Ατομική Διπλωματική Εργασία

ΚΑΤΑΝΕΜΗΜΕΝΗ ΣΥΜΦΩΝΙΑ ΜΕ ΕΓΩΙΣΤΙΚΟΥΣ ΠΡΑΚΤΟΡΕΣ

Θεοδώρα Μενελάου

ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΥΠΡΟΥ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Μάιος 2018

ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΥΠΡΟΥ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Κατανεμημένη Συμφωνία με Εγωιστικούς Πράκτορες

Θεοδώρα Μενελάου

Επιβλέποντες Καθηγητές

Μάριος Μαυρονικόλας

Χρύσης Γεωργίου

Άννα Φιλίππου

Η Ατομική Διπλωματική Εργασία υποβλήθηκε προς μερική εκπλήρωση των απαιτήσεων απόκτησης του πτυχίου Πληροφορικής του Τμήματος Πληροφορικής του Πανεπιστημίου Κύπρου

Μάιος 2018

Ευχαριστίες

Η ολοκλήρωση της διπλωματικής μου εργασίας σηματοδοτεί και το τέλος των προπτυχιακών μου σπουδών και θα ήθελα να ευχαριστήσω κάποια άτομα που ήταν δίπλα μου όλα αυτά τα χρόνια.

Αρχικά, θα ήθελα να ευχαριστήσω τον κ. Μάριο Μαυρονικόλα, που ως επιβλέποντας καθηγητής της διπλωματικής εργασίας μου, με βοήθησε με την καθοδήγηση και τις πολύτιμες συμβουλές του. Καθ'όλη τη διάρκεια της προσπάθειας μου με στήριξε, προτείνοντας λύσεις σε τυχόν προβλήματα που προκύπταν, ακούγοντας τις ιδέες και τις προτάσεις μου και δείχνοντας πάντα ενδιαφέρον αλλά και εμπιστοσύνη στη δουλειά μου. Η συνεργασία μαζί του μου έμαθε πολλά και συνέβαλε στο να ανταποκριθώ με τον καλύτερο δυνατό τρόπο στις ανάγκες της παρούσας εργασίας.

Επίσης, θα ήθελα να ευχαριστήσω θερμά τον κ. Χρύση Γεωργίου και την κα. Άννα Φιλίππου για την επιπρόσθετη βοήθεια που μου πρόσφεραν καθ'όλη την διάρκεια της εκπόνησης της διπλωματικής μου εργασίας, δίνοντας μου σημαντικές συμβουλές έτσι ώστε να καταστεί δυνατή η ολοκλήρωση της εργασίας αυτής.

Αναμφισβήτητα, η σημαντικότερη βοήθεια προήλθε από την οικογένειά μου. Πάντα στο πλευρό μου, στις ευχάριστες αλλά κυρίως στις δύσκολες στιγμές, δεν σταμάτησαν να μου δείχνουν την αγάπη, την πίστη και την υποστήριξή τους όλα αυτά τα χρόνια των σπουδών μου.

Ασφαλώς ένα μεγάλο ευχαριστώ οφείλω και στους φίλους μου, οι οποίοι ήταν πάντα δίπλα μου και στη διάθεσή μου όταν τους είχα ανάγκη, πρόθυμοι να με ακούσουν, να με συμβουλεύσουν και να με ενθαρρύνουν.

Περίληψη

Το πρόβλημα της Κατανεμημένης Συμφωνίας, όπου n επεξεργαστές, μερικοί από τους οποίους μπορεί να είναι ελαττωματικοί, πρέπει να συμφωνήσουν στην ίδια τιμή έχει μελετηθεί πολύ. Πιο πρόσφατα, υπάρχει ένα αυξανόμενο ενδιαφέρον για το μοντέλο του κατανεμημένου υπολογισμού με εγωιστικούς πράκτορες, οι οποίοι μπορεί να αποκλίνουν από έναν κατανεμημένο αλγόριθμο προκειμένου να επηρεάσουν το αποτέλεσμα του αλγορίθμου. Η κύρια ανησυχία σχετικά με τον κατανεμημένο υπολογισμό είναι να παρέχει έναν αλγόριθμο, ο οποίος να είναι ανθεκτικός σε εγωιστικούς πράκτορες, πράγμα που σημαίνει ότι οι ορθολογικοί πράκτορες δεν θα κερδίσουν τίποτα αποφεύγοντας το πρωτόκολλο με σκοπό να ξεγελάσουν κάποια ομάδα πρακτόρων της επιλογής τους.

Μια πρώτη μελέτη του προβλήματος της κατανεμημένης συμφωνίας με τους εγωιστικούς πράκτορες έγινε από τους Afek et al., οι οποίοι προτείνουν έναν αλγόριθμο που παρέχει μια $(n-1)$ - Ισχυρή - Ισορροπία, καθώς και μια $(n-2)$ - Ισχυρή - Ισορροπία σε ένα σύγχρονο δακτύλιο, όπου οι εισόδοι των πρακτόρων ακολουθούν την ομοιόμορφη κατανομή.

Στην παρούσα εργασία, μελετήθηκε ένα συγκεκριμένο μοντέλο κατανεμημένης λήψης αποφάσεων με την σχεδίαση τριών αλγορίθμων που βασίζονται στον αλγόριθμο των Afek et al. Ο πρώτος αλγόριθμος παρέχει μια $(n-2)$ - Ισχυρή - Ισορροπία σε ένα σύγχρονο δακτύλιο με την υπόθεση ότι οι εισόδοι των πρακτόρων δεν ακολουθούν την ομοιόμορφη κατανομή. Ο δεύτερος αλγόριθμος, παρέχει μια $(n-2)$ - Ισχυρή - Ισορροπία σε ένα ασύγχρονο δακτύλιο, με ομοιόμορφη κατανομή στις εισόδους των πρακτόρων. Τέλος, ο τρίτος αλγόριθμος, παρέχει μια $(n-2)$ - Ισχυρή - Ισορροπία σε ένα ασύγχρονο δακτύλιο με την υπόθεση ότι οι εισόδοι των πρακτόρων δεν ακολουθούν την ομοιόμορφη κατανομή. Η υλοποίηση του τρίτου αλγορίθμου βασίζεται στην υλοποίηση του δεύτερου με κάποιες μικρές αλλαγές τόσο στο σχεδιασμό του αλγορίθμου όσο και στην απόδειξη της ορθότητας και εγκυρότητάς του. Και στους τρείς αλγορίθμους, ο αριθμός των πρακτόρων n είναι ένας ζυγός αριθμός, καθώς πρόκειται για μια $(n-2)$ - Ισχυρή - Ισορροπία και όπως θα δούμε και παρακάτω, μια τέτοια Ισορροπία δεν επιτυγχάνεται όταν το n είναι μονός αριθμός, καθώς παραβιάζεται η συνθήκη εγκυρότητας των τριών πιο πάνω αλγορίθμων.

Περιεχόμενα

Κεφάλαιο **1** **Εισαγωγή**..... **1**

1.1 Κίνητρο 1

1.2 Στόχος Ατομικής Διπλωματικής Εργασίας 2

1.3 Μεθοδολογία 3

1.4 Οργάνωση 4

Κεφάλαιο **2** **Θεωρητικό Υπόβαθρο**..... **7**

2.1 Τεχνολογία Νοήμονων Πρακτόρων 8

2.1.1 Εισαγωγή στους πράκτορες 8

2.1.2 Ορισμός 9

2.1.3 Χαρακτηριστικά 11

2.1.4 Είδη πρακτόρων 12

2.2 Πολυπρακτορικά Συστήματα 15

2.2.1 Εισαγωγή 15

2.2.2 Επικοινωνία, συνεργασία και σύναψη συμφωνιών 17

2.2.3 Πολυπρακτορικά συστήματα και διαχείριση πόρων 19

2.3 Κατανεμημένα Συστήματα 20

2.3.1 Εισαγωγή 20

2.3.2 Βασικά χαρακτηριστικά γνωρίσματα 21

2.3.3 Μοντέλο ανταλλαγής μηνυμάτων 24

2.3.3.1 Επικοινωνία 24

2.3.3.2 Χρονισμός 25

2.4 Θεωρητική Ανάλυση της Θεωρίας Παιγνίων 26

2.4.1 Τι είναι η Θεωρία Παιγνίων 26

2.4.2 Εφαρμογές στην καθημερινή ζωή 27

2.4.3 Βασικές έννοιες της Θεωρίας Παιγνίων 28

2.4.4 Κατηγορίες παιγνίων 32

2.4.5 Προσέγγιση της Ισορροπίας Nash 35

Κεφάλαιο **3** **Μοντέλο Επίλυσης Κατανεμημένων Προβλημάτων και Προηγούμενη Εργασία**..... **36**
3.1 Μοντέλο Υπολογισμού 36
3.2 Θεωρητικό Μοντέλο Παιγνίων 39
3.3 Ενεργοποίηση (ξύπνημα) Πρακτόρων 42
3.3.1 Πρωτόκολλο ενεργοποίησης πρακτόρων σε δακτύλιο 44
3.4 Διαμοίραση Γνώσης 45
3.4.1 Διαμοίραση γνώσης σε ένα σύγχρονο δακτύλιο 48
3.4.2 Διαμοίραση γνώσης σε ένα ασύγχρονο δακτύλιο 48
3.5 Εκλογή Αρχηγού Πρακτόρων 49
3.6 Μετονομασία 49
3.7 Συμφωνία 51
3.8 Προηγούμενες Εργασίες 53
3.8.1 Η μελέτη των Afek et al. 53
3.8.2 Προηγούμενη Διπλωματική Εργασία 56

Κεφάλαιο **4** **Μια $(n-2)$ - Ισχυρή - Ισορροπία σε Σύγχρονο Δακτύλιο**..... **61**
4.1 Εισαγωγή 61
4.2 Αλγόριθμος $(n-2)$ - Ισχυρής - Ισορροπίας 62
4.3 Ανάλυση αλγορίθμου $(n-2)$ - Ισχυρής - Ισορροπίας 69

Κεφάλαιο **5** **Μια $(n-2)$ - Ισχυρή - Ισορροπία σε Ασύγχρονο Δακτύλιο.**..... **84**
5.1 Εισαγωγή 84
5.2 Αλγόριθμος $(n-2)$ - Ισχυρής - Ισορροπίας με Ομοιόμορφες Εισόδους 85
5.3 Ανάλυση Αλγορίθμου $(n-2)$ - Ισχυρής - Ισορροπίας με Ομοιόμορφες εισόδους 92
5.4 Αλγόριθμος $(n-2)$ - Ισχυρής - Ισορροπίας με Μη Ομοιόμορφες Εισόδους 101
5.5 Ανάλυση Αλγορίθμου $(n-2)$ - Ισχυρής - Ισορροπίας με Μη Ομοιόμορφες Εισόδους 106

Κεφάλαιο **6 Επίλογος** ……...……... **117**

6.1 Σύνοψη 117

6.2 Συμπεράσματα 118

6.3 Μελλοντική Εργασία 119

Β ι β λ ι ο γ ρ α φ ί α ……**120**

Π α ρ ά ρ τ η μ α Α……..…..…….…… **Α-1**

Κεφάλαιο 1

Εισαγωγή

1.1 Κίνητρο 1
1.2 Στόχος Ατομικής Διπλωματικής Εργασίας 2
1.3 Μεθοδολογία 3
1.4 Οργάνωση 4

1.1 Κίνητρο

Η παρούσα διπλωματική εργασία διαπραγματεύεται το πρόβλημα της Κατανεμημένης Συμφωνίας μεταξύ μιας ομάδας πρακτόρων. Έχουμε n πράκτορες που πρέπει να συμφωνήσουν σε μια τιμή (0 ή 1), η οποία αποτελεί και είσοδο σε ένα από αυτούς. Πρόκειται για ένα παίγνιο ανάμεσα σε n πράκτορες όπου όμως δεν υπάρχει κάποιος νικητής και κάποιος χαμένος πράκτορας, αλλά υπάρχει μια κοινή τιμή συμφωνίας. Το κίνητρο που οδήγησε στην περαιτέρω ανάλυση του προβλήματος της Κατανεμημένης Συμφωνίας ήταν η παροχή μιας $Ισχυρής\ Ισορροπίας$ σε τέτοια προβλήματα, ούτως ώστε με μια τιμή συμφωνίας να είμαστε σίγουροι ότι μεγιστοποιείται η ολική απόδοση ενός συστήματος πρακτόρων και ότι η τιμή αυτή συμφέρει στο μέγιστο δυνατό βαθμό κάθε πράκτορα με τον τρόπο της.

Μια $Ισχυρή\ Ισορροπία$ στη Θεωρία των Παιγνίων είναι μια συλλογή από στρατηγικές, μια για κάθε παίκτη, όπου δεν υπάρχει όφελος για οποιονδήποτε παίκτη να αλλάξει στρατηγικές. Σε αυτή την κατάσταση, όλοι οι παίκτες στο παιχνίδι είναι ικανοποιημένοι με τις επιλογές τους την ίδια χρονική στιγμή, επομένως, το παιχνίδι παραμένει σε ισορροπία. Στη παρούσα διπλωματική εργασία οι n πράκτορες είναι οι παίκτες ενός

παιχνιδιού και οι στρατηγικές είναι οι κινήσεις των πρακτόρων που δεν παραβιάζουν τη συνθήκη Ισχυρής Ισορροπίας, έτσι ώστε μέσα από ένα συγκεκριμένο αλγόριθμο Κατανεμημένης Συμφωνίας, να παρέχεται μια συμφέρουσα τιμή συμφωνίας για ολόκληρο το πολυπρακτορικό σύστημα.

Η $(n-2)$ - Ισχυρή - Ισορροπία που εξετάζουμε στη παρούσα διπλωματική εργασία είναι η $Ισχυρή\ Ισορροπία$ που δεν επιτυγχάνεται ανάμεσα σε όλους τους n πράκτορες, αλλά μόνο ανάμεσα στους $n-2$ πράκτορες. Δηλαδή, εάν έχουμε ένα κατανεμημένο αλγόριθμο A και P είναι το σύνολο των πρακτόρων, τότε ο A φθάνει σε $(n-2)$ - Ισχυρή - Ισορροπία εάν και μόνο εάν $\forall C \subseteq P$, έτσι ώστε $|C| \leq k$, για κάθε πράκτορα p που ανήκει στο C ισχύει ότι το κέρδος του p με το να αποκλίνει από τον αλγόριθμο συμφωνίας είναι μικρότερο ή ίσο από το κέρδος του εάν δεν είχε αποκλίνει.

Η παροχή μιας $Ισχυρής\ Ισορροπίας$ αποτελεί θεμέλιο λίθο για πολλά προβλήματα της καθημερινής μας ζωής και έτσι η ανάλυση της πάντα αποτελούσε ένα εξαιρετικά ενδιαφέρον πεδίο έρευνας στον κλάδο της Πληροφορικής.

Επιπλέον, ένα επιπρόσθετο κίνητρο για την περαιτέρω ανάλυση του προβλήματος της Κατανεμημένης Συμφωνίας έχει αποτελέσει μια προηγούμενη εργασία που μου έχει δοθεί [12], που διαπραγματεύεται την $Ισχυρή\ Ισορροπία$ για την Κατανεμημένη Συμφωνία σε πολυπρακτορικά συστήματα. Έτσι, μου δόθηκε η ευκαιρία να εμπλουτίσω αυτή την εργασία με περεταίρω έρευνα στο πρόβλημα της Κατανεμημένης Συμφωνίας με Εγωιστικούς Πράκτορες.

1.2 Στόχος Ατομικής Διπλωματικής Εργασίας

Το πρόβλημα της Κατανεμημένης Συμφωνίας με Εγωιστικούς Πράκτορες είναι ένα πρόβλημα κατανεμημένου υπολογισμού. Η κατανεμημένη επίλυση προβλημάτων ασχολείται με το πώς ένα συγκεκριμένο πρόβλημα μπορεί να επιλυθεί από ένα αριθμό επεξεργαστικών μονάδων που συνεργάζονται διαμοιράζοντας γνώση για το πρόβλημα, καθώς και τις επιμέρους λύσεις. Επειδή οι πράκτορες είναι ορθολογικοί, δηλαδή εγωιστικοί, μπορούν να παρεκκλίνουν από ένα κατανεμημένο αλγόριθμο για να επηρεάσουν το αποτέλεσμα του, καθώς ο κάθε πράκτορας παίρνει αποφάσεις μέσω

λογικής επαγωγής και θέλει να μεγιστοποιήσει την αναμενόμενη απόδοση του με μεγαλύτερο κίνητρο να εκλεγεί αρχηγός των άλλων πρακτόρων.

Στόχος λοιπόν της διπλωματικής αυτής εργασίας είναι να κατασκευαστεί κάποιος αλγόριθμος όπου η τιμή συμφωνίας μεταξύ n πρακτόρων θα παρέχεται από μια $(n-2)$ - Ισχυρή Ισορροπία, δηλαδή οι πράκτορες που προσπαθούν να ξεγελάσουν άλλους πράκτορες (Cheaters), να μην κερδίζουν τίποτα από την παρέκκλιση τους από το πρωτόκολλο, έτσι ώστε να μην έχουν κανένα απολύτως κίνητρο να ξεγελάσουν τους υπόλοιπους πράκτορες (συμπαίχτες τους). Στην παρούσα διπλωματική εργασία παρατίθενται τρείς τέτοιοι αλγόριθμοι όπου ο πρώτος παρέχει μια $(n-2)$ - Ισχυρή - Ισορροπία σε ένα σύγχρονο δακτύλιο με μη ομοιόμορφη κατανομή στις εισόδους, ο δεύτερος παρέχει μια $(n-2)$ - Ισχυρή - Ισορροπία σε ένα ασύγχρονο δακτύλιο με ομοιόμορφη κατανομή στις εισόδους και ο τρίτος παρέχει μια $(n-2)$ - Ισχυρή - Ισορροπία σε ένα ασύγχρονο δακτύλιο με μη ομοιόμορφη κατανομή στις εισόδους.

1.3 Μεθοδολογία

Η μεθοδολογία που έχω ακολουθήσει για την επιτυχή ολοκλήρωση της διπλωματικής αυτής εργασίας, είναι αρχικά να βασιστώ στον τρόπο που αντιμετωπίζει η Θεωρία των Παιγνίων τα προβλήματα της Κατανεμημένης Συμφωνίας, καθώς η Θεωρία των Παιγνίων θεωρείται η βάση για την επίλυση των κατανεμημένων προβλημάτων. Ακολούθως, έχω βασιστεί σε μια προηγούμενη εργασία που ασχολείται με την παροχή Ισχυρής Ισορροπίας για την Κατανεμημένη Συμφωνία σε ένα πολυπρακτορικό σύστημα και στον τρόπο με τον οποίο οι ερευνητές που την έχουν ολοκληρώσει έχουν διαχειριστεί τις αποδείξεις των θεωρημάτων τους.

Σαν πρώτο στάδιο ανάπτυξης της διπλωματικής μου εργασίας έγινε μια μελέτη για την Τεχνολογία των Νοήμονων Πρακτόρων, για τα Κατανεμημένα Συστήματα και την Θεωρία των Παιγνίων. Έχοντας αυτά κατά νου και μελετώντας την εργασία που μου είχε δοθεί, ήμουν σε θέση να την εμπλουτίσω με περεταίρω αποδείξεις θεωρημάτων που αφορούσαν την παροχή μιας $(n-2)$ - Ισχυρής - Ισορροπίας στο πρόβλημα της Κατανεμημένης Συμφωνίας με Εγωιστικούς Πράκτορες. Οι αποδείξεις των θεωρημάτων μου βασίστηκαν στον αλγόριθμο που έχουν προτείνει οι Afek et al. [1], ο οποίος

αλγόριθμος παρέχει μια $(n-1)$ - Ισχυρή Ισορροπία, καθώς και μια $(n-2)$ - Ισχυρή - Ισορροπία σε ένα σύγχρονο δακτύλιο ζυγού μεγέθους, υποθέτοντας ομοιόμορφη κατανομή πάνω στις εισόδους των πρακτόρων. Στο δεύτερο στάδιο επομένως της διπλωματικής μου εργασίας, έχω κατασκευάσει τρείς αλγορίθμους με βάση των αλγόριθμο που παραθέτουν οι Afek et al. στο [1], με την υπόθεση ότι ο αριθμός των πρακτόρων n είναι ένας ζυγός αριθμός. Ο πρώτος αλγόριθμος παρέχει μια $(n-2)$ - Ισχυρή - Ισορροπία σε ένα σύγχρονο δακτύλιο όπου οι είσοδοι που παρέχονται στους πράκτορες δεν ακολουθούν την ομοιόμορφη κατανομή ενώ ο δεύτερος αλγόριθμος λειτουργεί σε ένα ασύγχρονο δακτύλιο και οι είσοδοι που παρέχονται στους πράκτορες ακολουθούν την ομοιόμορφη κατανομή. Ο τρίτος αλγόριθμος λειτουργεί σε ένα ασύγχρονο δακτύλιο και οι είσοδοι που παρέχονται στους πράκτορες δεν ακολουθούν την ομοιόμορφη κατανομή. Και για τους τρείς αλγορίθμους παραθέτω την αντίστοιχη επεξήγηση και την απόδειξη παροχής μιας $(n-2)$ - Ισχυρής - Ισορροπίας μέσα από μια σειρά αποδείξεων κάποιων λημμάτων.

Τέλος, λαμβάνοντας υπόψη την ανάλυση που χρειάστηκε σε κάθε ένα από τους τρείς αλγορίθμους για να φτάσω στο επιθυμητό αποτέλεσμα, έχω παραθέσει τα συμπεράσματα μου και κάποια μελλοντική εργασία που μπορεί να γίνει για την επέκταση της δικής μου διπλωματικής εργασίας.

1.4 Οργάνωση

Στο Κεφάλαιο 2 γίνεται μια θεωρητική εισαγωγή στις έννοιες που αποτέλεσαν τη βάση για τον κατανεμημένο υπολογισμό. Έτσι, προσεγγίζονται οι έννοιες των πρακτόρων και των συστημάτων πολλαπλών πρακτόρων. Επίσης, δίνεται μια εισαγωγή στα κατανεμημένα συστήματα. Πρώτον, παρουσιάζουμε εν συντομία την έννοια των κατανεμημένων συστημάτων και τα χαρακτηριστικά τους γνωρίσματα. Στη συνέχεια, εξηγούμε το μοντέλο ανταλλαγής μηνυμάτων σε ένα κατανεμημένο σύστημα πρακτόρων, το πώς δηλαδή διεξάγεται η επικοινωνία μέσα σε ένα τέτοιο σύστημα. Επιπλέον, στο Κεφάλαιο 2, αναλύουμε τη Θεωρία των Παιγνίων, καθώς και την έννοια της Ισορροπίας Nash πάνω στα οποία βασίζεται η όλη ιδέα υλοποίησης των αλγορίθμων της παρούσας διπλωματικής εργασίας που θα παράξουν μια τιμή συμφωνίας για τους πράκτορες μας, η οποία τιμή θα βασίζεται στην ύπαρξη μιας k - Ισχυρής Ισορροπίας.

Εξηγούμε τι ακριβώς είναι η Θεωρία των Παιγνίων, καθώς και την χρησιμότητά της με αναφορές σε διάφορες εφαρμογές της στη καθημερινή μας ζωή. Επίσης, έχω προσκομίσει κάποιες βασικές έννοιες όσο αφορά τα παίγνια. Συμπληρωματικά, έχω εξηγήσει διάφορες κατηγορίες παιγνίων με ιδιαίτερη αναφορά στην κατηγορία στην οποία ανήκει το πρόβλημα που διαπραγματεύεται η διπλωματική μου εργασία.

Στο Κεφάλαιο 3 περιγράφεται το γενικό πλαίσιο ανάπτυξης ενός αλγορίθμου συμφωνίας μεταξύ n πρακτόρων με τη χρήση της Θεωρίας των Παιγνίων. Περιγράφονται τα διάφορα στάδια εξέλιξης ενός κατανεμημένου αλγορίθμου μέχρι να φτάσει στο σημείο συμφωνίας που προϋποθέτει την επιτυχή παροχή μιας k - Ισχυρής - Ισορροπίας. Επίσης, περιγράφω εν συντομία την ανάλυση που έχει γίνει στην διπλωματική εργασία που μου έχει δοθεί και που έχω εμπλουτίσει ανάλογα.

Στο Κεφάλαιο 4 παρουσιάζουμε ένα κατανεμημένο αλγόριθμο, τον αλγόριθμο $do\_consensus_i^{uni}$, που έχει ως βάση του (α) τις έννοιες που κατακλύζουν τα πολυπρακτορικά συστήματα, (β) τον κατανεμημένο υπολογισμό, (γ) τις έννοιες που κατακλύζουν τα παίγνια συνεργασίας πρακτόρων και (δ) την ιδέα της Ισορροπίας Nash. Ο αλγόριθμος αυτός, παρέχει μια $(n-2)$ - Ισχυρή - Ισορροπία σε ένα σύγχρονο δακτύλιο ζυγού μεγέθους και μονής κατεύθυνσης με την προϋπόθεση ότι οι εισόδοι που παρέχονται στους πράκτορες δεν ακολουθούν την ομοιόμορφη κατανομή. Παρουσιάζεται μια σειρά αποδείξεων της εγκυρότητας του αλγορίθμου.

Στο Κεφάλαιο 5 παρουσιάζουμε τον κατανεμημένο αλγόριθμο $Asynchronous\_do\_consensus_i^{uni}$, ο οποίος είναι παρόμοιος με τον αλγόριθμο $do\_consensus_i^{uni}$. Η διαφορά είναι ότι ο αλγόριθμος $Asynchronous\_do\_consensus_i^{uni}$ παρέχει μια $(n-2)$ - Ισχυρή - Ισορροπία σε ένα ασύγχρονο αυτή τη φορά δακτύλιο ζυγού μεγέθους και μονής κατεύθυνσης και με την προϋπόθεση αυτή τη φορά ότι οι εισόδοι που παρέχονται στους πράκτορες ακολουθούν την ομοιόμορφη κατανομή. Παρουσιάζεται μια σειρά αποδείξεων της εγκυρότητας του αλγορίθμου. Επίσης, σε αυτό το κεφάλαιο, παρουσιάζουμε τον κατανεμημένο αλγόριθμο $Asynchronous\_non\_uni\_do\_consensus_i^{non\_uni}$, ο οποίος είναι παρόμοιος με τον αλγόριθμο $Asynchronous\_do\_consensus_i^{uni}$. Η διαφορά είναι ότι ο αλγόριθμος

$Asynchronous\_non\_uni\_do\_consensus_i^{non\_uni}$ παρέχει μια $(n-2)$ - Ισχυρή - Ισορροπία σε ένα ασύγχρονο δακτύλιο ζυγού μεγέθους όπου οι εισόδοι που παρέχονται στους πράκτορες δεν ακολουθούν την ομοιόμορφη κατανομή. Παρουσιάζεται μια σειρά αποδείξεων της εγκυρότητας του αλγορίθμου.

Τέλος, στο Κεφάλαιο 6 γίνεται μια σύνοψη των πιο πάνω με την παροχή κάποιων συμπερασμάτων που απορρέουν από την υλοποίηση των αλγορίθμων:
(α) $do\_consensus_i^{uni}$
(β) $Asynchronous\_do\_consensus_i^{uni}$ και
(γ)$Asynchronous\_non\_uni\_do\_consensus_i^{non\_uni}$.
Παρουσιάζεται επίσης κάποια μελλοντική εργασία που μπορεί να γίνει μέσα στο πεδίο επιστημονικής έρευνας της Κατανεμημένης Συμφωνίας με Εγωιστικούς Πράκτορες.

