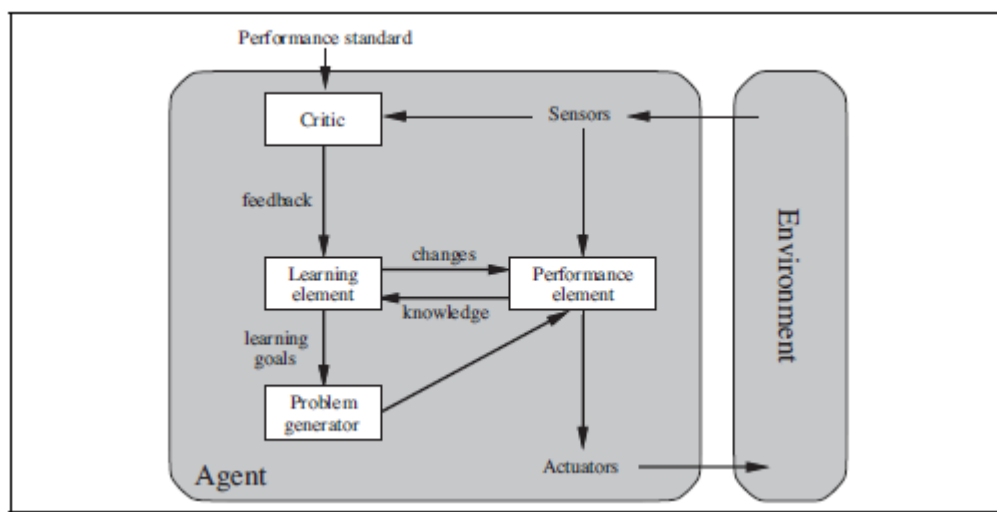
Σχήμα 2.5 Σχηματική απεικόνιση πράκτορα βασισμένου στη χρησιμότητα [1]

Στην περίπτωση αυτή ο πράκτορας το μόνο που έχει να κάνει είναι να επιλέξει την ενέργεια η οποία του μεγιστοποιεί τη συνάρτηση χρησιμότητας, δηλαδή την πιο επιθυμητή για αυτόν κατάσταση.

- Πράκτορες που μαθαίνουν (learning agents)

Πράκτορες που μαθαίνουν ονομάζονται οι πράκτορες που έχουν την ικανότητα να μαθαίνουν μέσω των ενεργειών τους. Οι πράκτορες αυτοί αποκτούν κριτική ικανότητα και πείρα για το περιβάλλον τους όσο περισσότερο δρουν σε αυτό (Σχήμα 2.6).

Επιπροσθέτως το να εξοπλίσουμε ένα περιβάλλον με τέτοιους πράκτορες θεωρείται η βέλτιστη επιλογή καθώς οι πράκτορες αυτοί μπορούν και γίνονται πιο ανταγωνιστικοί με το πέρασ του χρόνου, διότι πέραν την αρχική γνώση που ίσως κατέχουν για το περιβάλλον, μπορούν και αναπτύσσουν επιμέρους δική τους άποψη (κριτική).



Σχήμα 2.6 Σχηματική απεικόνιση πράκτορα που μαθαίνει [1]

Οι πράκτορες που μαθαίνουν εμφανίζουν στοιχεία βελτίωσης σε λήψη αποφάσεως συγκριτικά με τους απλούς πράκτορες. Να σημειωθεί ότι καθένας από τους παραπάνω τύπους πρακτόρων μπορεί να αναχθεί σε πράκτορα που μαθαίνει.

2.1.2 Περιβάλλον - Environment

Περιβάλλον ή αλλιώς περιβάλλον εργασίας (task environment) είναι το περιβάλλον μέσα στο οποίο τοποθετείται ένας η παραπάνω πράκτορες. Συχνά ο όρος περιβάλλον εργασίας ταυτίζεται με τον όρο "προβλήματα" στα οποία οι ορθολογικοί πράκτορες (rational agents) θα πρέπει

να ταυτίζονται ως "λύση". Συνήθως χρειάζονται επιμέρους στοιχεία για ένα περιβάλλον έτσι ώστε να μπορέσουμε να τα διαχωρίσουμε σε κατηγορίες. Αυτά τα στοιχεία που καθορίζουν το είδος του περιβάλλοντος είναι γνωστά ως PEAS όπου Performance, Environment, Actuators, Sensors είναι τα στοιχεία που χρειάζεται να ληφθούν υπ' όψη. Αναλυτικά οι κατηγορίες περιβαλλόντων είναι οι εξής:

- **Πλήρως ή Μερικώς Παρατηρήσιμο Περιβάλλον**

Πλήρως παρατηρήσιμο περιβάλλον είναι το περιβάλλον στο οποίο οι αισθητήρες του πράκτορα ή των πρακτόρων μπορούν και παρέχουν πληροφορίες για την κατάσταση ολόκληρου του περιβάλλοντος. Αντίστοιχα μερικώς παρατηρήσιμο περιβάλλον είναι εκείνο το περιβάλλον στο οποίο οι αισθητήρες του πράκτορα ή των πρακτόρων δεν έχουν ανά πάσα στιγμή πληροφορίες για το σύνολο του χώρου.

- **Αιτιοκρατικό ή Στοχαστικό Περιβάλλον**

Αιτιοκρατικό ονομάζεται το περιβάλλον του οποίου οι καταστάσεις εξαρτώνται από τις ενέργειες του πράκτορα. Συγκεκριμένα στα αιτιοκρατικά περιβάλλοντα οι επόμενες καταστάσεις εξαρτώνται πλήρως από την υπάρχουσα κατάσταση του περιβάλλοντος και από την ενέργεια του πράκτορα πάνω σε αυτό. Υποκατηγορία των περιβαλλόντων αυτών αποτελούν τα στρατηγικά περιβάλλοντα στα οποία οι ενέργειες άλλων πρακτόρων δεν παίζουν ρόλο για την επόμενη κατάσταση.

- **Στατικό ή Δυναμικό Περιβάλλον**

Στατικά είναι τα περιβάλλοντα στα οποία ο χώρος παραμένει σταθερός, δηλαδή δεν αυξομειώνεται με το χρόνο. Αντίστοιχα δυναμικά είναι τα περιβάλλοντα στα οποία ο χώρος μπορεί και μεταβάλλεται.

- **Διακριτό ή Συνεχές Περιβάλλον**

Η διαφορά σε αυτές τις κατηγορίες περιβάλλοντος είναι το κατά πόσο οι καταστάσεις του είναι διακριτές ή συνεχείς. Αυτό καθορίζεται από μονάδες όπως ο χρόνος που κυλάει στο περιβάλλον ή οι αντιλήψεις και οι ενέργειες των πρακτόρων που επιδρούν σ' αυτό.

- **Μονοκρατορικό ή Πολυπρακτορικό Περιβάλλον**

Τα περιβάλλοντα χωρίζονται σε αυτές τις δυο κατηγορίες ανάλογα με τον αριθμό των πρακτόρων που βρίσκονται και επιδρούν σε αυτά. Στην περίπτωση των πολυπρακτορικών περιβαλλόντων έχουμε δυο υποκατηγορίες οι οποίες καθορίζονται από τους πράκτορες. Συγκεκριμένα αν σε ένα πολυπρακτορικό σύστημα οι πράκτορες έχουν κοινούς στόχους, τότε το περιβάλλον είναι συνεργατικό (cooperative) και οι πράκτορες συνεργάζονται για να πετύχουν τους στόχους τους. Στην αντίθετη περίπτωση τα περιβάλλοντα ονομάζονται ανταγωνιστικά (competitive) ακριβώς διότι οι πράκτορες έχουν διαφορετικούς στόχους και πολλές φορές εμφανίζονται διαμάχες (conflicts) ανάμεσά τους οι οποίες συνήθως οφείλονται στη χρησιμότητα του καθενός.

2.2 Βασικές έννοιες κατανεμημένων συστημάτων

Ένας πράκτορας ορίζεται είτε ως μια φυσική, είτε ως μια εικονική ύπαρξη που μπορεί να δρα, να αντιλαμβάνεται το περιβάλλον του (μερικώς ή εξ' ολοκλήρου) και να επικοινωνεί με άλλους, όντας παρόλα αυτά αυτόνομος. Επίσης έχει ικανότητες που του επιτρέπουν να επιτυγχάνει τους στόχους του. Μιλώντας για ένα πολυπρακτορικό σύστημα και το περιβάλλον του, στο οποίο συμπεριλαμβάνονται αντικείμενα και πράκτορες (μόνο οι πράκτορες διαθέτουν την δυνατότητα επικοινωνίας-δράσης), οι σχέσεις μεταξύ όλων αυτών εμφανίζουν κοινά στοιχεία με τις σχέσεις που εμφανίζονται στα κατανεμημένα συστήματα. [2]

Ένα κατανεμημένο σύστημα δεν είναι τίποτε άλλο παρά ένας πράκτορας, ο οποίος παρόλα αυτά δεν εμφανίζει λογική και αυτονομία. Συνεπώς η υλοποίηση ενός πράκτορα (implementation) εμφανίζει πολλά κοινά στοιχεία με την υλοποίηση των κατανεμημένων συστημάτων. Η διαφορά έγκειται μόνο στο ότι τα κατανεμημένα συστήματα χρησιμοποιούνται για να παράγουν ρητά ένα αποτέλεσμα, ενώ χρησιμοποιώντας πράκτορες και μέσω της συνάρτησης χρησιμότητας που διαθέτουν, το αποτέλεσμα που θα εξαχθεί μπορεί να ποικίλει.

2.2.1 Βασικές λειτουργίες των κατανεμημένων συστημάτων

Κατανεμημένο ονομάζεται ένα σύστημα το οποίο απαρτίζεται από ξεχωριστές ενεργές οντότητες που εκτελούνται ταυτόχρονα αλλά ανεξάρτητα μεταξύ τους, παρουσιάζουν βλάβες, δεν διαθέτουν κοινή μνήμη ούτε κοινό ρολόι αλλά μπορούν και επικοινωνούν με μηνύματα πάνω από ένα δίκτυο με σκοπό την εξυπηρέτηση συγκεκριμένων στόχων. [3]

Όπως διαφαίνεται, βλάβες σε κατανεμημένα συστήματα μπορεί να υπάρξουν σε πολλά στάδια υλοποίησης. Μερικά εξ' αυτών είναι:

- Μη αξιόπιστη επικοινωνία μεταξύ των οντοτήτων του συστήματος
- Λάθη Συγχρονισμού, λόγω των διαφορετικών ρολογιών

- Βυζαντινές Βλάβες, συνδυασμός βλαβών - "κακόβουλοι" κόμβοι του συστήματος [4]

Οι λύσεις σε αυτά τα προβλήματα δεν αποτελούν τμήμα της συγκεκριμένης έρευνας αλλά αναφέρονται ενδεικτικά μερικοί ορισμοί-τρόποι υπερπήδησης των συγκεκριμένων εμποδίων:

- Αξιόπιστη επικοινωνία (reliable multicast) η οποία επιτυγχάνεται από το implementation-middleware του εκάστοτε πράκτορα-κόμβου του συστήματος.
- Χρήση χρονο-σφραγίδων (timestamps) ή εναλλακτικών μεθόδων (διανυσματικά ρολόγια κ.α) προσαρτημένα πάλι στο implementation του εκάστοτε πράκτορα για εξάλειψη των βλαβών συγχρονισμού.
- Οι **Βυζαντινές Βλάβες** είναι το χειρότερο είδος βλάβης και το πιο δύσκολο να αναγνωριστεί από το σύστημα. Η πιο συνηθισμένη κατηγορία βυζαντινών βλαβών είναι η δυσλειτουργία ενός ή περισσοτέρων κόμβων του συστήματος. Συγκεκριμένα ένας κόμβος ονομάζεται βυζαντινός αν στέλνει λανθασμένες πληροφορίες στο σύστημα ή γενικότερα υπονομεύει με τη μη ορθή συμπεριφορά του τη γενικότερη λειτουργία του συστήματος. Ο εντοπισμός των κόμβων που εμφανίζουν βυζαντινή βλάβη δεν είναι εύκολος διότι το σύστημα μπορεί να χρειαστεί πολλούς κύκλους ανταλλαγής μηνυμάτων για να εντοπίσει αυτή τη δυσλειτουργία. [4]

Για την παρούσα διπλωματική, το πολυπρακτορικό σύστημα που θα διερευνηθεί εμφανίζει επακριβώς αυτά τα στοιχεία των κατανεμημένων συστημάτων.

2.3 Βασικές έννοιες από το πεδίο των δικτύων.

Όπως διαφαίνεται από την παραπάνω ενότητα περί των χαρακτηριστικών των κατανεμημένων συστημάτων, δημιουργείται η ανάγκη να διερευνήσουμε τη δομή που εμφανίζει το κάθε σύστημα. Οι πράκτορες-κόμβοι του συστήματος οι οποίοι επικοινωνούν μεταξύ τους εμφανίζουν μια τοπολογία στο χώρο η οποία δεν παραμένει σταθερή, διότι οι πράκτορες-κόμβοι μπορούν και μετακινούνται στο χώρο, χωρίς παρόλα αυτά το γεγονός αυτό να επηρεάζει τη λειτουργία του συστήματος.

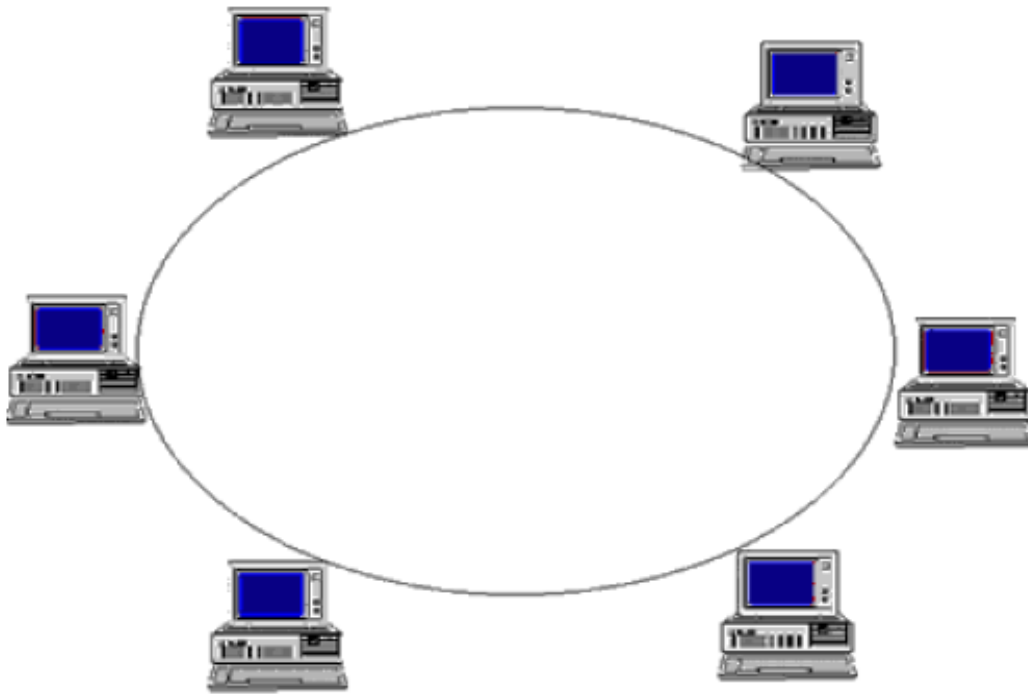
- Δίκτυο ονομάζεται οποιαδήποτε συλλογή από ανεξάρτητα υπολογιστικά συστήματα τα οποία επικοινωνούν μεταξύ τους πάνω από ένα κοινό αγωγό-δίκτυο μεταφοράς δεδομένων. [5]

Υπάρχουν δύο μεγάλες κατηγορίες δικτύων οι οποίες είναι οι εξής:

- Peer-to-peer
- Server-Based

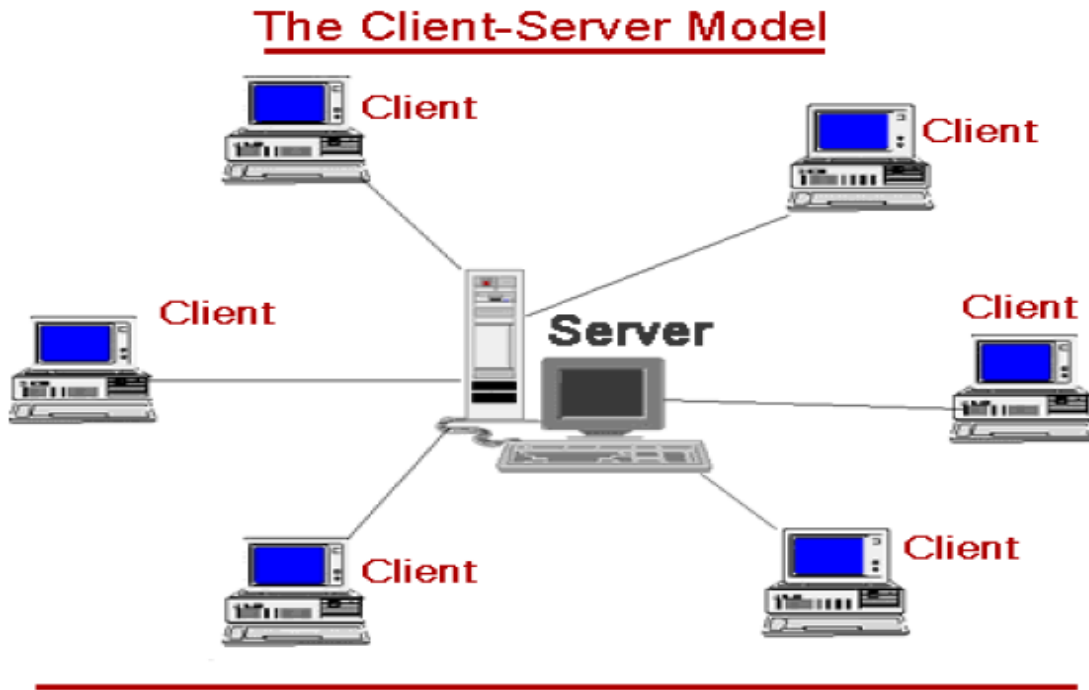
Στην κατηγορία peer-to-peer δικτύων (Σχήμα 2.7) δεν υφίσταται καμία ιεραρχική δόμηση μεταξύ των κόμβων του δικτύου. Όλοι οι κόμβοι είναι "ίσοι" μεταξύ τους και δεν υπάρχει κάποιος κόμβος υπεύθυνος για την λειτουργία του δικτύου. Τέτοιου είδους δίκτυα είναι καλή επιλογή για περιπτώσεις συστημάτων όπου οι κόμβοι βρίσκονται σε σχετικά μικρή απόσταση και δεν μπορούν να απομακρυνθούν. Σε αυτά τα δίκτυα δεν υπάρχουν προβλήματα ασφαλείας.