Κεφάλαιο 2

Θεωρητικό Υπόβαθρο

2.1 Τεχνολογία Νοήμονων Πρακτόρων 8
2.1.1 Εισαγωγή στους πράκτορες 8
2.1.2 Ορισμός 9
2.1.3 Χαρακτηριστικά 11
2.1.4 Είδη πρακτόρων 12
2.2 Πολυπρακτορικά Συστήματα 15
2.2.1 Εισαγωγή 15
2.2.2 Επικοινωνία, συνεργασία και σύναψη συμφωνιών 17
2.2.3 Πολυπρακτορικά συστήματα και διαχείριση πόρων 19
2.3 Κατανεμημένα Συστήματα 20
2.3.1 Εισαγωγή 20
2.3.2 Βασικά χαρακτηριστικά γνωρίσματα 21
2.3.3 Μοντέλο ανταλλαγής μηνυμάτων 24
2.3.3.1 Επικοινωνία 24
2.3.3.2 Χρονισμός 25
2.4 Θεωρητική Ανάλυση της Θεωρίας Παιγνίων 26
2.4.1 Τι είναι η Θεωρία Παιγνίων 26
2.4.2 Εφαρμογές στην καθημερινή ζωή 27
2.4.3 Βασικές έννοιες της Θεωρίας Παιγνίων 28
2.4.4 Κατηγορίες παιγνίων 32
2.4.5 Προσέγγιση της Ισορροπίας Nash 35

2.1 Τεχνολογία Νοήμονων Πρακτόρων

2.1.1 Εισαγωγή στους πράκτορες

Η εμφάνιση των πρακτόρων (agents) στον κλάδο της Τεχνητής Νοημοσύνης και, γενικότερα, στην περιοχή της Επιστήμης των Υπολογιστών υπόσχεται ριζικές αλλαγές στην επικοινωνία μεταξύ χρήστη και λογισμικού στο σημερινό διασυνδεδεμένο και δικτυωμένο ψηφιακό κόσμο. Ήδη, γίνεται αισθητή η παρουσία των πρακτόρων σε πληθώρα εφαρμογών, όπως είναι η αναζήτηση και το φιλτράρισμα των πληροφοριών στο διαδίκτυο, η παροχή έξυπνων υπηρεσιών βοήθειας και εξυπηρέτησης πελατών, καθώς και ο έλεγχος σωστής λειτουργίας μεγάλων εργοστασιακών μονάδων.

Η ιστορία των πρακτόρων ξεκίνησε στις αρχές της δεκαετίας του ΄90 με την εμφάνιση διαφορετικών αντιλήψεων και τεχνολογιών σχετικά με αυτούς. Οι αντιλήψεις αυτές σήμερα έχουν καταλήξει σε δύο κύρια είδη πρακτόρων, στους έξυπνους και κινητούς πράκτορες. Στις αρχές της εμφάνισης των πρακτόρων, η έννοια αυτή δεν ήταν σαφώς ορισμένη, και αυτό επειδή πολλοί από αυτούς που όριζαν ως «πράκτορες» δε βασίζονταν πάντοτε σε έννοιες και τεχνολογίες των πρακτόρων, όπως αυτές είναι γνωστές σήμερα. Για παράδειγμα, ο όρος «πράκτορας» έχει χρησιμοποιηθεί για πολλά χρόνια στον τομέα των κατανεμημένων υπολογιστικών συστημάτων, όπου αναφέρθηκε σε συγκεκριμένες οντότητες που χρησιμοποιήθηκαν για την υλοποίηση διεργασιών σε ένα κατανεμημένο υπολογιστικό σύστημα.

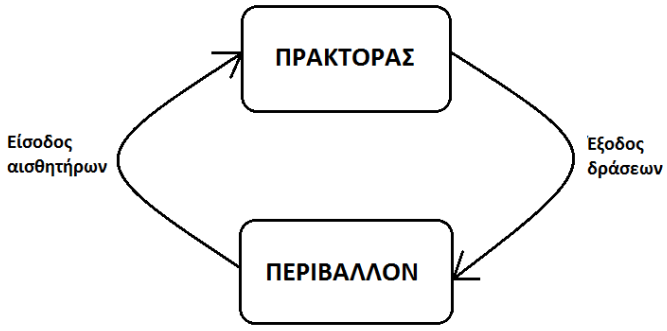
Το ενδιαφέρον για τους έξυπνους πράκτορες ξεκίνησε με την αύξηση της χρήσης των συστημάτων πολλαπλών πρακτόρων. Τα συστήματα πολλαπλών πρακτόρων έχουν ως βασική ιδέα τον καταμερισμό των πολύπλοκων διαδικασιών. Οι διαδικασίες αυτές μπορούν να αποτελέσουν το γινόμενο των επιμέρους ενεργειών ανεξαρτήτων προγραμματιστικών οντοτήτων, που ονομάζονται πράκτορες. Ένα σύστημα πολλαπλών πρακτόρων μπορεί να οριστεί ως ένα σύνολο πρακτόρων, οι οποίοι αλληλεπιδρούν μεταξύ τους και με το περιβάλλον τους ώστε να λύσουν ένα συγκεκριμένο πρόβλημα. Έτσι, η επικοινωνία και η συνεργασία των πρακτόρων αποτελούν ένα σημαντικό θέμα μελέτης.

Τέλος, η άνθιση της εποχής της γλώσσας προγραμματισμού Java ευνόησε την ανάπτυξη των πρακτόρων και γενικότερα αυτή η γλώσσα προγραμματισμού αποτελεί σήμερα τη βάση για τα περισσότερα συστήματα πολλαπλών πρακτόρων.

2.1.2 Ορισμός

Υπάρχουν διάφοροι ορισμοί και κατηγοριοποιήσεις για τους πράκτορες. Μέρος της δυσκολίας που συναντάται στο να δοθεί ένας καθολικά αποδεκτός ορισμός για τον πράκτορα είναι ότι τα χαρακτηριστικά του γνωρίσματα, που σχετίζονται με τη δράση του, έχουν διαφορετική σημασία για τους διάφορους τομείς που υποστηρίζει. Ωστόσο, ο ορισμός που φαίνεται να είναι περισσότερο αποδεκτός είναι αυτός που δίνει ο Wooldridge [26] σύμφωνα με τον οποίο ο πράκτορας είναι ένα υπολογιστικό σύστημα που βρίσκεται σε κάποιο περιβάλλον, και το οποίο σύστημα είναι ικανό για αυτόνομη δράση μέσα σε αυτό το περιβάλλον προκειμένου να ικανοποιήσει τους σχεδιαστικούς του στόχους.

Πιο κάτω, στο Σχήμα 2.1 βλέπουμε μία αφαιρετική άποψη ενός πράκτορα. Απεικονίζεται ο πράκτορας και το περιβάλλον στο οποίο βρίσκεται. Ο πράκτορας, ως αυτόνομη οντότητα, παρατηρεί και λαμβάνει είσοδο από το περιβάλλον του μέσω αισθητήρων, και παράγει ως έξοδο δράσεις που επηρεάζουν το περιβάλλον.

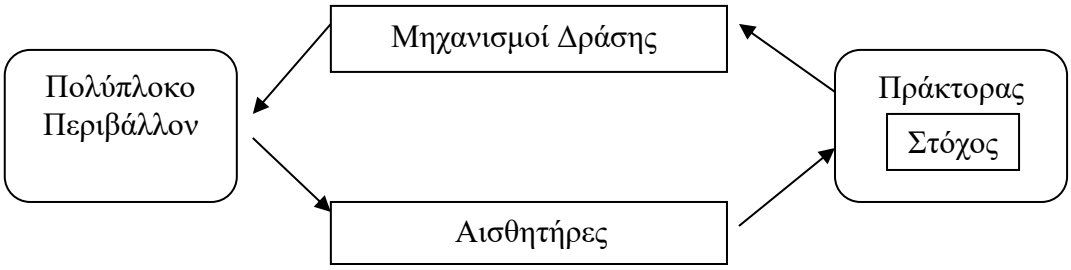


Σχήμα 2.1 Αφαιρετική άποψη ενός πράκτορα

Όπως έχουμε αναφέρει και πιο πάνω υπάρχουν διάφοροι αξιόλογοι ορισμοί που έχουν διατυπωθεί για τους πράκτορες. Η Pattie Maes, του Media Lab στο MIT, είναι από τους πρωτοπόρους της έρευνας στο χώρο των πρακτόρων. Στον ορισμό της για το τι είναι

πράκτορας προσθέτει ένα κρίσιμο χαρακτηριστικό: οι πράκτορες πρέπει να δρουν αυτόνομα, ώστε να «συνειδητοποιούν ένα σύνολο στόχων». Το μοντέλο πρακτόρων Maes [25], εκτός της αυτονομίας και της ύπαρξης στόχων, δίνει έμφαση και στο ότι το περιβάλλον είναι πολύπλοκο και πιθανώς δυναμικό:

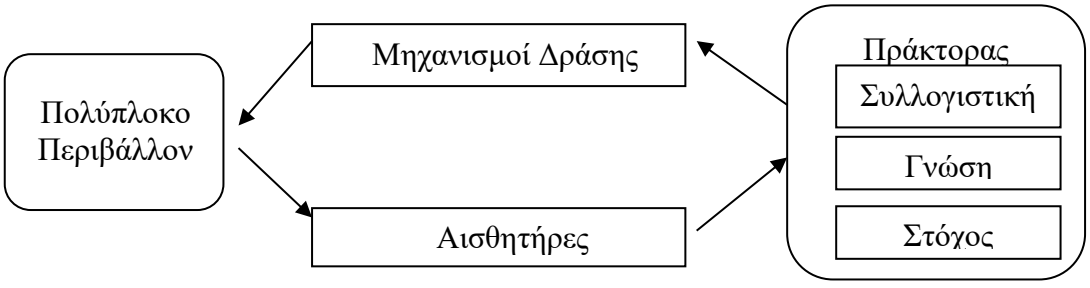
> «Οι πράκτορες είναι υπολογιστικά συστήματα που δρουν σε ένα πολύπλοκο περιβάλλον, αντιλαμβάνονται και δρουν αυτόνομα σε αυτό. Με τον τρόπο αυτό, επιτυγχάνουν ένα σύνολο από στόχους και εκτελούν καθήκοντα για τα οποία έχουν σχεδιαστεί.»



Σχήμα 2.2 Ο βασικός πράκτορας κατά Maes

Τέλος, στην περίπτωση κατά την οποία αναφερόμαστε σε ευφυείς πράκτορες, ένας ορισμός που ανταποκρίνεται στα χαρακτηριστικά τους είναι το μοντέλο Hayes-Roth, που δίνει έμφαση στη συλλογιστική:

> «Οι ευφυείς πράκτορες συνεχώς εκτελούν τρείς λειτουργίες: αντιλαμβάνονται τις δυναμικές συνθήκες του περιβάλλοντος, δρουν στο περιβάλλον, ώστε να το αλλάξουν, και συλλογίζονται, ώστε να ερμηνεύσουν αυτά που αντιλαμβάνονται, να λύσουν προβλήματα, να εξάγουν συμπεράσματα, για να καθορίσουν τη δράση τους.»



Σχήμα 2.3 Ο βασικός πράκτορας κατά Hayes – Roth

2.1.3 Χαρακτηριστικά

Βασικό κοινό χαρακτηριστικό των πρακτόρων είναι η αυτονομία τους. Ως αυτόνομες οντότητες ελέγχουν τις καταστάσεις και τις συμπεριφορές τους, χωρίς να κατευθύνονται εξωγενώς βήμα προς βήμα. Άλλα κοινά χαρακτηριστικά που διαθέτουν είναι η κοινωνικότητα και η δυνατότητα συνεργασίας με άλλους πράκτορες:

1. Αυτονομία: Οι πράκτορες ενεργούν αυτόνομα χωρίς άμεση παρέμβαση από χρήστες ή άλλους πράκτορες, με πλήρη έλεγχο των πράξεών τους και της εσωτερικής τους κατάστασης.
2. Κοινωνικότητα: Οι πράκτορες αλληλεπιδρούν με τους χρήστες για την επίτευξη των στόχων τους και με άλλους πράκτορες μέσω μίας κοινά κατανοητής γλώσσας. Έτσι, επιτυγχάνεται επικοινωνία μεταξύ των πρακτόρων για την ολοκλήρωση των ανεξάρτητων στόχων του καθενός ξεχωριστά και ενός κοινού στόχου με συνεργασία μεταξύ τους .
3. Ορθολογικότητα: Αφορά την υπόθεση ότι ένας πράκτορας θα κάνει πάντα το σωστό, δηλαδή θα δρα καταλλήλως για την εκπλήρωση των στόχων του και όχι με τρόπο ο οποίος αποτρέπει την επίτευξή τους.

Οι ευφυείς πράκτορες διαθέτουν επιπλέον χαρακτηριστικά που αφορούν το βαθμό νοημοσύνης που διαθέτουν, όπως:

1. Αντιδραστικότητα: Αφορά τον τρόπο με τον οποίο αντιλαμβάνονται το περιβάλλον και ανταποκρίνονται σε τυχόν αλλαγές του, εντός συγκεκριμένων χρονικών πλαισίων.
2. Προνοητικότητα: Οι πράκτορες δεν ανταποκρίνονται απλώς στις αλλαγές του περιβάλλοντός τους, αλλά είναι ικανοί να συμπεριφερθούν κατάλληλα σε αυτές τις αλλαγές ορίζοντας επιμέρους στόχους, δηλαδή, αναλαμβάνουν πρωτοβουλίες.
3. Γνώση: Συγκεντρωμένη γνώση σχετική με τον τρόπο με τον οποίο λειτουργεί το περιβάλλον στο οποίο δρα ο πράκτορας, η οποία έχει κατάλληλα αναπαρασταθεί, για να υποστηρίζει τη λήψη αποφάσεων.
4. Πεποιθήσεις: Αποτελούν την άποψη του πράκτορα για το περιβάλλον του μια δεδομένη χρονική στιγμή, η οποία άποψη ενδέχεται να είναι εσφαλμένη.

5. Επιθυμίες: Αφορούν την κρίση του πράκτορα για τις μελλοντικές καταστάσεις του περιβάλλοντός του, όπως, για παράδειγμα, αν μια μελλοντική κατάσταση είναι επιθυμητή ή όχι. Παρά ταύτα, δεν εξετάζεται αν μία επιθυμία του πράκτορα είναι εφικτή ή συγκρούεται με κάποια άλλη.
6. Προθέσεις: Οι προθέσεις είναι υποσύνολο των στόχων, τους οποίους ο πράκτορας προσπαθεί να επιτύχει τη δεδομένη χρονική στιγμή. Δεδομένου ότι δεν είναι δυνατή η ταυτόχρονη επίτευξη όλων των στόχων, επιλέγεται ένα υποσύνολό τους, βάσει ορισμένων κριτηρίων ιεράρχησης.
7. Υποχρεώσεις: Αφορούν την υποχρέωση του πράκτορα να υπακούει σε ένα σύνολο κανόνων και να δρα σε ένα γενικότερο πλαίσιο, ώστε να επιτύχει το σκοπό για τον οποίο σχεδιάστηκε.
8. Προσαρμοστικότητα: Ο πράκτορας προσαρμόζεται στο περιβάλλον του αποκτώντας την ικανότητα να μαθαίνει.

Εκτός των χαρακτηριστικών που προαναφέρθηκαν υπάρχουν και επιπλέον χαρακτηριστικά που απαντούν σε συγκεκριμένες κατηγορίες πρακτόρων προσφέροντας μια «ανθρωπόμορφη αρχιτεκτονική» λογισμικού, που επιτρέπει:

1. Κινητικότητα: Είναι η ικανότητα ενός πράκτορα να μετακινείται ελεύθερα σε ένα φυσικό χώρο ή σε ένα δίκτυο.

2. Συνεργασία μεταξύ πρακτόρων που βασίζεται σε:
   (α) Φιλαλήθεια: Οι πράκτορες δε δίνουν εσκεμμένα λάθος πληροφορίες.
   (β) Αγαθή προαίρεση: Ο κάθε πράκτορας προσπαθεί να επιτύχει τους δικούς του στόχους, οι οποίοι βρίσκονται σε αρμονία με τους στόχους των υπολοίπων πρακτόρων του συστήματος.

2.1.4 Είδη πρακτόρων

Τα κύρια είδη πρακτόρων είναι οι έξυπνοι πράκτορες και οι κινητοί πράκτορες:

(α) Έξυπνοι πράκτορες: Το σημαντικότερο στοιχείο που διακρίνει αυτόν τον τύπο πρακτόρων από τους υπόλοιπους τύπους των διεργασιών λογισμικού, είναι ότι μπορεί να

συνεργάζεται κατά κάποιο τρόπο με άλλους χρήστες (ανθρώπινους ή άλλους πράκτορες λογισμικού) είτε για να κάνει χρήση των δυνατοτήτων τους, είτε για να συνεισφέρει στη δράση τους και να επιτευχθεί ο κοινός σκοπός τους. Έτσι, συχνά καλούνται και ως ευφυής συνεργάσιμοι πράκτορες. Τα βασικά χαρακτηριστικά αυτής της ομάδας πρακτόρων είναι η επικοινωνία και η συνεργασία.

Η επικοινωνία επιτρέπει στους πράκτορες συστημάτων πολλαπλών πρακτόρων να ανταλλάσσουν πληροφορίες και να προσαρμόζουν τη συμπεριφορά και τη δράση τους σύμφωνα με αυτές. Γενικά οι πράκτορες λογισμικού επικοινωνούν με άλλους πράκτορες ώστε να γίνει η δράση τους πιο ευέλικτη. Για να γίνει εφικτή η κοινή δράση των πρακτόρων είναι απαραίτητο να αλληλεπιδρούν, να ανταλλάσσουν πληροφορίες ακόμα και να διαπραγματεύονται. Αυτά γίνονται μέσω ενός προκαθορισμένου πρωτοκόλλου επικοινωνίας, που ονομάζεται ACL (Agent Communicative Language) και είναι μια γλώσσα με αυστηρά καθορισμένο συντακτικό και σημασιολογία (semantics). Ωστόσο, σημειώνεται ότι η επικοινωνία των πρακτόρων πραγματοποιείται μέσω της χρήσης τριών συστατικών: ACL, Γλώσσα περιεχομένου και Οντολογία.

Η Οντολογία περιλαμβάνει τους όρους που συνιστούν το πεδίο της εφαρμογής και αναπαριστά ένα λεξικό δεδομένων σε ένα κλασσικό σύστημα πληροφορίας. Η γλώσσα περιεχομένου χρησιμοποιείται για το συνδυασμό των όρων της Οντολογίας σε προτάσεις οι οποίες είναι χρήσιμες στους πράκτορες. Ορισμένες φορές η γλώσσα περιεχομένου και η οντολογία είναι τόσο στενά δεμένες μεταξύ τους ώστε είναι πολύ δύσκολο να τις ξεχωρίσουμε. Τέλος, η ACL δρα σαν ένα πρωτόκολλο, δίνοντας τη δυνατότητα ανάπτυξης διαλόγων που περιλαμβάνουν προτάσεις της γλώσσας περιεχομένου μεταξύ πρακτόρων και καθορίζει τη σημασιολογία για τη συμπεριφορά των πρακτόρων που παίρνουν μέρος σε τέτοιους διαλόγους.

Στους έξυπνους πράκτορες ανήκουν και οι ορθολογικοί πράκτορες τους οποίους και χρησιμοποιούμε για την επίλυση του προβλήματος που διαπραγματεύεται η παρούσα διπλωματική εργασία. Οι ορθολογικοί πράκτορες λαμβάνουν αποφάσεις και δρουν βάσει μιας διαδικασίας λογικής απόδειξης. Στηρίζονται σε μία βάση γνώσης, χάρη στην οποία διατηρούν την αντίληψή τους για το περιβάλλον με μορφή λογικών προτάσεων, και σε ένα σύνολο κανόνων συμπερασμού, για να δράσουν. Έτσι, οι δράσεις που θα

εκτελεστούν ανάγονται σε προβλήματα απόδειξης της μαθηματικής λογικής και πολλές φορές χρησιμοποιούνται συμπερασματικές μηχανές, για να επιτευχθούν.

Οι ορθολογικοί πράκτορες εκτελούν συνεχώς τις εξής τρεις λειτουργίες:

1. Αντιλαμβάνονται τις δυναμικές συνθήκες του περιβάλλοντος.
2. Δρουν στο περιβάλλον, ώστε να το αλλάξουν και
3. Συλλογίζονται, ώστε να ερμηνεύσουν αυτά που αντιλαμβάνονται, να λύσουν προβλήματα, να συμπεράνουν και να καθορίσουν τη δράση τους.

Οι ορθολογικοί πράκτορες χρησιμοποιούν μεθόδους και τεχνικές που έχουν καθορισμένη και απλή σημασιολογία. Συνεπώς, είναι περισσότερο κατάλληλοι για στατικά περιβάλλοντα, ενώ καθίσταται ιδιαίτερα δύσκολη η εξαγωγή συμπερασμάτων σε μεταβαλλόμενα (δυναμικά) περιβάλλοντα, όπου απαιτείται επανασχεδιασμός των δράσεων του πράκτορα και μάλιστα σε περιορισμένα χρονικά όρια. Επίσης, η δράση ενός ορθολογικού πράκτορα σε ένα δυναμικό περιβάλλον απαιτεί ακριβή και ικανοποιητική (αν όχι πλήρη) συμβολική περιγραφή του πραγματικού κόσμου, κάτι το οποίο είναι ανέφικτο σε πρακτικό επίπεδο.

(β) Κινητοί πράκτορες: Οι κινητοί πράκτορες, συχνά αποκαλούμενοι και ως μετακινήσιμοι βασίζονται στην αρχή της δυνατότητας μετακίνησης κώδικα. Ο μετακινούμενος κώδικας βελτιώνει το κλασσικό client/server πρότυπο κάνοντας αλλαγές κατά μήκος δύο ορθογωνίων αξόνων. Τα βασικά ερωτήματα που γεννιούνται από την υιοθέτηση των κινητών πρακτόρων έχουν να κάνουν με το πού είναι εγκατεστημένο το know how της υπηρεσίας και το ποιος παρέχει τους υπολογιστικούς πόρους για την υποστήριξη της υπηρεσίας.

Έχουν οριστεί τρία βασικά πρότυπα για τους κινητούς υπολογισμούς, τα οποία είναι η απομακρυσμένη αξιολόγηση, ο κώδικας έπειτα από αίτηση και οι κινητοί πράκτορες. Αυτά τα πρότυπα διαφέρουν ως προς το πώς το know how, ο επεξεργαστής και οι πόροι είναι κατανεμημένοι γύρω από τα συστατικά ενός κατανεμημένου συστήματος. Το know how εκπροσωπεί τον κώδικα που είναι απαραίτητος για την υλοποίηση του υπολογισμού. Οι πόροι (π.χ. δεδομένα) είναι αυτοί που είναι εγκατεστημένοι στη μηχανή που θα εκτελέσει τον υπολογισμό.

2.2 Πολυπρακτορικά Συστήματα

2.2.1 Εισαγωγή

Στον κλάδο της Τεχνητής Νοημοσύνης, η τεχνολογία των συστημάτων βασισμένων σε πράκτορες, αποτελεί ένα νέο παράδειγμα σύλληψης, σχεδίασης και υλοποίησης συστημάτων λογισμικού. Οι πράκτορες είναι οντότητες που δρουν αυτόνομα και για λογαριασμό των χρηστών τους σε ανοιχτά και κατανεμημένα περιβάλλοντα, ώστε να επιλύσουν σύνθετα προβλήματα. Ωστόσο, υπάρχουν ολοένα και περισσότερες εφαρμογές, οι οποίες απαιτούν πολλαπλούς πράκτορες, οι οποίοι μπορούν να δουλέψουν μαζί στο ίδιο περιβάλλον. Ένα πολυπρακτορικό σύστημα είναι ουσιαστικά ένα δίκτυο από πράκτορες λογισμικού που αλληλεπιδρούν, δηλαδή συνεργάζονται, επικοινωνούν και διαπραγματεύονται, για να λύσουν ένα πρόβλημα.

Ένα πολυπρακτορικό σύστημα στοχεύει στη διασύνδεση και λειτουργία ήδη υπαρχόντων συστημάτων, καθώς και στην επίλυση προβλημάτων:

1. που είναι πέρα των δυνατοτήτων και της γνώσης ενός μόνο πράκτορα
2. τα οποία είναι από τη φύση τους κατανεμημένα. Τα πολυπρακτορικά συστήματα αποτελούν βασικό τομέα της Κατανεμημένης Τεχνητής Νοημοσύνης από πλευράς χαλαρής θεώρησης των πρακτόρων και
3. όπου η σχετική γνώση είναι κατανεμημένη σε διακριτές πηγές, όπως για παράδειγμα η υπάρχουσα εμπειρία στα επιμέρους γραφεία ενός οργανισμού.

Στα πολυπρακτορικά συστήματα συναντώνται συνήθως δύο διαφορετικές περιπτώσεις. Η πρώτη περιλαμβάνει πράκτορες, οι οποίοι ανταλλάσσουν πληροφορίες, αλλά εργάζονται αυτόνομα προς την κατεύθυνση επίτευξης δικών τους ανεξάρτητων στόχων. Η δεύτερη περιλαμβάνει πράκτορες οι οποίοι επικοινωνούν και συνεργάζονται, ώστε οι λύσεις που δίνουν σε επιμέρους προβλήματα να συνδυαστούν σε μία τελική λύση για κάποιο πιο σύνθετο πρόβλημα. Το στοιχείο εκείνο που χαρακτηρίζει τους πράκτορες της δεύτερης περίπτωσης είναι η δυνατότητα που έχουν να χρησιμοποιούν μία κατάλληλη και κοινά αποδεκτή μεταξύ τους γλώσσα επικοινωνίας, ώστε να διαπραγματευτούν, να επιλύσουν ενδεχόμενες συγκρούσεις που εμφανίζονται και τελικά να καταλήξουν σε κάποια συμφωνία, που οδηγεί στην καλύτερη δυνατή αντιμετώπιση ενός προβλήματος.

Ένα πολυπρακτορικό σύστημα παρουσιάζει τα εξής πλεονεκτήματα σε σχέση με τη χρήση ενός μοναδικού πράκτορα ή την επιλογή κάποιας κεντρικοποιημένης προσέγγισης:

1. Μπορεί να κατανείμει υπολογιστικούς πόρους και δυνατότητες σε ένα δίκτυο από διασυνδεδεμένους πράκτορες. Σε αντίθεση με ένα κεντρικοποιημένο σύστημα, το οποίο μπορεί να αντιμετωπίζει περιορισμούς σε πόρους, σημεία συμφόρησης στην απόδοση ή κρίσιμες αποτυχίες, ένα πολυπρακτορικό σύστημα δεν «υποφέρει» από το πρόβλημα του «μοναδικού σημείου αποτυχίας» .
2. Επιτρέπει τη συμμετοχή ετερογενών πρακτόρων σχεδιασμένων από διαφορετικές ομάδες και σχεδιαστές.
3. Παρέχει τη δυνατότητα κλιμάκωσης με τη δυναμική διείσδυση ή έξοδο πρακτόρων από το σύστημα.
4. Μοντελοποιεί προβλήματα βάσει της ιδέας των αυτόνομων πρακτόρων που αλληλεπιδρούν. Αυτό αποτελεί ένα πιο φυσικό και κατανοητό τρόπο για την αναπαράσταση προβλημάτων που σχετίζονται με την κατανομή εργασιών, τον προγραμματισμό μιας ομάδας, τις διαφορετικές προτιμήσεις χρηστών, τα ανοιχτά περιβάλλοντα, κλπ.
5. Ανακτά, φιλτράρει, συνδυάζει και συντονίζει πληροφορίες που μπορεί να προέρχονται από πηγές χωρικά κατανεμημένες.
6. Παρέχει λύσεις σε καταστάσεις όπου η εμπειρία και η τεχνογνωσία είναι χωρικά και προσωρινά κατανεμημένη.
7. Βελτιώνει τη συνολική απόδοση του συστήματος και πιο συγκεκριμένα σε επίπεδο υπολογιστικής απόδοσης, αξιοπιστίας, επεκτασιμότητας, δυνατότητας συντήρησης, απόκρισης, ευελιξίας και επαναχρησιμοποίησης.

Εκτός όμως από τα παραπάνω πλεονεκτήματα, υπάρχουν και κάποιες δυσκολίες κατά το σχεδιασμό και την υλοποίηση ενός πολυπρακτορικού συστήματος, στις οποίες πρέπει να δοθεί ιδιαίτερη προσοχή ώστε να αντιμετωπιστούν όσο γίνεται πιο αποτελεσματικά. Αυτές οι δυσκολίες σχετίζονται κυρίως με την επικοινωνία μεταξύ των πρακτόρων και τον τρόπο συνεργασίας τους, αλλά και με την ικανότητα μάθησης ώστε να μπορούν να αλλάζουν κατάλληλα τη συμπεριφορά τους και να προσαρμόζονται στις νέες συνθήκες που επικρατούν στο περιβάλλον τους. Η δυνατότητα μάθησης είναι πολύ σημαντική και

κρίσιμη, γιατί κατά το σχεδιασμό των πρακτόρων είναι αδύνατο να προβλεφθούν όλες οι δυνατές καταστάσεις που μπορεί να αντιμετωπίσουν στο περιβάλλον τους, ώστε να προσδιοριστεί και να τους αποδοθεί εκ των προτέρων η ιδανική συμπεριφορά για κάθε μία από αυτές.

2.2.2 Επικοινωνία, συνεργασία και σύναψη συμφωνιών

Για την επικοινωνία μεταξύ των πρακτόρων, έχουν αναπτυχθεί διάφοροι μηχανισμοί, οι κυριότεροι από τους οποίους περιλαμβάνουν:

1. Ανταλλαγή μηνυμάτων με χρήση κάποιας γλώσσας επικοινωνίας πρακτόρων: Υπάρχουν γλώσσες που αναπτύχθηκαν ειδικά για την επικοινωνία μεταξύ των πρακτόρων και βασίζονται στην ανταλλαγή μηνυμάτων. Μία τέτοια γλώσσα είναι η KQML (Knowledge Query and Manipulation Language). Αυτή η γλώσσα ορίζει μία κοινή μορφή για τα μηνύματα, τα οποία μπορούν να θεωρηθούν ως αντικείμενα με την έννοια του αντικειμενοστραφούς προγραμματισμού, καθώς διαθέτουν ένα τελεστικό (που μπορεί να θεωρηθεί ως κλάση) και έναν αριθμό από παραμέτρους (που μπορούν να θεωρηθούν ως μεταβλητές στιγμιότυπου). Αν και η KQML υιοθετήθηκε σε μεγάλο βαθμό από την κοινότητα των ερευνητών και χρησιμοποιήθηκε ως βάση για πολλές υλοποιήσεις, διάφοροι λόγοι οδήγησαν στην ανάπτυξη μίας νέας γλώσσας από την ομάδα FIPA. Το Ίδρυμα για Ευφυείς Φυσικούς Πράκτορες (Foundation for Intelligent Physical Agents, FIPA) ανέπτυξε μία γλώσσα επικοινωνίας πρακτόρων που αποκαλείται FIPA ACL [27]. Αυτή είναι παρόμοια με την KQML, με μόνη σημαντική διαφορά το σύνολο των τελεστικών που παρέχει κάθε μία από αυτές.
2. Επικοινωνία με χρήση του μοντέλου του μαυροπίνακα: Το μοντέλο του μαυροπίνακα είναι ένας από τους πιο σημαντικούς προδρόμους στην ιστορία των πολυπρακτορικών συστημάτων. Σύμφωνα με αυτό, μία ομάδα πρακτόρων, προκειμένου να επιλύσουν συλλογικά κάποιο πρόβλημα, χρησιμοποιούν μια κοινή δομή δεδομένων που ονομάζεται μαυροπίνακας.

Η βασική διαφορά ανάμεσα στους δύο μηχανισμούς επικοινωνίας, είναι ότι στην περίπτωση της ανταλλαγής μηνυμάτων, οι πράκτορες ανταλλάσουν μεταξύ τους

προσωπικά, ιδιωτικά μηνύματα, ενώ στα μοντέλο του μαυροπίνακα όλα τα μηνύματα αποθηκεύονται σε ένα χώρο που είναι κοινός για όλους τους πράκτορες του συστήματος, οπότε αυτά είναι ορατά και προσπελάσιμα από τον καθένα. Στη παρούσα διπλωματική εργασία χρησιμοποιείται ο πρώτος μηχανισμός επικοινωνίας.

Σε μία κοινότητα πρακτόρων, εκτός από το μηχανισμό επικοινωνίας, είναι πολύ σημαντικό να οριστεί και το λεγόμενο πρωτόκολλο επικοινωνίας. Το πρωτόκολλο μπορεί να καθορίσει τον τύπο, τη δομή, το περιεχόμενο και τη σημασιολογία των μηνυμάτων που ανταλλάσουν οι πράκτορες, αλλά και να ορίσει το σύνολο των επιτρεπτών ενεργειών σε κάθε περίσταση. Όπως σε κάθε ανθρώπινη κοινωνία, έτσι και στα πολυπρακτορικά συστήματα υπάρχει αλληλεπίδραση μεταξύ των μελών τους, τα οποία μπορεί να λειτουργούν ιδιοτελώς. Σε μία τέτοια περίπτωση, θα πρέπει οι πράκτορες να επισυνάψουν μεταξύ τους κάποιου είδους αμοιβαία επωφελούς συμφωνίας για ζητήματα κοινού ενδιαφέροντος. Η σύναψη συμφωνιών αποτελεί θεμελιώδη ικανότητα των αυτόνομων ευφυών πρακτόρων και η δυνατότητα διαπραγμάτευσης και επιχειρηματολογίας έχουν κεντρικό ρόλο σε αυτή. Μία ειδική κατηγορία πρωτοκόλλων επικοινωνίας πρακτόρων είναι τα πρωτόκολλα διαπραγμάτευσης. Αυτά ορίζουν τους «κανόνες της αναμέτρησης» ανάμεσα στους πράκτορες και σχεδιάζονται έτσι ώστε μία συγκεκριμένη διαπραγμάτευση να έχει κάποιες επιθυμητές ιδιότητες. Η παρούσα διπλωματική εργασία ασχολείται με ένα τέτοιο μοντέλο.

Υπάρχουν περιπτώσεις προβλημάτων στα οποία γίνεται η υπόθεση της αγαθής προαίρεσης, ότι δηλαδή οι πράκτορες ενός συστήματος μοιράζονται έμμεσα ένα κοινό στόχο και έτσι δεν υπάρχει ενδεχόμενο διένεξης μεταξύ τους. Σε αυτή την υπόθεση στηρίζεται και η Συνεργατική Κατανεμημένη Επίλυση Προβλήματος, ΣΚΕΠ, (Cooperative Distributed Problem Solving, CDPS) που ξεκίνησε με δουλειά του Lesser και των συναδέλφων του και την οποία έχω χρησιμοποιήσει σε αυτή η διπλωματική εργασία. Η ΣΚΕΠ μελετά τον τρόπο με τον οποίο ένα χαλαρά συνδεδεμένο δίκτυο επιλυτών προβλημάτων (πρακτόρων) μπορεί να συνεργαστεί για να επιλύσει προβλήματα που είναι πέρα από τις ατομικές ικανότητες των επιλυτών αυτών [28]. Μπορεί να θεωρηθεί ως μία δραστηριότητα τριών σταδίων:

1. Αποσύνθεση προβλήματος
2. Επίλυση προβλήματος

3. Σύνθεση λύσης

Με δεδομένο αυτό το γενικό πλαίσιο για τη ΣΚΕΠ, υπάρχουν δύο συγκεκριμένες συνεργατικές δραστηριότητες επίλυσης προβλημάτων:

1. Κοινοχρησία εργασιών: Πραγματοποιείται όταν ένα πρόβλημα αποσυντίθεται σε μικρότερα υποπροβλήματα που ανατίθενται σε διαφορετικούς πράκτορες. Το βασικό ζήτημα είναι ο τρόπος με τον οποίο γίνεται η ανάθεση εργασιών σε κάθε πράκτορα, το οποίο εξαρτάται σε μεγάλο βαθμό από το αν οι πράκτορες είναι ομοιογενείς ή μη ως προς τις ικανότητές τους. Υπάρχουν και οι περιπτώσεις, όπου οι πράκτορες είναι εντελώς αυτόνομοι και κατά συνέπεια μπορούν να αρνηθούν την εκτέλεση εργασιών. Τότε, δεν ισχύει η υπόθεση της αγαθής προαίρεσης που αναφέρθηκε παραπάνω και είναι αναγκαία η σύναψη συμφωνιών μεταξύ των πρακτόρων για να πραγματοποιηθεί η ανάθεση των εργασιών.
2. Κοινοχρησία αποτελεσμάτων: Απαιτεί οι πράκτορες να μοιράζονται πληροφορίες σχετικά με τα υποπροβλήματά τους. Αυτές οι πληροφορίες μπορεί να διαμοιράζονται ενεργητικά (ένας πράκτορας στέλνει πληροφορίες σε κάποιον άλλο πράκτορα επειδή πιστεύει ότι θα ενδιαφερθεί για αυτές) ή αντιδραστικά (ένας πράκτορας στέλνει πληροφορίες σε κάποιον άλλο ως απάντηση σε μία αίτηση που ο ίδιος έλαβε νωρίτερα).

Σε αυτή τη διπλωματική εργασία χρησιμοποιείται η κοινοχρησία αποτελεσμάτων. Φυσικά, υπάρχουν και περιπτώσεις προβλημάτων, στα οποία γίνεται ένας συνδυασμός κοινοχρησίας εργασιών και κοινοχρησίας αποτελεσμάτων.

2.2.3 Πολυπρακτορικά συστήματα και διαχείριση πόρων

Η κατανομή και διαχείριση πόρων είναι ένα πρόβλημα, που συναντάται σε πολλούς τομείς, από απλές καθημερινές δραστηριότητες, μέχρι και εφαρμογές μεγαλύτερου επιστημονικού ενδιαφέροντος. Οι πόροι μπορεί να είναι οποιασδήποτε φύσης: φυσικοί, ανθρώπινοι, πόροι σε κτίρια, σε αυτόματα συστήματα σπιτιών, σε παραγωγικά συστήματα, υλικοί ή άυλοι, υπολογιστικοί, βιοτικοί ή αβιοτικοί, ανανεώσιμοι ή μη, κλπ. Για την ευφυή διαχείρισή τους έχουν επινοηθεί πολλών ειδών αλγόριθμοι και έχουν

κατασκευαστεί συστήματα με δυνατότητες ανάλογες με τις απαιτήσεις του κάθε προβλήματος και με τις ανάγκες που παρουσιάζονται σε κάθε τομέα εφαρμογής.

Η Κατανεμημένη Τεχνητή Νοημοσύνη είναι ένας κλάδος στα πλαίσια του οποίου έχει μελετηθεί ιδιαίτερα το πρόβλημα της διαχείρισης πόρων και έχουν επινοηθεί πολύ αξιόλογες και χρήσιμες λύσεις με αξιοποίηση των τεχνικών και δυνατοτήτων που αυτός παρέχει. Τα Συστήματα Κατανεμημένης Τεχνητής Νοημοσύνης, τα οποία αποτελούνται από ένα σύνολο από κατανεμημένους, έξυπνους πράκτορες λήψης απόφασης, χαρακτηρίζονται από δύο βασικά ζητήματα: το συντονισμό και τη συνεργασία. Σε πολλές περιπτώσεις μελέτης προβλημάτων που σχετίζονται με τη διαχείριση των πόρων και το πώς αυτή μπορεί να γίνει με ευφυή τρόπο για την επίτευξη ενός επιθυμητού αποτελέσματος, γίνεται η προσπάθεια να δοθεί απάντηση στο εξής ερώτημα: πώς ένα σύνολο από γεωγραφικά κατανεμημένους πράκτορες μπορεί να κατανείμει σωστά μία συλλογή από εργασίες για την επίτευξη ενός στόχου, λύνοντας τις οποιεσδήποτε συγκρούσεις που μπορεί να εμφανιστούν, αλλά και ικανοποιώντας συγκεκριμένους περιορισμούς που πιθανώς να έχουν τεθεί στα πλαίσια του συγκεκριμένου προβλήματος.

2.3 Κατανεμημένα Συστήματα

2.3.1 Εισαγωγή

Ορίζουμε σαν κατανεμημένο σύστημα μια συλλογή από αυτόνομους υπολογιστές που επικοινωνούν μεταξύ τους και είναι συνδεδεμένοι μέσω ενός δικτύου με λογισμικό σχεδιασμένο τέτοιο ώστε να παράγει ολοκληρωμένες υπολογιστικές υπηρεσίες. Το λογισμικό κατανεμημένου συστήματος επιτρέπει στις υπολογιστικές μονάδες να συντονίσουν τις δραστηριότητες τους και να διαμοιράσουν τους πόρους του συστήματος (υλικό, λογισμικό και δεδομένα). Οι χρήστες ενός καλά σχεδιασμένου κατανεμημένου συστήματος θα πρέπει να αντιλαμβάνονται μια απλή, ολοκληρωμένη υπολογιστική υπηρεσία, έστω κι αν αυτή έχει υλοποιηθεί από πολλές υπολογιστικές μονάδες σε διαφορετικές τοποθεσίες.

Η ανάπτυξη των κατανεμημένων συστημάτων ακολούθησε την ανάπτυξη των υψηλής ταχύτητας τοπικών δικτύων υπολογιστών στην αρχή της δεκαετίας του 1970. Πιο

πρόσφατα, η διαθεσιμότητα των υψηλής απόδοσης προσωπικών υπολογιστών, σταθμών εργασίας και εξυπηρετητών (servers) είχε σαν αποτέλεσμα μια μέγιστη μετατόπιση προς τη μεριά των κατανεμημένων συστημάτων και απομάκρυνση από τα κεντρικοποιημένα και τα υπολογιστικά συστήματα πολλών χρηστών. Αυτή η κατεύθυνση επιταχύνθηκε με την ανάπτυξη λογισμικού κατανεμημένων συστημάτων, σχεδιασμένου ώστε να υποστηρίζει την ανάπτυξη κατανεμημένων εφαρμογών.

2.3.2 Βασικά χαρακτηριστικά γνωρίσματα

Τα βασικά χαρακτηριστικά γνωρίσματα που είναι πρωταρχικά υπεύθυνα για τη χρησιμότητα των κατανεμημένων συστημάτων είναι τα εξής [29]:

1. Διαμοίραση πόρων: Ο όρος πόρος είναι μάλλον αφαιρετικός, αλλά χαρακτηρίζει καλύτερα την έκταση των πραγμάτων που μπορούν εύχρηστα να διαμοιραστούν σε ένα κατανεμημένο σύστημα. Η έκταση εκτείνεται από στοιχεία υλικού όπως δίσκοι και εκτυπωτές έως στοιχεία λογισμικού όπως αρχεία, παράθυρα, βάσεις δεδομένων και άλλα αντικείμενα δεδομένων.

   Τα πλεονεκτήματα της διαμοίρασης πόρων είναι τα εξής:
   - Συσκευές υλικού όπως εκτυπωτές, μεγάλοι οδηγοί δίσκων και άλλα περιφερειακά διαμοιράζονται κατάλληλα για να ελαττωθεί το κόστος.
   - Η διαμοίραση δεδομένων είναι μια φυσική απαίτηση σε πολλές εφαρμογές υπολογιστών.

2. Επεκτασιμότητα: Η επεκτασιμότητα ενός υπολογιστικού συστήματος είναι το χαρακτηριστικό που καθορίζει εάν ένα σύστημα μπορεί να επεκταθεί με πολλούς τρόπους. Ένα σύστημα μπορεί να είναι ανοικτό ή κλειστό ανάλογα με τις προεκτάσεις (πρόσθεση περιφερειακών, μνήμης ή interfaces επικοινωνίας) ή ανάλογα με τις επεκτάσεις λογισμικού (πρόσθεση χαρακτηριστικών λειτουργικών συστημάτων, πρωτοκόλλων επικοινωνίας και υπηρεσίες διαμοίρασης πόρων). Η επεκτασιμότητα των κατανεμημένων συστημάτων καθορίζεται πρωταρχικά από το βαθμό στον οποίο υπηρεσίες διαμοίρασης πόρων μπορούν να προστεθούν χωρίς σπάσιμο ή διπλασιασμό υπαρχόντων υπηρεσιών.

3. Παραλληλία: Όταν υπάρχουν αρκετές διεργασίες σε ένα υπολογιστή λέμε ότι εκτελούνται ταυτόχρονα. Εάν ο υπολογιστής είναι εξοπλισμένος μόνο με ένα απλό κεντρικό επεξεργαστή, αυτή επιτυγχάνεται με την εναλλαγή της εκτέλεσης τμημάτων της κάθε διεργασίας. Εάν ο υπολογιστής έχει N επεξεργαστές, τότε πάνω από N διεργασίες μπορούν να εκτελούνται ταυτόχρονα, δηλαδή παράλληλα, επιτυγχάνοντας έτσι μια μεγάλη βελτίωση στην υπολογιστική απόδοση.

   Στα κατανεμημένα συστήματα υπάρχουν πολλοί υπολογιστές κάθε ένας με ένα ή περισσότερους κεντρικούς επεξεργαστές. Εάν υπάρχουν K υπολογιστές σε ένα κατανεμημένο σύστημα με ένα κεντρικό επεξεργαστή ο καθένας, τότε περισσότερες από K διεργασίες μπορούν να τρέχουν παράλληλα εξαιτίας του ότι οι διεργασίες τοποθετούνται σε διαφορετικούς υπολογιστές. Σε ένα κατανεμημένο σύστημα που βασίζεται στο μοντέλο διαμοίρασης πόρων, δυνατότητες για παράλληλη εκτέλεση συμβαίνουν για δύο λόγους:

   1. Πολλοί χρήστες ταυτόχρονα καλούν εντολές ή αλληλεπιδρούν με προγράμματα εφαρμογών.
   2. Πολλές διεργασίες-εξυπηρετητές τρέχουν ταυτόχρονα όπου καθεμία αποκρίνεται σε διαφορετικές αιτήσεις από διεργασίες - πελάτες.

   Ο λόγος 1 πηγάζει από μία ή περισσότερες διεργασίες εφαρμογών που τρέχουν για κάθε ένα ενεργό χρήστη. Στις περισσότερες αρχιτεκτονικές κατανεμημένων συστημάτων οι διεργασίες εφαρμογών τρέχουν στο σταθμό εργασίας ενός χρήστη και δεν συγκρούονται για πόρους επεξεργασίας με τις διεργασίες εφαρμογών άλλων χρηστών. Εάν ο σταθμός εργασίας έχει μόνο ένα απλό επεξεργαστή και υπάρχουν περισσότερες από μια διεργασίες εφαρμογών που τρέχουν σε αυτόν, εκτελούνται με ένα εναλλακτικό τρόπο. Οι σταθμοί εργασίας με πολλαπλούς επεξεργαστές επιτρέπουν στους χρήστες να εκτελέσουν προγράμματα εφαρμογών που απασχολούν περισσότερους από ένα επεξεργαστές.

   Ο λόγος 2 πηγάζει από την ύπαρξη μίας ή περισσότερων διεργασιών-εξυπηρετητών για κάθε τύπο πόρου. Αυτές κανονικά τρέχουν σε πρόσθετους υπολογιστές, επιτρέποντας σε κάθε διεργασία-εξυπηρετητή να προχωρά παράλληλα με άλλους

εξυπηρετητές και μαζί με τις διεργασίες που τρέχουν στους σταθμούς εργασίας. Πολυεπεξεργαστικοί υπολογιστές είναι συγκεκριμένα αποτελεσματικοί για την υλοποίηση πολυχρησιμοποιημένων εξυπηρετητών, αφού ένας εξυπηρετητής N επεξεργαστών έχει την ικανότητα να χειρίζεται πάνω από N αιτήσεις πελατών χωρίς καθυστέρηση. Αιτήσεις για πρόσβαση σε πόρους σε ένα συγκεκριμένο εξυπηρετητή μπαίνουν στην ουρά αυτού και επεξεργάζονται ακολουθιακά ή αρκετοί μπορεί να επεξεργαστούν ταυτόχρονα από πολλαπλά αντίγραφα της διεργασίας διαχείρισης πόρου. Όταν αρκετά προγράμματα που χρησιμοποιούν πόρους προσπελαύνουν τον ίδιο πόρο ταυτόχρονα, η διεργασία-εξυπηρετητής πρέπει να συγχρονίσει τις ενέργειες της για να επιβεβαιώσει ότι δεν θα συγκρουστούν. Ο συγχρονισμός πρέπει να σχεδιαστεί προσεκτικά έτσι ώστε να μη χαθούν οι ωφέλειες της παραλληλίας.

Συμπερασματικά, ταυτόχρονη και παράλληλη εκτέλεση αποτελούν σημαντικό θέμα στα κατανεμημένα συστήματα λόγω των ξεχωριστών ενεργειών των χρηστών, της ανεξαρτησίας των πόρων και της τοποθέτησης των διεργασιών-εξυπηρετητών σε διαφορετικούς υπολογιστές. Ο διαχωρισμός αυτών των ενεργειών επιτρέπει στην επεξεργασία να προχωρά παράλληλα σε διαφορετικούς υπολογιστές. Ταυτόχρονες προσπελάσεις και ενημερώσεις σε διαμοιρασμένους πόρους πρέπει να συγχρονίζονται.

4. Διαβαθμισιμότητα: Τα κατανεμημένα συστήματα λειτουργούν αποτελεσματικά και αποδοτικά σε πολλές διαφορετικές διαβαθμίσεις. Το λιγότερο πρακτικό κατανεμημένο σύστημα είναι δυνατό να αποτελείται από δύο σταθμούς εργασίας και ένα εξυπηρετητή αρχείων, ενώ ένα κατανεμημένο σύστημα κατασκευασμένο πάνω σε ένα απλό τοπικό δίκτυο μπορεί να περιλαμβάνει αρκετές εκατοντάδες σταθμών εργασίας και πολλούς εξυπηρετητές αρχείων, εξυπηρετητές εκτύπωσης και άλλους εξυπηρετητές ειδικού σκοπού. Αρκετά τοπικά δίκτυα συχνά διασυνδέονται για να σχηματίσουν υπερσύνολα δικτύων, και αυτά μπορούν να περιλαμβάνουν πολλές χιλιάδες υπολογιστών που σχηματίζουν ένα απλό κατανεμημένο σύστημα επιτρέποντας στους πόρους να μοιράζονται ανάμεσα σε όλους. Η απαίτηση για διαβαθμισιμότητα στα κατανεμημένα συστήματα οδήγησε σε μια σχεδιαστική φιλοσοφία στην οποία κανένας απλός πόρος δεν θεωρείται ότι είναι περιορισμένης

διαθεσιμότητας. Αντίθετα, καθώς η απαίτηση για ένα πόρο αυξάνεται, θα πρέπει να είναι δυνατό να επεκταθεί το σύστημα για να ανταποκριθεί.

5. Ανεκτικότητα βλαβών: Τα υπολογιστικά συστήματα μερικές φορές παθαίνουν βλάβες. Όταν συμβαίνουν βλάβες στο υλικό ή το λογισμικό, τα προγράμματα θα παράγουν λάθος αποτελέσματα ή θα σταματήσουν πριν τελειώσουν τον υπολογισμό που τους έχει ανατεθεί.

   Ο σχεδιασμός ενός υπολογιστικού συστήματος με ανεκτικότητα σε βλάβες βασίζεται σε δύο προσεγγίσεις όπου και οι δύο πρέπει να υιοθετηθούν για τη διαχείριση των σφαλμάτων:
   1. Πλεονασμός υλικού: Η χρήση επιπλέον στοιχείων.
   2. Ανάκαμψη λογισμικού: Ο σχεδιασμός προγραμμάτων για ανάκαμψη από βλάβες.

   Στα κατανεμημένα συστήματα ο πλεονασμός μπορεί να σχεδιαστεί με πιο έξυπνο τρόπο. Αυτόνομοι εξυπηρετητές που είναι ουσιώδεις για τη συνεχόμενη λειτουργία κρίσιμων περιοχών μπορούν να αναπαράγονται.

6. Διαφάνεια: Στη διαφάνεια ορίζεται η απόκρυψη από το χρήστη και τον προγραμματιστή εφαρμογών των ξεχωριστών στοιχείων σε ένα κατανεμημένο σύστημα ώστε το σύστημα να θεωρείται σαν ένα ενιαίο σώμα παρά σαν μια συλλογή ανεξάρτητων στοιχείων. Οι συνέπειες της διαφάνειας είναι ένα μέγιστο θέμα στο σχεδιασμό του λογισμικού συστήματος.

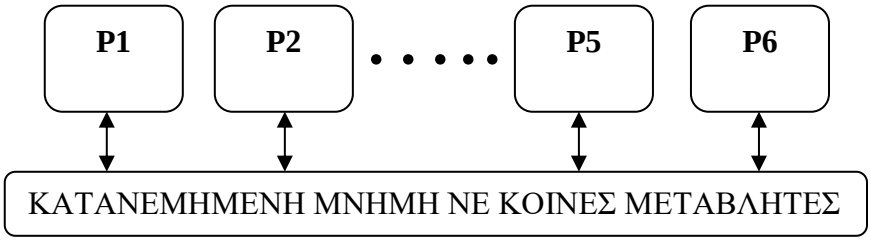
2.3.3 Μοντέλο ανταλλαγής μηνυμάτων

2.3.3.1 Επικοινωνία

Οι επεξεργαστές σε ένα κατανεμημένο σύστημα επικοινωνούν μέσω μιας κοινής μνήμης που περιλαμβάνει ένα σύνολο από κοινές μεταβλητές.
Τύποι μεταβλητών**:**

- Ανάγνωσης και γραφής (read/write)

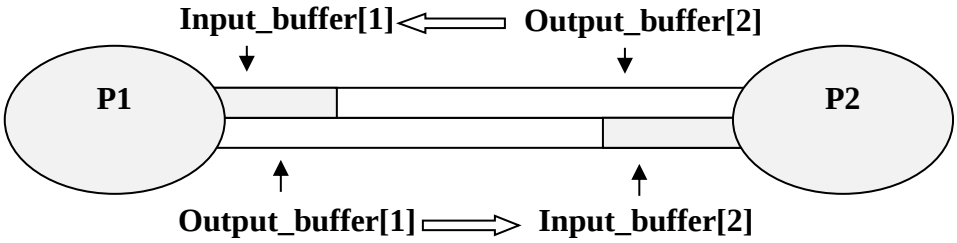
- Ανάγνωσης-Μετατροπής-Γραφής (read-modify-write)
- Ελέγχου και Αλλαγής (test and set)
- Σύγκρισης και Αντιμετάθεσης (compare and swap)



Οι επεξεργαστές επικοινωνούν εσωκλείοντας πληροφορία σε μηνύματα που στέλνονται στο δίκτυο.

Κάθε επεξεργαστής διαθέτει:

1. Λίστα εισερχόμενων μηνυμάτων
2. Λίστα εξερχόμενων μηνυμάτων



2.3.3.2 Χρονισμός

Έχουμε δύο είδη χρονισμού σε ένα κατανεμημένο σύστημα πρακτόρων, το σύγχρονο μοντέλο επικοινωνίας και το ασύγχρονο μοντέλο. Για την επίλυση του προβλήματος που διαπραγματεύεται η διπλωματική αυτή εργασία θα ασχοληθούμε και με τα δύο είδη χρονισμού:

- Σύγχρονο Μοντέλο:
    - Συγχρονισμένα βήματα επεξεργαστών.
    - Ο υπολογισμός προχωράει σε γύρους.