The Peer-to-Peer Model



Σχήμα 2.7 Σχηματική απεικόνιση των peer-to-peer δικτύων [5]

Η κατηγορία των Sever-Based δικτύων (Σχήμα 2.8) είναι εξ' ολοκλήρου δομημένη σε ένα μοντέλο ιεραρχικής δόμησης, όπου υπεύθυνος για την ομαλή λειτουργία-επικοινωνία του δικτύου και των κόμβων του είναι ο server.



Σχήμα 2.8 Σχηματική απεικόνιση των Server-Based δικτύων [5]

Σε αυτό το μοντέλο δικτύου είναι βασισμένοι πολλοί αλγόριθμοι και εξυπηρέτησης-επικοινωνίας οι οποίοι έχουν παρόμοια βάση αλλά διαφορετικά χαρακτηριστικά. Για τις ανάγκες της συγκεκριμένης εργασίας θα αναφερθούν ορισμοί για τα δίκτυα με τοπολογία και επίλυση σε μορφή clustering καθώς και βασικές ιδιότητες των δικτύων αυτών.

2.3.1 Clustering method

Cluster ονομάζεται μια υποδομή ενός δικτύου η οποία έχει αφηρημένη μορφή και αποσκοπεί σε δύο τομείς:

- Τοπικές αλλαγές στην τοπολογία του δικτύου να μην γίνονται γνωστές σε ολόκληρο το δίκτυο
- Η πληροφορία που δίδεται από τους κόμβους του δικτύου να μην συγκεντρώνεται σε μια μόνο κεντρική μονάδα. [6]

Clustering είναι η διαδικασία ορισμού αυτών των υποδομών μέσα σε ολόκληρη την τοπολογία του δικτύου. Οι κόμβοι διακρίνονται σε clusterheads, gateways και ordinal nodes.

Clusterheads είναι οι κόμβοι οι οποίοι είναι υπεύθυνοι να κινούν τα νήματα σε ένα cluster. Δεν διαφέρουν στο implementation από τους υπόλοιπους κόμβους του δικτύου απλά από τη στιγμή που θα οριστούν ως clusterheads ενός cluster αποκτούν περισσότερες αρμοδιότητες. Για την αναγωγή ενός κόμβου σε clusterhead έχουν σχεδιαστεί αρκετοί αλγόριθμοι με πιο βασικό τον Distributed Clustering Algorithm (DCA). [7]

Δημιουργώντας αυτές τις δομές επιτυγχάνουμε μια ιεράρχηση στη μορφολογία του δικτύου η οποία εμφανίζει χρονικές βελτιστοποιήσεις σε περιπτώσεις τόσο καλής λειτουργίας του δικτύου, όσο και σε περιπτώσεις βλάβης.

Gateways ονομάζονται οι κόμβοι οι οποίοι συνορεύουν με γειτονικά clusters, δηλαδή βρίσκονται κοντά σε ένα γειτονικό cluster. Οι κόμβοι αυτοί εμφανίζουν ενδιαφέρον διότι πρέπει να είναι ενήμεροι για τις επιτρεπτές κινήσεις που μπορούν να κάνουν χωρίς να ξεφύγουν από τα όρια του cluster τους. Ordinal nodes ονομάζονται όλοι οι κόμβοι του δικτύου.

2.3.2 Clustering και κατανεμημένα συστήματα

Τα κατανεμημένα συστήματα με τη σειρά τους εμφανίζουν ιεραρχική δόμηση. Παραλληλίζοντας τους δύο τομείς μπορούμε να θεωρήσουμε πως ο όρος clusterhead από τη θεωρία δικτύων και ο όρος coordinator για κατανεμημένα συστήματα είναι ο ίδιος αναφορικά με τη λειτουργία τους.

Ένας coordinator κόμβος σε ένα ιεραρχικά δομημένο κατανεμημένο σύστημα είναι εκείνος που θα "αποφασίζει" για τους κόμβους του συστήματος. Η μόνη διαφορά με τον όρο cluster-

head έγκειται στον τρόπο με τον οποίο αποφασίζει το εκάστοτε κατανεμημένο σύστημα να εκλέξει έναν κόμβο του ως coordinator κόμβο σε περίπτωση βλάβης.[3]

Σε περιπτώσεις βλάβης του coordinator κόμβου τα κατανεμημένα συστήματα μπαίνουν σε μια διαδικασία εκλογών για τον ορισμό νέου coordinator. Ας πάρουμε ως παράδειγμα ένα κατανεμημένο σύστημα το οποίο απαρτίζεται από αυτόνομους κόμβους, οι οποίοι επικοινωνούν μεταξύ τους και έχουν ένα κοινό σκοπό την επίλυση ενός προβλήματος. Αν σε αυτό το σύστημα στερήσουμε τον κόμβο διαχείρισης, τότε δημιουργούμε στο σύστημα μια τρύπα. Το σύστημα με οποιοδήποτε τρόπο πρέπει να δώσει λύση στο πρόβλημα και αυτό γίνεται με τη διεξαγωγή εκλογών ανάμεσα στα ενεργά μέλη-κόμβους του συστήματος, τα οποία θα εκλέξουν έναν νέο coordinator. Η διαδικασία εξαγωγής αποτελέσματος επίλυσης του εκάστοτε προβλήματος παγώνει και το πρόβλημα επιλύεται εφόσον ανακηρυχθεί ο νέος coordinator. [3]

ΚΕΦΑΛΑΙΟ 3

Περιβάλλον διεκπεραίωσης των ναυτικών επιχειρήσεων

Στο κεφάλαιο αυτό θα αναφερθούν κάποια βασικά χαρακτηριστικά του περιβάλλοντος καθώς και παράμετροι που μπορούν να το επηρεάσουν και κατ' επέκταση να επηρεάσουν τη λειτουργία των πρακτόρων που δρουν σε αυτό.

3.1 Η φύση του περιβάλλοντος

Οποιοδήποτε είδος προβλήματος που προκύπτει σε θάλασσες και γενικά υδάτινες περιοχές απαιτεί προσκόμιση εξωτερικής βοήθειας για την επίλυσή του. Στην παρούσα μελέτη εστιάζουμε στις περιπτώσεις διάσωσης σε ναυτικές επιχειρήσεις, συνεπώς η φύση του περιβάλλοντος είναι μια θαλάσσια ζώνη καθώς και όλα τα μορφολογικά στοιχεία που συναντούμε σε αυτή.

3.2 Προσβασιμότητα

Η προσβασιμότητα των πρακτόρων σε οποιοδήποτε μέρος συνδέεται άμεσα με τη φύση του εκάστοτε περιβάλλοντος. Στις περιπτώσεις διάσωσης σε ναυτικές επιχειρήσεις όπου τα προβλήματα εντοπίζονται σε θαλάσσιους χώρους η προσβασιμότητα θεωρείται φαινομενικά εύκολη δηλαδή οι πράκτορες διάσωσης φαινομενικά μπορούν εύκολα να αποφύγουν τυχόν εμπόδια που εμφανίζονται όπως ύφαλοι ή βράχια κ.α. και να φτάσουν στο σημείο του ενδιαφέροντος. Αυτό όπως θα δούμε σε επόμενα κεφάλαια γίνεται εφικτό μέσω της αντίληψης που εμφανίζουν οι πράκτορες τόσο για το χώρο γύρω τους όσο και σχεδιαστικά με τον αλγόριθμο

υλοποίησής τους. Η προσβασιμότητα επίσης θεωρείται εύκολη διότι μιλάμε για πράκτορες-πλοία τα οποία είναι ιδανικά, ενεργειακά ανεξάρτητα και δεν έχουν ανάγκη εφοδιασμού, δηλαδή επιστροφής στην ξηρά.

3.3 Γεωγραφική Έκταση

Όπως αναφέρθηκε παραπάνω η πρόσβαση στις περιπτώσεις ανάγκης σε θαλάσσιους χώρους μπορεί να θεωρηθεί εύκολη, παρόλα αυτά η έκταση των θαλάσσιων ζωνών είναι υπερμεγέθης με αποτέλεσμα πολλές φορές η αμεσότητα της βοήθειας να μην θεωρείται δεδομένη. Στις περιπτώσεις αυτές (μεγάλων εκτάσεων) ο τρόπος επίλυσης του προβλήματος πρέπει να συμπεριλαμβάνει στις παραμέτρους του και το χρόνο απόκρισης. Συγκεκριμένα απαιτείται μια εικονική χαρτογράφηση των περιοχών που καλύπτονται από την εκάστοτε ομάδα με σκοπό την αποτελεσματικότερη και βέλτιστη αποστολή βοήθειας. Η χαρτογράφηση γίνεται σταδιακά από τους πράκτορες του συστήματος. Αυτό καθίσταται εφικτό για πράκτορες που εμφανίζουν στοιχεία εκμάθησης και που κρατούν ιστορικό για τις κινήσεις ή και τον χώρο αντίληψής τους. Ιδανικά το σύστημα μέσω της πληροφορίας των πρακτόρων που το απαρτίζουν, μετά το πέρας κάποιου χρονικού διαστήματος, θεωρείται πλήρως παρατηρήσιμο.

3.4 Καιρικές Συνθήκες

Οι καιρικές συνθήκες στις θαλάσσιες αποστολές διάσωσης παίζουν πρωταγωνιστικό ρόλο και είναι και εκείνες που συνήθως, πέραν των τεχνικών ή μηχανικών προβλημάτων που τυχόν εμφανιστούν στους πράκτορες, τις δημιουργούν. Έως και σήμερα η μετεωρολογική υπηρεσία εκδίδει δελτία καιρού που είναι πολλές φορές απαγορευτικά για τον απόπλου πλοίων διότι η επικινδυνότητα αυξάνεται ανάλογα των καιρικών συνθηκών (κυρίως λόγω ανέμων, χαμηλής ορατότητας κλπ). Οι άνεμοι μπορούν να προκαλέσουν προβλήματα στους πράκτορες καθώς θα δημιουργούν κύματα στη θάλασσα και το έργο τους θα είναι αρκετά δυσκολότερο τόσο στον εντοπισμό των διασωθέντων, όσο και στην προσκόμιση αυτών σε ασφαλές σημείο. Χα-

μηλή ορατότητα δημιουργείται από τυχόν καιρικά φαινόμενα όπως βροχές-καταιγίδες καθώς σε συνδυασμό με τους ανέμους μπορούν και καθιστούν το έργο των πρακτόρων του περιβάλλοντος δυσκολότερο. Λαμβάνοντας υπόψη τα παραπάνω, οι πράκτορες του συστήματος θα πρέπει να σχεδιαστούν ανάλογα, δηλαδή να μπορούν κατασκευαστικά να ανταπεξέρχονται σε οποιαδήποτε κατάσταση περιβάλλοντος. Όπως μπορούμε να καταλάβουμε ο καιρός παίζει πρωταγωνιστικό ρόλο για την διεξαγωγή των επιχειρήσεων διάσωσης. Ακριβώς για αυτό πρέπει να συμπεριληφθεί ως παράμετρος του συστήματος και του περιβάλλοντος.

ΚΕΦΑΛΑΙΟ 4

Σχεδιασμός πράκτορα για τις ναυτικές επιχειρήσεις διάσωσης

Με δεδομένα όσα έχουν αναφερθεί στο προηγούμενο κεφάλαιο για τη φύση του περιβάλλοντος δράσης των πρακτόρων είναι εύκολο να προσδιορίσουμε τόσο τον τύπο των πρακτόρων, όσο και την κατασκευή τους.

4.1 Η συνεισφορά της Τεχνητής Νοημοσύνης

Στην υποενότητα αυτή θα παρατεθούν μερικά από τα οφέλη της χρήσης Τεχνητής Νοημοσύνης σε ένα ζήτημα που αφορά την διάσωση ανθρώπινων ζωών. Υπήρχαν και υπάρχουν πολλές προκαταλήψεις περί της χρήσης ευφυών υπολογιστικών συστημάτων σε ποικίλους τομείς, πόσο μάλλον σε ένα τομέα όπου η παραμικρή χρονική απώλεια (delay) μπορεί να κοστίσει ανθρώπινες ζωές.

Σε αντίθεση με αυτές τις απόψεις, καθώς και με την ταυτόχρονη εξέλιξη των ευφυών συστημάτων τόσο σε επίπεδο λογισμικού (software) όσο και σε επίπεδο υλικού (hardware), μπορούμε πλέον με ορθό σχεδιασμό και διαδικασίες δοκιμών να υλοποιήσουμε ασφαλή ευφυή συστήματα και να εκμεταλλευτούμε όλα τα οφέλη που μας παρέχουν συγκριτικά με τις ανθρώπινες μεθόδους ανταπόκρισης που κυριαρχούν όλα τα χρόνια.

Παρακάτω αναφέρονται μερικά χαρακτηριστικά της Τεχνητής Νοημοσύνης που δεν συναντάμε σε άλλα συστήματα άμεσης δράσης:

- Ασφάλεια του ανθρώπινου δυναμικού διάσωσης. Οι πράκτορες είναι ανεξάρτητα όντα και θα αποστέλλονται σε όλες τις περιπτώσεις κινδύνου.
- Πράκτορες φτιαγμένοι με υλικά που θα αντέχουν στις χειρότερες καιρικές συνθήκες και καιρικές ανωμαλίες (όπως kevlar) με σκοπό την άμεση και ασφαλή προσέγγιση στο σημείο του προβλήματος.
- Μεγαλύτερη ακρίβεια σε υπολογισμούς εν ώρα δράσης, όπως αλλαγή κατεύθυνσης ή υπολογισμός ανθρωπίνων ζωών και χώρος που χρειάζεται για την ασφαλή τοποθέτησή τους.

4.2 Χαρακτηριστικά Πράκτορα

Οι πράκτορες αποτελούνται από την αρχιτεκτονική (όπως για παράδειγμα ένας υπολογιστής) και από το πρόγραμμα το οποίο "τρέχει" πάνω στην αρχιτεκτονική αυτή. [8]

Τα χαρακτηριστικά ενός πράκτορα είναι οι ιδιότητες που εμφανίζει μέσω της αρχιτεκτονικής του σχεδίασης. Τα χαρακτηριστικά αυτά αποτελούν μέρος της μηχανικής σχεδίασης του και είναι ξεχωριστά από το προγραμματιστικό τμήμα του το οποίο ευθύνεται για τη λειτουργία και τη λήψη αποφάσεων. Τα χαρακτηριστικά πρέπει να προσαρμόζονται ανάλογα το περιβάλλον στο οποίο ενεργεί ο πράκτορας καθώς και στις ενέργειες που είναι προδιαγεγραμμένο ότι πρέπει να διαθέτει στο ρεπερτόριό του. Ακολουθούν κάποια βασικά χαρακτηριστικά που κρίνονται απαραίτητα για το μηχανικό σχεδιασμό ενός πράκτορα που δρα σε ναυτικές επιχειρήσεις διάσωσης (emergency agent).

4.2.1 Περιγραφή πράκτορα διάσωσης

Η αρχιτεκτονική των πρακτόρων του συγκεκριμένου συστήματος βασίζεται σε ένα μοντέλο όμοιο με εκείνο των πλοίων. Συγκεκριμένα οι πράκτορες θα έχουν τη μορφή σκάφους, αφού ο

σχεδιασμός τους, όπως είναι προφανές, είναι ο καταλληλότερος για το περιβάλλον στο οποίο θα ενεργούν. Οι πράκτορες θα πρέπει να διαθέτουν, πέρα από τους αισθητήρες τους οποίους θα δούμε αναλυτικά, υδραυλικά μέρη τα οποία θα ανοίγουν και θα εμφανίζουν το εσωτερικό μέρος του πλοίου για την προσκόμιση των διασωθέντων καθώς και μηχανισμούς ρίψης σωσιβίων για τους ανθρώπους που βρίσκονται στη θάλασσα. Ταυτόχρονα είναι προφανές πως με τη χρήση της ρομποτικής ταχύτητας των πρακτόρων αυτών, η ανταπόκριση σε περιπτώσεις ανάγκης είναι βέλτιστη. Η ρομποτική ταχύτητα είναι ίσως το πιο σημαντικό προνόμιο που διαθέτουν οι πράκτορες-πλοία συγκριτικά με την αποστολή επανδρωμένων πλοίων.

Ένα επιπρόσθετο χαρακτηριστικό των πρακτόρων-πλοίων διάσωσης θα πρέπει να είναι η αυτονομία τους. Είναι μείζονος σημασίας ζήτημα να μην τίθεται θέμα ανεφοδιασμού καυσίμων στους πράκτορες αυτούς και η χρήση τους να είναι εφικτή ανά πάσα στιγμή. Τέτοια μοντέλα πρακτόρων έχουν ήδη κάνει την εμφάνισή τους αν και παρόλα αυτά βρίσκονται σε πολύ βασικό στάδιο υλοποίησης όσον αφορά την χρήση τους ως μη επανδρωμένα πλοία (Σχήμα 4.1).