- Υπάρχει γνωστό όριο τερματισμού κάποιου γύρου και όλα τα μηνύματα παραδίδονται με το τέλος του γύρου.

- Ασύγχρονο Μοντέλο:
  - Αυθαίρετες διαφορές στις ταχύτητες των επεξεργαστών.
  - Αυθαίρετες καθυστερήσεις μηνυμάτων/Αυθαίρετη καθυστέρηση πρόσβασης της κοινόχρηστης μνήμης.

2.4 Θεωρητική Ανάλυση της Θεωρίας Παιγνίων

2.4.1 Τι είναι η Θεωρία Παιγνίων

Η Θεωρία Παιγνίων είναι η επίσημη μελέτη της σύγκρουσης και της συνεργασίας, την οποία συνεργασία διαπραγματεύεται η διπλωματική αυτή εργασία. Οι ιδέες της Θεωρίας Παιγνίων εφαρμόζονται κάθε φορά που οι ενέργειες πολλών πρακτόρων είναι αλληλοεξαρτώμενες. Αυτοί οι πράκτορες μπορεί να είναι άτομα, ομάδες, επιχειρήσεις ή οποιοσδήποτε συνδυασμός αυτών. Οι έννοιες της Θεωρίας Παιγνίων παρέχουν μια γλώσσα για τη διαμόρφωση, τη δομή, την ανάλυση και την κατανόηση στρατηγικών σεναρίων [30].

Η εσωτερική συνοχή και οι μαθηματικές βάσεις της Θεωρίας Παιγνίων την κάνουν ένα εξαιρετικό εργαλείο για τη μοντελοποίηση και το σχεδιασμό διαδικασιών λήψης αποφάσεων, οι οποίες είναι αυτοματοποιημένες και λαμβάνουν χώρα σε διαδραστικά περιβάλλοντα. Μπορεί κάποιος, για παράδειγμα, να ήθελε αποτελεσματικούς κανόνες υποβολής προσφορών για μια ιστοσελίδα δημοπρασιών ή απαραβίαστες αυτόματες διαπραγματεύσεις για την αγορά εύρους ζώνης επικοινωνίας. Έρευνα σε αυτές τις εφαρμογές της Θεωρίας Παιγνίων είναι το θέμα της πρόσφατης διάσκεψης ή των άρθρων των περιοδικών, αλλά εξακολουθεί να είναι σε εκκολαπτόμενο στάδιο [30].

Ως ένα μαθηματικό εργαλείο για την λήψη αποφάσεων, η δύναμη της Θεωρίας Παιγνίων είναι η μεθοδολογία που παρέχει τη διάρθρωση και την ανάλυση των προβλημάτων της στρατηγικής επιλογής. Η διαδικασία της τυπικής μοντελοποίησης μια κατάστασης σαν ένα παίγνιο, απαιτεί από τον λήπτη αποφάσεων να απαριθμήσει με σαφήνεια τους

παίκτες καθώς και τις στρατηγικές τους επιλογές, και να λαμβάνει υπόψη του τις προτιμήσεις και τις αντιδράσεις αυτών των παικτών. Η πειθαρχία που εμπλέκεται στην κατασκευή ενός τέτοιου μοντέλου έχει ήδη τη δυνατότητα του να παρέχει ο λήπτης αποφάσεων μια σαφέστερη και ευρύτερη άποψη της κατάστασης. Αυτή είναι μια «καθοδηγητική» εφαρμογή της Θεωρίας Παιγνίων που έχει ως στόχο τη βελτίωση της στρατηγικής της λήψης αποφάσεων. Έχοντας, λοιπόν, αυτήν την προοπτική κατά νου, το άρθρο «Game Theory» των Theodore L. Turocy & Bernhard von Stengel (2001) [30], εξηγεί τις βασικές αρχές της Θεωρίας Παιγνίων ως μία εισαγωγή σε έναν αναγνώστη ο οποίος ενδιαφέρεται να μάθει αλλά δεν έχει κάποιο υπόβαθρο στα οικονομικά [30].

Η Θεωρία Παιγνίων, όπως προαναφέραμε, ασχολείται με τις ενέργειες των φορέων λήψης αποφάσεων οι οποίοι συναισθάνονται ότι οι ενέργειές επηρεάζουν η μία την άλλη. Όταν, για παράδειγμα, υπάρχουν δύο εκδότες σε μία πόλη και μόνο αυτοί οι δύο επιλέγουν τιμές για τις εφημερίδες τους και γνωρίζουν ότι οι πωλήσεις τους καθορίζονται από κοινού, είναι παίκτες σε ένα παίγνιο το οποίο παίζεται μεταξύ τους. Δεν είναι ένα παίγνιο με τους αναγνώστες που αγοράζουν τις εφημερίδες, γιατί κάθε αναγνώστης αγνοεί την επίδρασή του στον εκδότη. Η Θεωρία Παιγνίων δεν είναι χρήσιμη όταν οι αποφάσεις που λαμβάνονται αγνοούν τις αντιδράσεις των άλλων ή όταν τις αντιμετωπίζουν ως απρόσωπες δυνάμεις της αγοράς.

2.4.2 Εφαρμογές στην καθημερινή ζωή

Όπως είδαμε μέχρι τώρα και θα δούμε και παρακάτω, η Θεωρία Παιγνίων έχει μεγάλη γκάμα εφαρμογών. Θα λέγαμε πως όλα έχουν κάποια σχέση με την Θεωρία Παιγνίων αφού έχει εφαρμογές στην οικονομία, στις επιχειρήσεις, στην πληροφορική, στις τηλεπικοινωνίες, στην πολιτική, στην κοινωνιολογία, στη βιολογία και φυσικά στην καθημερινότητα [13]. Μια σύγχρονη μαθηματική θεωρία μπορεί να αναλύσει κάθε είδος αναμέτρησης, από την ντάμα και το σκάκι μέχρι τον τζόγο ή έναν πυρηνικό πόλεμο, και να προβλέψει τον νικητή.

Οι οικονομολόγοι εδώ και πολύ καιρό χρησιμοποιούν τη Θεωρία Παιγνίων για να αναλύσουν διάφορους κλάδους όπως για παράδειγμα η βιομηχανική οργάνωση, ο σχεδιασμός μηχανισμών με υποκλάδο τις δημοπρασίες, τις συμφωνίες, τα ολιγοπώλια,

τα μονοπώλια, (ο Γάλλος μαθηματικός Κουρνό το 1838 έγραψε το πρώτο μοντέλο δυοπωλίου) [14] τα συστήματα για να μπορεί κάποιος να ψηφίσει και πολλά άλλα. Οι έρευνες αυτές για να πραγματοποιηθούν εστιάζουν στην ισορροπία που υπάρχει στα παιχνίδια, την οποία θα σχολιάσουμε παρακάτω.

Επιπρόσθετα η Θεωρία των Παιγνίων παίζει σημαντικό ρόλο στην παγκόσμια διπλωματία και στις πολεμικές στρατηγικές, επηρεάζοντας τη μοίρα των διαφόρων χωρών ακόμη και αν αυτό δεν είναι άμεσα ορατό. Χρησιμοποιείται όμως και στην Πολιτική Οικονομία και ειδικά στη θεωρία της συλλογικής δράσης, όπου εξηγεί ενδεχόμενα συνεργασίας μεταξύ των παικτών. Αυτό βρίσκεται σε άμεση συσχέτιση με τον ρόλο του κράτους και των θεσμών σε θέματα συνεργασίας. Χαρακτηριστικό παράδειγμα είναι η παροχή δημόσιων αγαθών και η φορολογία [15].

Στη βιολογία, η Θεωρία των Παιγνίων έχει χρησιμοποιηθεί για να κατανοήσουμε διάφορα φαινόμενα. Πρωτοχρησιμοποιήθηκε για να εξηγήσει την εξέλιξη (και την σταθερότητα) της αναλογίας 1 προς 1 στα φύλα. Ο Ronald Fisher πρότεινε ότι αυτή η αναλογία είναι αποτέλεσμα εξελικτικών δυνάμεων που δρουν μεμονωμένα, προσπαθώντας να μεγιστοποιήσουν τον αριθμό των εγγονιών [31]. Συμπληρωματικά, οι επιστήμονες προσπάθησαν να εξηγήσουν την εμφάνιση της επικοινωνίας στα ζώα, ενώ ανέλυσαν και την επιθετική συμπεριφορά τους.

Είναι ξεκάθαρο ότι μπορούμε να αναφέρουμε άπειρες εφαρμογές της Θεωρίας Παιγνίων σε διάφορους τομείς ακόμη και στην καθημερινότητα μας, από τα πιο πολύπλοκα έως τα πιο απλά όπως για παράδειγμα το ποιο αυτοκίνητο να αγοράσουμε, που θα πάμε το βράδυ ή τι θα φορέσουμε [16].

2.4.3 Βασικές έννοιες της Θεωρίας Παιγνίων

Θεμέλιο λίθο στην Θεωρία Παιγνίων αποτελούν τα βασικά χαρακτηριστικά του παιγνίου. Ως στοιχεία του παιγνίου θεωρούνται το σύνολο των παικτών, το σύνολο των πιθανών ενεργειών που θα πραγματοποιήσουν οι παίκτες (οι στρατηγικές τους), οι πληροφορίες που υπάρχουν κατά τη διάρκεια του παιχνιδιού, τα αποτελέσματα που μπορεί να αποκομίσει ο παίκτης για κάθε ενέργειά του, καθώς επίσης και οι προτιμήσεις των

παικτών με βάσει τα αποτελέσματα. Το αποτέλεσμα που μπορεί να αποκομίσει ο παίκτης, εξαρτάται από τις στρατηγικές που θα ακολουθήσει και από τις αποδόσεις που μπορεί να λάβει. Η απόδοση, είναι η αριθμητική αποτίμηση των στόχων του, η χρησιμότητα που θα αποκτήσει όταν το παιχνίδι θα τελειώσει [17].

Με τον όρο στρατηγική ορίζουμε το σύνολο των κανόνων σχετικά με το ποια επιλογή πρέπει να ακολουθήσει ο παίκτης, ποιες είναι οι επιλογές του στο κάθε παίγνιο ξεχωριστά, έχοντας όμως υπόψη του και όλες τις κινήσεις του αντιπάλου. Μια διάκριση που μπορεί να γίνει στις στρατηγικές είναι σε αμιγείς και σε μεικτές στρατηγικές. Μια αμιγής (καθαρή) στρατηγική είναι εκείνη στην οποία κάθε μία από τις δυνατές επιλογές που έχει ο παίκτης επιλέγεται στο ακέραιο. Αντίθετα μεικτή είναι η στρατηγική η οποία περιλαμβάνει συνδυασμό επιλογών, από τις οποίες τουλάχιστον μία επιλέγεται με μη ακέραιες τιμές. Οι μεικτές στρατηγικές δηλαδή, καθορίζουν ότι η στρατηγική που θα διαλέξει ο παίκτης θα επιλεγεί τυχαία από το σύνολο των καθαρών στρατηγικών που έχει, με κάποια πιθανότητα. Επομένως μια μεικτή στρατηγική είναι μια κατανομή πιθανοτήτων πάνω στις καθαρές στρατηγικές που έχει ο παίκτης [22]. Ένα παίγνιο στο οποίο οι παίκτες παίζουν ταυτόχρονα, μπορεί να απεικονιστεί ως «κανονική» (normal) ή «στρατηγική» (strategic) μορφή χρησιμοποιώντας έναν πίνακα ο οποίος συσχετίζει τις στρατηγικές των παικτών με τις αποδόσεις που θα έχουν [18].

Ένα στρατηγικό παιχνίδι είναι ένα μοντέλο όπου έχουμε N παίκτες, καθένας από τους οποίους διαλέγει μόνο μία στρατηγική, η οποία δεν αλλάζει. Σε ένα στρατηγικό παιχνίδι υπάρχουν διάφορες συμπεριφορές παικτών:

• Το παιχνίδι παίζεται μόνο μία φορά.

• Κάθε παίκτης «ξέρει» το παιχνίδι, δηλαδή κάθε παίκτης γνωρίζει όλες τις κινήσεις και τις αποδόσεις του παιχνιδιού.

• Οι παίκτες είναι ορθολογικοί. Ένας ορθολογικός παίκτης είναι ένας παίκτης που παίζει εγωιστικά, θέλοντας να μεγιστοποιήσει το κέρδος του στο παιχνίδι, ενώ ταυτόχρονα γνωρίζει πως και οι αντίπαλοι του είναι ορθολογιστές.

• Όλοι οι παίκτες διαλέγουν τις κινήσεις τους ταυτόχρονα χωρίς όμως να γνωρίζουν τις επιλογές των άλλων παικτών [19].

Για να κατανοήσουμε καλύτερα την κανονική μορφή των παιγνίων, παραθέτουμε το πιο κάτω παίγνιο, το οποίο θα χρησιμοποιήσουμε σαν παράδειγμα για να εξηγήσουμε τα στρατηγικά παίγνια.

Πίνακας 2.1 Παίγνιο κυριαρχίας κινδύνου «Risk Dominance» [20]

A \ B	β1	β2
α1	5,5	-100,4
α2	0,1	0,0

Το συγκεκριμένο παίγνιο είναι δύο γραμμών επί δύο στηλών και έχουμε δύο παίκτες, τον Α και τον Β. Ο Α παίκτης ονομάζεται «παίκτης γραμμής», ενώ ο Β «παίκτης στήλης». Οι επικεφαλίδες των στηλών και των γραμμών είναι οι στρατηγικές του κάθε παίκτη. Η πρώτη στρατηγική επιλογή του παίκτη Α είναι η πρώτη γραμμή, η οποία ονομάζεται α1, ενώ η δεύτερη στρατηγική του είναι η α2. Ομοίως για τον παίκτη Β η πρώτη στρατηγική επιλογή του είναι η πρώτη στήλη, δηλαδή η β1, ενώ η δεύτερη στρατηγική του είναι η δεύτερη στήλη, η β2. Στα κελιά του κάθε πίνακα υπάρχουν αριθμοί που δείχνουν το κέρδος (όφελος) κάθε παίκτη για κάθε συνδυασμό στρατηγικών. Το πρώτο νούμερο σε κάθε κελί αντιστοιχεί στον παίκτη γραμμής, ενώ το δεύτερο ανήκει στον παίκτη στήλης.

Το παιχνίδι ξεκινάει και οι παίκτες διαλέγουν ταυτόχρονα μία στρατηγική. Το κελί που αντιστοιχεί στο σημείο τομής των δύο επιλογών δείχνει το κέρδος που έχουν οι δύο παίκτες. Αν για παράδειγμα, ο παίκτης Α διαλέξει την πρώτη στρατηγική επιλογή (α1) και ο Β επίσης την πρώτη (β1) τότε το κέρδος τους θα είναι 5 μονάδες για τον καθένα. Οι παίκτες πριν πάρουν κάποια απόφαση και διαλέξουν ποια στρατηγική θα ακολουθήσουν, κοιτάνε ποια στρατηγική πραγματικά τους ωφελεί, με ποια θα έχουν το μεγαλύτερο δυνατό κέρδος ό,τι και να κάνει ο αντίπαλος τους. Σε αυτό το σημείο η επιλογή γίνεται με βάση την κυριαρχία των στρατηγικών.

Μια στρατηγική λέμε ότι είναι κυρίαρχη «dominant» εάν για όλους τους συνδυασμούς στρατηγικών των άλλων παικτών έχει το μεγαλύτερο όφελος σε σχέση με τις υπόλοιπες. Είναι πάντα καλύτερη ό,τι και να κάνει ο άλλος παίκτης αφού έχει το μεγαλύτερο κέρδος σε σχέση με τις άλλες εναλλακτικές επιλογές του. Αντιθέτως μια στρατηγική

χαρακτηρίζεται ως κυριαρχούμενη «dominated» όταν υπάρχει κάποια άλλη στρατηγική που είναι πάντα καλύτερη ό,τι και να κάνει ο άλλος παίκτης [21].

Στο παραπάνω παράδειγμα βλέπουμε πως για τον παίκτη Β η στρατηγική β1 κυριαρχεί της στρατηγικής β2, αφού (5 > 4) και (1 > 0), δηλαδή αν ο παίκτης Α διαλέξει την α1 στρατηγική, ο Β θα επιλέξει την β1και το ίδιο θα κάνει αν ο Α διαλέξει την α2. Επομένως, η καλύτερη κίνηση του είναι να επιλέξει την β1 στρατηγική. Για τις στρατηγικές του παίκτη Α όμως δεν παρατηρούμε το ίδιο. Αυτό, γιατί αν ο Α ξέρει πως ο Β θα επιλέξει την β1 στρατηγική, τον συμφέρει να διαλέξει την α1, αφού (5 > 0) εάν όμως ο Β διαλέξει την β2, ο Α δεν θα επιλέξει πάλι την α1 αλλά την α2 αφού (-100 < 0). Επομένως, για τον Α παίκτη καμιά στρατηγική δεν κυριαρχεί της άλλης. Αν κάποιος παίκτης έχει κυρίαρχη στρατηγική την ακολουθεί και τότε το παιχνίδι έχει λύση κυρίαρχης στρατηγικής. Όπως είδαμε όμως είναι πολύ πιθανό να μην υπάρχουν πάντα κυρίαρχες στρατηγικές αλλά να υπάρχουν ασθενείς κυριαρχίες.

Μια στρατηγική κυριαρχεί ασθενώς «weakly dominates» εάν για κάθε μία από τις εναλλακτικές στρατηγικές του παίκτη έχει τουλάχιστον ίση απολαβή για όλους τους συνδυασμούς στρατηγικών των υπολοίπων παικτών και καλύτερη απολαβή για τουλάχιστον έναν συνδυασμό στρατηγικών των άλλων παικτών. Όλες οι άλλες εναλλακτικές στρατηγικές ονομάζονται ασθενώς κυριαρχούμενες «weakly dominated strategy». Στο παραπάνω παίγνιο η στρατηγική α1 κυριαρχεί ασθενώς της α2 αφού (5 > -100) και (0 = 0). Ο συνδυασμός των στρατηγικών που επιλέχθηκαν από κάθε παίκτη μας δίνει την έννοια της ισορροπίας «equilibrium». Η ισορροπία στο παίγνιο δηλαδή προέρχεται από τις καλύτερες στρατηγικές μία για κάθε παίκτη στο παιχνίδι [22]. Στο παράδειγμα μας η ισορροπία βρίσκεται στο κελί (α1, β1) δηλαδή στη λύση (5, 5) αφού η καλύτερη επιλογή για τον Α παίκτη είναι η α1, για τον Β παίκτη η β1 και η τομή τους είναι το κελί (α1, β1). Για να βρούμε αυτήν την ισορροπία εάν υπάρχει κυρίαρχη στρατηγική για κάποιον παίκτη τότε επιλέγεται, όπως αναφέραμε και παραπάνω. Σε περίπτωση όμως που δεν υπάρχει, ο περιορισμός των κυριαρχούμενων στρατηγικών «dominated» μπορεί να οδηγήσει στη δημιουργία νέων κυριαρχούμενων στρατηγικών, οι οποίες με τη σειρά τους θα απαλειφθούν κι αυτές. Ξεκινώντας το παιχνίδι διαγράφονται μία μία οι ασθενώς κυριαρχούμενες στρατηγικές από τις επιλογές του παίκτη και αυτό συνεχίζεται μέχρι να βρεθεί μόνο μία στρατηγική για κάθε παίκτη. Η

διαδικασία αυτή ονομάζεται απαλοιφή κυριαρχούμενων στρατηγικών «Iterated Elimination of Dominated Strategies, IEDS». Η διαδικασία αυτή είναι απολύτως λογική αφού και οι παίκτες είναι λογικοί και γνωρίζουν πως και οι αντίπαλοι τους είναι λογικοί γεγονός που δείχνει ότι κανένας από αυτούς δεν θα επιλέξει μια στρατηγική η οποία είναι ασθενώς κυριαρχούμενη. Αν απαλείψουμε μόνο κυριαρχούμενες στρατηγικές, η σειρά της απαλοιφής δεν επηρεάζει το αποτέλεσμα. Ο κίνδυνος υπάρχει μόνο αν απαλείψουμε με λάθος σειρά ασθενώς κυριαρχούμενες στρατηγικές, οδηγώντας μας σε λάθος αποτέλεσμα. Σωστή σειρά θεωρείται η ταυτόχρονη απαλοιφή για όλους τους παίκτες σε κάθε γύρο.

Η σημαντικότερη έννοια ισορροπίας στη Θεωρία Παιγνίων είναι η ισορροπία Nash που θα αναλύσουμε στη συνέχεια του κεφαλαίου αυτού.

2.4.4 Κατηγορίες παιγνίων

Η συγκεκριμένη διπλωματική εργασία διαπραγματεύεται ένα παίγνιο συνεργασίας πρακτόρων (επεξεργαστών) κανονικής μορφής. Πιο κάτω υπάρχει η ανάλυση των δύο κύριων κατηγοριών των παιγνίων που είναι τα παίγνια συνεργασίας ή μη συνεργασίας και τα παίγνια στρατηγικής ή εκτεταμένης μορφής.

1. Παίγνια συνεργασίας και μη συνεργασίας: Τα παίγνια μπορούν να περιγραφούν επίσημα σε διάφορα επίπεδα λεπτομέρειας. Ένα παίγνιο συμμαχίας, ή αλλιώς συνεργασίας είναι μια περιγραφή υψηλού επιπέδου που καθορίζει μόνο τις ανταμοιβές που κάθε πιθανή ομάδα ή συμμαχία μπορεί να αποκτήσει μέσα από τη συνεργασία των μελών της. Αυτό που δεν γίνεται σαφές είναι η διαδικασία με την οποία διαμορφώνεται η συμμαχία. Ως παράδειγμα μπορούμε να πούμε τους παίκτες που μπορεί να είναι διάφορα κόμματα του κοινοβουλίου. Κάθε κόμμα έχει μια διαφορετική δύναμη με βάση τον αριθμό των εδρών που κατέχουν τα μέλη του κόμματος. Το παίγνιο περιγράφει ποιες συμμαχίες κομμάτων μπορούν να σχηματίσουν μια πλειοψηφία αλλά δεν οριοθετείται, για παράδειγμα, η διαδικασία της διαπραγμάτευσης μέσω της οποίας επιτυγχάνεται μια συμφωνία για να ψηφίσουν ομαδικά, ως σύνολο [30].

Η θεωρία παιγνίων συνεργασίας διερευνά τέτοια παίγνια συμμαχίας με σεβασμό προς τις σχετικές ποσότητες δύναμης που κατέχουν οι διάφοροι παίκτες ή το πώς μια επιτυχημένη συμμαχία θα πρέπει να διαιρέσει τα έσοδα των παικτών που την αποτελούν. Αυτό εφαρμόζεται πιο φυσικά σε καταστάσεις που προκύπτουν στην πολιτική επιστήμη ή στις διεθνείς σχέσεις όπου ιδέες, όπως η δύναμη, είναι πιο σημαντικές. Ο Nash, για παράδειγμα, πρότεινε μία λύση για την κατανομή των κερδών μέσα από τη συμφωνία σε ένα πρόβλημα διαπραγμάτευσης το οποίο εξαρτάται αποκλειστικά από τις σχετικές δυνάμεις της διαπραγματευτικής θέσης των δύο μερών. Το ποσό της δύναμης που έχει η μια πλευρά, καθορίζεται συνήθως από το αναποτελεσματικό αποτέλεσμα που προκύπτει όταν οι διαπραγματεύσεις καταρρεύσουν. Το μοντέλο του Nash ταιριάζει εντός του πλαισίου συνεργασίας, υπό την έννοια ότι δεν οριοθετεί ένα συγκεκριμένο χρονοδιάγραμμα των προσφορών και αντιπροσφορών, αλλά επικεντρώνεται αποκλειστικά και μόνο στο αποτέλεσμα της διαπραγματευτικής διαδικασίας [30].

Από την άλλη πλευρά, υπάρχει η θεωρία παιγνίων μη-συνεργασίας. Η θεωρία των παιγνίων αυτών ασχολείται με την ανάλυση των στρατηγικών επιλογών. Οι λεπτομέρειες της παραγγελίας και το χρονοδιάγραμμα των επιλογών των παικτών, παραδειγματίζουν τη θεωρία παιγνίων μη-συνεργασίας και είναι ζωτικής σημασίας για τον προσδιορισμό του αποτελέσματος ενός παιγνίου. Σε αντίθεση με το μοντέλο συνεργασίας του Nash, ένα μοντέλο διαπραγμάτευσης μη-συνεργασίας τοποθετεί μια συγκεκριμένη διαδικασία στην οποία είναι προκαθορισμένο το ποιος μπορεί να κάνει μια προσφορά σε μια δεδομένη στιγμή. Κάτι το οποίο γίνεται συχνά είναι το γεγονός ότι η συνεργασία μπορεί να προκύψει σε μοντέλα παιγνίων μη-συνεργασίας, όταν οι παίκτες βρουν ότι το να συνεργαστούν είναι προς το δικό τους συμφέρον [30].

Συγκεντρώνοντας τα παραπάνω, καταλήγουμε στο συμπέρασμα ότι ένα παίγνιο συνεργασίας είναι ένα παίγνιο στο οποίο οι παίκτες μπορούν να διαπράξουν δέσμευση, σε αντίθεση με ένα παίγνιο μη-συνεργασίας στο οποίο δεν μπορούν. Ο ορισμός αυτός σκιαγραφεί τη διαφορά που υπάρχει ανάμεσα στις δύο θεωρίες παιγνίων, αλλά η πραγματική διαφορά βρίσκεται στην προσέγγιση του μοντέλου. Και οι δύο θεωρίες ξεκινούν με τους κανόνες των παιγνίων, αλλά διαφέρουν στο είδος της λύσης που εφαρμόζεται. Η θεωρία παιγνίων συνεργασίας είναι αποφθεγματική,

συχνά προσφεύγει στην ορθολογικότητα του Pareto, είναι δίκαιη και αμερόληπτη. Η θεωρία παιγνίων μη-συνεργασίας βασίζεται σε ιδέες των οποίων η λύση βασίζεται στο να μεγιστοποιήσουν οι παίκτες τη δική τους συνάρτηση χρησιμότητας, η οποία στηρίζεται σε κάποιους περιορισμούς [32].

Οι υποθέσεις που υπάρχουν στους κλάδους της Θεωρίας Παιγνίων επίσης διαφέρουν. Υπάρχει όμως μια κεντρική υπόθεση σε πολλές παραλλαγές της Θεωρίας Παιγνίων και αυτή είναι το ότι οι παίκτες είναι ορθολογικοί. Ως ορθολογικός χαρακτηρίζεται ένας παίκτης ο οποίος επιλέγει πάντα μία ενέργεια η οποία του δίνει το αποτέλεσμα που προτιμά περισσότερο, δεδομένου του τι αναμένει να κάνουν οι αντίπαλοί του. Όπως, λοιπόν, γίνεται αντιληπτό, ο στόχος της θεωρητικής ανάλυσης των παιγνίων σε αυτούς τους κλάδους είναι να προβλέψει το πώς το παίγνιο θα παιχτεί από τους ορθολογικούς παίκτες ή να δώσει συμβουλές για το ποιος είναι ο καλύτερος τρόπος για να παίξει κάποιος το παίγνιο ενάντια στους αντιπάλους του, οι οποίοι είναι και αυτοί ορθολογικοί [30].

2. Παίγνια στρατηγικής και εκτεταμένης μορφής: Η Θεωρία Παιγνίων μπορεί να χρησιμοποιήσει δύο τρόπους για την αναπαράσταση των παιγνίων. Ο ένας τρόπος είναι με μήτρες, δηλαδή με στρατηγική μορφή, η οποία αποτελεί την πιο συνηθισμένη μορφή του παιγνίου και ο άλλος τρόπος είναι με δέντρα, δηλαδή με εκτεταμένη μορφή. Ας δούμε αναλυτικά τους δύο αυτούς τρόπους αναπαράστασης.