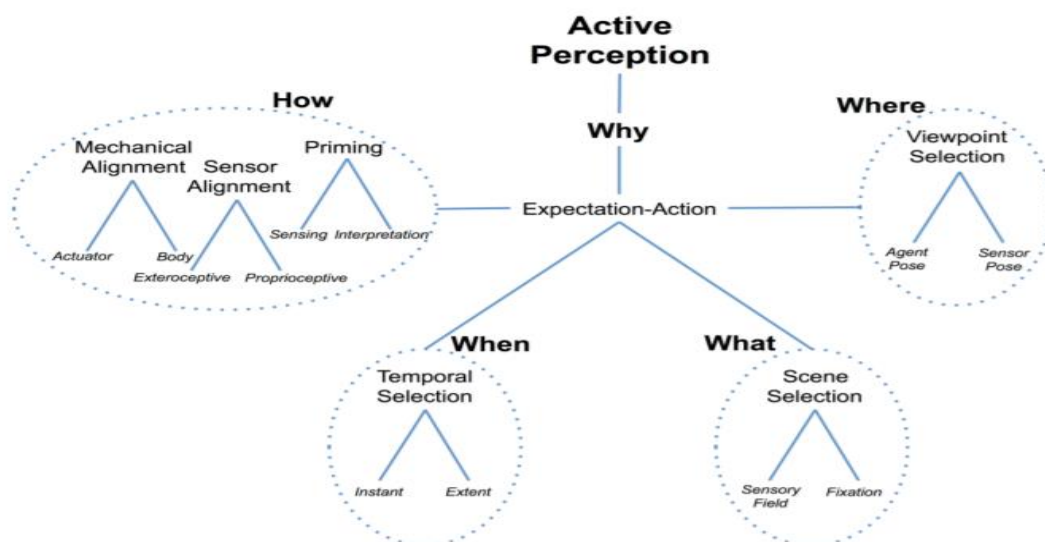


Σχήμα 4.1 Εικόνα ενός αυτόνομου πλοίου με χρήση ηλιακής ενέργειας για κινητήρια δύναμη [9]

4.2.2 Αντίληψη του χώρου και των πράξεων των πρακτόρων (Perception of Region & Action in Agents)

Ένας πράκτορας είθισται να έχει ένα συγκεκριμένο χώρο δράσης στο περιβάλλον στο οποίο βρίσκεται. Ο χώρος δράσης αυτός παρουσιάζει, ανάλογα τον πράκτορα που μελετάμε, κάποια συγκεκριμένη σχηματική μορφή (κυκλική, τετραγωνοειδής κλπ) και το κέντρο της είναι ο πράκτορας. Πιο πρακτικά εάν ένας πράκτορας μπορεί να παρατηρεί και να δρα σε ακτίνα ενός μέτρου προς πάσα κατεύθυνση από το σημείο που βρίσκεται, τότε το πεδίο αντίληψής του είναι ένας κύκλος με ακτίνα ένα μέτρο. Όμοια μπορούμε να ανάγουμε την παραπάνω θεώρηση και για οποιοδήποτε σχηματική μορφή. [1] [10]

Η γενικότερη μορφή αντιλήψεων ενός πράκτορα καθορίζεται κυρίως από τα ερεθίσματα που λαμβάνει από το εξωτερικό περιβάλλον. Την αντίληψη αυτή του χώρου, του την παρέχουν οι αισθητήρες. Όταν ένας πράκτορας λαμβάνει από τους αισθητήρες του μια κατάσταση του χώρου στον οποίο βρίσκεται, έρχεται αντιμέτωπος με μια σειρά ερωτημάτων για το πώς θα πρέπει να ερμηνεύσει τα στοιχεία που έλαβε και έπειτα για το τι πρέπει να πράξει. Μια απεικόνιση του τρόπου με τον οποίο ένας πράκτορας αντιλαμβάνεται το περιβάλλον φαίνεται στο Σχήμα 4.2.[11]



Σχήμα 4.2 Σχηματική απεικόνιση των ερωτημάτων που καλείται να απαντήσει ένας πράκτορας καθώς λάβει τα εξωτερικά ερεθίσματα [11]

Με βάση τα παραπάνω μπορούμε να χαρακτηρίσουμε ένα περιβάλλον πλήρως παρατηρήσιμο, αν οι αισθητήρες των πρακτόρων που δρουν σε αυτό έχουν την εμβέλεια και το καλύπτουν εξ' ολοκλήρου. Σε κάθε άλλη περίπτωση το περιβάλλον είναι τμηματικά παρατηρήσιμο. [1]

Ανάγοντας τα παραπάνω στο αντικείμενο της συγκεκριμένης εργασίας, οι πράκτορες του πολυπρακτορικού συστήματος μπορεί να εμφανίζουν είτε πλήρη, είτε τμηματική γνώση για το περιβάλλον στο οποίο δρουν. Το συγκεκριμένο θέμα δε θα απασχολήσει την παρούσα έρευνα καθώς για οποιοδήποτε πρόβλημα σε κάποιο μη παρατηρήσιμο τμήμα του χώρου υποθέτουμε πως το σύστημα ενημερώνεται. (Συγκεκριμένα αυτό συμβαίνει και στην πραγματικότητα είτε μέσω δορυφόρων, είτε μέσω ασυρμάτου από τα πλοία που εκπέμπουν σήμα βοήθειας).

4.2.3 Ρεπερτόριο ενεργειών πράκτορα διάσωσης

Το ρεπερτόριο ενεργειών των πρακτόρων για τις περιπτώσεις ναυτικών επιχειρήσεων διάσωσης είναι το σύνολο των δυνατών ενεργειών που μπορούν να εκτελεστούν ανά πάσα στιγμή από κάθε πράκτορα. Αυτές χωρίζονται στις εξής κατηγορίες:

- Δράσεις κίνησης : μετακίνηση ουσιαστικά του πράκτορα στο σημείο που εμφανίζεται το εκάστοτε πρόβλημα (για παράδειγμα Μετακινήσου βόρεια ή Στρίψε Δεξιά κλπ).
- Δράσεις επικοινωνίας : επικοινωνία με τους πράκτορες για συντονισμό και κατάστροψη σχεδίου. Αυτές οι δράσεις συμβαίνουν πριν τις δράσεις κίνησης.
- Δράσεις Διάσωσης : ρίψη σωσιβίων και έπειτα ασφαλής προσκόμιση των διασωθέντων στο εσωτερικό του πλοίου.

4.3 Αισθητήρες - Sensors

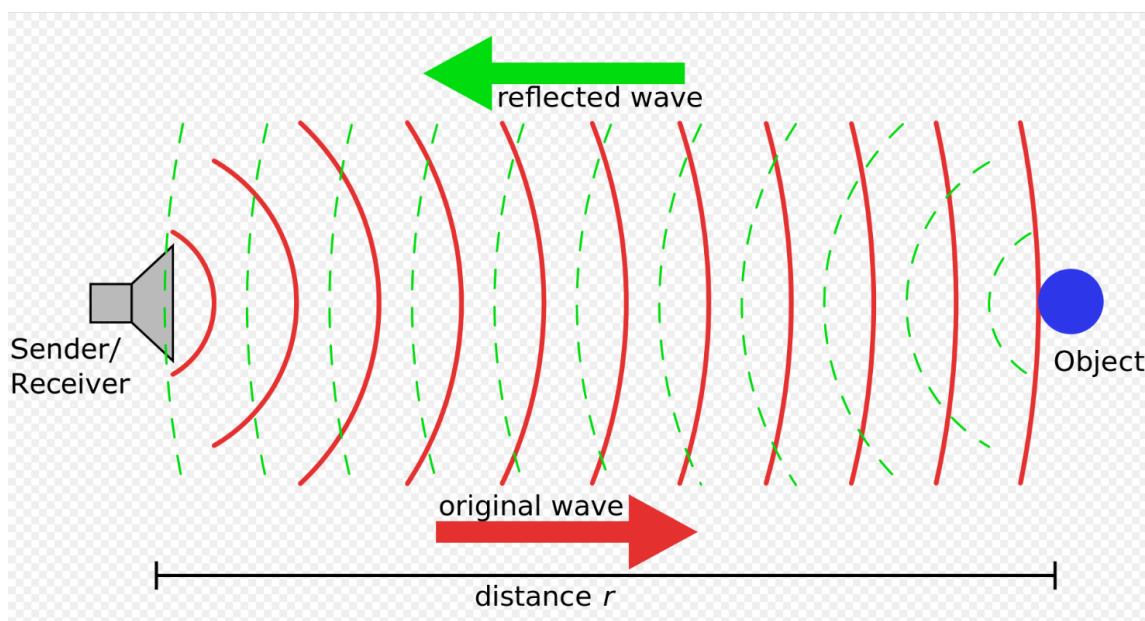
Όπως έχει αναφερθεί σε αρκετά σημεία της εργασίας έως τώρα οι αισθητήρες (sensors) οι οποίοι θα είναι τοποθετημένοι στους πράκτορες του συστήματος, διαδραματίζουν ίσως το σημαντικότερο ρόλο για τη λειτουργία τους. Αυτό καθίσταται προφανές, εφόσον οι αισθητήρες είναι εκείνοι οι οποίοι δίνουν τα πρωταρχικά ερεθίσματα-αντιλήψεις στους πράκτορες τόσο για το περιβάλλον στο οποίο βρίσκονται, όσο και για την ύπαρξη προβλήματος σε αυτό. Συνεπώς καταλαβαίνουμε πως η συλλογή δεδομένων καθώς και η λήψη αποφάσεων από τους πράκτορες λαμβάνονται πρωταρχικά με βάση τις ενδείξεις των αισθητήρων τους.

Για την αντιμετώπιση του συγκεκριμένου προβλήματος οι αισθητήρες που θα πρέπει να διαθέτει ένας πράκτορας του συστήματος είναι οι εξής :

- **Αισθητήρες τοποθεσίας - χώρου.** Οι αισθητήρες αυτοί είναι απαραίτητοι για να γνωρίζουμε ανά πάσα στιγμή την τοποθεσία του κάθε πράκτορα στο χώρο, καθώς και το πόσο απέχει από ενδεχόμενη κατάσταση προβλήματος σε οποιοδήποτε σημείο του χώρου. Οι αισθητήρες αυτοί μπορούν να θεωρηθούν ως ένα είδος GPS για το σύστημα, δηλαδή ο κάθε πράκτορας εμφανίζει ένα στίγμα ανάλογα την τοποθεσία του. Οι παραπάνω αισθητήρες πέραν της χρησιμότητας για τον εκάστοτε πράκτορα ως μονάδα, εμφανίζουν όπως θα δούμε στο επόμενο κεφάλαιο της εργασίας, μεγάλη χρησιμότητα για όλο το δίκτυο καθώς βοηθούν στη δημιουργία συγκεκριμένης τοπολογίας δικτύου μεταξύ των πρακτόρων (χαρτογράφηση).
- **Αισθητήρες περιβάλλοντος.** Οι αισθητήρες αυτοί δίνουν στον πράκτορα μια απεικόνιση του περιβάλλοντος στο οποίο βρίσκεται. Συγχρόνως οι αισθητήρες αυτοί είναι εκείνοι που πολλές φορές θα βοηθούν τον πράκτορα και κατ' επέκταση το σύστημα να αντιλαμβάνεται τυχόν προβλήματα σε κοντινή απόσταση από τον πράκτορα (Ακτίνα Δράσης του πράκτορα). Αισθητήρες τέτοιας φύσης είναι συνήθως κάμερες με ηχητική και εικονική κάλυψη οι οποίες σε περιπτώσεις ανάγκης μπορούν να μετατρέψουν την

εικόνα σε ένδειξη θερμικής προσομοίωσης για τον εντοπισμό ανθρώπων (περιπτώσεις χαμηλής ορατότητας λόγω καιρικών συνθηκών).

- **Αισθητήρες ανακάλυψης προβλημάτων και εμποδίων.** Τέτοιοι αισθητήρες χρησιμοποιούνται και σήμερα υπό τη μορφή ηχοπλοήγησης, είναι γνωστοί και ως SO.N.A.R, οι οποίοι είναι ουσιαστικά ηλεκτροακουστικές συσκευές που εκμεταλλεύονται τη διάδοση των κυμάτων ηχητικής ενέργειας μέσα σε θαλάσσιες ζώνες (Σχήμα 4.3). Σκοπός των εναλλακτικών αυτών αισθητήρων για την περίπτωση της παρούσας μελέτης είναι αρχικά ο εντοπισμός/ανίχνευση τυχόν ναυαγίων καθώς και η χαρτογράφηση της εκάστοτε περιοχής για τυχόν εμπόδια στη μορφολογία του περιβάλλοντος, όπως βράχια, ύφαλοι κ.α.



Σχήμα 4.3 Σχηματική απεικόνιση λειτουργίας SO.N.A.R [12]

- **Αισθητήρες κίνησης.** Οι συγκεκριμένοι αισθητήρες είναι από τους πιο συνηθισμένοι στη χρήση. Συνήθως πρόκειται για όργανα που δείχνουν την ταχύτητα, την επιτάχυνση και την πορεία του πράκτορα.

ΚΕΦΑΛΑΙΟ 5

Προτεινόμενη υλοποίηση και λειτουργία του συστήματος

Λαμβάνοντας υπ' όψη το θεωρητικό υπόβαθρο, τα χαρακτηριστικά τόσο του πράκτορα διάσωσης όσο και του κόσμου-περιβάλλοντος στο οποίο δρα και εφόσον έχουμε σχεδιάσει μηχανικά τους πράκτορες του συστήματος με σκοπό να πληρούν τις προϋποθέσεις που προαναφέρθηκαν στο προηγούμενο κεφάλαιο, είμαστε έτοιμοι να περάσουμε στο κομμάτι υλοποίησης του πράκτορα και κατ' επέκταση του συστήματος.

Ξεκινώντας θα πρέπει να σημειωθεί ότι η προτεινόμενη υλοποίηση χρησιμοποιεί συγκεκριμένο κόσμο-περιβάλλον και τυχόν αλλαγές σε αυτό ίσως απαιτούν επιπλέον προσθήκες/αναπροσαρμογές για την εξυπηρέτηση του σκοπού του συστήματος.

5.1 Περιγραφή περιβάλλοντος-κόσμου του συστήματος

5.1.1 Χαρακτηριστικά του Συστήματος

Στη συγκεκριμένη εργασία θα εξετάσουμε μια μορφή υλοποίησης για απλοποιημένη περίπτωση περιβάλλοντος. Συγκεκριμένα το περιβάλλον στο οποίο θα εργαστούμε είναι ένας δισδιάστατος κόσμος θαλάσσιων ζωνών. Το περιβάλλον επίσης είναι διακριτό και συγκεκριμένα ο χώρος αποτελείται από τετράγωνα με σκοπό την απλοποίηση του ρεπερτορίου κίνησης των πρακτόρων. Επίσης το περιβάλλον υλοποίησης είναι στρατηγικό και δυναμικό, ο αριθμός των πρακτόρων στο σύστημα είναι τυχαίος αλλά πάντοτε μεγαλύτερος του ενός (εφόσον αναφερόμαστε στην περίπτωση πολυπρακτορικού συστήματος). Ένα τελευταίο στοιχείο για το περιβάλλον είναι η παρατηρησιμότητα του, η οποία όπως τονίσαμε και στο κεφάλαιο 4 θα θεω-

ρηθεί δεδομένη για όλο το σύστημα. Συνεπώς το σύστημά μας θεωρείται πλήρως παρατηρήσιμο.

Συγκεντρωτικά στοιχεία περιβάλλοντος:

- Δυσδιάστατος Κόσμος Θαλάσσιων Ζωνών (2D)
- Διακριτό (Χωρισμένο σε τετράγωνα)
- Στρατηγικό
- Δυναμικό (Μπορεί να μεταβληθεί το μέγεθος του)
- Πολυπρακτορικό (Αριθμός πρακτόρων πάντα μεγαλύτερος του ενός)
- Πλήρως παρατηρήσιμο

5.1.2 Παράμετροι του συστήματος

Στο σύστημα οι παράμετροι όσον αφορά το μέγεθος του περιβάλλοντος διαμορφώνονται από τους πράκτορες για κάθε αλλαγή σε αυτό, πράγμα το οποίο θα συζητήσουμε και στη συνέχεια. Ο αριθμός των προβλημάτων που μπορεί να εμφανιστούν παρόλα αυτά είναι παραμετροποιημένος, υπό την έννοια ότι τα προβλήματα πρέπει να είναι σημαντικά λιγότερα από τον αριθμό των πρακτόρων. Το συγκεκριμένο θέμα θα καλυφθεί επίσης στη συνέχεια του κεφαλαίου με περισσότερη λεπτομέρεια. Τέλος όπως έχει αναφερθεί, το πολυπρακτορικό σύστημα που θα εξετάσουμε απαρτίζεται από πράκτορες οι οποίοι συνεργάζονται έχοντας κοινό στόχο την επίλυση προβλημάτων. Εξ' ορισμού λοιπόν θεωρείται πως δεν εμφανίζονται υπονομεύσεις (**conflicts**) μεταξύ των πρακτόρων του συστήματος ή με άλλα λόγια οι πράκτορες του συστήματος είναι "καλόβουλοι".

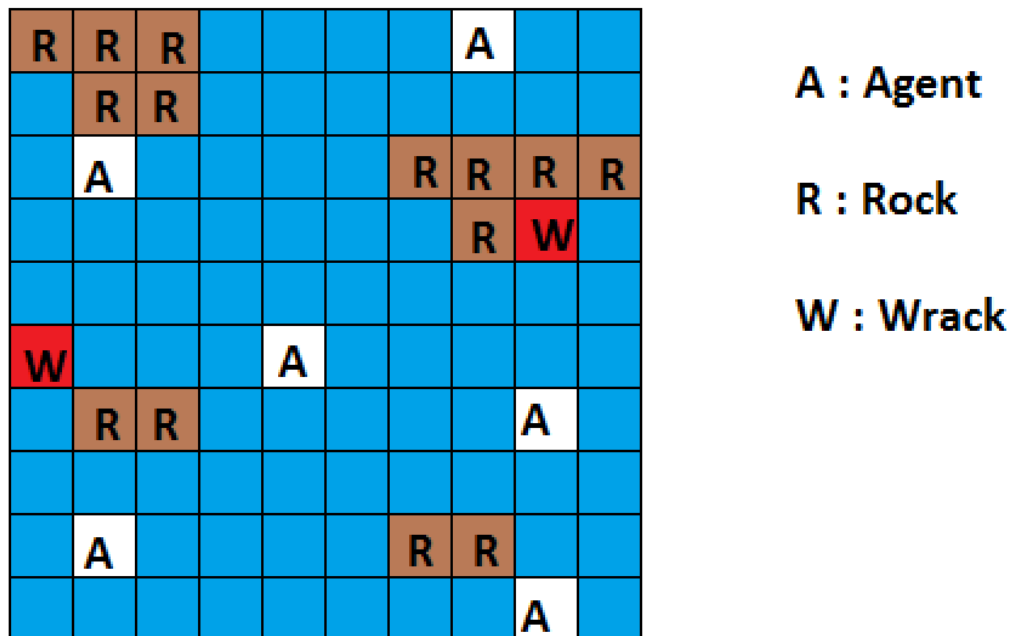
5.1.3 Αντικείμενα και οντότητες του συστήματος

Σε αυτό το σημείο πρέπει να ορισθούν τα αντικείμενα (objects) τα οποία βρίσκονται στο περιβάλλον που ορίσαμε, καθώς και οι οντότητες (entities) οι οποίες θα επιδρούν σε αυτό. Συγκεκριμένα ως οντότητες στο σύστημα μας θεωρούμε τους πράκτορες καθώς και τους ανθρώπους που θα πρέπει να διασωθούν σε περίπτωση ναυαγίου, ενώ αντικείμενα θεωρούνται τα τυχόν εμπόδια μορφολογικής φύσεως τα οποία μπορούμε να συναντήσουμε.

Συγκεντρωτικά οι οντότητες και αντικείμενα του συστήματος:

- **Οντότητες :** Πράκτορες οι οποίοι για χάρη απλοποίησης καταλαμβάνουν ένα τετράγωνο στο διακριτό χώρο που προαναφέραμε. Δεν μπορεί να υπάρξει περίπτωση δυο ή παραπάνω πράκτορες να βρίσκονται στον ίδιο χώρο-τετράγωνο.
- **Αντικείμενα :** Αντικείμενα θα θεωρήσουμε τα εμπόδια μορφολογικής φύσης του περιβάλλοντος, τα οποία ο πράκτορας πρέπει να αποφύγει για να φτάσει στο σημείο του εκάστοτε προβλήματος και τα προβλήματα-ναυάγια. Τα εμπόδια μορφολογικής φύσης μπορούν να καταλαμβάνουν χώρο από ένα έως και περισσότερα τετράγωνα του διακριτού χώρου, ενώ τα ναυάγια καταλαμβάνουν ένα μόνο τετράγωνο στον χώρο αυτό.

Ένα τυχαίο στιγμιότυπο του Συστήματος διαφαίνεται στη συνέχεια στο Σχήμα 5.1, όπου όπως μπορούμε να διακρίνουμε στο σύστημα υπάρχει ένας αριθμός από πράκτορες (Agents) καθώς και αρκετές ζώνες στις οποίες η μορφολογία του εδάφους απαγορεύει στους πράκτορες την πρόσβαση (Rocks). Συνάμα έχουμε εμφάνιση ναυαγίων (Wracks) τα οποία στη συγκεκριμένη αναπαράσταση είναι περισσότερα του ενός. Όπως θα αναλυθεί στη συνέχεια της εργασίας το προτεινόμενο μοντέλο υλοποίησης μπορεί και εξυπηρετεί πολλαπλά προβλήματα-ναυάγια ταυτόχρονα.

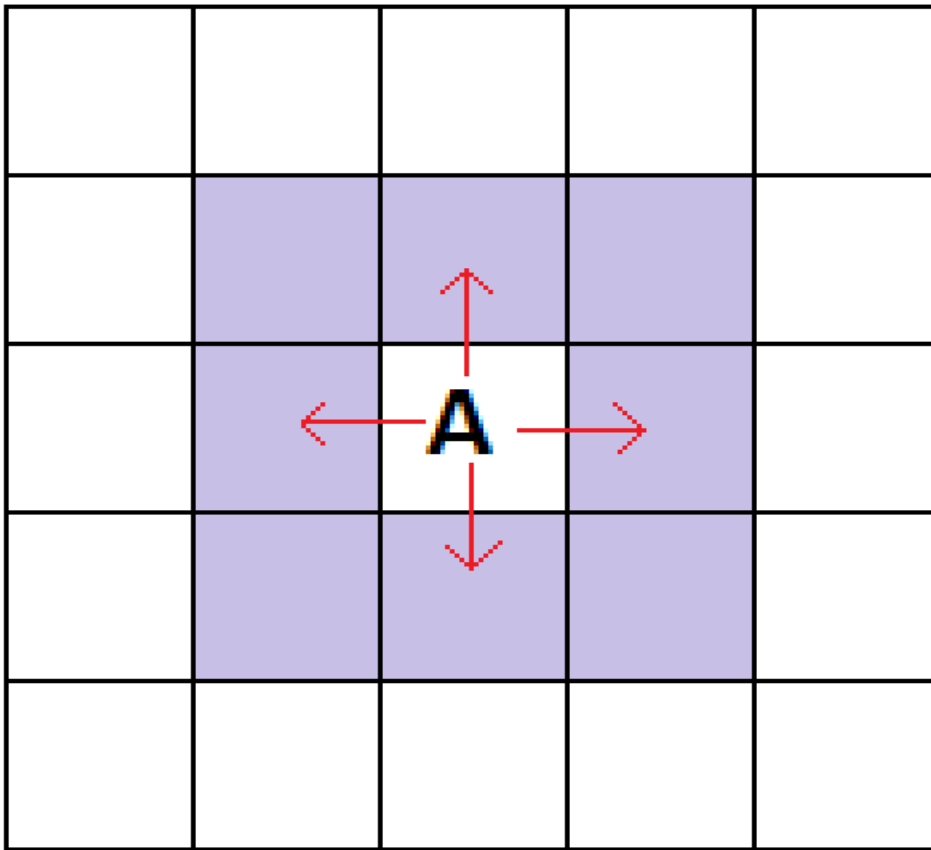


Σχήμα 5.1 Ενδεικτικό-τυχαίο στιγμιότυπο του περιβάλλοντος υλοποίησης

Όπως έχει αναφερθεί στα προηγούμενα κεφάλαια, οι πράκτορες είναι οι μοναδικές οντότητες στο σύστημά μας οι οποίες έχουν την ικανότητα μετακίνησης στο χώρο και την ικανότητα επικοινωνίας.

5.2 Κινησιολογία Πρακτόρων

Οι πράκτορες πρέπει να μπορούν να μετακινηθούν στο χώρο για την εξυπηρέτηση των προβλημάτων που συχνά προκύπτουν. Στη συγκεκριμένη απλοϊκή περίπτωση περιβάλλοντος (τετράγωνο) ο πράκτορας μπορεί να κινηθεί βηματικά, ένα τετράγωνο τη φορά. Συγκεκριμένα οι διαθέσιμες κινήσεις του είναι η περιοχή αντίληψης του κάθε πράκτορα όπως αναφέρθηκε στο κεφάλαιο 4. Ένα παράδειγμα δίδεται σχηματικά (Σχήμα 5.2) παρακάτω για πλήρη κατανόηση των διαθέσιμων ενεργειών κίνησης.



Σχήμα 5.2 Σχηματική απεικόνιση των επιτρεπτών κινήσεων ενός πράκτορα καθώς και σκιαγράφηση του πεδίου το οποίο μπορεί να ανιχνεύσει μέσω αισθητήρων

5.2.1 Ρεπερτόριο Ενεργειών Πράκτορα

Οι ενέργειες ενός πράκτορα απαρτίζονται από δυο κατηγορίες:

- **Απλές δράσεις κίνησης** , οι οποίες είναι όπως είδαμε και στο σχήμα 5.2, η απλή βηματική μετάβαση του πράκτορα σε ένα γειτονικό τετράγωνο. Οι απλές δράσεις κίνησης που επιτρέπεται να κάνει ο πράκτορας είναι οι κάτωθι:
 - **Move up**: Ο πράκτορας θα μετακινηθεί ένα τετράγωνο βόρεια από την τρέχουσα τοποθεσία του.

- **Move down**: Ο πράκτορας θα μετακινηθεί ένα τετράγωνο νότια από την τρέχουσα τοποθεσία του.
 - **Move right**: Ο πράκτορας θα μετακινηθεί ένα τετράγωνο ανατολικά από την τρέχουσα τοποθεσία του.
 - **Move left**: Ο πράκτορας θα μετακινηθεί ένα τετράγωνο δυτικά από την τρέχουσα τοποθεσία του.
- **Πολύπλοκες δράσεις κίνησης** , οι οποίες είναι δράσεις στις οποίες ο πράκτορας καλείται να μετακινηθεί περισσότερα από ένα βήματα. Σε αυτές τις περιπτώσεις η ενέργεια ονομάζεται **Move XY**.

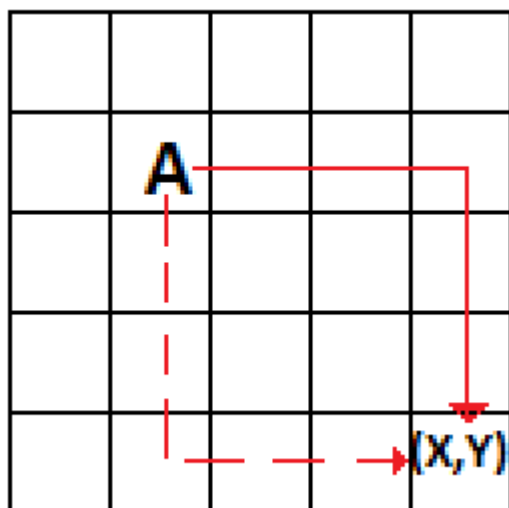
Ενέργεια Move XY

Η συγκεκριμένη εντολή-ενέργεια υποδηλώνει ότι ο πράκτορας θα μετακινηθεί στο τετράγωνο του χώρου με καρτεσιανές συντεταγμένες τις ορισμένες από την ενέργεια (X , Y). Στην προκειμένη περίπτωση αναλύουμε μια σύνθετη ενέργεια στην οποία ο εκάστοτε πράκτορας πρέπει να διαθέτει την ικανότητα να γνωρίζει τις δικές του συντεταγμένες, πράγμα εφικτό με τους αισθητήρες τοποθεσίας-χώρου που διαθέτει (GPS), και με βάση αυτές να υπολογίζει τη διαδρομή που πρέπει να κάνει μέσω των απλών δράσεων κίνησης για να φτάσει στο σημείο που ορίζεται από τις (X,Y) συντεταγμένες. Να σημειωθεί στο σημείο αυτό πως σε περίπτωση φυσικών εμποδίων που τυχόν υπάρχουν στο "δρόμο" προς το σημείο που δίνεται, ο πράκτορας πρέπει μέσω των απλών δράσεων κίνησης να επιλέξει αποφυγή των εμποδίων αυτών.

Μπορούμε εύκολα να ανάγουμε το πρόβλημα σε συνάρτηση η οποία θα έχει εισόδους τα (X,Y) του προορισμού και εμφανίζει ως έξοδο την ακολουθία βημάτων απλής δράσης κίνησης που θα πρέπει να εκτελέσει ο εκάστοτε πράκτορας για να μεταβεί στον προορισμό. Παρόλα αυτά σε περίπτωση εμποδίων η μορφή της συνάρτησης δε λειτουργεί χωρίς την προσθή-

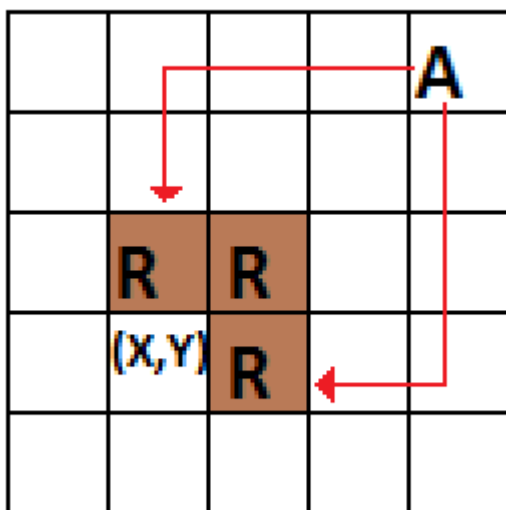
κη των ενδείξεων από τους αισθητήρες του εκάστοτε πράκτορα. Η συνάρτηση λοιπόν θα πρέπει να παραμετροποιηθεί με τρόπο έτσι ώστε να είναι μεν άπληστη (**greedy**) όσον αφορά την ελαχιστοποίηση των απλών κινήσεων αλλά και να αποφεύγει τυχόν εμπόδια ταυτόχρονα.

Το συγκεκριμένο πρόβλημα ανάγεται σε ένα πρόβλημα το οποίο εμφανίζουν όλες οι άπληστες υλοποιήσεις, οι οποίες σε περίπτωση εμποδίων μπορεί να "παγιδευτούν" και οι λύσεις που προτείνουν να μην είναι βέλτιστες. Εάν δεν υπήρχαν φυσικά εμπόδια προφανώς η βέλτιστη υλοποίηση θα ήταν η άπληστη συνάρτηση μετάβασης στον επιθυμητό κόμβο. Δίδεται μια σχηματική απεικόνιση του προβλήματος (Σχήμα 5.3) τόσο για περιβάλλον με εμπόδια όσο και για περιβάλλον χωρίς.



Σχήμα 5.3 Σχηματική απεικόνιση των βέλτιστων διαδρομών που μπορεί να κάνει ένας πράκτορας για την μετάβαση του στον επιθυμητό κόμβο(X,Y) χωρίς ύπαρξη εμποδίων

Στη συγκεκριμένη περίπτωση υπάρχουν δυο βέλτιστες διαδρομές για τη μετάβαση του πράκτορα στο επιθυμητό τετράγωνο. Σε τέτοιες περιπτώσεις, στις οποίες δηλαδή υπάρχουν διαδρομές με ίδιο αριθμό απλών δράσεων κίνησης, τότε ο πράκτορας μπορεί να επιλέξει μια από εκείνες τυχαία. Το ίδιο συμβαίνει και σε περιπτώσεις με εμπόδια όπως στο Σχήμα 5.4 που ακολουθεί.



Σχήμα 5.4 Σχηματική απεικόνιση των βέλτιστων διαδρομών που θα λάμβανε ο πράκτορας στην περίπτωση άνευ εμποδίων

Τέτοια προβλήματα ομοιάζουν με τα προβλήματα που συναντάμε σε αλγορίθμους τοπικής αναζήτησης όπως τον Hill Climbing. Συγκεκριμένα το πρόβλημα ανάγεται στο ότι ο αλγόριθμος αυτός τοπικής αναζήτησης δεν υπολογίζει σε βάθος χρόνου τις αποφάσεις του, αντ' αυτού επιλέγει να δράσει με βάση την καλύτερη απόφαση τη δεδομένη χρονική στιγμή.

Στο Σχήμα 5.4 φαίνεται πως ο πράκτορας θέλει να μετακινηθεί στο τετράγωνο (X,Y) αλλά η ενέργειά του είτε στην πρώτη περίπτωση (**Move down**) είτε στη δεύτερη (**Move left**), δε μπορεί να πραγματοποιηθεί λόγω της ύπαρξης εμποδίου (**rock**). Αυτό οδηγεί τον πράκτορα σε αδιέξοδο. Μόνη λύση στο πρόβλημα είναι η χρήση μνήμης στους πράκτορες καθώς και αισθητήρων (όπως SO.N.A.R) με σκοπό ο πράκτορας να εκτιμά κάθε φορά την ρεαλιστική απόσταση που απέχει από τον προορισμό του.

Με την μνήμη που θα κατέχει για ορισμένα μονοπάτια είναι βέβαιο ότι μπορεί σε μελλοντικές χρήσεις να εμφανίσει ενδεχομένως καλύτερη επιλογή μονοπατιού, δηλαδή συντομότερη. Ο πράκτορας έτσι μπορεί και εμφανίζει σημάδια αυτοβελτίωσης.

- **Ρίψη Σωσιβίων**, είναι η ενέργεια που θα πρέπει να κάνει ένας πράκτορας αφού φτάσει στο σημείο ενδιαφέροντος με σκοπό να αποτρέψει περιπτώσεις πνιγμών. Τα σωσίβια είναι συνδεδεμένα με το πλοίο με σκοπό να μπορούν να "τραβούν" τους διασωθέντες προς το πλοίο-πράκτορα.

- **Ανοιγμα υδραυλικής εισόδου για προσκόμιση των διασωθέντων**, είναι η ενέργεια η οποία έπεται μετά την ρίψη σωσιβίων. Σε αυτή την ενέργεια ανοίγει η καταπακτή και ο αυτόνομος πράκτορας-πλοίο συγκεντρώνει με τη βοήθεια των σωσιβίων τους διασωθέντες.
- **Δράση Αδράνειας (Idle)** , η συγκεκριμένη "δράση" εμπίπτει στην περίπτωση στην οποία οι πράκτορες δεν χρειάζεται να κάνουν τίποτα στον κόσμο, πέραν από τη συγκέντρωση πληροφοριών μέσω των αισθητήρων τους. Η συγκεκριμένη περίπτωση αναφέρεται σε ένα σενάριο περιβάλλοντος στο οποίο δεν υπάρχει ενδεχομένως ναυάγιο ή δεν του ανατίθεται, όπως θα δούμε εν συνεχεία, κάποια δράση.

Προϋποθέσεις Δράσεων

Για την εκτέλεση των παραπάνω ενεργειών από ένα πράκτορα εφόσον αποφασιστεί από το σύστημα, δεν υπάρχουν περαιτέρω δεσμεύσεις, καθώς οι πράκτορες είναι ενεργειακά αυτόνομοι. Η εκτέλεση των ενεργειών παρόλα αυτά πρέπει να αποφασιστεί όπως θα δούμε, από το σύστημα κάθε φορά, το οποίο εμφανίζει ιεραρχική δόμηση.

5.3 Εξυπηρέτηση των προβλημάτων - Λήψη αποφάσεων

Όπως είδαμε σε προηγούμενα κεφάλαια, οι πράκτορες είναι ουσιαστικά αυτόνομα καταναεμημένα συστήματα τα οποία επικοινωνούν με σκοπό τη λήψη απόφασης για την επίλυση ενός συγκεκριμένου προβλήματος. Η επικοινωνία τους γίνεται πάνω από ένα αξιόπιστο δίκτυο στο οποίο υποθέτουμε πως τα μηνύματα φτάνουν με μηδενική καθυστέρηση και χωρίς βλάβες (**reliable multicast**). Το περιβάλλον του συστήματος όπως αναφέραμε είναι δυναμικό, οπότε το δίκτυο θα είναι ταυτόχρονα δυναμικού μεγέθους. Αυτό αποτελεί πρόβλημα στη συγκέντρωση πληροφορίας όπως είδαμε στο κεφάλαιο 2 για μεγάλα όχι peer-to-peer δίκτυα (θυμίζουμε πως στα peer-to-peer δίκτυα οι κόμβοι δεν μπορούν να μετακινηθούν σημαντικά και η δόμηση είναι μη ιεραρχική).

5.3.1 Υλοποίηση Clustering

Στο κεφάλαιο 2 αναφέραμε τα πλεονεκτήματα της ιεραρχικής δόμησης, τα οποία στην περίπτωση της συγκεκριμένης υλοποίησης είναι ζωτικής σημασίας για την βελτιστοποίηση της αποτελεσματικότητας του συστήματος.

Παρόλα αυτά λόγω του περιβάλλοντος, δεν είναι εφικτή η ιεράρχηση ολόκληρου του συστήματος με μόνο ένα διαχειριστή, όπως είναι προφανές. Η συσσώρευση πληροφορίας σε ένα διαχειριστή μπορεί να προκαλέσει καθυστερήσεις οι οποίες μπορεί στο συγκεκριμένο ζήτημα να κοστίσουν τη ζωή σε ανθρώπους. Συνεπώς προτείνεται μια υλοποίηση Clustering η οποία δυναμικά μπορεί και διαχωρίζει το δίκτυο σε νοητά υπο-δίκτυα, στα οποία ορίζονται Coordinators-Clusterheads, οι οποίοι είναι οι διαχειριστές του εκάστοτε υπο-δικτύου.