   Η στρατηγική μορφή, που ονομάζεται επίσης και κανονική μορφή, είναι ο βασικός τύπος παιγνίου που μελετάται στη θεωρία παιγνίων μη-συνεργασίας. Ένα παίγνιο σε στρατηγική μορφή παραθέτει τις στρατηγικές του κάθε παίκτη καθώς και τα αποτελέσματα που προκύπτουν από κάθε πιθανό συνδυασμό των στρατηγικών του. Αυτό μπορεί να γίνει με τη μορφή μήτρας, όπου σε κάθε γραμμή υπάρχει η στρατηγική του Παίκτη Ι και σε κάθε στήλη η στρατηγική του Παίκτη ΙΙ. Ένα αποτέλεσμα αντιπροσωπεύεται από μια ανταμοιβή, η οποία είναι ξεχωριστή για κάθε παίκτη και είναι ένας αριθμός ο οποίος μετρά το κατά πόσο αρέσει στον παίκτη το αποτέλεσμα που προέκυψε [33].

Από την άλλη πλευρά, η εκτεταμένη μορφή, που ονομάζεται επίσης και αναλυτική μορφή, είναι πιο λεπτομερής από ότι η στρατηγική μορφή παιγνίου. Πρόκειται για μια πλήρη περιγραφή για το πώς παίζεται ένα παίγνιο με την πάροδο του χρόνου και εμφανίζει όλες τις λεπτομέρειες της αλληλεπίδρασης των παικτών. Περιλαμβάνει τη σειρά με την οποία οι παίκτες αναλαμβάνουν ενέργειες, τις πληροφορίες που οι παίκτες έχουν κατά το χρόνο στον οποίο πρέπει να λάβουν τις εν λόγω ενέργειες καθώς και τις ώρες κατά τις οποίες έχει επιλυθεί η αβεβαιότητα της κατάστασης. Ένα παίγνιο σε εκτεταμένη μορφή μπορεί να αναλυθεί άμεσα ή μπορεί να μετατραπεί σε ένα ισοδύναμο παίγνιο στρατηγικής μορφής.

2.4.5 Προσέγγιση της ισορροπίας Nash

Η ισορροπία Nash είναι μια θεμελιώδης ιδέα στη Θεωρία των Παιγνίων και η πιο ευρέως χρησιμοποιούμενη μέθοδος πρόβλεψης του αποτελέσματος μιας στρατηγικής αλληλεπίδρασης στις κοινωνικές επιστήμες. Ένα παιχνίδι (σε στρατηγική ή κανονική μορφή) αποτελείται από τα ακόλουθα τρία στοιχεία: ένα σύνολο παικτών, ένα σύνολο ενεργειών (ή καθαρών στρατηγικών) που διατίθενται σε κάθε παίκτη και μια λειτουργία πληρωμής (ή χρησιμότητα) για κάθε παίκτη. Οι λειτουργίες πληρωμής αντιπροσωπεύουν τις προτιμήσεις κάθε παίκτη σε σχέση με τα προφίλ δράσης, όπου ένα προφίλ δράσης είναι απλά μια λίστα ενεργειών, μία για κάθε παίκτη. Μια ισορροπία Nash με καθαρή στρατηγική είναι ένα προφίλ δράσης με την ιδιότητα ότι κανένας παίκτης δεν μπορεί να αποκτήσει υψηλότερη απόδοση αποκλίνοντας μονομερώς από αυτό το προφίλ. Η ιδέα αυτή μπορεί να γίνει καλύτερα κατανοητή εξετάζοντας μερικά παραδείγματα. Σκεφτείτε αρχικά ένα παιχνίδι που περιλαμβάνει δύο παίκτες, καθένα από τους οποίους έχει δύο διαθέσιμες ενέργειες, τις οποίες ονομάζουμε Α και Β. Αν οι παίκτες επιλέξουν διαφορετικές ενέργειες, παίρνουν πληρωμή 0. Αν και οι δύο επιλέξουν Α, παίρνουν ο καθένας 2 και αν και οι δύο επιλέξουν Β, παίρνουν ο καθένας 1. Αυτό το παιχνίδι «συντονισμού» μπορεί να εκπροσωπείται ως εξής, όπου ο παίκτης 1 επιλέγει μια σειρά, ο παίκτης 2 επιλέγει μια στήλη και οι προκύπτουσες αποδόσεις (πληρωμές) παρατίθενται σε παρενθέσεις, με το πρώτο στοιχείο να αντιστοιχεί στην πληρωμή του παίκτη 1: Το προφίλ δράσης (Β, Β) είναι μια ισορροπία, αφού μια μονομερής απόκλιση από τον Α από οποιονδήποτε παίκτη θα είχε ως αποτέλεσμα χαμηλότερη απόδοση για τον αποκλίνοντα παίκτη. Ομοίως, το προφίλ δράσης (Α, Α) είναι επίσης μια ισορροπία.

Κεφάλαιο 3

Μοντέλο Επίλυσης Κατανεμημένων Προβλημάτων και Προηγούμενη Εργασία

3.1 Μοντέλο Υπολογισμού 36
3.2 Θεωρητικό Μοντέλο Παιγνίων 39
3.3 Ενεργοποίηση (ξύπνημα) Πρακτόρων 42
3.3.1 Πρωτόκολλο ενεργοποίησης πρακτόρων σε δακτύλιο 44
3.4 Διαμοίραση Γνώσης 45
3.4.1 Διαμοίραση γνώσης σε ένα σύγχρονο δακτύλιο 48
3.4.2 Διαμοίραση γνώσης σε ένα ασύγχρονο δακτύλιο 48
3.5 Εκλογή Αρχηγού Πρακτόρων 49
3.6 Μετονομασία 49
3.7 Συμφωνία 51
3.8 Προηγούμενες Εργασίες 53
3.8.1 Η μελέτη των Afek at al. 53
3.8.2 Προηγούμενη Διπλωματική Εργασία 56

3.1 Μοντέλο Υπολογισμού

Το μοντέλο μας αποτελείται από n επεξεργαστές σε ένα κατανεμημένο δίκτυο. Κάθε επεξεργαστής είναι ένας ορθολογικός πράκτορας και έτσι έχει προτιμήσεις ως προς το αποτέλεσμα του πρωτοκόλλου. Κάθε φορά που ένας επεξεργαστής μπορεί να αυξήσει την αναμενόμενη χρησιμότητά του, θα αποκλίνει από οποιοδήποτε δεδομένο πρωτόκολλο, προκειμένου να επωφεληθεί. Προκειμένου τα πρωτόκολλα να είναι ανθεκτικά σε ορθολογικούς πράκτορες, τους ζητάμε να φτάσουν σε ισορροπία, έτσι ώστε

οι πράκτορες να μην έχουν κίνητρο να αποκλίνουν από το πρωτόκολλο. Για να επιτευχθεί αυτό, απαιτούμε από τους πράκτορες να έχουν μια συνάρτηση χρησιμότητας που να ικανοποιεί την προτίμηση λύσης, έτσι ώστε οι πράκτορες να μην προτιμούν περισσότερο ένα αποτέλεσμα του πρωτοκόλλου όπου δεν αποτελεί λύση στο πρόβλημα παρά ένα αποτέλεσμα στο οποίο υπάρχει μια λύση.

Χρησιμοποιούμε το μοντέλο ανταλλαγής μηνύματος, όπου όλοι οι επεξεργαστές είναι ορθολογικοί πράκτορες. Το δίκτυο είναι ένα απλό, ισχυρά συνδεδεμένο, πεπερασμένο γράφημα. Κάθε κόμβος αντιπροσωπεύει έναν πράκτορα (δηλαδή έναν επεξεργαστή) και οι ακμές αντιπροσωπεύουν συνδέσεις επικοινωνίας μέσω των οποίων τα ζεύγη πρακτόρων ανταλλάσσουν μηνύματα. Οι πράκτορες μπορούν να στέλνουν μηνύματα μέσω των εξερχόμενων συνδέσεών τους και μπορούν να αναγνωρίσουν τη σύνδεση από την οποία έχουν λάβει ένα εισερχόμενο μήνυμα. Τα μηνύματα που αποστέλλονται μέσα σε ένα δίκτυο πρακτόρων ακολουθούν την πολιτική FIFO, δηλαδή το πρώτο μήνυμα που εισέρχεται του δικτύου και στέλνεται από τον πράκτορα που αναλαμβάνει να ξεκινήσει τη διαδικασία επιλογής μιας τιμής συμφωνίας, είναι και το πρώτο που εξέρχεται από τον τελευταίο πράκτορα του δικτύου. Στη παρούσα διπλωματική εργασία η τοπολογία του δικτύου των πρακτόρων που χρησιμοποιείται είναι ένας δακτύλιος. Δηλώνουμε σαν n τον συνολικό αριθμό των πρακτόρων στο δίκτυο. Κάθε πράκτορας έχει μια μοναδική ταυτότητα id που λαμβάνεται από ένα σύνολο των φυσικών αριθμών. Υποθέτουμε ότι η τοπολογία του δικτύου και το n είναι κοινές πληροφορίες, επομένως είναι γνωστές σε όλους. Κάθε πράκτορας γνωρίζει επιπλέον την ταυτότητα id και την είσοδο $input$ του, αλλά όχι την ταυτότητα id ή την είσοδο $input$ οποιουδήποτε άλλου πράκτορα.

Σε αυτή την διπλωματική εργασία, εξετάζονται τόσο τα σύγχρονα όσο και τα ασύγχρονα μοντέλα δικτύου. Σε ένα σύγχρονο δίκτυο, οι πράκτορες εκτελούν το πρωτόκολλο σε γύρους. Κάθε γύρος r αποτελείται από πράκτορες που λαμβάνουν όλα τα μηνύματα που στέλνονται στους εισερχόμενους συνδέσμους τους στο γύρο $r-1$ (αν $r>0$), εκτελούν οποιονδήποτε υπολογισμό και ενημέρωση εσωτερικών μεταβλητών και στέλνουν μηνύματα στους εξερχόμενους συνδέσμους τους. Στο ασύγχρονο δίκτυο η καθυστέρηση μετάδοσης μηνυμάτων είναι απεριόριστη αλλά πεπερασμένη. Τόσο στο σύγχρονο όσο και στο ασύγχρονο δίκτυο, κάθε πράκτορας υποχρεούται να απαντήσει σε ένα μήνυμα-ερώτηση του προκατόχου του με ένα μήνυμα-απάντηση, ανεξάρτητα από το αν θέλει να

αποκλίνει από τον αλγόριθμο παροχής μιας Ισχυρής Ισορροπίας ή όχι. Ένα παράδειγμα απόκλισης του από τον αλγόριθμο είναι όταν το μήνυμα που υποχρεούται να στείλει, το στέλνει κενό ή τοποθετεί στο μήνυμα του μια τιμή εισόδου $input$ που δεν ανήκει στο σύνολο $\{0,1\}$.

Τρία βασικά κατανεμημένα προβλήματα πληροφορικής εξετάζονται εδώ: μετονομασία, συμφωνία και εκλογή αρχηγού πρακτόρων (ηγέτη, $leader$).

Το πρόβλημα της συμφωνίας: κάθε πράκτορας p ξεκινάει με μια είσοδο i_p, και η έξοδος του o_p αρχικοποιείται σε $\perp$. Οι επεξεργαστές πρέπει να συμφωνήσουν σχετικά με την έξοδο. Δηλαδή, οι επεξεργαστές επικοινωνούν, και στο τέλος της εκτέλεσης του πρωτοκόλλου πρέπει να ικανοποιούνται τα τρία πιο κάτω κριτήρια:

(1) Πληρότητα (Integrity): $\exists j \forall i \in \{1, \dots, n\}, o_i \neq \perp$ και παρέχεται μια κοινή τιμή συμφωνίας, δηλαδή για κάθε επεξεργαστή p, $o_p = v$ για κάποια τιμή v.

(2) Εγκυρότητα (Validity): αν για $o_j = v$, τότε υπάρχει κάποιος επεξεργαστής k για τον οποίο $i_k = v$.

(3) Τερματισμός (Termination): Ο αλγόριθμος τερματίζει για οποιεσδήποτε τιμές εισόδου που δίνονται στους πράκτορες.

Το πρόβλημα της εκλογής $leader$ είναι το ίδιο με το πρόβλημα της συμφωνίας όπου το $input$ κάθε πράκτορα είναι το id του.

Το πρόβλημα μετονομασίας: κάθε πράκτορας έχει ένα μοναδικό id και πρέπει να επιλέξει ένα μοναδικό όνομα από το εύρος $1 \dots n$. Ορίζουμε το o_p να είναι η έξοδος του πράκτορα. Κάθε πράκτορας p γράφει το νέο όνομά του στο o_p, ικανοποιώντας τις ακόλουθες απαιτήσεις:

(1) Συμφωνία: Δεν υπάρχουν δύο επεξεργαστές που να αποκτούν το ίδιο νέο όνομα.
$\rightarrow \forall_{x,y} o_x \neq o_y$

(2) Ισχύς: Κάθε νέο όνομα είναι ένας ακέραιος αριθμός στο σύνολο $[1 \dots . n]$.

3.2 Θεωρητικό Μοντέλο Παιγνίων

Οι ορθολογικοί πράκτορες δεν είναι ελαττωματικοί και έχουν προτιμήσεις όσο αφορά το αποτέλεσμα του πρωτοκόλλου. Εάν ένας πράκτορας μπορεί να βελτιώσει τις προτιμήσεις του, θα αποκλίνει από το πρωτόκολλο, δηλαδή θα εξαπατήσει τους υπόλοιπους πράκτορες συμπαίχτες του). Για παράδειγμα, ένας ορθολογικός πράκτορας p μπορεί να προτιμά να στέλνει λιγότερα μηνύματα στο πρωτόκολλο. Σε μια τέτοια περίπτωση, μπορεί να είναι τεμπέλης και να μην συμμετέχει πλήρως στο πρωτόκολλο ή να αποκλίνει με οποιονδήποτε τρόπο έτσι ώστε ο αναμενόμενος αριθμός μηνυμάτων που αποστέλλονται να είναι όσο το δυνατόν λιγότερος, αρκεί να μην προκαλέσει την αποτυχία του πρωτοκόλλου.

Τυπικά, κάθε πράκτορας p έχει μια συνάρτηση χρησιμότητας u_p στις τελικές καταστάσεις του πρωτοκόλλου. Η τελική κατάσταση του πρωτοκόλλου περικλείει την εκτέλεση που οδηγεί στην κατάσταση που μπορεί να δει ο p και στην έξοδο του πρωτοκόλλου. Η συνάρτηση χρησιμότητας αντιπροσωπεύει το πόσο κερδοφόρο είναι το αποτέλεσμα σύμφωνα με την άποψη του p. Όσο μεγαλύτερη είναι η χρησιμότητα, τόσο το καλύτερο. Ένας πράκτορας μπορεί να έχει οποιαδήποτε συνάρτηση χρησιμότητας που να αντιπροσωπεύει οποιαδήποτε προτίμησή του. Μπορεί να προτιμήσει τη τιμή εξόδου του αλγορίθμου, τον αριθμό των μηνυμάτων που έχει στείλει, το ποσό του υπολογισμού που κάνει ή οποιαδήποτε άλλη προτίμηση ή συνδυασμό προτιμήσεων, εφόσον ο αλγόριθμος τερματίζεται σε μια νόμιμη κατάσταση, ικανοποιώντας τη προτίμηση της λύσεως (ορισμός 3.2.1 πιο κάτω) [1].

Έστω c η κατάσταση του p στην εκτέλεση του πρωτοκόλλου και έστω το S_p να είναι το σύνολο όλων των πιθανών τελικών καταστάσεων που μπορεί να είναι προσπελάσιμες και συνεπείς με την τρέχουσα κατάσταση του p, υποθέτοντας ότι όλοι οι πράκτορες εκτός από τον p ακολουθούν το δεδομένο πρωτόκολλο και το επόμενο βήμα του p είναι το o_p, η οποία πράξη μπορεί να αποκλίνει από το πρωτόκολλο (δηλαδή, εξαπάτηση). Για κάθε τελική κατάσταση $s \in S_p$, με x_s να είναι η πιθανότητα, υπολογιζόμενη από το p, ότι το s επιτυγχάνεται λαμβάνοντας το βήμα o_p. Η αναμενόμενη χρησιμότητα του p μετά τη λήψη του βήματος o_p στην κατάσταση c είναι τότε: $E_{c,op}[u_p] = \sum_{\forall s \in S_p} x_s . u_p(s)$.

Εάν με το να αποκλίνει από το πρωτόκολλο κάνοντας ένα βήμα o_p, ο p μπορεί να αυξήσει την αναμενόμενη χρησιμότητα του, τότε λέμε ότι ο πράκτορας p έχει κίνητρο να αποκλίνει από το πρωτόκολλο (δηλαδή να εξαπατήσει).

Για παράδειγμα, στο πρόβλημα της συμφωνίας, ένας πράκτορας p μπορεί να έχει προτίμηση πάνω στη τιμή της συμφωνίας, έτσι ώστε να προτιμά μια συμφωνία στη τιμή 1 έναντι μιας συμφωνίας στη τιμή 0. Η συνάρτηση χρησιμότητάς του μπορεί να είναι ως εξής:

$$u_p = \begin{cases} 1, & \text{Αποφασίζεται το 1} \\ 0, & \text{Αποφασίζεται το 0 ή το πρωτόκολλο μας δεν φθάνει σε κάποια συμφωνία} \end{cases}$$

Σε κάθε σημείο κατά τη διάρκεια εκτέλεσης του πρωτοκόλλου, ο πράκτορας p υπολογίζει την πιθανότητα για κάθε πιθανό αποτέλεσμα του πρωτοκόλλου, σύμφωνα με τις γνώσεις του, ενώ υποθέτει ότι όλοι οι άλλοι πράκτορες ακολουθούν το δεδομένο πρωτόκολλο. Έστω x η πιθανότητα υπολογιζόμενη από τον p, ότι το αποτέλεσμα του πρωτοκόλλου έχει ως αποτέλεσμα μια συμφωνία με τιμή 1. Τότε η αναμενόμενη χρησιμότητα του p είναι η ακόλουθη:

$$E[u_p] = x * u_p(1) + (1 - x) * u_p(0)$$

Δεδομένου ότι το p είναι ένας ορθολογικός πράκτορας, θα αποκλίνει από το πρωτόκολλο όποτε μπορεί να αυξήσει την αναμενόμενη χρησιμότητα του από οποιαδήποτε θετική τιμή.

Για να εμποδίσουμε τους ορθολογικούς πράκτορες να προκαλέσουν την αποτυχία του αλγορίθμου, υποθέτουμε ότι οι πράκτορες έχουν συναρτήσεις χρησιμότητας που τιμωρούν τους πράκτορες εάν το πρωτόκολλο σφάλλει. Η προτίμηση λύσης εγγυάται ότι ένας πράκτορας δεν έχει ποτέ κίνητρο να προκαλέσει την αποτυχία του πρωτοκόλλου [1].

ΟΡΙΣΜΟΣ 3.2.1 (ΠΡΟΤΙΜΗΣΗ ΛΥΣΗΣ): Έστω O να είναι το σύνολο όλων των πιθανών εκτελέσεων του πρωτοκόλλου. Έστω $o_L \in O$ να είναι μια νόμιμη εκτέλεση του

πρωτοκόλλου (δηλαδή να παράγει μια νόμιμη έξοδο) και έστω $o_E \in O$ να είναι μια εσφαλμένη εκτέλεση του πρωτοκόλλου (δηλαδή να παράγει μια παράνομη έξοδο). Για να ικανοποιηθεί η προτίμηση λύσης, η συνάρτηση χρησιμότητας ενός πράκτορα p πρέπει να ικανοποιεί το εξής:

$$\forall_{o_L,o_E}: u_p(o_L) \geq u_p(o_E)$$

Αυτό σημαίνει ότι η συνάρτηση χρησιμότητας u_pενός πράκτορα p δεν αποδίδει ποτέ μεγαλύτερη χρησιμότητα σε ένα αποτέλεσμα του πρωτοκόλλου στο οποίο δεν υπάρχει λύση ή η λύση του είναι εσφαλμένη παρά σε ένα αποτέλεσμα στο οποίο υπάρχει μια νόμιμη λύση. Λέμε ότι ένα πρωτόκολλο είναι ανθεκτικό στην ορθολογική συμπεριφορά εάν φτάσει στην ισορροπία Nash. Ένα πρωτόκολλο που φτάνει στην ισορροπία Nash είναι εγγυημένο ότι εκτελείται σωστά απέναντι σε ορθολογικούς πράκτορες, χωρίς κανένας πράκτορας να μπορεί να βελτιώσει τη χρησιμότητά του με απόκλιση από το πρωτόκολλο [1].

ΟΡΙΣΜΟΣ 3.2.2 (ΠΡΩΤΟΚΟΛΛΟ ΙΣΟΡΡΟΠΙΑΣ NASH): Ένα πρωτόκολλο λέγεται ότι φθάνει σε ισορροπία Nash εάν κανένας πράκτορας δεν μπορεί μονομερώς να αυξήσει την αναμενόμενη χρησιμότητά του με απόκλιση από το πρωτόκολλο ενώ υποθέτει ότι όλοι οι άλλοι πράκτορες ακολουθούν το δεδομένο πρωτόκολλο.

Ενώ η ισορροπία Nash ασχολείται με έναν μόνο πράκτορα που αποκλίνει από το πρωτόκολλο, μερικές φορές οι πράκτορες βρίσκονται σε συμμαχίες, συνασπισμούς (coalitions), δουλεύοντας μαζί για να βελτιώσουν τις υπηρεσίες τους (δηλαδή τις συναρτήσεις χρησιμότητας τους). Ορίζουμε έναν συνασπισμό μεγέθους t ως ένα σύνολο από t ορθολογικούς πράκτορες. Λέμε ότι ένα πρωτόκολλο είναι ανθεκτικό ενάντια στους συνασπισμούς αν φτάσει σε Ισχυρή Ισορροπία. Ένα πρωτόκολλο που φτάνει σε
t - Ισχυρή - Ισορροπία είναι εγγυημένο να εκτελεστεί σωστά, χωρίς συνασπισμό με k πράκτορες, όπου $k \leq t$, έχοντας τη δυνατότητα να βελτιώσει τη χρησιμότητά του με απόκλιση από το πρωτόκολλο, ακόμη και με ένα συντονισμένο τρόπο [1].

ΟΡΙΣΜΟΣ 3.2.3 (ΠΡΩΤΟΚΟΛΛΟ t - ΙΣΧΥΡΗΣ - ΙΣΟΡΡΟΠΙΑΣ): Ένα πρωτόκολλο φτάνει σε t - Ισχυρή - Ισορροπία αν δεν υπάρχει συνασπισμός μεγέθους $k \leq t$ που να μπορεί να αυξήσει την αναμενόμενη χρησιμότητα ενός ή περισσοτέρων πρακτόρων στον

συνασπισμό με απόκλιση από το πρωτόκολλο, υποθέτοντας ότι όλοι οι πράκτορες που δεν ανήκουν στον συνασπισμό ακολουθούν το δεδομένο πρωτόκολλο.

Στη Θεωρία των Παιγνίων, ένα παιχνίδι απαιτεί τρία πράγματα: τους παίκτες, τις συναρτήσεις χρησιμότητας των παικτών και τις διαθέσιμες στρατηγικές για κάθε παίκτη. Το κατανεμημένο μοντέλο με τους ορθολογικούς πράκτορες που συζητούμε σε αυτή τη διπλωματική εργασία, εκπληρώνει αυτό τη θεωρία αυτή με τον ακόλουθο τρόπο: 1) Το σύνολο των παικτών P είναι το σύνολο όλων των πρακτόρων (δηλαδή των επεξεργαστών) στο δίκτυο. 2) Κάθε παίκτης $p \in P$ έχει μια συνάρτηση χρησιμότητας u_p, η οποία μπορεί να είναι οποιαδήποτε λειτουργία που ικανοποιεί την προτίμηση του λήμματος (ορισμός 3.2.1). 3) Η στρατηγική κάθε παίκτη p καθορίζει τα μηνύματα που στέλνει ο p. Ο παίκτης p μπορεί να επιλέξει να στείλει κανένα, ένα ή περισσότερα μηνύματα σε οποιοδήποτε σημείο του πρωτοκόλλου και εκτελεί την επιλεγμένη στρατηγική του [1].

3.3 Ενεργοποίηση (ξύπνημα) Πρακτόρων

Σε πολλά κατανεμημένα πρωτόκολλα υπολογιστών, υποθέτουμε ότι όλοι οι πράκτορες ξεκινούν το πρωτόκολλο ταυτόχρονα. Ωστόσο, αυτό δεν συμβαίνει συχνά. Συνήθως, οι επεξεργαστές είτε ξυπνούν αυθόρμητα σε αυθαίρετα χρονικά σημεία από τον προγραμματιστή είτε όταν λαμβάνουν το πρώτο μήνυμα μέσω ενός από τους συνδέσμους εισόδου τους.

Το block αφύπνισης ενός πρωτοκόλλου ασχολείται με την αφύπνιση όλων των πρακτόρων. Το block αυτό μπορεί να εκτελεστεί ως η πρώτη φάση οποιουδήποτε πρωτοκόλλου, εξασφαλίζοντας έτσι ότι όλοι οι πράκτορες είναι ξύπνιοι και ξεκινούν την λειτουργία τους από κοινού. Ένα πρωτόκολλο που αποσκοπεί σε συμφωνία ανάμεσα σε n πράκτορες και παρέχει μια k - Ισχυρή - Ισορροπία, ξεκινά με την αφύπνιση των πρακτόρων σε αυθαίρετα χρονικά σημεία από τον προγραμματιστή. Οι πράκτορες πρέπει να επικοινωνούν έτσι ώστε, στο τέλος του πρωτοκόλλου, όλοι οι πράκτορες να είναι ξύπνιοι. Το πρωτόκολλο διασφαλίζει ότι, κατά τον τερματισμό, οι πράκτορες γνωρίζουν επίσης τα αναγνωριστικά ids όλων των άλλων πρακτόρων. Επιπλέον, το πρωτόκολλο πρέπει να πετύχει απέναντι σε ορθολογικούς πράκτορες και έτσι φτάνει σε t - Ισχυρή -

Ισορροπία για $t < n$ [1]. Το πρωτόκολλο μπορεί να αποτύχει παρουσία τεμπέληδων ορθολογικών πρακτόρων, που προτιμούν να στέλνουν λιγότερα μηνύματα, με τον ακόλουθο τρόπο:

Έστω μια διαδικασία αφύπνισης που απαιτεί από τους πράκτορες να στέλνουν και να προωθούν μηνύματα γύρω από το δακτύλιο, όπου προωθούνται μόνο μηνύματα του πράκτορα με το υψηλότερο id. Έστω ένας πράκτορας p με προτίμηση για την αποστολή λιγότερων μηνυμάτων που έχει την ακόλουθη συνάρτηση χρησιμότητας, όπου m_p είναι ο αριθμός των μηνυμάτων που ο p στέλνει:

$$u_p = \begin{cases} 1 - \dfrac{m_p}{n} & \text{Το πρωτόκολλο αφύπνισης έγινε με επιτυχία} \\ 0 & \text{Το πρωτόκολλο αφύπνισης έγινε με αποτυχία} \end{cases}$$

Στη διαδικασία, κάθε πράκτορας δεν στέλνει περισσότερα από n μηνύματα, για να ικανοποιεί την προτίμηση του λύματος (ορισμός 3.2.1). Όταν ο p ξυπνά, προτιμά να μην στείλει κανένα μήνυμα αφύπνισης. Στηρίζεται σε έναν άλλο πράκτορα για να ξυπνήσει τους άλλους, αυξάνοντας έτσι την αναμενόμενη χρησιμότητα του. Επιπλέον, εάν ο p είναι ο μόνος πράκτορας που ξυπνά αυθόρμητα, δεν θα επιτευχθεί λύση και το πρωτόκολλο θα αποτύχει. Παρόλο που είναι πιθανό ότι δεν θα επιτευχθεί λύση, η χρησιμότητα κάποιου πράκτορα p u_p εξυπηρετεί την προτίμηση λύσης (ορισμός 3.2.1). Αυτό οφείλεται στο ότι αυξάνεται μόνο η αναμενόμενη χρησιμότητα του p, οπότε ο p «θα πάρει την ευκαιρία» αποτυχίας του πρωτοκόλλου και δεδομένης της θετικής πιθανότητας να μην αποτύχει, ο p στο τέλος θα επωφεληθεί. Το πρωτόκολλο αφύπνισης πρέπει να είναι ανθεκτικό στη συνάρτηση χρησιμότητας που περιγράφεται στο παραπάνω παράδειγμα και σε οποιαδήποτε άλλη λειτουργία χρησιμότητας που ικανοποιεί την προτίμηση λύσης (ορισμός 3.2.1).