Η λύση των cluster ανάγεται στις λύσεις προβλημάτων διαχείρισης τοπολογίας με την διαδικασία Divide and Conquer (Διαίρει και Βασίλευε). Το σύστημα των πρακτόρων είναι υπεύθυνο να δημιουργεί κάθε στιγμή τα cluster με βάση τον αλγόριθμο DCA (Distributed Clustering Algorithm).

5.3.2 Distributed Clustering Algorithm (DCA)

Ο DCA αλγόριθμος, θεωρείται από τους πιο βασικούς online αλγόριθμους για την δημιουργία cluster σε ένα δίκτυο. Θεωρείται online διότι για οποιαδήποτε προσθαφαίρεση πρακτόρων ή αυξομείωση του περιβάλλοντος μπορεί να τρέξει από την αρχή και να επανακαθορίσει τα clusters. Όσο εκτελείται ο αλγόριθμος η τοπολογία του συστήματος δεν μπορεί να μεταβληθεί (στην περίπτωση μας οι πράκτορες δεν μπορούν να κινηθούν πριν τη δημιουργία-προκαθορισμό των cluster).

Εξ' ορισμού κάθε κόμβος θα έχει ένα μοναδικό **ID** και ένα μοναδικό Βάρος (**Weight**). Κόμβος του συστήματος στην προκειμένη περίπτωση θεωρείται ο κάθε πράκτορας του συστήματος. **ΔΕΝ** υπάρχουν δυο όμοια βάρη στο σύστημα. Στόχοι του αλγορίθμου είναι:

- Κάθε απλός κόμβος (**Ordinal Node**), να έχει ως "γείτονα" τουλάχιστον ένα Cluster-Head (**CH**).

- Κάθε απλός κόμβος συσχετίζεται με τον γειτονικό ClusterHead που έχει το μεγαλύτερο βάρος (δηλαδή κάθε απλός κόμβος θα είναι μέρος του cluster που δημιουργεί ο ClusterHead γείτονάς του με το μεγαλύτερο βάρος).
- Δύο ClusterHeads δεν μπορούν να γειτνιάζουν.

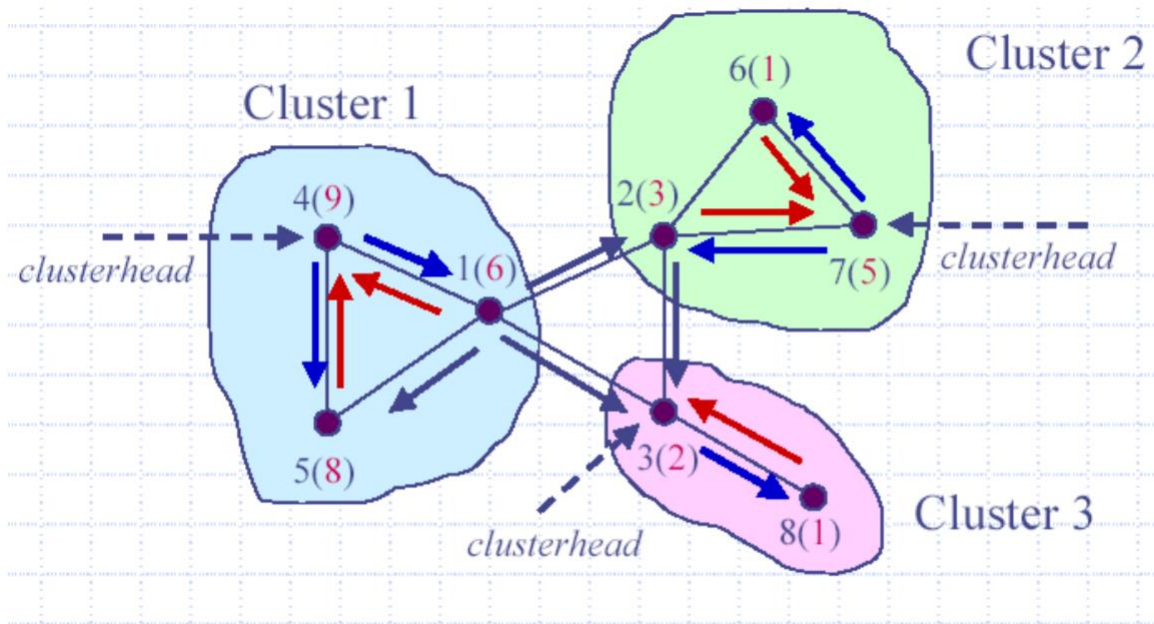
Ο αλγόριθμος εκτελείται γνωρίζοντας το **ID** και το **Weight** του κόμβου, καθώς και τα **Weights** των γειτονικών κόμβων.

Όταν όλοι οι κόμβοι γνωρίζουν τα παραπάνω ξεκινά ο αλγόριθμος. Ο κάθε κόμβος έχει αναλόγως τα βάρη που έχει συλλέξει για εκείνον και τη γειτονιά του μπορεί να δει αν υπάρχει μεγαλύτερο βάρος από το δικό του ή όχι. Αν δεν υπάρχει μεγαλύτερο βάρος στη γειτονιά του τότε στέλνει το μήνυμα **CH(v)**, όπου **v** το **ID** του, στους γείτονές του γνωστοποιώντας τους έτσι ότι είναι ClusterHead. Αυτό συμβαίνει σε όλους τους κόμβους ταυτόχρονα. Σε περίπτωση που το βάρος του κόμβου που εξετάζουμε είναι μικρότερο από κάποιο-α βάρη από γειτονικούς κόμβους, τότε απλά περιμένουμε να σταλθεί μήνυμα **CH(v)** από κάποιον-ους κόμβους της γειτονιάς μας (Αυτό θα συμβαίνει πάντα αφού τα βάρη είναι μοναδικά). Εφόσον λάβουμε μήνυμα **CH** από ένα κόμβο τότε :

- **On Receiving CH** , ελέγχουμε αν έχουμε λάβει μηνύματα από όλους τους γειτονικούς κόμβους οι οποίοι τυχόν έχουν μεγαλύτερο βάρος από τον κόμβο που μας έστειλε το μήνυμα **CH(v)**. Συγκεκριμένα εάν οι γείτονές του με βάρος μεγαλύτερο του **CH(v)** έχουν εκπέμψει και εκείνοι μήνυμα **CH()** τότε ο κόμβος θα κάνει **Join()** στο cluster με ClusterHead αυτόν με το μεγαλύτερο βάρος.
- **On Receiving Join** , όταν ένας κόμβος **v**, ο οποίος έχει στείλει μήνυμα **CH(v)** λαμβάνει ένα μήνυμα **Join(u,v)** από ένα γείτονά του, τότε γνωρίζει ότι εκείνος είναι μέλος του cluster του. Ο αλγόριθμος τερματίζει εφόσον όλοι οι γείτονες κάθε ClusterHead αποστείλουν μήνυμα επιθυμίας ένταξης σε κάποιο cluster. Τότε όλοι οι clusterheads του συστήματος γνωρίζουν τα μέλη του cluster τους.

- **On Receiving Join case:** Εάν ένας κόμβος v ΔΕΝ έχει στείλει μήνυμα **CH(v)** για τον εαυτό του, πριν αποφασίσει για το ρόλο του πρέπει να περιμένει να δει τι αποφάσισαν οι γείτονές του με βάρος μεγαλύτερο από το δικό του. Εάν όλοι αυτοί οι κόμβοι του έχουν στείλει μηνύματα **Join()** τότε αυτό σημαίνει ότι εκείνοι έχουν ενσωματωθεί σε άλλα clusters ως απλοί κόμβοι και πως ο v πλέον έχει το μεγαλύτερο βάρος στη γειτονιά του. Στην περίπτωση αυτή ο κόμβος v θα γίνει ClusterHead κόμβος και θα ακολουθήσει την διαδικασία που ακολουθούν οι Clusterhead κόμβοι για τα μέλη τους που δεν έχουν προσαρτηθεί σε άλλα clusters. Σε περίπτωση που όλοι οι γείτονές του έχουν προσαρτηθεί σε clusters, τότε ο v θα είναι clusterhead με μοναδικό μέλος του cluster του, τον εαυτό του.

Έπεται Σχηματική απεικόνιση του αλγορίθμου για μια απλή περίπτωση τοπολογίας όπου στις παρενθέσεις βρίσκονται τα Βάρη του εκάστοτε κόμβου και δίπλα το ID του (Σχήμα 5.5). Τα διακεκομμένα βέλη μας δείχνουν τους clusterheads για το εκάστοτε cluster, ενώ τα κανονικά βέλη φανερώνουν τα μηνύματα (transactions) που ανταλλάσσονται μεταξύ των κόμβων σε πρώτο ή δεύτερο χρόνο.



Σχήμα 5.5 Σχηματική υλοποίηση του DCA για τη δημιουργία cluster σε ένα σύστημα 8 κόμβων

Αναλύοντας την πολυπλοκότητα του αλγορίθμου μπορούμε να καταλάβουμε ότι σε συνθήκες ρεαλιστικές θα υπάρχουν χρόνοι αναμονής για κόμβους οι οποίοι περιμένουν να λάβουν μηνύματα με τις αποφάσεις των γειτόνων τους. Στην περίπτωση της δικής μας μελέτης, του δικού μας συστήματος, έχουμε υποθέσει ότι τα μηνύματα στέλνονται χωρίς καθυστερήσεις και συνεπώς χωρίς αναμονές. Άρα στο ιδανικό σενάριό ο αλγόριθμος τερματίζει σε χρόνο σχεδόν μηδενικό.

5.3.3 Επικοινωνία των ClusterHead με τα μέλη του Cluster και εξυπηρέτηση των προβλημάτων

Όπως αναφέρθηκε προηγουμένως και εφόσον έχουν δημιουργηθεί τα clusters στο σύστημά μας, το μόνο που απομένει είναι η λήψη απόφασης και η απόκριση του συστήματος σε περιπτώσεις ανάγκης (ναυαγίου).

Το περιβάλλον του συστήματος όπως διευκρινίστηκε στην αρχή του κεφαλαίου θεωρείται πλήρως παρατηρήσιμο, συνεπώς η εμφάνιση ναυαγίου μπορεί να εντοπιστεί άμεσα από το σύστημα, είτε μέσω αισθητήρων από κάποιον πράκτορα σε κοντινή εμβέλεια, είτε από δορυφόρους ή από ενημέρωση του πληρώματος του πλοίου που εμφανίζει το πρόβλημα. Μπορούμε λοιπόν να διαχωρίσουμε την ενημέρωση για τυχόν ναυάγιο σε δυο μεγάλες κατηγορίες :

- Ενημέρωση από εξωγενείς παράγοντες και
- Ενημέρωση από πράκτορες-κόμβους του συστήματος.

Στην περίπτωση ενημέρωσης από εξωγενείς παράγοντες η πληροφορία για την τοποθεσία του ναυαγίου αποστέλλεται στους ClusterHeads απευθείας από τους εξωγενείς παράγοντες, ενώ στο επόμενο σενάριο οι πράκτορες είναι υπεύθυνοι να ενημερώσουν τον ClusterHead του cluster στο οποίο ανήκουν για την τοποθεσία του ναυαγίου.

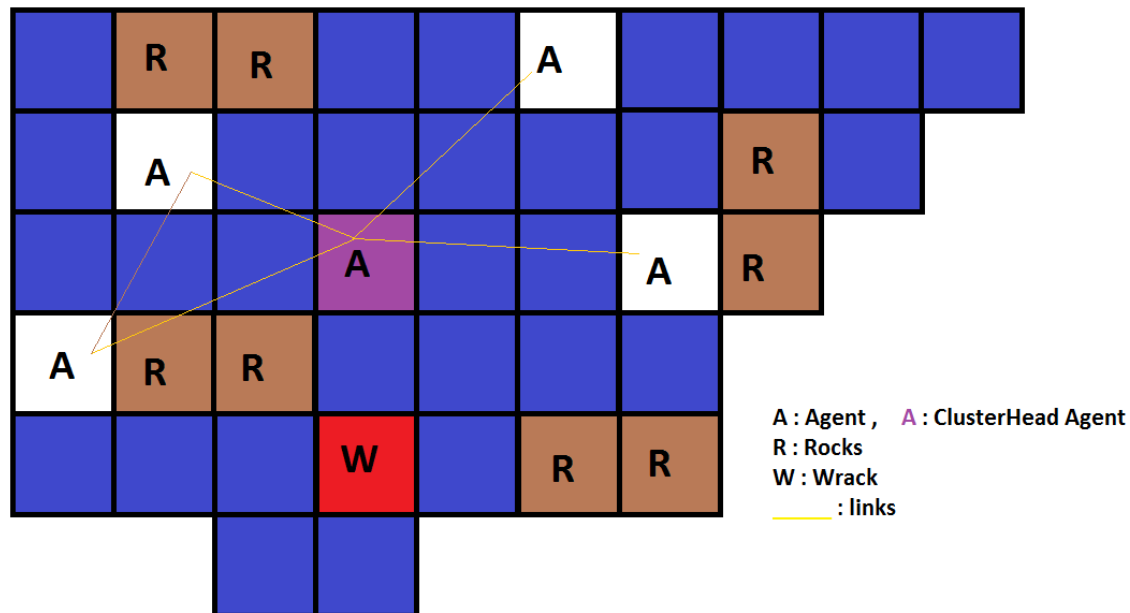
Η επικοινωνία όπως προαναφέρθηκε γίνεται πάνω από δίκτυο αξιόπιστο και με μηδενική χρονοκαθυστέρηση (**delay**).

Ο ClusterHead είναι υπεύθυνος να "αποφασίσει" πιο μέλος του cluster βρίσκεται σε κοντινότερη τοποθεσία με το ναυάγιο. Αυτό γίνεται με μια σειρά ανταλλαγής μηνυμάτων ανάμεσα σε αυτόν και τα μέλη του cluster. Συγκεκριμένα τα βήματα που τηρούνται έχουν ως εξής:

- Εντοπισμός ναυαγίου (όπως είδαμε) και ενημέρωση του Clusterhead στο cluster του οποίου βρίσκεται το ναυάγιο.
- Ο clusterhead λειτουργώντας ως co-ordinator του συστήματος, ζητά από τα μέλη του cluster του να υπολογίσουν πόση απόσταση απέχουν από την τοποθεσία του ναυαγίου (όπως αναφέρθηκε οι πράκτορες διαθέτουν μνήμη και αισθητήρες για αποτελεσματικότερο υπολογισμό απόστασης ακόμη και στην περίπτωση εμποδίων) .
- Οι agents-κόμβοι απαντούν στον Clusterhead. Η απάντηση είναι σε κοινή μονάδα μέτρησης για το σύστημα, όπου στη συγκεκριμένη περίπτωση θεωρείται η απλή δράση κίνησης ή αλλιώς η κίνηση για κάθε τετράγωνο.
- Ο ClusterHead αφού συλλέξει την πληροφορία από το σύνολο των μελών του cluster, μπορεί και συγκρίνει τις απαντήσεις. Επιλέγει τον πράκτορα - κόμβο με την μικρότερη απόσταση από το ναυάγιο και τον ενημερώνει για την αποστολή του.
- Έπειτα και εφόσον ο πράκτορας που έχει επιλεγεί λύσει το πρόβλημα, το cluster αναδιαμορφώνεται λόγω της μετακίνησης του πράκτορα. Συγκεκριμένα ίσως ο κόμβος που μετακινήθηκε να έχει πλέον νέους γειτονικούς κόμβους και να κρίνεται απαραίτητος ο επαναπροσδιορισμός των cluster για το σύστημα.

Επισημαίνεται πως με τη συγκεκριμένη υλοποίηση μπορούν να εξυπηρετηθούν περισσότερα από ένα ναυάγια ταυτόχρονα. Συγκεκριμένα μπορούν να εξυπηρετηθούν από τα μέλη ενός cluster τόσα ναυάγια όσος και ο αριθμός των μελών του.

Μια απλή περίπτωση απεικόνισης του προβλήματος στα πλαίσια ενός cluster παρουσιάζεται εικονικά στο Σχήμα 5.6 που ακολουθεί.



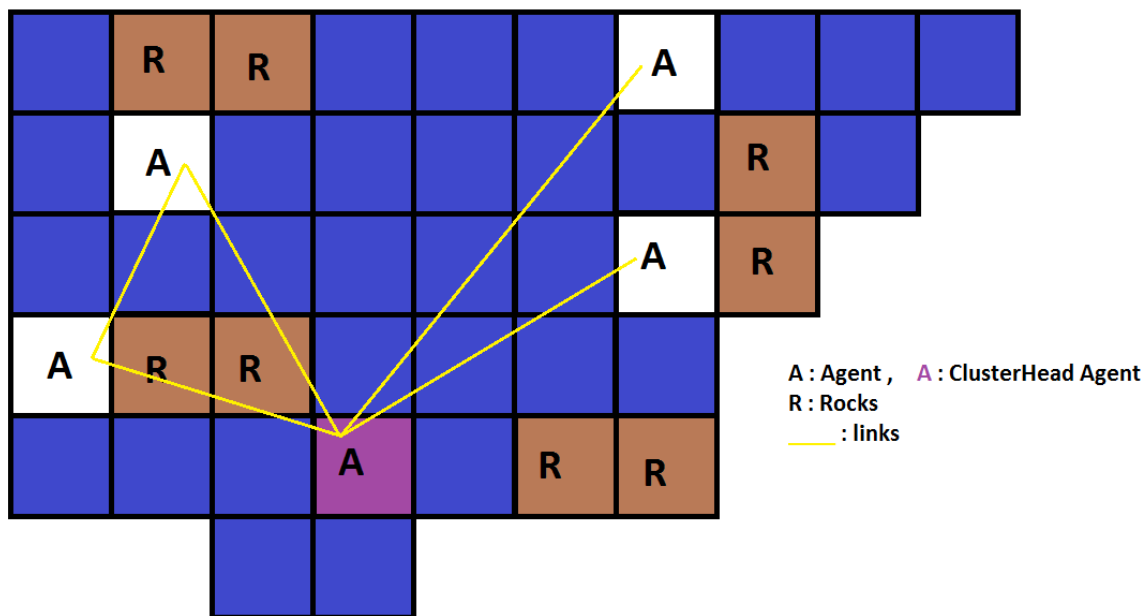
Σχήμα 5.6 Σχηματική απεικόνιση ενός cluster το οποίο έχει δημιουργηθεί με τον DCA αλγόριθμο και των συνδέσεων δικτύου επικοινωνίας μεταξύ των κόμβων του

Η επίλυση του προβλήματος που παρουσιάζεται στο Σχήμα 5.6, σύμφωνα με τα βήματα που παρουσιάστηκαν προηγουμένως θα έχει ως εξής:

- Ενημέρωση του ClusterHead agent για την ακριβή τοποθεσία (X,Y) του ναυαγίου. (σημειώνεται πως σε ρεαλιστικές υλοποιήσεις το περιβάλλον είναι 3D συνεπώς θα χρειαζόμασταν περισσότερες συντεταγμένες)
- Επικοινωνία του ClusterHead με τα μέλη του cluster. Η επικοινωνία καθίσταται εφικτή μέσω των links που διαφαίνονται και έχει σκοπό τη γνωστοποίηση των συντεταγμένων (X,Y) του προβλήματος καθώς και τον επιμέρους υπολογισμό από κάθε μέλος του cluster (συμπεριλαμβανομένου και του ClusterHead agent) της απόστασης από το σημείο ενδιαφέροντος.