Μια άλλη πιθανή προτίμηση που μπορεί να έχει ένας πράκτορας είναι η προτίμηση πάνω στις ιδιότητες του id του. Ενώ το ίδιο το block αφύπνισης αγνοεί τα ids των πρακτόρων, μπορεί να χρησιμοποιηθεί από ένα πρωτόκολλο που δεν τα αγνοεί. Για παράδειγμα, σκεφτείτε ένα πρωτόκολλο εκλογής $leader$ που τρέχει πάνω από το block αφύπνισης, όπου ο πράκτορας με την υψηλότερη ταυτότητα idεκλέγεται ηγέτης. Αν εκτελέσουμε το

πρωτόκολλο, ένας πράκτορας μπορεί να έχει κίνητρο να πει ψέματα για την ταυτότητά του. ‘Εστω ένας πράκτορα p με την ακόλουθη συνάρτηση χρησιμότητας:

$$u_p = \begin{cases} 1 & \text{Ο } p \text{ εκλέγεται αρχηγός} \\ 0 & \text{Ο } p \text{ δεν εκλέγεται αρχηγός ή δεν εκλέγεται κανένας αρχηγός} \end{cases}$$

Ο πράκτορας p θα αποκλίνει από το πρωτόκολλο λέγοντας ψέματα για την ταυτότητά του. Προτιμά να στέλνει μηνύματα και να προσποιείται ότι έχει το υψηλότερο id. Παρόλο που είναι πιθανό ότι ένας άλλος πράκτορας θα έχει την ίδια ταυτότητα για την οποία ο p είπε ψέματα, σύμφωνα με τη u_p , ο p αυξάνει την αναμενόμενη χρησιμότητα του αυξάνοντας τις πιθανότητες του να εκλεγεί ηγέτης [1].

Ακολουθώντας αυτό το παράδειγμα, θεωρούμε ότι τα πρωτόκολλα που είναι χτισμένα με βάση αυτό το block αφύπνισης αγνοούν τα ids των πρακτόρων. Με αυτά στο μυαλό, παρουσιάζουμε πρωτόκολλα για το block αφύπνισης. Σε ένα σύγχρονο δίκτυο, όλοι οι πράκτορες ολοκληρώνουν το πρωτόκολλο στον ίδιο γύρο. Τα πρωτόκολλα φθάνουν στην t - Ισχυρή Ισορροπία για $t < n$, για n ορθολογικούς πράκτορες με οποιαδήποτε συνάρτηση χρησιμότητας που ικανοποιεί την προτίμηση λύσης (ορισμός 3.2.1).

Η ιδέα πίσω από τη διαδικασία μας προκειμένου να επιτύχουμε αυτές τις προαναφερθέντες ιδιότητες και να εμποδίσουμε έναν πράκτορα από το να μην στέλνει μηνύματα αφύπνισης όταν κανονικά θα έπρεπε, είναι να απαιτεί από όλους τους πράκτορες να στέλνουν μηνύματα αφύπνισης. Ένας πράκτορας δεν ολοκληρώνει το πρωτόκολλο πριν λάβει μηνύματα αφύπνισης από όλους τους άλλους πράκτορες. Έτσι, ένας πράκτορας που δεν στέλνει ένα μήνυμα αφύπνισης προκαλεί την αποτυχία του πρωτοκόλλου, σε αντίθεση με την προτίμηση λύσης (ορισμός 3.2.1) [1].

3.3.1 Πρωτόκολλο ενεργοποίησης πρακτόρων σε δακτύλιο

Το πρωτόκολλο για την αφύπνιση σε ένα δακτύλιο έχει ως εξής: κατά τη φάση αφύπνισης του, κάθε πράκτορας στέλνει ένα μήνυμα αφύπνισης που περνά γύρω από το δαχτυλίδι. Ένας πράκτορας ολοκληρώνει το πρωτόκολλο όταν έχει λάβει ένα μήνυμα αφύπνισης από όλους τους άλλους πράκτορες. Το μήνυμα αφύπνισης περιέχει τόσο την ταυτότητα

id του δημιουργού του μηνύματος όσο και έναν μετρητή λυκίσκου k αρχικοποιημένο σε $n-1$ από τον δημιουργό και μειωμένο κατά 1 από κάθε διάδοχο. Αυτός ο μετρητής k αντιπροσωπεύει την απόσταση του μηνύματος από τον τρέχοντα παραλήπτη στον δημιουργό [1].

3.4 Διαμοίραση Γνώσης

Σε πολλούς κατανεμημένους αλγόριθμους, κάθε διαδικασία πρέπει να γνωρίζει τις ιδιωτικές τιμές όλων των άλλων διαδικασιών προκειμένου να πραγματοποιήσει τον επιθυμητό υπολογισμό. Το block ανταλλαγής γνώσεων ασχολείται με την ανταλλαγή ιδιωτικών τιμών μεταξύ όλων των πρακτόρων. Αντιμετωπίζοντας ορθολογικούς πράκτορες, δημιουργείται ένα πρόβλημα. Όταν ένας πράκτορας ξέρει όλες τις άλλες τιμές των άλλων πρακτόρων, μπορεί να πει ψέματα για τη δική του τιμή για να αυξήσει τη χρησιμότητά του. Έτσι, τα πρωτόκολλα που παρουσιάζονται για το block ανταλλαγής γνώσεων διασφαλίζουν ότι κάθε πράκτορας δεσμεύεται στη δική του τιμή πριν μάθει τις τιμές όλων των άλλων πρακτόρων.

Κάθε πράκτορας p ξεκινά με μια τιμή v_p και χρειάζεται να μάθει τις τιμές όλων των άλλων πρακτόρων $V = \{v1. \dots, v_n\}$. Για αυτό το block, υποθέτουμε ότι οι πράκτορες γνωρίζουν τα ids όλων των άλλων πρακτόρων. Εάν όχι, μπορούμε απλώς να προσθέσουμε το block αφύπνισης. Μια λύση για την ανταλλαγή γνώσεων φαίνεται τετριμμένη: κάθε πράκτορας στέλνει την τιμή του σε όλους τους άλλους πράκτορες. Μόλις ένας πράκτορας λάβει τις τιμές V όλων των άλλων πρακτόρων, τερματίζεται. Ωστόσο, η λύση αυτή δεν επιτυγχάνει ισορροπία. Ένας πράκτορας μπορεί εύκολα να πει ψέματα για την τιμή του προκειμένου να αυξήσει τη χρησιμότητά του. Για παράδειγμα, έστω ένα πρωτόκολλο εκλογής $leader$ που εκτελείται χρησιμοποιώντας αυτό το block, όπου ο ηγέτης εκλέγεται με τον ακόλουθο τρόπο: Η τιμή, η αξία κάθε πράκτορα είναι ένας τυχαίος αριθμός και το άθροισμα όλων των αριθμών $mod\, n$ είναι η τάξη του id εκλεγμένου ηγέτη. Εάν εκτελέσουμε το πρωτόκολλο στο μοντέλο θεωρίας παιγνίων, ένας πράκτορας μπορεί να έχει κίνητρο να πει ψέματα για την αξία του.

Έστω ένας πράκτορας p με την ακόλουθη συνάρτηση χρησιμότητας:

$$u_p = \begin{cases} 1 & \text{Ο } p \text{ εκλέγεται αρχηγός} \\ 0 & \text{Ο } p \text{ δεν εκλέγεται αρχηγός ή δεν εκλέγεται κανένας αρχηγός} \end{cases}$$

Ο πράκτορας p μπορεί να προτιμά να καθυστερήσει τη συμμετοχή του στο πρωτόκολλο, έτσι ώστε να γνωρίζει τις αξίες όλων των άλλων πρακτόρων. Μόλις ο p γνωρίζει όλες τις τιμές V, τότε μπορεί να πει ψέματα για τη δική του αξία v_p εξασφαλίζοντας ότι εκλέγεται ως ηγέτης. Για να ξεπεραστεί αυτό το ζήτημα, διασφαλίζουμε ότι κανένας πράκτορας δεν μαθαίνει τις αξίες άλλων πρακτόρων προτού αποστείλει τη δική του αξία. Επιπλέον, όταν αντιμετωπίζουμε συμμαχίες, συνασπισμούς από $cheaters$, τα πρωτόκολλα διασφαλίζουν ότι οι πράκτορες εκτός του συνασπισμού μαθαίνουν τις αξίες όλων των πρακτόρων του συνασπισμού πριν οι πράκτορες του συνασπισμού μάθουν τις τιμές όλων των άλλων πρακτόρων [1].

Εάν το block χρησιμοποιείται από ένα πρωτόκολλο που δεν είναι αληθές, ένας πράκτορας μπορεί να έχει κίνητρο να πει ψέματα για την αξία του, ανεξάρτητα από το πρωτόκολλο ανταλλαγής γνώσεων. Ένα πρωτόκολλο λέγεται ότι είναι αληθές εάν κανένας πράκτορας που συμμετέχει στο πρωτόκολλο δεν έχει κίνητρο να πει ψέματα για την είσοδό του, ή οποιοδήποτε άλλο στοιχείο που μοιράζεται ως μέρος του πρωτοκόλλου. Η χρησιμότητα ενός πράκτορα για ένα αποτέλεσμα που επιτυγχάνεται από το ψέμα δεν είναι μεγαλύτερη από τη χρησιμότητά του που επιτυγχάνεται λέγοντας την αλήθεια.

Για παράδειγμα, έστω ένα πρωτόκολλο συμφωνίας που στηρίζεται πάνω σε αυτό το block, το οποίο αποφασίζει για τη μέγιστη αξία όλων των πρακτόρων. Εάν εκτελέσουμε το πρωτόκολλο στο μοντέλο θεωρίας παιγνίων, ένας πράκτορας μπορεί να έχει κίνητρο να πει ψέματα για την αξία του. Έστω ένας πράκτορας p με τιμή $v_p \in \{0,1\}$, με την ακόλουθη χρησιμότητα:

$$u_p = \begin{cases} 1 - d & \text{Η συμφωνία γίνεται με απόφαση } d \\ 0 & \text{Δεν δημιουργείται καμία συμφωνία} \end{cases}$$

Σε μια εκτέλεση όπου $v_p = 1$, ο πράκτορας p προτιμά τη μετάδοση $v_p = 0$. Στην περίπτωση που ο p είναι ο μόνος πράκτορας με είσοδο 1, το πρωτόκολλο αποφασίζει 0 αντί για 1 και ο p ωφελείται. Διαφορετικά, η απόφαση παραμένει η ίδια και ο p δεν ωφελείται, αλλά ούτε και χάνει. Επομένως, ο p αυξάνει την αναμενόμενη χρησιμότητά του με απόκλιση από το πρωτόκολλο, και έτσι το πρωτόκολλο δεν φτάνει σε ισορροπία.

Επιπλέον, έστω μια εκτέλεση όπου όλοι οι πράκτορες έχουν τη χρησιμότητα u_p, αλλά όλοι έλαβαν 1 ως είσοδο. Οι πράκτορες όλοι μεταδίδουν την τιμή 0, η οποία αποφασίζεται. Η τιμή 0 δεν ήταν η εισροή οποιουδήποτε πράκτορα και έτσι παραβιάζεται η απαίτηση εγκυρότητας. Αυτό δείχνει ότι, ενώ το block ανταλλαγής γνώσεων δεν παραβιάζει την ειλικρίνεια, εξαρτάται από την ιδιότητα για ύπαρξη μιας πλήρης γνώσης των πρωτοκόλλων που το χρησιμοποιούν. Ως εκ τούτου, σύμφωνα με τον ορισμό 3.4.1, κάθε πρωτόκολλο που χρησιμοποιεί το block ανταλλαγής γνώσεων πρέπει να διασφαλίζει ότι ικανοποιεί την ιδιότητα πλήρους γνώσης για ολόκληρο τον αλγόριθμο, έτσι ώστε ο αλγόριθμος να είναι αληθής [1].

ΟΡΙΣΜΟΣ 3.4.1 (ΙΔΙΟΤΗΤΑ ΓΙΑ ΠΛΗΡΗ ΓΝΩΣΗ): Για κάθε πράκτορα που δεν γνωρίζει τις τιμές όλων των άλλων πρακτόρων $V = \{v_1, \dots, v_n\}$, οποιαδήποτε έξοδος του πρωτοκόλλου εξακολουθεί να είναι εξίσου δυνατή [1].

Για παράδειγμα, αν και το προηγούμενο παράδειγμα δείχνει ότι το πρωτόκολλο συμφωνίας αποτυγχάνει με αυτό το block, είναι δυνατόν να επιτευχθεί συμφωνία με αυτό το block. Το πρωτόκολλο συμφωνίας που παρουσιάζεται στο παράδειγμα αυτό δεν ικανοποιεί τον ορισμό 3.4.1, καθώς ένας πράκτορας μπορεί να πει ψέματα για την αξία του προκειμένου να επηρεάσει το αποτέλεσμα του πρωτοκόλλου, χωρίς να γνωρίζει τις τιμές άλλων πρακτόρων, αυξάνοντας έτσι την αναμενόμενη χρησιμότητα του. Ωστόσο, στο 3.6, παρουσιάζουμε ένα πρωτόκολλο που καταλήγει σε συμφωνία, ενώ ικανοποιεί τον ορισμό 3.4.1, χρησιμοποιώντας σωστά το block ανταλλαγής γνώσης.

Με αυτά στο μυαλό, παρουσιάζουμε πρωτόκολλα για το block ανταλλαγής γνώσεων. Το block προϋποθέτει τα εξής: κάθε πράκτορας p ξεκινά με μια τιμή v_p, οι πράκτορες γνωρίζουν τα ids όλων των άλλων πρακτόρων και, σε σύγχρονα δίκτυα, όλοι οι πράκτορες ξεκινούν το πρωτόκολλο μαζί.

Τα πρωτόκολλα φτάνουν σε t - Ισχυρή - Ισορροπία, όπου $t < n$ για σύγχρονα δίκτυα και $t < \frac{n}{2}$ για ασύγχρονα δίκτυα, για n ορθολογικούς πράκτορες με οποιαδήποτε προτίμηση που ικανοποιεί τον ορισμό 3.2.1 και οποιαδήποτε τιμή που ικανοποιεί τον ορισμό 3.4.1. Σε ένα σύγχρονο δίκτυο, όλοι οι πράκτορες ολοκληρώνουν το πρωτόκολλο στον ίδιο γύρο [1].

3.4.1 Διαμοίραση γνώσης σε ένα σύγχρονο δακτύλιο

Στον γύρο 1, κάθε πράκτορας p στέλνει ένα μήνυμα με την τιμή v_p. Αυτά τα μηνύματα διαβιβάζονται γύρω από τον δακτύλιο από όλους τους πράκτορες, έως ότου κάθε μήνυμα επιστρέψει στον δημιουργό του. Σε αυτό το σημείο, όλοι οι πράκτορες έχουν λάβει όλα τα δεδομένα. Εάν ένας πράκτορας p δεν λαμβάνει ένα μήνυμα από τον προκάτοχό του σε κάθε γύρο ή λαμβάνει ένα διπλότυπο μήνυμα που έλαβε πριν, ορίζει $o_p = \perp$, εξασφαλίζοντας έτσι τη σωστή συμμετοχή όλων λόγω του ορισμού 3.2.1. Η πολυπλοκότητα του μηνύματος είναι το n^2 και η πολυπλοκότητα του χρόνου είναι n γύροι [1].

3.4.2 Διαμοίραση γνώσης σε ένα ασύγχρονο δακτύλιο

Η λύση σύγχρονου δακτυλίου δεν λειτουργεί σε ασύγχρονο δακτύλιο. Οι πράκτορες δεν μπορούν να καθορίσουν πότε να στείλουν και να προωθήσουν μηνύματα και μπορούμε να φτάσουμε σε μια κατάσταση στην οποία ένας πράκτορας μπορεί να μάθει τις αξίες όλων των άλλων πρακτόρων προτού αποστείλει τη δική του αξία στον διάδοχό του. Ο πράκτορας θα μπορούσε τότε να έχει κίνητρο να πει ψέματα για την αξία του, όπως περιγράφεται στο παράδειγμα εκλογής $leader$ παραπάνω. Για να λυθεί αυτό το ζήτημα, προτείνουμε ένα πρωτόκολλο για εκλογές ηγέτη σε ασύγχρονο δακτύλιο. Έχουμε τον πράκτορα p με το χαμηλότερο id που είναι ο δημιουργός, ο οποίος ξεκινάει το πρωτόκολλο στέλνοντας την αξία του στον διάδοχό του. Ο διάδοχός του στη συνέχεια στέλνει ένα μήνυμα με δική του αξία στον δικό του διάδοχο και ούτω καθεξής.

Τα μηνύματα διαβιβάζονται με τον ίδιο τρόπο σε όλο το δαχτυλίδι. Όταν ο δημιουργός λάβει ένα μήνυμα από τον προκάτοχό του, αυτό είναι ένδειξη ότι ο πρώτος κύκλος, στον

οποίο ο κάθε πράκτορας έμαθε την αξία του προκατόχου του, έχει ολοκληρωθεί και ο αρχικός δημιουργός του μηνύματος ξεκινά τον δεύτερο γύρο μηνυμάτων στέλνοντας την τιμή του προκατόχου του στο διάδοχο του. Τα μηνύματα διαβιβάζονται πάλι γύρω από ολόκληρο το δαχτυλίδι ξανά, με τον ίδιο τρόπο, όπου κάθε πράκτορας στέλνει τα δεδομένα του προκατόχου του. Από τους κύκλους 2 έως $n-1$, κάθε πράκτορας στέλνει την τιμή που έχει λάβει στον προηγούμενο κύκλο. Στο γύρο v ο πράκτορας u στέλνει την τιμή εισόδου του πράκτορα $u-v+1 \, mod \, n$. Είναι εύκολο να δούμε ότι η πολυπλοκότητα του μηνύματος και του χρόνου του πρωτοκόλλου εί $\nu \ \alpha$ ι n $(n-1)$ [1].

3.5 Εκλογή Αρχηγού Πρακτόρων

Το πρωτόκολλο για τις εκλογές των ηγετών δείχνει πώς να προσαρμόσουμε ένα πρόβλημα για να ικανοποιεί τις απαιτήσεις των blocks που έχουμε αναφέρει. Εδώ παραθέτουμε ένα block για τον αλγόριθμο εκλογής $leader$:

ΠΡΩΤΟΚΟΛΛΟ 3.5.1 (ΕΚΛΟΓΗ LEADER ΒΑΣΗ ΤΗΣ ΘΕΩΡΙΑΣ ΠΑΙΓΝΙΩΝ): Για κάθε πράκτορα p:
(1) Αρχικοποιείται το block αφύπνισης.
(2) Έστω $v_p \ = \ random \ (1 \ldots . n)$.
(3) Block ανταλλαγής γνώσης για την εκμάθηση $V \ = \ \{v_1, \ldots , v_n\}$.
(4) Για κάθε k, αν $(v_k < \ 1)$ ή $(v_k \ > \ n)$, τότε $o_p \ = \perp$ και τερματίζουμε.
(5) Υπολογίζουμε $(N = \sum_{k=1}^{n} v_k \, mod \, n)$, και την έξοδο: $o_p \leftarrow$ το νιοστό id

Είναι εύκολο να δούμε ότι το πρωτόκολλο 3.5.1 ακολουθεί τις απαιτήσεις των blocks και εκλέγει σωστά έναν ηγέτη ενώ φτάνει στην ισορροπία [1].

3.6 Μετονομασία

Έχοντας μια λύση στο πρόβλημα της εκλογής ηγέτη που φτάνει στην ισορροπία κάτω από το μοντέλο της θεωρίας παιγνίων, φαίνεται να υπάρχει ένα πρωτότυπο πρωτόκολλο μετονομασίας: εκτελούμε την εκλογή του ηγέτη και τότε ο ηγέτης στέλνει τυχαία τα νέα ονόματα σε όλους τους πράκτορες. Ωστόσο, αυτό το πρωτότυπο πρωτόκολλο δεν φτάνει

σε ισορροπία. Ο ηγέτης μπορεί να προτιμήσει μια συγκεκριμένη ανάθεση ονομάτων και μπορεί να τα αναθέσει όπως του συμφέρει.

Έχοντας αυτό κατά νου, ένα άλλο πιθανό πρωτόκολλο είναι να εκτελέσουμε το πρωτόκολλο εκλογής ηγέτη n φορές. Σε κάθε εκτέλεση, ο αρχηγός λαμβάνει ένα όνομα από $[1 \ldots . n]$, με σειρά και καταργείται από τη λίστα πιθανών ηγετών για την επόμενη εκτέλεση. Ενώ αυτό το πρωτόκολλο επιλύει το πρόβλημα, έχει πολύ μεγάλη πολυπλοκότητα μηνύματος και χρόνου. Επιπλέον, ίσως η απαίτηση του ορισμού 3.2.1 είναι πολύ ισχυρή. Μπορούμε να σκεφτούμε μια περίπτωση όπου ένας πράκτορας νοιάζεται μόνο για ορισμένα ονόματα (ή ένα μόνο όνομα) και όχι για ολόκληρη τη λύση.

Για παράδειγμα, έστω ένας πράκτορας p με την ακόλουθη χρησιμότητα:

$$u_p = \begin{cases} 1 - \frac{m_p}{n^2} & \text{Ο } p \text{ γνωρίζει το νέο του όνομα και έστειλε } m_p \text{ μηνύματα} \\ 0 & \text{Ο } p \text{ δεν γνωρίζει το νέο του όνομα} \end{cases}$$

Αν και το u_p δεν ικανοποιεί την προτίμηση λύσης (ορισμός 3.2.1), εξακολουθεί να είναι μια λογική χρησιμότητα. Έτσι, όταν οι πράκτορες επιλέξουν ένα νέο όνομα για τον p, ο p θα σταματήσει να στέλνει μηνύματα και δεν θα συμμετέχει πλέον στο πρωτόκολλο, αυξάνοντας έτσι τη χρησιμότητά του. Έτσι, το πρωτόκολλο δεν φτάνει σε ισορροπία. Επιπλέον, το πρωτόκολλο αποτυγχάνει και δεν επιλέγει νέα ονόματα μετά το νέο όνομα του p. Προκειμένου να επιλυθεί αυτό το ζήτημα, προτείνουμε ένα πρωτόκολλο όπου είτε επιλέγονται όλα τα νέα ονόματα είτε κανένα. Με αυτό το πρωτόκολλο, ένας πράκτορας μπορεί να μάθει το νέο του όνομα μόνο αν όλα τα νέα ονόματα έχουν εκχωρηθεί σε όλους τους πράκτορες. Έτσι, το πρωτόκολλο φτάνει σε ισορροπία, ακόμα και όταν αντιμετωπίζει έναν πράκτορα με την προτίμηση που παρουσιάζεται στο παραπάνω παράδειγμα. Το πρωτόκολλο είναι παρόμοιο με το πρωτόκολλο που παρουσιάζεται στο σημείο 3.5. Ωστόσο, αντί να τυχαιοποιήσει έναν μόνο αριθμό, κάθε πράκτορας τυχαιοποιεί n αριθμούς, ουσιαστικά τρέχοντας n εκλογές ηγέτη ταυτόχρονα. Οι πράκτορες μοιράζονται τότε όλους τους n^2 αριθμούς και αναθέτουν τα νέα ονόματα. Το πρωτόκολλο λειτουργεί ως εξής:

ΠΡΩΤΟΚΟΛΛΟ 3.6 (ΜΕΤΟΝΟΜΑΣΙΑ ΜΕ ΒΑΣΗ ΤΗ ΘΕΩΡΙΑ ΤΩΝ ΠΑΙΓΝΙΩΝ):

Για κάθε πράκτορα p:

(1) Αρχικοποιείται το block αφύπνισης.

(2) Επιλέγουμε ένα σύνολο n τυχαίων αριθμών ως τιμή: $\forall k \in \{1, \dots, n\}$: $v_p[k] \leftarrow random(1, \dots, k)$

(3) Block ανταλλαγής γνώσης για την εκμάθηση $V = \{v_1, \dots, v_n\}$.

(4) Για κάθε k, j, αν $(v_k[j] < 1)$ ή $(v_k[j] > j)$, τότε $o_p = \perp$ και τερματίζουμε.

(5) Υπολογίζουμε $\forall j \in \{1,\dots,n\}$:$N_j = \sum_{k=1}^{n} v_k[j] (mod\ j)$

Ο πράκτορας με το N_n^{th} ψηλότερο id έχει πάρει το νέο όνομα n, και στη συνέχεια αφαιρείται από τη λίστα των ids, επομένως αγνοείται στις επόμενες επαναλήψεις. Ο πράκτορας με το N_{n-1}^{th}υψηλότερο id τότε παίρνει το νέο όνομα $(n-1)$ και αφαιρείται από τη λίστα των ids και ούτω καθεξής.

(6) Θέτουμε το o_p να ισούται με το νέο όνομα του p όπως έχει υπολογιστεί στο προηγούμενο βήμα.

Αυτή η τυχαιοποίηση εξασφαλίζει ότι κάθε επιλεγμένο όνομα είναι τυχαίο, ικανοποιώντας έτσι την ιδιότητα πλήρους γνώσης (ορισμός 3.4.1). Και πάλι, είναι εύκολο να διαπιστώσουμε ότι ικανοποιεί όλες τις απαιτήσεις για τα blocks και αναθέτει νέα ονόματα ενώ επιτυγχάνεται ισορροπία [1].

3.7 Συμφωνία

Το πρόβλημα της συμφωνίας μπορεί να επιλυθεί παρουσία ορθολογικών πρακτόρων με την εκλογή ενός ηγέτη και έχοντας τον ηγέτη να επιλέξει τη τιμή συμφωνίας να είναι η δική του η είσοδος. Ωστόσο, αυτός ο αλγόριθμος δεν επιτυγχάνει ισορροπία και μπορεί να μην υποστηρίζει την απαίτηση εγκυρότητας, ακόμη και αν υπάρχει μόνο ένας ορθολογικός πράκτορας. Ο εκλεγμένος ηγέτης μπορεί να έχει προτίμηση σε σχέση με την τιμή της συμφωνίας και, ως εκ τούτου, έχει κίνητρο να πει ψέματα για την είσοδό του. Παίρνει την ευκαιρία που του παρουσιάζεται να μην επιτευχθεί συμφωνία, αλλά η αναμενόμενη χρησιμότητα του αυξάνεται συνολικά και έτσι επωφελείται.