- Αφού ΟΛΑ τα μέλη του cluster απαντήσουν την απόσταση σε απλές δράσης κίνησης που απέχουν από το σημείο (X,Y) στον ClusterHead, εκείνος με τη σειρά του αποφασίζει πιο μέλος του Cluster είναι το κατάλληλο για να αποσταλεί στο σημείο του ναυαγίου.
- Μετά την επίλυση του ναυαγίου έχουμε επανεκκίνηση του DCA αλγορίθμου για το-τα εκάστοτε cluster. (όπως είπαμε μπορεί να εξυπηρετηθεί αριθμός προβλημάτων τόσο στα πλαίσια του cluster όσο και σε διαφορετικά cluster παράλληλα)

Στο συγκεκριμένο παράδειγμα ο ClusterHead αφού συγκεντρώσει την πληροφορία θα μπορέσει να καταλάβει πως στο σημείο (X,Y) του ναυαγίου βρίσκεται πιο κοντά εκείνος, οπότε και θα μετακινηθεί για να το επιλύσει (Σχήμα 5.7). Όπως λοιπόν καταλαβαίνουμε η υλοποίηση των πρακτόρων ανεξαρτήτως ρόλου παραμένει ίδια, με μόνη διαφορά να εμφανίζεται στον τρόπο με τον οποίο αποφασίζει το σύστημα το πως θα δράσει συλλογικά.



Σχήμα 5.7 Σχηματική απεικόνιση του περιβάλλοντος του cluster μετά την επίλυση του προβλήματος

Όπως παρατηρούμε η τοπολογία του cluster αναδιαμορφώνεται μετά την εξυπηρέτηση προβλημάτων. Επίσης είναι προφανές πως οι διασωθέντες πρέπει να προσκομιστούν σε ασφαλές μέρος, συνεπώς οι πράκτορες που επιλύουν τα προβλήματα εγκαταλείπουν το cluster τους και πηγαίνουν στη βάση, εκτός συστήματος, για το σκοπό αυτό. Τα links και το cluster αλλάζουν. Συνεπώς για οποιαδήποτε αφαίρεση Agent από cluster και για οποιαδήποτε προσθήκη agent στο σύστημα (ανεξαρτήτως cluster στο οποίο θα προσαρτηθεί), είναι προφανές πως θα δημιουργηθούν νέα links και πως πρέπει να τρέξει από την αρχή ο αλγόριθμος DCA για τον ορισμό νέων cluster.

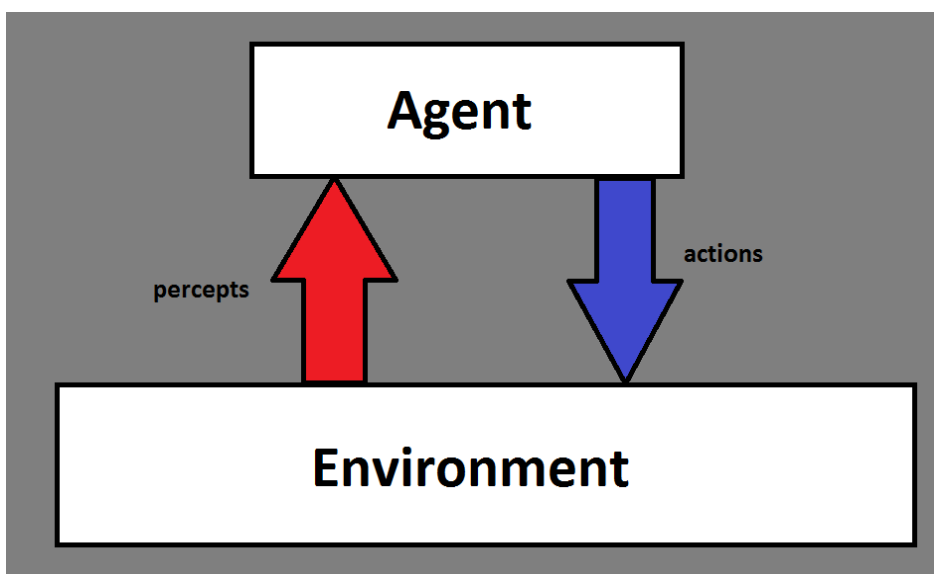
5.3.4 Περιπτώσεις ναυαγίων

Σε ορισμένες περιπτώσεις πολλαπλών ναυαγίων, υπάρχει η πιθανότητα τα ναυάγια να βρίσκονται κοντά σε έναν πράκτορα του συστήματος, ενώ οι υπόλοιποι να απέχουν αρκετά από αυτά. Εάν τα ναυάγια εντοπισθούν ταυτόχρονα από το σύστημα τότε ο ClusterHead είναι υπεύθυνος να ορίσει το βέλτιστο (δηλαδή) τον ελάχιστο αριθμό βημάτων που απαιτείται από οποιοδήποτε δυνατό συνδυασμό πρακτόρων για να τα επιλύσει. Σε περίπτωση που οι αποστάσεις είναι ίδιες μεταξύ διαφορετικών κόμβων η επιλογή γίνεται όπως και προηγουμένως τυχαία. Σε περίπτωση που ένα ναυάγιο εντοπιστεί ενώ ένας πράκτορας έχει ήδη αποφασισθεί πως θα καλύψει μια άλλη περίπτωση ναυαγίου τότε το σύστημα δεν τον υπολογίζει, δηλαδή ο clusterhead θα θεωρήσει πως ο πράκτορας είναι κατειλημμένος και δε θα λάβει υπ' όψη την απόστασή του από το νέο-α ναυάγιο.

5.4 Πρόγραμμα Πράκτορα

Ως πρόγραμμα ενός πράκτορα του συστήματος θεωρούμε το λογισμικό σύμφωνα με το οποίο εκείνος πραγματοποιεί τα βήματα που απαιτούνται για τη συγκεκριμένη υλοποίηση. Ταυτόχρονα το πρόγραμμα του πράκτορα όπως είδαμε, θα πρέπει να είναι κατάλληλο να μεταφράσει τα ερεθίσματα από τους αισθητήρες που διαθέτει καθώς και να εμφανίζει δυνατότητα

μνήμης με σκοπό την βελτίωση των επιλογών του. Αυτή η ικανότητα δίνει στον πράκτορα τη δυνατότητα να διαθέτει μια εσωτερική κατάσταση (**internal state**), η οποία του επιτρέπει να έχει επίγνωση για το περιβάλλον γύρω του. Μπορούμε να χαρακτηρίσουμε λοιπόν το πρόγραμμα του πράκτορα ως μια συνάρτηση σύμφωνα με την οποία ο πράκτορας βάσει των ερεθισμάτων που αποκτά μπορεί και εξάγει αποτελέσματα-ενέργειες οι οποίες μεταβάλλουν το περιβάλλον γύρω του (Σχήμα 5.8).



Σχήμα 5.8 Ο πράκτορας υλοποιημένος ως συνάρτηση

Όπως διαφαίνεται το συγκεκριμένο τμήμα του πράκτορα δεν αποτελεί μέρος της αρχιτεκτονικής του δομής.

$$\text{Πράκτορας} = \text{Αρχιτεκτονική} + \text{Πρόγραμμα}$$

Οι πράκτορες που χρησιμοποιούνται από τη συγκεκριμένη υλοποίηση είναι πράκτορες βασισμένοι στη χρησιμότητα και εμπίπτουν στην κατηγορία **BDI** (Belief - Desire - Intention) πρακτόρων. Όπως αναφέραμε και στο κεφάλαιο 2, οι πράκτορες αυτοί εξασφαλίζουν την επίτευξη του στόχου με τον βέλτιστο δυνατό τρόπο. Στη συγκεκριμένη περίπτωση Βέλτιστος Τρόπος επίτευξης του στόχου θεωρείται η εύρεση συντομότερου μονοπατιού και κατ' επέκταση η ελαχιστοποίηση του χρόνου απόκρισης από το σημείο του ναυαγίου. Το **utility function** λοιπόν των πρακτόρων του συστήματος θα καθορίζει πόσο επιθυμητή είναι μια κατάσταση ή όχι με βάση τη βελτιστοποίηση που αναφέρθηκε.

5.4.1 Βάση γνώσης πρακτόρων

Βάση γνώσης ενός πράκτορα ονομάζεται η δομή η οποία εμπεριέχει τη γνώση που κατέχει ο πράκτορας για το περιβάλλον του. Στη βάση γνώσης των πρακτόρων μπορούμε να βρούμε πληροφορίες τόσο για τον ίδιο τον πράκτορα (**ID-Weight, role, position etc**) όσο και μοντέλα για το πως αντιλαμβάνεται ο ίδιος το περιβάλλον πριν ή και μετά την επίδραση του σε αυτό. Στην περίπτωση των πρακτόρων του συστήματος μας, η βάση γνώσης όπως είναι προφανές θα έχει μια σύνθετη δομή βάσης δεδομένων καθώς συμπεριλαμβάνει ποικίλες αντιλήψεις, καταστάσεις και γενικότερα δεδομένα για τον πράκτορα και το περιβάλλον του.

Μερικά εκ των πεδίων που θα συμπεριλαμβάνονται και θα αποθηκεύονται στη δομή αυτή των πρακτόρων που αφορούν το πολυπρακτορικό σύστημά μας είναι τα παρακάτω:

- **ID** , όπως καταλαβαίνουμε το προσωπικό αναγνωριστικό του κάθε πράκτορα, δηλαδή με άλλα λόγια η ταυτότητά του, ακριβώς λόγω της χρησιμότητάς που παρουσιάζει (Clustering , διαχωρισμός πρακτόρων κλπ) πρέπει να αποθηκεύεται σε μια δομή εσωτερικά του πράκτορα, έτσι ώστε να μπορεί να ανακτηθεί όταν χρειαστεί.
- **Weight** , όπως και στην περίπτωση του ID, το βάρος του πράκτορα είναι απαραίτητο και αναπόσπαστο κομμάτι της συγκεκριμένης υλοποίησης για το τρέξιμο του αλγορίθμου DCA και τη δημιουργία των cluster.
- **Role** , η θέση του πράκτορα υπό ιεραρχικής σκοπιάς, είναι απαραίτητη για την αναγωγή ρόλων στους πράκτορες, καθώς άλλη αρμοδιότητα παρουσιάζουν οι clusterhead agents και άλλη οι ordinal nodes agents του συστήματος.
- **Position** , η θέση του πράκτορα στο χώρο ανά πάσα στιγμή πρέπει να είναι γνωστή. Λόγω της μετακίνησης των πρακτόρων πρέπει να κρατείται ιστορικό μνήμης (όπως έχει τονιστεί) τόσο για την ακρίβεια της τοποθεσίας μετά από κίνηση, όσο και για την εύρεση ελάχιστων μονοπατιών (αυτοβελτίωση). Η θέση του πράκτορα στη συγκεκριμένη υλοποίηση δηλώνεται με τη χρήση καρτεσιανών συντεταγμένων (X,Y).

- **Cluster Limits** , είναι τα όρια που πρέπει να γνωρίζει κάθε μέλος ενός cluster με σκοπό να μη τα ξεπερνά αλλά και να μην αναφέρει τυχόν προβλήματα που μπορεί να εντοπίσει και δεν ανήκουν στο cluster. Τα όρια του κάθε cluster σχηματίζονται μετά την δημιουργία των cluster και η πληροφορία πρέπει να υπάρχει σε όλους τους πράκτορες που το απαρτίζουν.
- **Current state** , είναι η κατάσταση στην οποία ο πράκτορας έχει μια συγκεκριμένη αντίληψη για το περιβάλλον γύρω του και την αποκτά μέσω των αισθητήρων του. Μπορεί να αναπαρασταθεί με μια μορφή πίνακα στον οποίο θα σημειώνονται τα χαρακτηριστικά του περιβάλλοντος την συγκεκριμένη χρονική στιγμή.
- **Goal state** , είναι αντίστοιχα η επιθυμητή κατάσταση περιβάλλοντος για τους πράκτορες. Στην περίπτωση μας είναι η εξάλειψη των προβλημάτων που εμφανίζονται.
- **Distance_Function** , είναι η συνάρτηση σύμφωνα με την οποία κάθε πράκτορας μπορεί, εφόσον του αιτηθεί, να υπολογίζει (με τη βοήθεια και της μνήμης "Position") πόσο απέχει από ένα σημείο στο cluster.

Στην παραπάνω δομή δε χρειάζεται να συμπεριληφθεί ο μηχανισμός επικοινωνίας μεταξύ των πρακτόρων (links) καθώς θα θεωρηθεί, για τους σκοπούς της συγκεκριμένης μελέτης γνωστός και δεδομένος. Σε ρεαλιστικά σενάρια όπως προαναφέρθηκε τόσο μηχανισμοί επικοινωνίας καθώς και τυχόν καθυστερήσεις ή βλάβες (π.χ. βυζαντινές βλάβες) που μπορεί να εμφανιστούν στο σύστημα πρέπει να αναφέρονται και να επιλύονται. Συνεπώς θα πρέπει να σημειωθεί πως μελετάμε μια απλοποιημένη μορφή συστήματος με περιορισμούς. Η τεχνική αυτή, της αναγωγής του προβλήματος σε σενάρια απλοϊκά, συνηθίζεται στους σχεδιαστές συστημάτων. Ξεκινώντας από απλές μορφές προβλημάτων μπορούμε σταδιακά να χτίσουμε νέες εκδοχές με μεγαλύτερη πολυπλοκότητα και ανοχή σε περισσότερα ρεαλιστικά σενάρια.

ΚΕΦΑΛΑΙΟ 6

Πλατφόρμα υλοποίησης του συστήματος

Συνοψίζοντας όλα όσα αναφέρθηκαν στο προηγούμενο κεφάλαιο σε επίπεδο υλοποίησης τόσο των πρακτόρων όσο και της λειτουργίας του συστήματος, δημιουργήθηκε ένα βασικό πρόγραμμα προσομοίωσης σε περιβάλλον Python. Η συγκεκριμένη πλατφόρμα παρέχει γραφικές αναπαραστάσεις και χρησιμοποιήθηκε με σκοπό την απεικόνιση του κόσμου, μιας ενδεικτικής γραφικής αναπαράστασης του πολυπρακτορικού συστήματος (Σχήμα 6.1) και της δράσης του σε περιπτώσεις ναυαγίων.

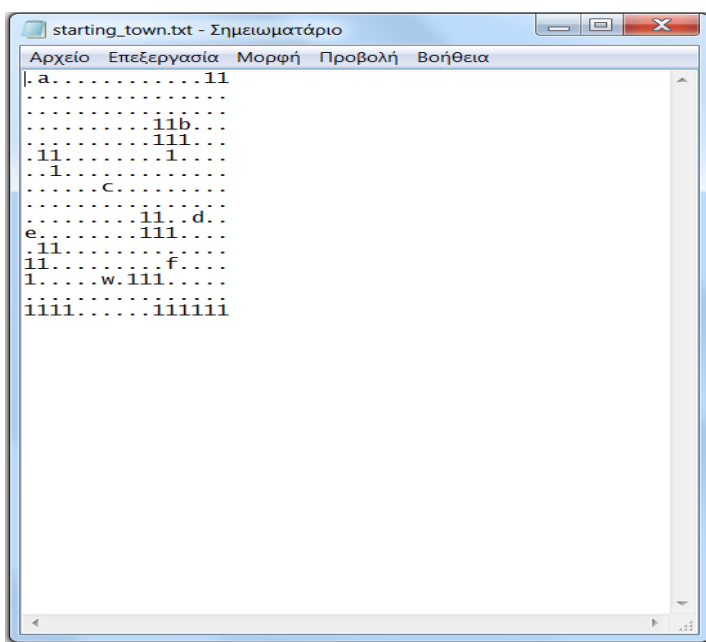


Σχήμα 6.1 Στιγμιότυπο προσομοίωσης του συστήματος

6.1 Μοντέλα συστημάτων βασισμένα σε πράκτορες

Τα ευφυή συστήματα τα οποία απαρτίζονται από αυτόνομους πράκτορες εμφανίζουν δυσκολίες στη δομή και την απεικόνιση, καθώς συνηθίζεται να χρησιμοποιούνται δομές πινάκων από τους σχεδιαστές τους, προκειμένου να προσδιορίζεται η ακριβής τοπολογία του συστήματος. Για τις ανάγκες της παρούσας διπλωματικής εργασίας και εφόσον εξετάζουμε σύστημα πολλών πρακτόρων, δημιουργήθηκε μια ενδεικτική πλατφόρμα προσομοίωσης υλοποιημένη σε python. Τέτοιου είδους προσομοιώσεων είναι γνωστές ως μοντέλα συστημάτων βασισμένα σε πράκτορες και προσομοιώνουν τη συμπεριφορά των πρακτόρων σε ένα εικονικό παραμετροποιήσιμο περιβάλλον για ένα είδος προβλήματος-ων. Στα πλαίσια της τεχνητής νοημοσύνης η εικονική προσομοίωση τέτοιων συστημάτων είναι απαραίτητη για τους σχεδιαστές τους, με σκοπό την περαιτέρω κατανόηση των ιδιοτήτων καθώς και την εξαγωγή συμπερασμάτων περί της συμπεριφοράς των πρακτόρων που το απαρτίζουν.

Ένα εξίσου σημαντικό σχεδιαστικό στοιχείο που πρέπει να ληφθεί υπ' όψη για τον σχεδιασμό τέτοιων προσομοιώσεων, είναι η ακρίβεια αναφορικά με τις θέσεις των αντικειμένων και των οντοτήτων που υπάρχουν σε αυτό. Συγκεκριμένα για την παρούσα πλατφόρμα χρησιμοποιήθηκε ως είσοδος (input) για την κατάσταση του συστήματος, ένας παραμετροποιήσιμος δισδιάστατος πίνακας (Σχήμα 6.2) που όπως αναφέρθηκε είναι και αυτός που χρησιμοποιείται κατά κόρον από την πληθώρα των σχεδιαστών. Η μορφή της εισόδου είναι η εξής :



Σχήμα 6.2 Απεικόνιση της εισόδου για την προσομοίωση ενός στιγμιότυπου του συστήματος

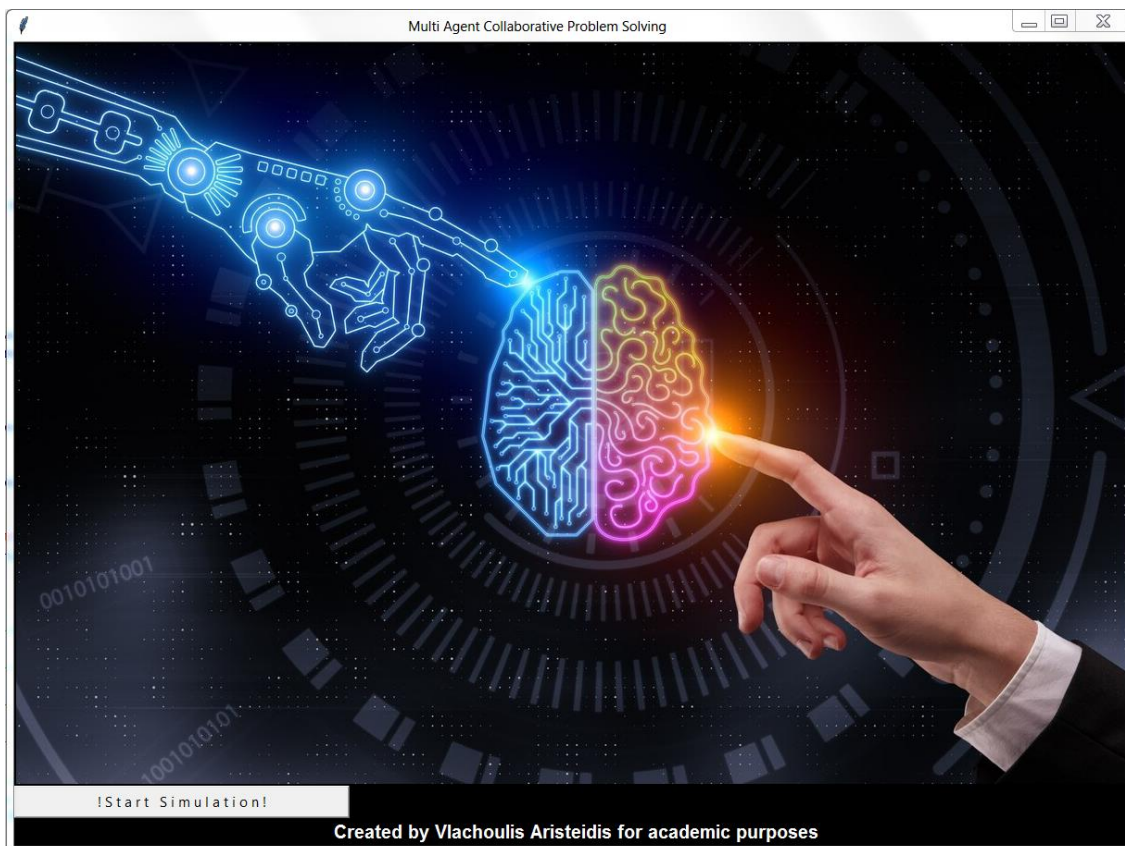
Η είσοδος είναι παραμετροποιήσιμη, καθώς οποιαδήποτε αλλαγή στο αρχείο .txt θα διαφανεί στην απεικόνιση. Συγκεκριμένα έχουμε δημιουργήσει ένα περιβάλλον στο οποίο εμφανίζεται ένα cluster του συστήματος. Οι πράκτορες παρουσιάζονται ως γράμματα του αγγλικού αλφάβητου με **ID** (a, b, c, d, e, f etc), τα φυσικά εμπόδια (στην εν λόγω περίπτωση βράχια) συμβολίζονται με τον αριθμό 1, ενώ ο στόχος, ο οποίος στην προκειμένη περίπτωση είναι η εξυπηρέτηση των ναυαγίων (wracks), συμβολίζεται με "w". Να τονιστεί επίσης πως οι τελείες (.) στο αρχείο είναι η κωδικοποίηση για την γραφική απεικόνιση του υδάτινου περιβάλλοντος στο οποίο το σύστημα λαμβάνει χώρα. Όπως διαφαίνεται, ο εν λόγω πίνακας έχει διαστάσεις 16 επί 16 ενώ όπως προαναφέρθηκε μπορεί να αναπροσαρμοστεί τόσο σε μορφολογία (θέση πρακτόρων, εμποδίων ή ναυαγίου) όσο και σε μέγεθος, με την προσθήκη επιπλέον σειρών και στηλών. Η απεικόνιση του συστήματος όπως αυτό παρουσιάζεται στο Σχήμα 6.2 , δίνεται από το Σχήμα 6.1 στην αρχή του κεφαλαίου.

6.2 Γλώσσα προγραμματισμού και εργαλεία υλοποίησης πλατφόρμας

Όπως προαναφέρθηκε η συγκεκριμένη υλοποίηση σχεδιάστηκε και προγραμματίστηκε σε περιβάλλον γλώσσας προγραμματισμού Python. Αξίζει να αναφερθεί πως απαραίτητη προϋπόθεση για τη λειτουργία της πλατφόρμας είναι η εισαγωγή της βιβλιοθήκης pygame [13], της βιβλιοθήκης numpy [14] καθώς και της tkinter [15]. Συγκεκριμένα η pygame, η υλοποίηση της οποίας ανάγεται σε αυτή της θεωρίας ανάπτυξης παιγνίων, είναι μια βιβλιοθήκη της python η οποία χρησιμοποιείται για την απεικόνιση ενός κόσμου, η numpy με τη σειρά της είναι μια βιβλιοθήκη η οποία διευκολύνει πράξεις και στην παρούσα υλοποίηση χρησιμοποιήθηκε για πράξεις πινάκων καθώς και υπολογισμό αποστάσεων. Τέλος η tkinter χρησιμοποιήθηκε για τη δημιουργία του γραφικού περιβάλλοντος διεπαφής του χρήστη (Game User Interface). Τέλος, τα γραφικά που χρησιμοποιήθηκαν επιλέχθηκαν από την ιστοσελίδα OpenGameArt.org [16] .

6.3 Οδηγίες πλατφόρμας προσομοίωσης

Η πλατφόρμα εκκινεί τρέχοντας το αρχείο στο οποίο έχει σχεδιαστεί. Συγκεκριμένα κατά την εκτέλεση του αρχείου το πρώτο παράθυρο που εμφανίζεται στο χρήστη είναι το κάτωθι :



Σχήμα 6.3 Παράθυρο διεπαφής εισόδου του χρήστη στην πλατφόρμα

Στο αρχικό παράθυρο, Σχήμα 6.3, ο εκάστοτε χρήστης πρέπει να πιάσει το κουμπί της έναρξης με σκοπό να εισέλθει στον χώρο-περιβάλλον της προσομοίωσης της οποίας έχει επιλέξει με βάση τα στοιχεία που έχει θέσει στο σύστημα (βλέπε Σχήμα 6.2).

Έπειτα, εφόσον τρέξει η προσομοίωση το συγκεκριμένο παράθυρο χάνεται και τη θέση του παίρνει το παράθυρο στο οποίο αναπαρίσταται το εκάστοτε περιβάλλον που έχει ορίσει ο χρήστης (Σχήμα 6.1).

Υλοποίηση προγράμματος

Ο αλγόριθμος που χρησιμοποιήθηκε στη συγκεκριμένη πλατφόρμα, είναι ο αλγόριθμος που αναλυτικά παρουσιάστηκε στο Κεφάλαιο 5 ως προτεινόμενη υλοποίηση.

Χειρισμός πράκτορα

Ο χειρισμός πράκτορα γίνεται με τη χρήση χειροκίνητης ρητής εντολής από το χρήστη. Παρόλα αυτά, το σύστημα όπως είδαμε και στο Κεφάλαιο 5, μπορεί και επιλέγει κάθε φορά τον πράκτορα ο οποίος θα εμφανίζει τη μέγιστη χρησιμότητα, για να επιλύσει το πρόβλημα και στη συγκεκριμένη περίπτωση επιλέγεται από το σύστημα ο πράκτορας ο οποίος απέχει μικρότερη απόσταση από το ναυάγιο. Αναλυτικά το σύστημα είναι υπεύθυνο να ορίσει τον πράκτορα που θα πρέπει να λύσει το πρόβλημα. Έπειτα ο χρήστης είναι υπεύθυνος να τον μετακινήσει στην θέση που εμφανίζεται το πρόβλημα. Η μετακίνηση αυτή πραγματοποιείται ως εξής :

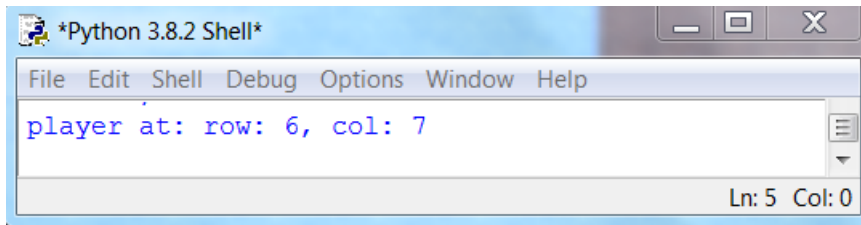
- **Move Up** : Χρήση πλήκτρου ↑
- **Move Down** : Χρήση πλήκτρου ↓
- **Move Left** : Χρήση πλήκτρου ←
- **Move Right** : Χρήση πλήκτρου →

Η προσομοίωση ακολουθεί βηματική κίνηση του πράκτορα με τη βοήθεια του χρήστη έως τη στιγμή που ο πράκτορας φτάνει στο σημείο του ναυαγίου. Να σημειωθεί πως εάν η κίνηση που ο χρήστης θα επιβάλλει στον επιλεγόμενο από το σύστημα πράκτορα δεν είναι επιτρεπτή, πχ σύγκρουση με βράχια, τότε ο πράκτορας παραμένει στάσιμος έως να του επιβληθεί επιτρεπτή κίνηση.



Σχήμα 6.4 Στιγμιότυπο του περιβάλλοντος μετά τη μετακίνηση του πράκτορα στο ναυάγιο

Για είσοδο που φαίνεται στο Σχήμα 6.2, η αρχική κατάσταση του κόσμου παρουσιάζεται στο Σχήμα 6.1, ενώ η κατάσταση τερματισμού (επιθυμητή κατάσταση) φαίνεται στο Σχήμα 6.4. Το τερματικό του συστήματος το οποίο τρέχει παράλληλα με τη χρήση της προσομοίωσης, μας ενημερώνει για τις κινήσεις του πράκτορα στο χώρο καθώς και για την επίτευξη του στόχου (Σχήμα 6.4). Η μορφή της ενημέρωσης μέσω του τερματικού για το συγκεκριμένο παράδειγμα βήμα-βήμα είναι η εξής :



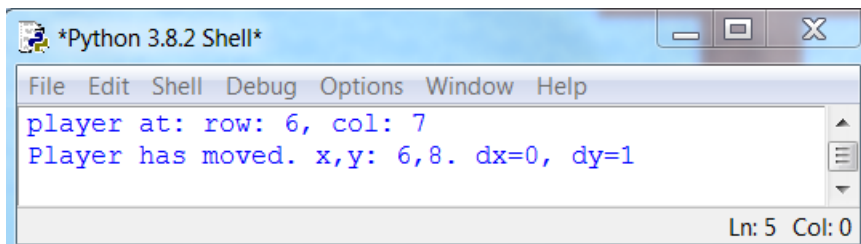
Σχήμα 6.5 Αρχική παρουσίαση θέσης πράκτορα μέσω τερματικού

Εικονικά:



Σχήμα 6.6 Σχηματική απεικόνιση του πράκτορα στο σημείο που κατονομάζει το τερματικό

Με την μετακίνηση του πράκτορα από τον χρήστη το τερματικό και συνάμα η απεικόνιση του πράκτορα στον κόσμο αλλάζει. Στα σχήματα που ακολουθούν (Σχήμα 6.7, 6.8, 6.9, 6.10, 6.11, 6.12, 6.13, 6.14, 6.15, 6.16) εμφανίζεται η πληροφορία μετά από μετακινήσεις του πράκτορα προς το ναυάγιο, τόσο σε μορφή τερματικού όσο και γραφικά.



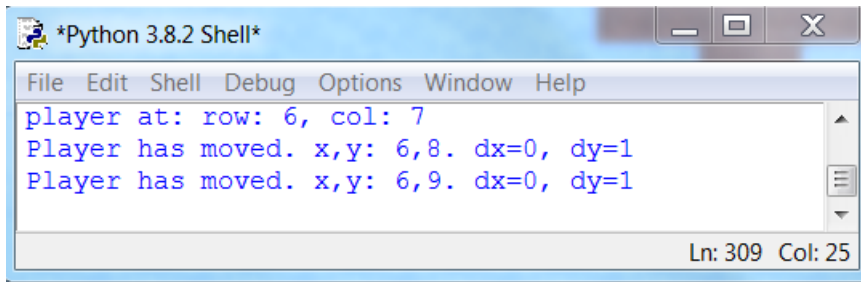
```
*Python 3.8.2 Shell*
File Edit Shell Debug Options Window Help
player at: row: 6, col: 7
Player has moved. x,y: 6,8. dx=0, dy=1
Ln: 5 Col: 0
```

Σχήμα 6.7 Παρουσίαση θέσης πράκτορα μετά από 1 μετακίνηση μέσω τερματικού

Εικονικά:



Σχήμα 6.8 Σχηματική απεικόνιση πράκτορα μετά την 1η μετακίνηση



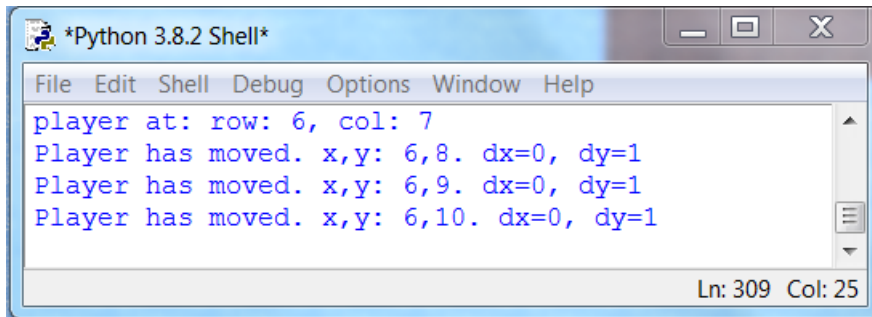
```
*Python 3.8.2 Shell*
File Edit Shell Debug Options Window Help
player at: row: 6, col: 7
Player has moved. x,y: 6,8. dx=0, dy=1
Player has moved. x,y: 6,9. dx=0, dy=1
Ln: 309 Col: 25
```

Σχήμα 6.9 Παρουσίαση θέσης πράκτορα μετά από 2 μετακινήσεις μέσω τερματικού

Εικονικά:



Σχήμα 6.10 Σχηματική απεικόνιση πράκτορα μετά τη 2η μετακίνηση



```
*Python 3.8.2 Shell*
File Edit Shell Debug Options Window Help
player at: row: 6, col: 7
Player has moved. x,y: 6,8. dx=0, dy=1
Player has moved. x,y: 6,9. dx=0, dy=1
Player has moved. x,y: 6,10. dx=0, dy=1
Ln: 309 Col: 25
```

Σχήμα 6.11 Παρουσίαση θέσης πράκτορα μετά από 3 μετακινήσεις μέσω τερματικού

Εικονικά:



Σχήμα 6.12 Σχηματική απεικόνιση πράκτορα μετά την 3η μετακίνηση


```
*Python 3.8.2 Shell*
File Edit Shell Debug Options Window Help
player at: row: 6, col: 7
Player has moved. x,y: 6,8. dx=0, dy=1
Player has moved. x,y: 6,9. dx=0, dy=1
Player has moved. x,y: 6,10. dx=0, dy=1
Player has moved. x,y: 6,11. dx=0, dy=1
Ln: 309 Col: 25
```

Σχήμα 6.13 Παρουσίαση θέσης πράκτορα μετά από 4 μετακινήσεις μέσω τερματικού

Εικονικά:



Σχήμα 6.14 Σχηματική απεικόνιση πράκτορα μετά την 4η μετακίνηση

```
*Python 3.8.2 Shell*
File Edit Shell Debug Options Window Help
player at: row: 6, col: 7
Player has moved. x,y: 6,8. dx=0, dy=1
Player has moved. x,y: 6,9. dx=0, dy=1
Player has moved. x,y: 6,10. dx=0, dy=1
Player has moved. x,y: 6,11. dx=0, dy=1
Player has moved. x,y: 6,12. dx=0, dy=1
Ln: 309 Col: 25
```