Μπορούμε να εκτελέσουμε τα blocks, όπου η αξία για την ανταλλαγή γνώσεων είναι η είσοδος κάθε πράκτορα στο πρόβλημα της συμφωνίας. Ωστόσο, ο υπολογισμός της αποφασισμένης τιμής μετά την ανταλλαγή γνώσεων είναι κρίσιμος. Για παράδειγμα, η συμφωνία πάνω στη μέγιστη τιμή όλων των πρακτόρων δεν λειτουργεί παρουσία ορθολογικών πρακτόρων και δεν επιτυγχάνει ισορροπία. Ένας πράκτορας p με προτίμηση στη τιμή 1 για τη συμφωνία έχει ένα σαφές κίνητρο να μοιραστεί μια τιμή $v_p = 1$, ακόμα κι αν η είσοδος του στη συμφωνία ήταν 0. Έτσι, προτείνουμε το ακόλουθο πρωτόκολλο για μια αληθή συμφωνία:

ΠΡΩΤΟΚΟΛΛΟ 3.7 (ΣΥΜΦΩΝΙΑ ΜΕ ΒΑΣΗ ΤΗ ΘΕΩΡΙΑ ΤΩΝ ΠΑΙΓΝΙΩΝ): Για κάθε πράκτορα p:

(1) Αρχικοποιείται το block αφύπνισης.

(2) Block ανταλλαγής γνώσης για την εκμάθηση $V = \{v_1, \dots, v_n\}$.

(3) Για κάθε k, αν $v_k \notin \{0,1\}$, τότε $o_p = \perp$ και τερματίζουμε.

(4) Αν ο συνολικός αριθμός των 1 είναι μονός ή ίσος με n, τότε $o_p = 1$, αλλιώς $o_p = 0$.

Πρώτον, είναι σαφές ότι χωρίς ορθολογικούς πράκτορες, το πρωτόκολλο επιλύει μια συμφωνία. Επιπλέον, αφού αποφασίζουν σύμφωνα με την ισοτιμία όλων των αξιών, οι πράκτορες δεν μπορούν να επηρεάσουν το αποτέλεσμα. Ένας πράκτορας μπορεί να μάθει τις αξίες όλων των πρακτόρων και δεν έχει κανένα κίνητρο να πει ψέματα για την αξία του, καθώς δεν μπορεί να καθορίσει την επιρροή που θα έχει στην απόφαση, ικανοποιώντας έτσι την ιδιότητα πλήρους γνώσης (ορισμός 3.4.1) .

Όταν ο αριθμός των πρακτόρων n είναι μονός, το πρωτόκολλο είναι αληθές, αφού κανένας πράκτορας δεν μπορεί να ωφεληθεί λέγοντας ψέματα για την είσοδό του. Ωστόσο, όταν το n είναι ζυγός αριθμός, το πρωτόκολλο δεν είναι ικανοποιητικό. Εάν, για παράδειγμα, ένας συνασπισμός από $n-1$ πράκτορες έχει την προτίμηση να συμφωνήσει σε 1, κάθε πράκτορας p στο συνασπισμό μπορεί να πει ψέματα έτσι ώστε $v_p = 1$. Σε μια τέτοια περίπτωση, η συμφωνία θα είναι 1, ανεξάρτητα από την είσοδο του απομείναντα πράκτορα και έτσι δεν φτάνουμε στην ισορροπία και δεν ικανοποιούμε την απαίτηση εγκυρότητας στην περίπτωση που η είσοδος όλων των πρακτόρων είναι 0.

Για να χειριστούμε αυτό το πρόβλημα, για δίκτυα όπου το n είναι ζυγός αριθμός, η είσοδος κάθε πράκτορα για την ανταλλαγή γνώσεων επίσης περιλαμβάνει έναν τυχαίο αριθμό, όπως στο πρωτόκολλο εκλογής $leader$ 3.5.1, εκλέγοντας έτσι έναν ηγέτη ταυτόχρονα. Η είσοδος του ηγέτη στη συνέχεια λαμβάνεται υπόψη δύο φορές στον υπολογισμό, καθιστώντας έτσι το πρωτόκολλο αληθές και για δίκτυα όπου το μέγεθος τους είναι ένας ζυγός αριθμός.

Το πρωτόκολλο ικανοποιεί όλες τις απαιτήσεις για τα blocks και επιτυγχάνει συμφωνία κατά την επίτευξη ισορροπίας [1].

3.8 Προηγούμενες Εργασίες

3.8.1 Η μελέτη των Afek et al.

Το πρόβλημα της κατανεμημένης συμφωνίας, όπου n επεξεργαστές, μερικοί από τους οποίους μπορεί να είναι ελαττωματικοί, πρέπει να συμφωνήσουν στην ίδια αξία, τιμή που αποτελεί και είσοδο σε ένα από αυτούς, έχει μελετηθεί καλά [3-7]. Πιο πρόσφατα, υπάρχει αυξανόμενο ενδιαφέρον για το μοντέλο κατανεμημένων υπολογιστών με ορθολογικούς πράκτορες, οι οποίοι μπορεί να αποκλίνουν από έναν κατανεμημένο αλγόριθμο προκειμένου να επηρεάσουν το αποτέλεσμα του αλγορίθμου [8-12].

Η κύρια ανησυχία σχετικά με τον κατανεμημένο υπολογισμό με τους ορθολογικούς πράκτορες είναι να παράσχει έναν αλγόριθμο ο οποίος είναι ανθεκτικός σε ορθολογικούς πράκτορες, πράγμα που σημαίνει ότι οι ορθολογικοί πράκτορες δεν θα κερδίσουν τίποτα αποφεύγοντας το πρωτόκολλο. Προκειμένου να αντιμετωπιστεί αυτό πιο τυπικά, οι Afek et al. χρησιμοποίησαν τους ακόλουθους ορισμούς (ανατρέξτε στο [1] για πλήρεις λεπτομέρειες):

Ορισμός 1 (Συνάρτηση χρησιμότητας): Έστω A είναι ένας κατανεμημένος αλγόριθμος και O είναι το σύνολο όλων των πιθανών τελικών καταστάσεων του A. Έστω p είναι ένας πράκτορας, τότε $u_p: O \rightarrow \mathbb{R}$ είναι η συνάρτηση χρησιμότητας, έτσι ώστε $\forall o \in O, u_p\ (o)$ είναι το κέρδος που ο p θα κέρδιζε εάν έφθανε στην τελική κατάσταση o.

Ορισμός 2 (Προτίμηση λύσης): Έστω A είναι ένας κατανεμημένος αλγόριθμος και O είναι το σύνολο όλων των πιθανών τελικών καταστάσεων του A. Έστω p είναι ένας πράκτορας, και u_p είναι η συνάρτηση χρησιμότητάς του, τότε η u_p ικανοποιεί την προτίμηση λύσης εάν και μόνο εάν $\forall o_L, o_E \in O$ τέτοια ώστε το o_l είναι μια νόμιμη τελική κατάσταση του A και το o_E είναι μια λανθασμένη τελική κατάσταση του A, η $u_P\ (o_L)\ \geq\ u_P\ (o_E)$.

Ορισμός 3 (K- Ισχυρή - Ισορροπία): Έστω A είναι ένας κατανεμημένος αλγόριθμος και P είναι το σύνολο των επεξεργαστών, τότε ο A φτάνει στην k - Ισχυρή - Ισορροπία εάν και μόνο εάν $\forall C\ \subseteq P$, έτσι ώστε $|\ C\ |\ \leq\ k$, για κάθε $p\ \in C$ ισχύει το εξής:

$$\mathbb{E}\ [\ u_p|C \text{ αποκλίνει από τον } A] \leq \mathbb{E}\ [u_p|C \text{ τρέχει τον } A]$$

Χρησιμοποιώντας λοιπόν τους πιο πάνω ορισμούς, οι Afek et al., έχουν σχεδιάσει ένα αλγόριθμο που παρέχει μια $(n-1)$ - Ισχυρή – Ισορροπία για το πρόβλημα της Κατανεμημένης Συμφωνίας με n ορθολογικούς πράκτορες σε ένα σύγχρονο δακτύλιο όπου ο αλγόριθμος λειτουργεί εάν ο δακτύλιος είναι μονού μεγέθους [1]. Σε ένα σύγχρονο δακτύλιο, ο πράκτορας που κάνει μια ερώτηση αναστέλλει τη λειτουργία του μέχρι να πάρει την απάντηση που περιμένει. Ο αλγόριθμος αυτός παρουσιάζεται στο [1] και είναι ο εξής:

Algorithm 1 (Afek *et al*. [1])**:**

```
do_consensus (input, id):
1       my_rand:= rand ()
2       ids_array.append (<id, input>)
3       rand_sum:= my_rand
4       input_sum:= input
5       Send (<input, id, my_rand>, out_interface)
6       for i = 1,…, n-1:
7           tmp: = Recv (in_interface)
8           if tmp = null or not (tmp.input ∈{0, 1}):
9               REPORT CHEATER
10          ids_array.append (<tmp.id, tmp.input>)
11          rand_sum += tmp.rand
12          input_sum += tmp.input
13          Send (tmp, out_interface)
14      end for
15      tmp: = Recv (in_interface)
```

```
16              if tmp.input ≠ input or tmp.id ≠ id or tmp.rand
                ≠ my_rand:
17                   REPORT CHEATER
18              sort ids_array by id
19              if ∃id ∈ ids_array with ids_array.count(id) ≠ 1:
20                   REPORT CHEATER
21              leader: = ids_array [rand_sum % n]
22              input_sum += leader.input
23              if input_sum % 2 == 1:
24                   return 1
25              else:
26                   return 0
```

end do_consensus

Επεξήγηση αλγορίθμου:

Σύμφωνα με τον αλγόριθμο των Afek et al., κάθε πράκτορας i που παίρνει ένα σήμα για να ξεκινήσει τη διαδικασία επίλυσης του προβλήματος της Κατανεμημένης Συμφωνίας, αρχικά δημιουργεί μια my_rand τιμή που αντιστοιχεί στην τυχαία του επιλογή για το ποιος θα έπρεπε να εκλεγεί ως ηγέτης στο δακτύλιο. Υπάρχει ένας πίνακας ids_array, ο οποίος χρησιμοποιείται για την αποθήκευση των ταυτοτήτων (ids), των εισόδων ($inputs$) και των τυχαίων επιλογών των πρακτόρων (my_rands). Χρησιμοποιούνται δύο μεταβλητές, η $rand\_sum$ και η $input\_sum$. Η μεταβλητή $rand\_sum$ χρησιμοποιείται για να κρατεί το άθροισμα των τυχαίων επιλογών των πρακτόρων όπου το άθροισμα αυτό θα χρησιμοποιηθεί μετά για την εκλογή του τελικού ηγέτη του δακτυλίου. Η μεταβλητή $input\_sum$ χρησιμοποιείται για να κρατεί το συνολικό άθροισμα των εισόδων των πρακτόρων, το οποίο θα χρησιμοποιηθεί μετά μαζί με την είσοδο του τελικού ηγέτη για την απόφαση της τιμής συμφωνίας του συστήματος. Ο πράκτορας i ξεκινά την αφύπνιση των πρακτόρων στέλνοντας ένα μήνυμα με το id, my_rand και $input$ του στον δακτύλιο, έτσι ώστε ξυπνώντας όλους τους πράκτορες να μπορέσει να μάθει και τις δικές τους τιμές για να μπορεί να καταλήξει στο τέλος σε μια τιμή συμφωνίας. Κάθε πράκτορας του δακτυλίου που λαμβάνει το μήνυμα του πράκτορα i, ελέγχει εάν ο i ανήκει σε κάποιο συνασπισμό από πράκτορες C που τείνουν να αποκλίνουν από τον αλγόριθμο (cheaters) για παροχή Ισχυρής Ισορροπίας. Εάν το μήνυμα που λάβει ένας πράκτορας p από τον πράκτορα i είναι κενό ή η είσοδος του i δεν ανήκει στο πεδίο τιμών $\{0,1\}$, τότε ο p καταλαβαίνει ότι ο i ανήκει στο C και έτσι ο αλγόριθμος τερματίζεται χωρίς την ύπαρξη συμφωνίας. Εάν δεν εντοπιστεί κάποιος $cheater$, τότε κάθε πράκτορας p αποθηκεύει τις δικές του τιμές στο πίνακα ids_array

και ενημερώνονται οι μεταβλητές $rand\_sum$ και $input\_sum$. Τότε κάθε πράκτορας i στέλνει τις δικές του τιμές γύρω στο δακτύλιο για να μαθευτούν όλες οι τιμές σε όλους τους πράκτορες.

Όταν μαθευτούν όλες οι τιμές, ο πράκτορας i λαμβάνει πίσω το αρχικό του μήνυμα και ελέγχει εάν υπήρχε κάποιος $cheater$ που το έχει τροποποιήσει για να ξέρει αν θα τερματίσει τον αλγόριθμο ή όχι. Επίσης ελέγχει εάν έχει πάρει κάποιο διπλότυπο μήνυμα από τον ίδιο πράκτορα. Σε έτσι περίπτωση, σημαίνει ότι κάποιος πράκτορας δεν έχει ενημερώσει τους υπόλοιπους πράκτορες για τις τιμές του, επομένως υπήρξε κάποιος $cheater$ και ο αλγόριθμος τερματίζεται. Εάν δεν υπήρξε κάποιος $cheater$, τότε υπολογίζεται ο ηγέτης με τρόπο έτσι ώστε οποιοσδήποτε πράκτορας στο δακτύλιο να έχει τις ίδιες πιθανότητες με όλους να εκλεγεί ως ηγέτης. Τέλος με βάση την είσοδο του ηγέτη και το $input\_sum$ υπολογίζεται η τελική τιμή συμφωνίας.

Συμπεράσματα:

1. Ο αλγόριθμος των Afek et al. λειτουργεί και παρέχει μια $(n-1)$ - Ισχυρή - Ισορροπία σε ένα σύγχρονο δακτύλιο μονού μεγέθους.
2. Σύμφωνα με τον τρόπο που υπολογίζεται η τιμή συμφωνίας στον αλγόριθμο των Afek et al., μια k - Ισχυρή - Ισορροπία ικανοποιείται εάν ο αριθμός k είναι ένας ζυγός αριθμός.

3.8.2 Προηγούμενη Διπλωματική Εργασία

Εισαγωγή:

Σε εργασία που αναπτύχθηκε από ερευνητές του Πανεπιστημίου Tel-Aviv με τίτλο «Impossibility of $(n-1)$ – Strong - Equilibrium for Distributed Consensus with Rational Agents» [12] παρουσιάζεται ένα μεγάλο ενδιαφέρον προς το πεδίο της Κατανεμημένης Συμφωνίας με εγωιστικούς πράκτορες.

Οι ερευνητές που έγραψαν την εργασία αυτή, παραθέτουν και αναλύουν κάποια θεωρήματα, που αφορούν τον αλγόριθμο που προτείνουν οι Afek et al. στο [1], έναν αλγόριθμο που παρέχει $(n-1)$ - Ισχυρή - Ισορροπία για το πρόβλημα της κατανεμημένης συμφωνίας με τους ορθολογικούς πράκτορες σε ένα σύγχρονο δακτύλιο

και για τον οποίο έχουμε συζητήσει στο Κεφάλαιο 3.8. Σκοπός της εργασίας αυτής ήταν η ανάλυση του αλγορίθμου που προτείνουν οι Afek et al., έτσι ώστε να δείξουν την ισχύ του στο να παρέχει μια k - Ισχυρή - Ισορροπία, σύμφωνα (α) με την τιμή που μπορεί να λάβει ο αριθμός k, (β) με το γεγονός εάν ο αριθμός των πρακτόρων είναι ζυγός ή μονός αριθμός και (γ) με την τοπολογία του δικτύου των πρακτόρων.

Χρησιμοποιώντας λοιπόν τον αλγόριθμο που προτείνουν οι Afek et al., οι ερευνητές απέδειξαν ότι αλγόριθμος αυτός αποτυγχάνει να παρέχει μια $(n-1)$ - Ισχυρή - Ισορροπία σε περίπτωση που ο αριθμός των πρακτόρων n στο δακτύλιο είναι ζυγός αριθμός. Επιπλέον, έχουν δείξει ότι μια $(n-1)$ - Ισχυρή - Ισορροπία είναι ανέφικτη όταν το n είναι ζυγός αριθμός σε κάθε τοπολογία του δικτύου. Παρόλα αυτά, έχουν επίσης αποδείξει ότι ο αλγόριθμος αυτός είναι ο μόνος που παράγει μια $(n-1)$ - Ισχυρή - Ισορροπία σε ένα σύγχρονο δακτύλιο όταν το n είναι μονός αριθμός. Τέλος, έχει αποδειχθεί ότι ο αλγόριθμος που προτείνουν οι Afek et al. παρέχει επίσης μια $(n-2)$ - Ισχυρή - Ισορροπία σε ένα σύγχρονο δακτύλιο ζυγού μεγέθους. Ο αλγόριθμος παρουσιάζεται στο Παράρτημα Α.

Θεωρήματα που έχουν αποδειχθεί:

- Θεώρημα 1: Υποθέτοντας ομοιόμορφη κατανομή πάνω στις εισόδους των πρακτόρων, δεν υπάρχει αλγόριθμος για κατανεμημένη συμφωνία σε κάθε τοπολογία δικτύου, παρέχοντας μια $(n-1)$ - Ισχυρή - Ισορροπία όταν το n (ο αριθμός των κόμβων) είναι ζυγός αριθμός.

Το θεώρημα αυτό έχει αποδειχθεί με τη μέθοδο της αντίφασης καταλήγοντας σε αντίφαση χρησιμοποιώντας τους πιθανούς συνασπισμούς των $cheaters\ C$ που μπορεί να δημιουργηθούν σε ένα σύγχρονο δακτύλιο.

- Θεώρημα 2: Υποθέτοντας ομοιόμορφη κατανομή πάνω στις εισόδους των πρακτόρων, ο αλγόριθμος που προτείνουν οι Afek et al. για κατανεμημένη συμφωνία είναι ο μόνος που παρέχει μια $(n-1)$ - Ισχυρή - Ισορροπία όταν το n (ο αριθμός των κόμβων) είναι μονός αριθμός.

Το θεώρημα αυτό έχει αποδειχθεί με τη μέθοδο της αντίφασης και χρησιμοποιώντας τους πιθανούς συνασπισμούς των $cheaters\ C$ που μπορεί να δημιουργηθούν σε ένα σύγχρονο δακτύλιο.

- Θεώρημα 3: Υποθέτοντας ομοιόμορφη κατανομή πάνω στις εισόδους των πρακτόρων και ένα σύγχρονο δακτύλιο ζυγού μεγέθους, ο αλγόριθμος που προτείνουν οι Afek et al. στο 1 λειτουργεί και παρέχει μια $(n-2)$ - Ισχυρή - Ισορροπία.

Για την απόδειξη αυτού του θεωρήματος χρησιμοποιήθηκε η υπόθεση ότι ο δακτύλιος είναι μονής κατεύθυνσης (unidirectional), δηλαδή τα μηνύματα κινούνται στο δακτύλιο προς μία και μόνο κατεύθυνση. Έχει αποδειχθεί ότι ο αλγόριθμος που προτείνουν οι Afek et al. διατηρεί τόσο την εγκυρότητα του όσο και την παροχή συμφωνίας. Επίσης, έχει αποδειχθεί ότι $P(\text{ring decides } 0) = P(\text{ring decides } 1) = \frac{1}{2}$, με μια απλή επαγωγή πάνω στον αριθμό των κόμβων n. Τέλος, για την απόδειξη ότι όντως ο αλγόριθμος των Afek et al. παρέχει μια $(n-2)$ - Ισχυρή - Ισορροπία, έχει αποδειχθεί ότι εάν υπάρχει ένας συνασπισμός από $cheaters\ C$ με $|C| = n-2$ και οι $u_1, u_2 \notin C$ είτε είναι είτε δεν είναι γείτονες στο δακτύλιο τότε το C δεν έχει κανένα κίνητρο να κλέψει. Για την απόδειξη αυτή χρησιμοποιήθηκε μια σειρά αποδείξεων λημμάτων.

Για τις αποδείξεις των λημμάτων αυτού του θεωρήματος, χρησιμοποιήθηκαν τα ακόλουθα, τα οποία θα χρησιμοποιηθούν και για τις αποδείξεις των θεωρημάτων της παρούσας διπλωματικής εργασίας:

- Κάθε κόμβος σε ένα δακτύλιο αρχικοποιείται με 5 τιμές: $< id, input, my\_rand, pos\_0, pos\_1 >$.
- Θέτουμε σαν $Trip(u)$ το σύνολο των $3^{\omega v}$ τιμών $< id, input, my\_rand >$ αρχικοποίησης του u.
- Θέτουμε σαν $Trip(v, u)$ το σύνολο των $3^{\omega v}$ πιο πάνω τιμών που λαμβάνει ο u από τον v.
- Έστω ότι $v \in C$ και $u \notin C$ και m ο αριθμός των κόμβων μεταξύ του v και του u συμπεριλαμβανομένου του u:

Αφού $v \in C$, τότε ο v μπορεί να πει ψέματα σε $(n-m)$ $triplets$ που λαμβάνει ο u, αλλά δεν μπορεί να επηρεάσει (να πει ψέματα) στα m $triplets$ που στέλνει στο $u(Trip(u) + m - 1)$.

Αφότου ο v έχει στείλει κάποια $triplets$, μετά δεν μπορεί να τα αλλάξει.

1. $R\ (v,u) = m$, το σύνολο των κόμβων μεταξύ v και u, συμπεριλαμβανομένου του u.
2. $Lie\ (u) = \{Trip\ (w,u)\ |w \notin R\ (v,u)\}$: Αυτά τα $triplets$ ελέγχονται πλήρως από το C.
3. $Truth\ (u) = \{Trip\ (w,u)\ |w \in R\ (v,u)\}$: Αυτά τα $triplets$ δεν ελέγχονται από το C.
4. $InputT(u) = \sum_{Trip \in Truth(u)} Trip.input$
5. $InputL(u) = \sum_{Trip \in Lie(u)} Trip.input$
6. $RandomT(u) = \sum_{Trip \in Truth(u)} Trip.random$
7. $RandomL(u) = \sum_{Trip \in Lie(u)} Trip.random$

Συμπεράσματα:

Στη εργασία «Impossibility of $(n-1)$ – Strong - Equilibrium for Distributed Consensus with Rational Agents» έχει αποδειχθεί ότι το πρόβλημα της κατανεμημένης συμφωνίας με έναν συνασπισμό $n-1$ λογικών πρακτόρων δεν μπορεί να λυθεί όταν το n είναι ένας ζυγός αριθμός, ανεξάρτητα από την τοπολογία του δικτύου. Επιπλέον, έχει αποδειχθεί ότι όταν το n είναι μονός αριθμός, μπορεί να υπάρξει μόνο μία μέθοδος για να επιτευχθεί συμφωνία , η ισοτιμία του συνόλου όλων των εισροών.

Επίσης, έχει αποδειχθεί ότι ο αλγόριθμος από το [1] λύνει το πρόβλημα της Κατανεμημένης Συμφωνίας σε ένα σύγχρονο δακτύλιο με έναν συνασπισμό
$n-2$ ορθολογικών πρακτόρων, όταν το n είναι ζυγός αριθμός. Έτσι, έχει δοθεί τόσο ένα ανώτερο όσο και ένα χαμηλότερο όριο στο μέγιστο μέγεθος ενός συνασπισμού που υποστηρίζεται από ένα σωστό πρωτόκολλο για το πρόβλημα της κατανεμημένης συμφωνίας με τους ορθολογικούς πράκτορες της τοπολογίας δακτυλίου.

Τέλος, λαμβάνοντας υπόψη τα αποτελέσματα της εργασίας αυτής, σύμφωνα με τους Afek et al., μια *k* - Ισχυρή - Ισορροπία ικανοποιείται εάν ο αριθμός *k* είναι ένας ζυγός αριθμός.

Κεφάλαιο 4

Μια $(n-2)$ – Ισχυρή - Ισορροπία σε Σύγχρονο Δακτύλιο

4.1 Εισαγωγή 61

4.2 Αλγόριθμος $(n-2)$ – Ισχυρής - Ισορροπίας 62

4.3 Ανάλυση αλγορίθμου $(n-2)$ - Ισχυρής - Ισορροπίας 69

4.1 Εισαγωγή

Σε αυτό το κεφάλαιο δείχνουμε ότι αυτός ο αλγόριθμος που προτείνουν οι Afek et al. στο [1] και που έχει χρησιμοποιηθεί και στην εργασία που περιγράφεται στο Κεφάλαιο 3.8 μπορεί να διαμορφωθεί για να παρέχει μια $(n-2)$ - Ισχυρή - Ισορροπία σε ένα σύγχρονο δακτύλιο μονής κατεύθυνσης (unidirectional) με την προϋπόθεση ότι οι εισόδοι που παρέχονται στους πράκτορες δεν ακολουθούν την ομοιόμορφη κατανομή και ότι ο αριθμός των πρακτόρων είναι ένας ζυγός αριθμός. Σύμφωνα με τους Afek et al., μια k - Ισχυρή - Ισορροπία ικανοποιείται εάν ο αριθμός k είναι ένας ζυγός αριθμός. Επομένως, εφόσον χρησιμοποιείται η περίπτωση παροχής μιας Ισχυρής Ισορροπίας για $k = n - 2$, σημαίνει ότι αναγκαστικά το n πρέπει να είναι ένας ζυγός αριθμός για να ικανοποιείται η συνθήκη εγκυρότητας του αλγορίθμου, το οποίο θα δείξουμε και παρακάτω στην ανάλυση του συγκεκριμένου αλγορίθμου. Επομένως, ο αλγόριθμος που παρουσιάζουμε σε αυτό το κεφάλαιο αποτυγχάνει να παρέχει μια $(n-2)$ - Ισχυρή - Ισορροπία όταν ο δακτύλιος είναι μονού μεγέθους. Όλα αυτά είναι επακόλουθα του αλγορίθμου των Afek et al. στον οποίο και έχουμε βασιστεί και ισχύουν για τον συγκεκριμένο αλγόριθμο παροχής μιας $(n-2)$ - Ισχυρής - Ισορροπίας. Σε ένα σύγχρονο δακτύλιο, ο πράκτορας που κάνει μια ερώτηση αναστέλλει τη λειτουργία του μέχρι να πάρει την απάντηση που περιμένει.

Για την κατασκευή της απόδειξης ότι ο αλγόριθμος που προτείνω παρακάτω παρέχει όντως μια $(n-2)$ - Ισχυρή - Ισορροπία σε ένα σύγχρονο δακτύλιο, χρησιμοποιώ τους ορισμούς που χρησιμοποιούν και οι Afek et al. στο [1] και που αναγράφονται στο Κεφάλαιο 3.8.1, καθώς και τους ορισμούς που παρατίθενται στο Κεφάλαιο 3.8.2.

4.2 Αλγόριθμος $(n-2)$ - Ισχυρής - Ισορροπίας

Με βάση τον αλγόριθμο που προτείνουν οι Afek et al. στο [1], θέλουμε να δείξουμε ότι μπορεί να κατασκευαστεί ένας νέος αλγόριθμος που να λύνει το πρόβλημα της Κατανεμημένης συμφωνίας παρέχοντας μια $(n-2)$ - Ισχυρή - Ισορροπία σε ένα σύγχρονο δακτύλιο ζυγού μεγέθους και με την προϋπόθεση ότι οι εισόδοι των πρακτόρων δεν ακολουθούν την ομοιόμορφη κατανομή. Στο κεφάλαιο αυτό, θα παρουσιάσουμε τον αλγόριθμο αυτό και θα εξηγήσουμε αναλυτικά τη λειτουργία του. Στο επόμενο κεφάλαιο (4.3), θα αποδείξουμε ότι ο αλγόριθμος αυτός είναι έγκυρος και δουλεύει σωστά παρέχοντας όντως μια $(n-2)$ - Ισχυρή Ισορροπία σε δακτύλιο ζυγού μεγέθους και θα δείξουμε γιατί δεν μπορεί να λειτουργήσει σε δακτύλιο μονού μεγέθους. Συνεπώς, παρέχοντας μια $(n-2)$ - Ισχυρή - Ισορροπία σε δακτύλιο ζυγού μεγέθους, ο αλγόριθμος αυτός θα είναι σε θέση να λύσει το πρόβλημα της Κατανεμημένης Συμφωνίας, παρέχοντας μια τιμή συμφωνίας στην οποία και θα συμφωνήσουν όλοι οι πράκτορες του δακτυλίου.