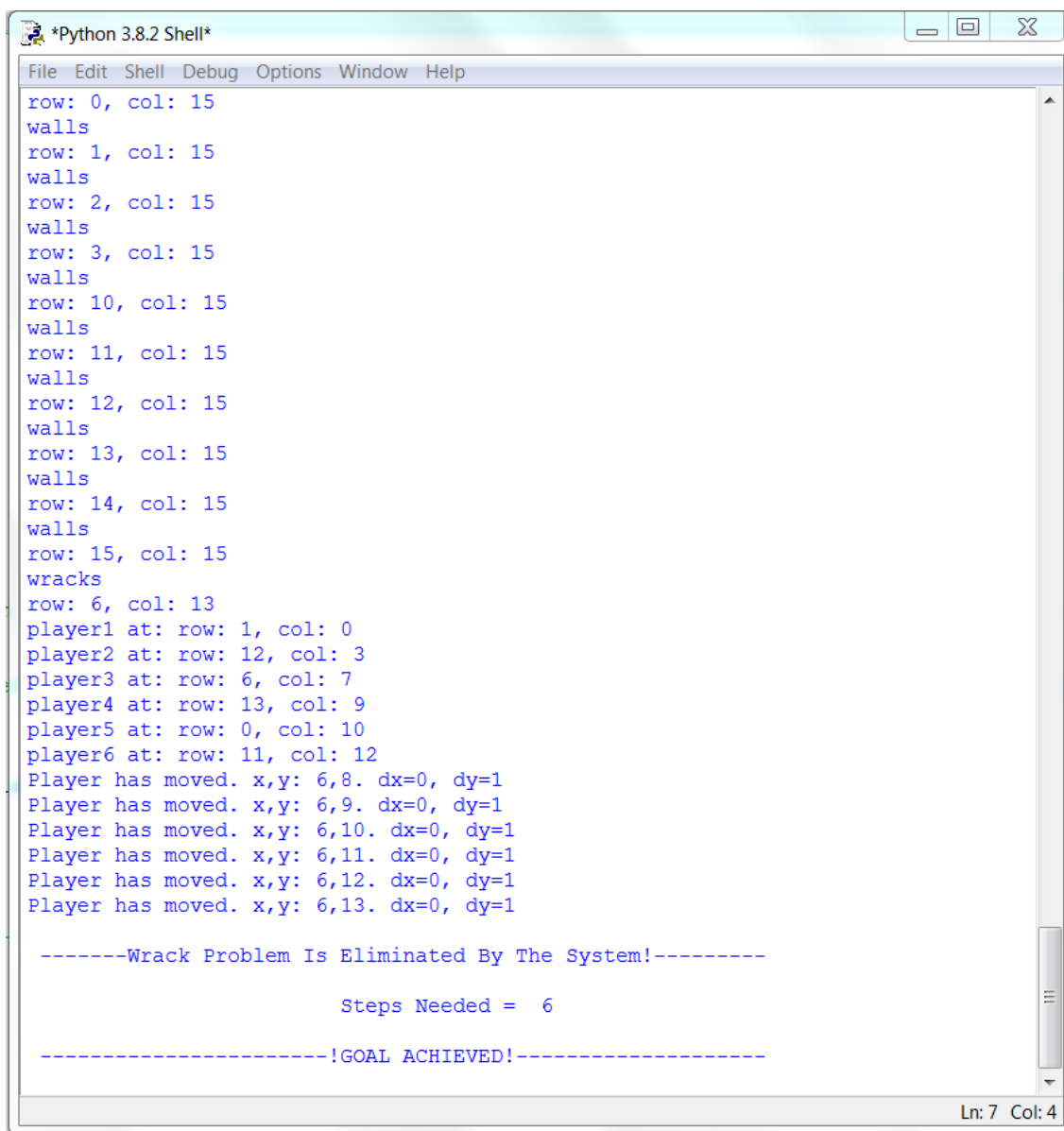
Σχήμα 6.15 Παρουσίαση θέσης πράκτορα μετά από 5 μετακινήσεις μέσω τερματικού

Εικονικά:



Σχήμα 6.16 Σχηματική απεικόνιση πράκτορα μετά την 5η μετακίνηση

Όπως βλέπουμε και στο Σχήμα 6.16, ο πράκτορας στην επόμενη μετακίνηση θα φτάσει στο στόχο του. Η εφαρμογή στο επόμενο βήμα εμφανίζει ολόκληρη την πληροφορία τόσο για τον πράκτορα που επιλέχθηκε, την τοποθεσία των εμποδίων, την τοποθεσία των άλλων πρακτόρων καθώς και την τοποθεσία του ναυαγίου όπως θα δούμε στο Σχήμα 6.17 που ακολουθεί:



```
*Python 3.8.2 Shell*
File Edit Shell Debug Options Window Help
row: 0, col: 15
walls
row: 1, col: 15
walls
row: 2, col: 15
walls
row: 3, col: 15
walls
row: 10, col: 15
walls
row: 11, col: 15
walls
row: 12, col: 15
walls
row: 13, col: 15
walls
row: 14, col: 15
walls
row: 15, col: 15
wracks
row: 6, col: 13
player1 at: row: 1, col: 0
player2 at: row: 12, col: 3
player3 at: row: 6, col: 7
player4 at: row: 13, col: 9
player5 at: row: 0, col: 10
player6 at: row: 11, col: 12
Player has moved. x,y: 6,8. dx=0, dy=1
Player has moved. x,y: 6,9. dx=0, dy=1
Player has moved. x,y: 6,10. dx=0, dy=1
Player has moved. x,y: 6,11. dx=0, dy=1
Player has moved. x,y: 6,12. dx=0, dy=1
Player has moved. x,y: 6,13. dx=0, dy=1

-----Wrack Problem Is Eliminated By The System!-----

                Steps Needed = 6

-----!GOAL ACHIEVED!-----
Ln: 7 Col: 4
```

Σχήμα 6.17 Σχηματική απεικόνιση του τερματικού του συστήματος κατά την λειτουργία της πλατφόρμας

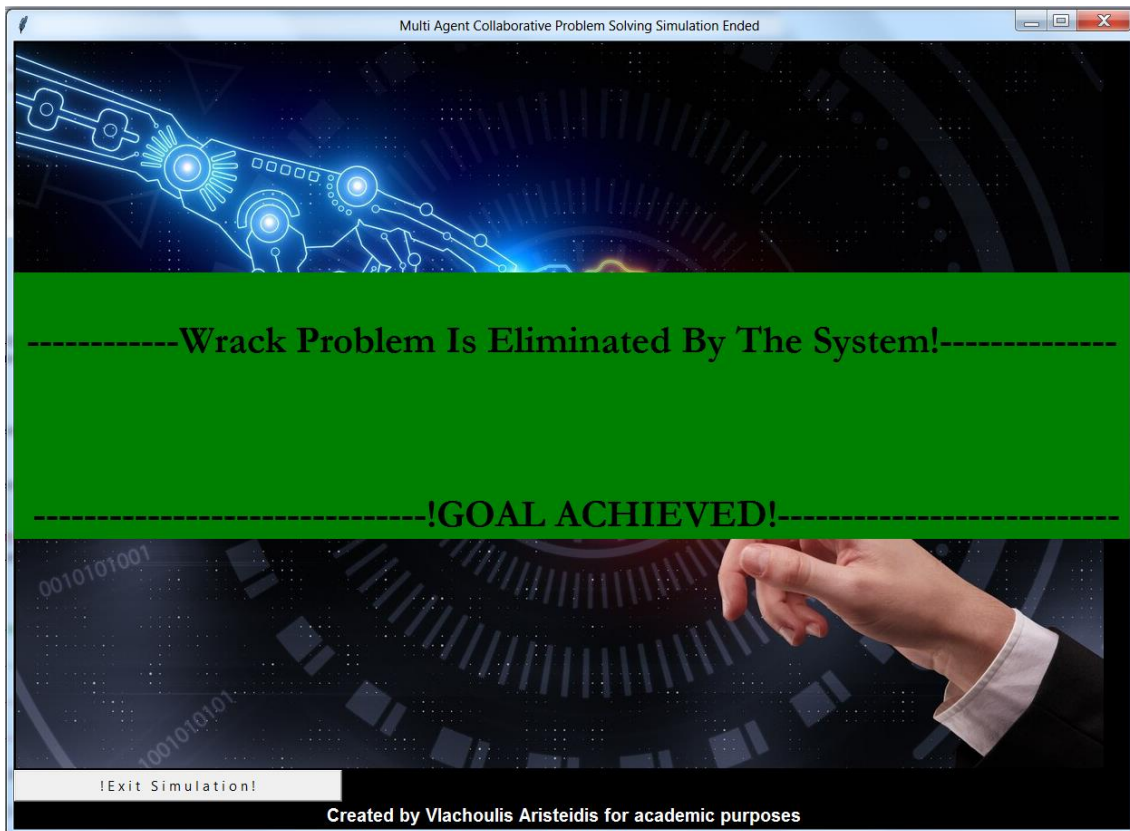
Εικονικά:



Σχήμα 6.18 Σχηματική απεικόνιση πράκτορα μετά την 6η μετακίνηση

Όπως βλέπουμε, στο τερματικό **εμφανίζεται όλη η πληροφορία** καθώς ο πράκτορας μετακινείται από το χρήστη. Εφόσον ο πράκτορας φτάσει στο σημείο του ναυαγίου (wreck), τότε το πρόγραμμα θα εμφανίσει στο τερματικό το παραπάνω μήνυμα, όπως φαίνεται στο Σχήμα 6.17. Στο μήνυμα αυτό **διαφαίνεται ο βέλτιστος αριθμός βημάτων (απόσταση)** που χρειάστηκε να κάνει ο πράκτορας.

Έπειτα κλείνει το παράθυρο απεικόνισης του περιβάλλοντος ενώ στη συνέχεια η εφαρμογή θα εμφανίσει το τελευταίο παράθυρο, που είναι το παράθυρο εξόδου (Σχήμα 6.19) .



Σχήμα 6.19 Παράθυρο διεπαφής εξόδου του χρήστη από την πλατφόρμα

Στο συγκεκριμένο παράθυρο εξόδου (Σχήμα 6.19) ο χρήστης έχει εξέλθει από το περιβάλλον του συστήματος και για να εξέλθει πλήρως από το περιβάλλον της πλατφόρμας καλείται να τερματίσει την προσομοίωση πατώντας το αντίστοιχο κουμπί.

6.4 Αποτελέσματα υλοποίησης

Όπως αναφέρθηκε και στο κεφάλαιο 5, η υλοποίηση ευφυών συστημάτων πολλών πρακτόρων με ομαδοποίηση και συνεργασία, στην περίπτωση μας μέσω clustering, εμφανίζει βέλτιστες λύσεις όσον αφορά κατανομή εργασίας και γενικά την επίτευξη στόχου. Η υλοποίηση της πλατφόρμας μας δίνει μια πιο ρεαλιστική εικόνα του προβλήματος που αντιμετωπίζει το σύστημα.

ΚΕΦΑΛΑΙΟ 7

Επίλογος

Έπεται μια σύνοψη των σημαντικότερων θεμάτων που αναλύθηκαν στη συγκεκριμένη διπλωματική εργασία. Τέλος, θα παρουσιαστούν ορισμένα παραδείγματα και ιδέες για εν δυνάμει μελλοντικές επεκτάσεις της μελέτης αυτής.

7.1 Σύνοψη και συμπεράσματα

Στην παρούσα διπλωματική εργασία μελετήθηκε η πολυπρακτορική συνεργατική επίλυση προβλημάτων. Συγκεκριμένα αναλύθηκε εις βάθος η περίπτωση των ναυτικών επιχειρήσεων διάσωσης όπου και αναφέρθηκαν παραστατικά όλες οι δυσκολίες και οι αρχές που θα πρέπει να τηρούνται σε τέτοιες περιπτώσεις (Κεφάλαιο 3). Εν συνεχεία παρουσιάστηκαν όλα τα κριτήρια και οι προϋποθέσεις οι οποίες θα πρέπει να πληρούνται για το σχεδιασμό λειτουργικών πρακτόρων, έτσι ώστε αυτοί να μπορούν να δρουν σε ένα περιβάλλον, όπως αυτό των ναυτικών επιχειρήσεων διάσωσης (Κεφάλαιο 4). Έπειτα στο Κεφάλαιο 5 προτάθηκε μια υλοποίηση υπό τη μορφή ενός συστήματος, στην οποία οι πράκτορες θα σχηματίζουν "ομάδες" (cluster) με ιεραρχική δόμηση, με σκοπό τη βέλτιστη εξυπηρέτηση του στόχου. Επίσης στο Κεφάλαιο 5, παραθέσαμε όλες τις λεπτομέρειες αρχιτεκτονικής σχεδίασης και σχεδίασης λογισμικού τόσο των πρακτόρων ατομικά αλλά και σε συλλογικό επίπεδο. Τέλος στο Κεφάλαιο 6, παρουσιάστηκε η πλατφόρμα προσομοίωσης που δημιουργήθηκε με σκοπό την αποτελεσματικότερη απεικόνιση του συστήματος καθώς και της προτεινόμενης υλοποίησης.

Τα συμπεράσματα που μπορούμε να εξάγουμε από την μελέτη αυτή είναι πως η αντιμετώπιση προβλημάτων σε ένα σύστημα πολλών ομογενών πρακτόρων, οι οποίοι εμφανίζουν κοινούς στόχους (έχουν δηλαδή κοινές επιθυμητές καταστάσεις), βελτιστοποιείται με τη δημιουργία "ομάδων" πρακτόρων. Αποδείχθηκε λοιπόν πως το παραπάνω ισχύει για σενάρια όπου η επι-

κοινωνία, ο συντονισμός και η ομαλή λειτουργία (δίχως βλάβες) μεταξύ των πρακτόρων είναι δεδομένη. Επίσης όπως παρουσιάστηκε, τα ναυάγια πάντα θα εξυπηρετούνται και μάλιστα βέλτιστα, δηλαδή ο στόχος του συστήματος υλοποίησης πάντα καλύπτεται από το σύστημα. Οι έννοιες και τα αποτελέσματα που εξάγουμε από αυτά τα ευφυή συστήματα λοιπόν, δεν θα πρέπει να εμφανίζουν αντίκτυπο μόνο υπό τη σκοπιά της τεχνητής νοημοσύνης, αλλά θα πρέπει να αποτελούν και γνώση για τον ίδιο τον άνθρωπο, ώστε να δρα αντίστοιχα σε παρόμοιες καταστάσεις.

7.2 Μελλοντικές επεκτάσεις

Ορισμένες μελλοντικές επεκτάσεις της παρούσας μελέτης είναι :

- Μελέτη περιστατικών διάσωσης για διαφορετικά περιβάλλοντα, με διαφορετικές απαιτήσεις, στόχους και δράσεις (πχ εκκενώσεις πόλεων σε περιπτώσεις καταστροφών από πυρηνικά ατυχήματα, σεισμούς, φωτιές κ.α.).
- Μελέτη της παρούσας θέσης για συνθήκες μη ιδανικές, αλλά ρεαλιστικές (βλάβες, μη αξιόπιστη επικοινωνία κλπ)
- Βελτίωση πλατφόρμας προσομοίωσης με σκοπό να εμφανίζει περισσότερα από ένα cluster πρακτόρων και εξυπηρέτηση πολλαπλών ναυαγίων

Βιβλιογραφία

- [1] Stuart Russel & Peter Norvig, “Artificial Intelligence: A Modern Approach”, 3rd edition.
- [2] Jacques Ferber, “Multi-Agent System: An Introduction to Distributed Artificial Intelligence”, ISBN 0-201-36048-9, 1999.
- [3] HECTOR GARCIA-MOLINA, “Elections in a Distributed Computing System”, IEEE TRANSACTIONS ON COMPUTERS, VOL C-31, NO.1, JANUARY 1982.
- [4] “Introduction to Distributed Systems”, Copyright 2007, Google, the content of this page is licensed under the Creative Commons Attribution 2.5 License. Available Online: https://pages.cs.wisc.edu/~zuyu/files/dist_systems.pdf , accessed February, 2020.
- [5] “Basic Networking Tutorial”, Compiled by Sangay Yeshi. Available Online: <https://www.mowhs.gov.bt/wp-content/uploads/2011/08/What-is-a-computer-Network.pdf>, accessed February, 2020.
- [6] G. Cormode, S. Muthukrishnan and W. Zhuang, "Conquering the Divide: Continuous Clustering of Distributed Data Streams," *2007 IEEE 23rd International Conference on Data Engineering*, Istanbul, 2007, pp. 1036-1045.
doi: 10.1109/ICDE.2007.368962.
- [7] Basagni, Stephano, “On the complexity of clustering multi-hop wireless networks.” Technical Report, UTD/EE-01-98, 2009.
- [8] “IAI : Building Intelligent Agents”, Available Online: <https://www.cs.bham.ac.uk/~jxb/IAI/w4.pdf>, accessed February, 2020.

[9] Available Online: <https://steelguru.com/logistic/eco-marine-power-applauds-introduction-of-poseidon-principles/542738> , accessed February,2020.

[10] ROBERT THIBADEAU, “Artificial Perception of Actions”, *COGNITIVE SCIENCE* 10, 117-149 (1986).

[11] Bajcsy, R., Aloimonos, Y., Tsotsos, J.K. “Revisiting active perception”, *Auton Robot* 42, 177–196, (2018).

[12] SO.N.A.R, Available Online:
<https://el.wikipedia.org/wiki/%CE%A3%CF%8C%CE%BD%CE%B1%CF%81>,
accessed February,2020.

[13] Pygame, Available Online: <https://www.pygame.org/>, accessed February,2020.

[14] Numpy, Available Online: <https://numpy.org/>, accessed February,2020.

[15]Tkinter, Available Online: <https://wiki.python.org/moin/TkInter>, accessed February,2020.

[16] OpenGameArt, Available Online: <https://opengameart.org/>, accessed February,2020.

Συντομογραφίες

κλπ	και λοιπά
κ.α.	και άλλα
πχ	παραδείγματος χάριν
GPS	Geographic Position System
DCA	Destributed Clustering Algorithm
SO.N.A.R	SOund Navigation And Ranging
PEAS	Performance Environment Actuators Sensors
BDI	Belief Desire Intention

Ορολογία - Γλωσσάρι

Ελληνικός Όρος

Αγγλικός Όρος

πράκτορας	agent
περιβάλλον	environment
πρόγραμμα πράκτορα	agent program
απλός αντανακλαστικός	πράκτορας simple reflex agent
πράκτορας με μοντέλο	model based agent
πράκτορας βασισμένοι σε στόχο	goal-based agent
πράκτορας χρησιμότητας	utility-based agent
πράκτορας μάθησης	learning agent
ελάχιστη ενιαία περιοχή αντίληψης	minimum united ROP
αντικείμενο περιβάλλοντος	environment objects
οντότητα περιβάλλοντος	environment entity
βάση γνώσης	knowledge base
δίκτυα βασισμένα σε εξυπηρετητή	server based model
δίκτυα χωρίς ιεραρχική δομή	peer to peer
διαχειριστής	coordinator - clusterhead
βυζαντινές βλάβες	byzantine fault
κατανεμημένα συστήματα	distributed systems
ναυάγιο	wrack
βράχια	rocks
λογικοί πράκτορες	rational agents
αισθητήρες	sensors
μηχανισμοί δράσης	actuators
αντιλήψεις πράκτορα	percepts

ενδιάμεσο κομμάτι λογισμικού

λογισμικό

υλικό

υπονομεύσεις

άπληστη συνάρτηση

ιδιωτικό αναγνωριστικό

βάρος

διαίρει και βασίλευε

implementation-middleware

software

hardware

conflicts

greedy function

ID

weight

divide and conquer