Προχωρώντας λοιπόν στην κατασκευή του νέου μας αλγορίθμου, για απλοποίηση, χρησιμοποιούμε την υπόθεση ότι ο δακτύλιος μας είναι μονής κατεύθυνσης (unidirectional). Θεωρούμε ότι ο αριθμός των πρακτόρων n είναι ένας ζυγός αριθμός. Η τοπολογία δακτυλίου είναι μια τοπολογία όπου κάθε πράκτορας συνδέεται με ακριβώς δύο άλλους πράκτορες σχηματίζοντας μια ενιαία συνεχή οδό για τα σήματα μέσω κάθε πράκτορα, σχηματίζοντας δηλαδή ένα δακτύλιο. Ένας δακτύλιος μονής κατεύθυνσης είναι μια τοπολογία δακτυλίου όπου όλα τα σήματα κινούνται είτε δεξιόστροφα είτε αριστερόστροφα γύρω από το δακτύλιο. Θα παρουσιάσουμε ένα αλγόριθμο που παρέχει μια $(n-2)$ - Ισχυρή - Ισορροπία σύμφωνα με τον αλγόριθμο που προτείνεται στο [1] σε ένα σύγχρονο δακτύλιο ζυγού μεγέθους και μονής κατεύθυνσης και με μη ομοιόμορφη κατανομή στις εισόδους των πρακτόρων:

Από τη στιγμή που οι εισόδοι των πρακτόρων δεν ακολουθούν την ομοιόμορφη κατανομή, θα θεωρήσουμε για ευκολία ότι είτε η πιθανότητα της εισόδου 1 θα είναι μεγαλύτερη είτε η πιθανότητα της εισόδου 0 θα είναι μεγαλύτερη. Επομένως, αρχικά δίνεται στους πράκτορες η είσοδος τους που ανήκει στο σύνολο $\{0,1\}$ τυχαία σύμφωνα με τη πιθανότητα εμφάνισης του $0 \Rightarrow p\ (input = 0)\ = x$ και τη πιθανότητα εμφάνισης του $1 \Rightarrow p\ (input = 1)\ = 1 - x$.

Ο αλγόριθμος για την παραχώρηση της εισόδου στους πράκτορες σύμφωνα με αυτές τις πιθανότητες είναι ο πιο κάτω:

- Έστω n το μέγεθος του δακτυλίου⇒ έχουμε n πράκτορες
- Έστω ότι η πιθανότητα η είσοδος να ισούται με 0 είναι 0.3 (30%) και η πιθανότητα η είσοδος να ισούται με 1 είναι 0.7 (70%)

Αλγόριθμος υπολογισμού εισόδων των πρακτόρων:

```
1  Random rng= new Random();
2  double arxiko=0.0;
3  /* 0.0 ≤ x < 0.3 ⇒ 0 (το εύρος διαστήματος είναι  (0.3 - 0.0) = 0.3, ή 30%)*/
4  /*  0.3 ≤ x < 1.0 ⇒ 1 (το εύρος διαστήματος είναι  (1.0 - 0.3) = 0.7, ή 70%)*/
5  double x=0.3;
6  double pos_0= x-arxiko;
7  double pos_1=1-x;
8  for(int i=1;i<=n; i++){
9           double rand_sum=rng.nextDouble();
10          if(rand_sum<pos_0){
11                ids_array.append(<id,0>);
12          }else if(rand_sum<pos_0+pos_1){
13                      Ids_array.aapend(<id,1>);
14           }
15 }
```

Ο αλγόριθμος που παρέχει μια $(n - 2)$ - Ισχυρή - Ισορροπία χωρίζεται σε 3 μέρη:

1. Block αφύπνισης
2. Block ανταλλαγής γνώσεων και

3. Block εκλογής ηγέτη

Αλγόριθμος 1: Πρωτόκολλο $do\_consensus\ _i^{uni}(input, id)$ που εκτελείται από κάθε πράκτορα i σύμφωνα με τον αλγόριθμο που προτείνουν οι Afek et al. στο [1]:

do_consensus $_i^{uni}$**(*input, id*):**

```
1  /*Block αφύπνισης πρακτόρων:*/
2  next_i=id_i //ποιος είναι ο επόμενος πράκτορας από τον οποί
   περιμένουμε το μήνυμα
3  count_i=n //Όταν στη συνέχεια περιμένουμε μήνυμα και για κάθε
   πράκτορα που έχει πάρει την απόφαση του, μειώνουμε το count_i
   κατά 1.
4  type=1 //το είδος του μηνύματος που περιμένουμε
5  my_rand:=rand() //τυχαίος ακέραιος αριθμός από 0…n-1
6  ids_array.append(<id,input,my_rand,pos_0,pos_1>)
   //αποθηκεύουμε το id (ταυτότητα) του κάθε κόμβου, για να
   ξέρει ο κάθε κόμβος από ποιον να περιμένει μήνυμα
7  rand_sum:=my_rand
8  input_sum:=input
9  Send(<input,id,my_rand,pos_0,pos_1>,out_interface)
   //Στέλνουμε το μήνυμα αφύπνισης σε όλους τους διαδόχους του
   πράκτορα i
10 for i=1, .., n-1: //για κάθε ένα από του διαδόχους του
   πράκτορα i
11    tmp:=Recv(<input,id, my_rand, pos_0, pos_1>,in_interface)
   //ο κάθε διάδοχος i λαμβάνει μήνυμα από τον προκάτοχό του
12    if tmp=null or not (tmp.input∈{0,1}) or
       (tmp.pos_0==tmp.pos_1==0.5){
13      REPORT CHEATER
14       System.exit(0)
15    }
16    input_sum+=tmp.input
```

```
17    rand_sum+=tmp.rand
18 /*Μετά την παραλαβή του μηνύματος
    <input,id,my_rand,pos_0,pos_1> από τον προκάτοχό του:*/
19 tmp:= Recv(<input, id, my_rand, pos_0, pos_1>, in_interface)
20    if tmp.id > next_i and type=1 then{
21            next_i=id_q
22            count_i:=n
23            ids_array.append(<tmp.id,tmp.input>)
24            Send (<tmp>, out_interface)
25     }
26     if tmp.id=next_i then{
27               if id≠id_i  or count≠0 or type≠1 then{
28                   leader_i:=⊥
29                   System.exit(0)
30               }
31               else{
32                    count:=n
33                    type=2
34                    ids_array_i=ids_array
35                    Send (<id_i, n, ids_array>, out_interface)
                      //out_interface=διαδόχοι
36               }
37     }
38 /*Μετά την παραλαβή του μηνύματος (id, k, ids_array) από τον
   προκάτοχο:*/
39      if count=0 and type=1 and tmp.id=highest_i then{
40         type=2
41         count=k-1
42         ids_array_i=ids_array
43         Send(<tmp.id,k-1, ids_array>,out_interface)
44      }
45       else if type=1 then {
```

```
                leader_i:=⊥
                System.exit(0)
            }
    if type=2 and count=0 then{
                my_rand_i =rand()
                count=n
                type=3
                Send (<id_i, my_rand_i>, out_interface)
            }
    if type=3 and count=q>0 then{
                if πάρει <id, my_rand'> από τον προκάτοχο
                του και το id είναι το id του q-οστού
                πράκτορα πριν τον i στο ids_array_i then {
                    Send (<id, my_rand'>, out_interface)
                    my_rand_q:= my_rand'
                    q=q-1
                }
                else{
                    leader_i:=⊥
                    System.exit (0)
                }
    }
end for
tmp:=Receive(in_interface)
if tmp.input≠input or tmp.id≠id or tmp.rand≠my_rand or
tmp.pos_0≠pos_0 or tmp.pos_1≠pos_1:
   REPORT CHEATER
sort ids_array by id
if ∃ id ∈ ids_array with ids_array.count(id) ≠1:
    REPORT CHEATER
leader:=ids.array[rand_sum%n] //Δεν χρειάζεται να γίνει
κάποια αλλαγή στο υπολογισμό του leader του πρωτοκόλλου στο
```

```
        [1], καθώς αυτή η αλλαγή υποδηλώνεται από το input_sum με
        μεγαλύτερη πιθανότητα να έχουμε 0 ή 1 ανάλογα με τις αρχικές
        πιθανότητες εισόδου που δίνονται στους πράκτορες από τον
        προγραμματιστή.
74      input_sum+=leader.input
75      if input_sum%2==1 then
76                  return 1
77      else
78                   return 0
```

end $\boldsymbol{do\_consensus_i^{uni}}$

Επεξήγηση αλγορίθμου:

Σε ένα σύγχρονο περιβάλλον επικοινωνίας πρακτόρων, κάθε πράκτορας γνωρίζει το μέγεθος του δακτυλίου. Αρχικά, στη φάση αφύπνισης των πρακτόρων, θέτουμε σαν $next_i$ να είναι το id του επόμενου πράκτορα από τον οποίο ο κάθε πράκτορας i περιμένει ένα μήνυμα. Χρησιμοποιούμε την μεταβλητή $count_i$ για να ξέρουμε τον αριθμό των μηνυμάτων που περιμένει να λάβει ο πράκτορας i. Επίσης χρησιμοποιούμε τη μεταβλητή $type$ για να ξέρουμε τι είδους μηνύματα περιμένει σε κάθε φάση του πρωτοκόλλου ένας πράκτορας i και αρχικοποιούμε αυτή την μεταβλητή σε 1. Ακολούθως, υπολογίζεται ο τυχαίος αριθμός που θα χρησιμοποιήσει στη συνέχεια ο πράκτορας i για την εκλογή του $leader$. Στο πίνακα ids_array αποθηκεύονται τα id, $input$, my_rand, pos_0 και pos_1 των πρακτόρων. Έχουμε ακόμα δύο μεταβλητές. Η $rand\_sum$ θα κρατά το άθροισμα των τυχαίων ακέραιων τιμών που θα επιλέγουν οι πράκτορες για την εκλογή του $leader$ και η $input\_sum$ θα κρατά το συνολικό άθροισμα των εισόδων όλων των πρακτόρων για την εύρεση της τιμής συμφωνίας στο τέλος του αλγορίθμου.

Ξεκινώντας με τον πρώτο γύρο στο πρωτόκολλο, κάθε πράκτορας i στέλνει το id του σε όλους τους πράκτορες του δακτυλίου. Πρώτα πρέπει ο κάθε πράκτορας να στείλει το δικό του id σε άλλους πράκτορες για να μπορέσει να μάθει τα ids καθώς και άλλες πληροφορίες των υπόλοιπων πρακτόρων του δακτυλίου. Ακολούθως, ο πράκτορας i λαμβάνει μήνυμα με τα στοιχεία του γείτονά του. Εάν ο πράκτορας i δεν λάβει κάποιο

μήνυμα ή εάν η τιμή εσόδου του γείτονα του δεν ανήκει στο σύνολο $\{0,1\}$, ή εάν η πιθανότητα εισόδου 0 ισούται με την πιθανότητα εισόδου 1, τότε το πρωτόκολλο μας εντοπίζει ένα $cheater$ (δηλαδή κάποιο πράκτορα που αποκλίνει από τον αλγόριθμο) και τερματίζει χωρίς την ύπαρξη κάποιας συμφωνίας. Ακολούθως, εάν ένας πράκτορας λάβει ένα δεύτερο διπλότυπο μήνυμα, το οποίο έφθασε από ένα διαφορετικό πράκτορα, το μήνυμα αγνοείται εάν ο τελευταίος πράκτορας που το έχει στείλει έχει μικρότερο id από τον προηγούμενο πράκτορα, αλλιώς το νέο μήνυμα στέλνεται γύρω στον δακτύλιο. Στη συνέχεια, ο δημιουργός του μηνύματος με το ψηλότερο id παίρνει πίσω το μήνυμα και έτσι ξέρει όλα τα ids όλων των πρακτόρων και το μήνυμα τότε στέλνεται για δεύτερη φορά στο δακτύλιο.

Ακολούθως, ενώ εξακολουθούμε να είμαστε στο δεύτερο γύρο, ο δημιουργός του μηνύματος παίρνει το μήνυμα για δεύτερη φορά και κάθε πράκτορας i επιλέγει ένα τυχαίο αριθμό και τον στέλνει γύρω στο δακτύλιο. Προχωρώντας στον τρίτο γύρο, εάν ένας πράκτορας i δεν λάβει ένα μήνυμα το οποίο περιμένει να λάβει, τότε θεωρούμε ότι υπάρχει κάποιος $cheater$ με αποτέλεσμα να μην μπορεί να εκλεγεί κάποιος $leader$ από την στιγμή που θα μπορούσε και έτσι ο αλγόριθμος θα τερματίσει χωρίς να προσφέρει κάποια τιμή συμφωνίας. Επίσης, για κάθε $n-1$ εκτελέσεις λήψης των μηνυμάτων των πρακτόρων, μετά που ο δημιουργός του μηνύματος πράκτορας i με το μεγαλύτερο id παίρνει πίσω το μήνυμα του για δεύτερη φορά, κάθε πράκτορας i πρέπει να πάρει ένα τυχαίο αριθμό από τον κατάλληλο πράκτορα. Δηλαδή, k γύρους μετά που ο δημιουργός του μηνύματος με το ψηλότερο id παίρνει πίσω το μήνυμα του για δεύτερη φορά, ο πράκτορας i θα πρέπει να πάρει τον τυχαίο αριθμό του πράκτορα q, όπου ο πράκτορας q είναι k βήματα πριν από τον i στο δακτύλιο.

Μετά από $n-1$ επαναλήψεις, όλοι οι πράκτορες θα ξέρουν όλους τους τυχαίους αριθμούς $my\_rand_1, \dots, my\_rand_n$ και τότε θα μπορεί να γίνει η εκλογή του $leader$ με το νιοστό μεγαλύτερο id. Πριν από αυτό όμως, ο πράκτορας i ελέγχει εάν οι τελικές τιμές που έχει λάβει τη τελευταία φορά από τους πράκτορες συμπίπτουν με προηγούμενες τιμές που έχει δει πιο νωρίς. Εάν οι τιμές αυτές δεν ισούνται μεταξύ τους, τότε έχουμε ξεκάθαρα ένα $cheater$ και έτσι ο αλγόριθμος τερματίζεται με αποτυχία συμφωνίας.

Στο τέλος του αλγορίθμου εκλέγεται κάποιος $leader$ και χρησιμοποιώντας το συνολικό άθροισμα όλων των εισόδων όλων των πρακτόρων στο δακτύλιο συν την είσοδο του $leader$, υπολογίζεται η τελική τιμή συμφωνίας του πράκτορα i.

4.3 Ανάλυση Αλγορίθμου $(n-2)$ – Ισχυρής - Ισορροπίας

Στο κεφάλαιο αυτό παρέχουμε την ανάλυση του αλγορίθμου του Κεφαλαίου 4.2 μέσα από μια σειρά αποδείξεων λημμάτων, με σκοπό να φτάσουμε στην απόδειξη του κύριου θεωρήματος που αφορά τον αλγόριθμο που έχουμε κατασκευάσει. Το κύριο αυτό θεώρημα θα το δούμε στο τέλος του κεφαλαίου αυτού.

Αρχικά θα αποδείξουμε μέσα από το λήμμα που ακολουθεί, ότι ο αλγόριθμος που έχουμε προτείνει είναι έγκυρος και παρέχει μια αξιόπιστη τιμή συμφωνίας όπου η πιθανότητα η τιμή συμφωνίας να είναι το 0 ισούται με την πιθανότητα η τιμή συμφωνίας να είναι το 1.

Λήμμα 1.1 Αν όλοι οι κόμβοι στο δακτύλιο ακολουθούν τον πιο πάνω αλγόριθμο $do\_consensus_i^{uni}$, τότε ο αλγόριθμος διατηρεί τόσο την εγκυρότητα του όσο και την ύπαρξη συμφωνίας και $P\,(απόφαση\ δακτυλίου = 0) = P\,(απόφαση\ δακτυλίου = 1) = \frac{1}{2}$

Απόδειξη. Απόδειξη εγκυρότητας- Στην περίπτωση μη-ομοιόμορφης κατανομής στις εισόδους, λόγω του ότι η πιθανότητα να πάρει ο πράκτορας τη τιμή 0 σαν είσοδο δεν ισούται με την πιθανότητα ο πράκτορας να πάρει τη τιμή 1 σαν είσοδο, σημαίνει ότι μπορούν όλοι οι κόμβοι να πάρουν 0 ή 1 σαν είσοδο, αλλά σε περίπτωση που δεν πάρουν όλοι 0 ή 1 σαν είσοδο, σίγουρα ένα από τα δύο θα υπερισχύει, δηλαδή είτε θα έχουμε περισσότερα μηδενικά είτε θα έχουμε περισσότερους άσσους, αλλά σίγουρα δεν πρόκειται να έχουμε ίσο αριθμό μηδενικών και άσσων.

- Αν κάθε κόμβος έπερνε 0 σαν είσοδο:

1. $pos\_0 > pos\_1$
2. $v.input = 0$ και $input\_sum = v.input = 0$
3. $input\_sum = input\_sum + tmp[i].input$
$= input\_sum + input\_sum\ (n-1)$

$= 0 + 0\ (n-1) = 0$

4. $leader.input = 0$
5. $input\_sum = input\_sum + leader.input$

   $= 0 + 0 = 0$
6. $decision_i = input\_sum\%2 = 0\%2 = 0 \Rightarrow return\ 0$

   ⇒Όταν όλοι οι κόμβοι παίρνουν σαν είσοδο το 0, τότε όλοι συμφωνούν στο 0.

- Αν κάθε κόμβος έπερνε 1 σαν είσοδο:

1. $pos\_1 > pos\_0$
2. $v.input = 1$ και $input\_sum = v.input = 1$
3. $input\_sum = input\_sum + tmp[i].input =$

   $= input\_sum + input\_sum\ (n-1)$

   $= 1 + (n-1) * 1$

   $= 1 + n - 1 = n$
4. $leader.input = 1$
5. $input\_sum = input\_sum + leader.input$

   $= n + 1$
6. $decision_i = input\_sum\%2 = (n\ +\ 1)\%2$

   (αφού το n είναι ζυγός αριθμός, σημαίνει ότι το $n\ +\ 1$ είναι μονός αριθμός)

   =1⇒ return 1

⇒Όταν όλοι οι κόμβοι παίρνουν σαν είσοδο το 1, τότε όλοι συμφωνούν στο 1.

<u>Απόδειξη παραβίασης συνθήκης εγκυρότητας σε ένα δακτύλιο μονού μεγέθους</u>:

Ο αλγόριθμος που έχουμε σχεδιάσει ακολουθεί την λογική υπολογισμού μιας τιμής συμφωνίας που ακολουθεί και ο αλγόριθμος που έχουν προτείνει οι Afek et al. Εάν το n μας ήταν μονός αριθμός, αυτό θα σήμαινε ότι το $n\ +\ 1$ θα ήταν ζυγός αριθμός με αποτέλεσμα το $(n + 1)\%2$ να είχε σαν αποτέλεσμα το 0 και σύμφωνα με τον αλγόριθμο μας το αποτέλεσμα της συμφωνίας σε τέτοια περίπτωση θα ήταν το 0, ένα αποτέλεσμα που

παραβιάζει τη συνθήκη εγκυρότητας του αλγορίθμου μας, καθώς όλοι οι κόμβοι έλαβαν σαν είσοδο το 1 και κανένας το 0. Επομένως, αφού και οι τρείς αλγόριθμοι που παρουσιάζονται στην διπλωματική αυτή εργασία έχουν ως βάση τον αλγόριθμο των Afek et al., αποδεικνύουμε την ύπαρξη μιας $(n-2)$ - Ισχυρής - Ισορροπίας και στους τρείς μας αλγορίθμους με την υπόθεση ότι το *n* είναι ένας ζυγός αριθμός. ■

- Όταν κάποιοι κόμβοι παίρνουν το 0 σαν είσοδο και άλλοι το 1:

1. Αν $P\,(input = 0) > P\,(input = 1)$, τότε κάθε αποτέλεσμα θα ήταν αποδεκτό όσο αφορά την εγκυρότητα.
2. Αν $P\,(input = 0) < P\,(input = 1)$, τότε επίσης κάθε αποτέλεσμα θα ήταν αποδεκτό όσο αφορά την εγκυρότητα.

Απόδειξη συμφωνίας- Αν όλοι οι κόμβοι ακολουθούν τον αλγόριθμο $do\_consensus_i^{uni}$, τότε μετά τη γραμμή 69, κάθε κόμβος θα έχει τις ίδιες τιμές για $rand\_sum$, $input\_sum$ και ids_array (σύμφωνα με μια απλή επαγωγή), που σημαίνει ότι όλοι οι κόμβοι θα συμφωνούν στην ίδια τιμή.

Απόδειξη του $P\,(ring\ decides\ 0) = P\,(ring\ decides\ 1) = 1/2$ – Σπάζουμε αυτή την απόδειξη σε αποδείξεις δύο ισχυρισμών:

Ισχυρισμός 1- Έστω οι κόμβοι $v_1, v_2, \dots, v_{4k}$ (Χρησιμοποιούμε το $4k$, για να δείξουμε στην απόδειξη ότι ο αριθμός των κόμβων είναι πάντα ένας ζυγός αριθμός):

$$P\left(\left[\left(\sum_{i=1}^{4k} v_i.input\right)\right] mod\ 2 = 0\right) = P\left(\left[\left(\sum_{i=1}^{4k} v_i.input\right)\right] mod\ 2 = 1\right) = \frac{1}{2}$$

- Η πιθανότητα το $input\_sum$ των εισόδων να είναι μονός ή ζυγός αριθμός είναι $\frac{1}{2}$.
- $P\,(input\_sum = 0) = P\,(input\_sum = 1) = \frac{1}{2}$.

Ισχυρισμός 2- Έστω οι κόμβοι $v_1, v_2, \dots, v_{4k}$:

$$P\left(\left[\left(\sum_{i=1}^{4k} v_i.input\right) + leader.input\right] mod\ 2 = 0\right) =$$

$$\left(\left[\left(\sum_{i=1}^{4k} v_i.input\right) + leader.input\right] mod\ 2 = 1\right) = \frac{1}{2}$$

- Η πιθανότητα το $input\_sum$ των εισόδων συν την είσοδο του $leader$ να είναι μονός ή ζυγός αριθμός είναι $\frac{1}{2}$.
- $P\ (input\_sum + leader.input = 0)$
  $= P\ (input\_sum + leader.input = 1) = \frac{1}{2}$.

Απόδειξη των πιο πάνω ισχυρισμών με επαγωγή στον αριθμό των κόμβων n:

1. Βάση επαγωγής: Έστω $n = 4 \Rightarrow (v_1, v_2, v_3, v_4)$:

1ος Ισχυρισμός. Υπάρχουν 4 συνδυασμοί εισόδων:

(α) $P\ (input = 1) > P\ (input = 0)$

- $< 1,1,1,0 >$: Άθροισμα=3$\Rightarrow \left[\left(\sum_{i=1}^{4k} v_i.input\right)\right] mod\ 2 = 1$
- $< 1,1,1,1 >$: Άθροισμα=4$\Rightarrow \left[\left(\sum_{i=1}^{4k} v_i.input\right)\right] mod\ 2 = 0$

(β) $P\ (input = 1) < P\ (input = 0)$

- $< 0,0,0,1 >$: Άθροισμα=1$\Rightarrow \left[\left(\sum_{i=1}^{4k} v_i.input\right)\right] mod\ 2 = 1$
- $< 0,0,0,0 >$: Άθροισμα=0$\Rightarrow \left[\left(\sum_{i=1}^{4k} v_i.input\right)\right] mod\ 2 = 0$

Παρατηρούμε ότι στις μισές περιπτώσεις (2) το $\left[\left(\sum_{i=1}^{4k} v_i.input\right)\right] mod\ 2 = 0$ και στις άλλες μισές (2) το $\left[\left(\sum_{i=1}^{4k} v_i.input\right)\right] mod\ 2 = 1$. Επομένως, ο πρώτος ισχυρισμός ισχύει με $P\left(\left[\left(\sum_{i=1}^{4k} v_i.input\right)\right] mod\ 2 = 0\right) = P\left(\left[\left(\sum_{i=1}^{4k} v_i.input\right)\right] mod\ 2 = 0\right) = \frac{1}{2}$ για $n = 4$.

2ος Ισχυρισμός. Υπάρχουν 8 συνδυασμοί εισόδων:

Πίνακας 4.1 Συνδυασμοί Εισόδων Πρακτόρων και Άθροισμα

$<\left(\sum_{i=1}^{4k} vi.input\right),$ $leader.input>$	Άθροισμα
<3,0>	$3\Rightarrow\left[\left(\sum_{i=1}^{4k} vi.input\right) + leader.input\right] mod\ 2 = 1$
<3,1>	$4\Rightarrow\left[\left(\sum_{i=1}^{4k} vi.input\right) + leader.input\right] mod\ 2 = 0$
<4,0>	$4\Rightarrow\left[\left(\sum_{i=1}^{4k} vi.input\right) + leader.input\right] mod\ 2 = 0$
<4,1>	$5\Rightarrow\left[\left(\sum_{i=1}^{4k} vi.input\right) + leader.input\right] mod\ 2 = 1$

<1,0>	$1 \Rightarrow \left[\left(\sum_{i=1}^{4k} vi.input\right) + leader.input\right] mod\ 2 = 1$
<1,1>	$2 \Rightarrow \left[\left(\sum_{i=1}^{4k} vi.input\right) + leader.input\right] mod\ 2 = 0$
<0,0>	$0 \Rightarrow \left[\left(\sum_{i=1}^{4k} vi.input\right) + leader.input\right] mod\ 2 = 0$
<0,1>	$1 \Rightarrow \left[\left(\sum_{i=1}^{4k} vi.input\right) + leader.input\right] mod\ 2 = 1$

Παρατηρούμε ότι στις μισές περιπτώσεις (4), το $\left[\left(\sum_{i=1}^{4k} v_i.input\right) + leader.input\right] mod\ 2 = 0$ και στις άλλες μισές (4) το $\left[\left(\sum_{i=1}^{4k} v_i.input\right) + leader.input\right] mod\ 2 = 1$.

Επομένως, ο δεύτερος ισχυρισμός ισχύει με
$P\left(\left[\left(\sum_{i=1}^{4k} v_i.input\right) + leader.input\right] mod\ 2 = 0\right) = P\left(\left[\left(\sum_{i=1}^{4k} v_i.input\right) + leader.input\right] mod\ 2 = 1\right) = \frac{1}{2}$ για $n = 4$.

2. Επαγωγική υπόθεση: Υποθέτουμε ότι οι 2 ισχυρισμοί ισχύουν για $n = 4\ (k-1)\ = \ 4k - 4$. Θα αποδείξουμε ότι ισχύουν και για $n = 4k$.

3. Επαγωγικό βήμα:

   1ος Ισχυρισμός. $< v_1, v_2, \dots, v_{4k-4}, v_{4k-3}, v_{4k-2}, v_{4k-1}, v_{4k} >$

   Χρησιμοποιώντας την ίδια ανάλυση με τη βάση επαγωγής και την επαγωγική υπόθεση για $<v_1, v_2, .., v_{4k-4}>$, ο ισχυρισμός αποδεικνύεται και ισχύει για *n*=4*k*.

   2ος Ισχυρισμός. $< v_1, v_2, \dots, v_{4k-1}, v_{4k} >$

   - Εάν τα $v_{4k-3}, v_{4k-2}, v_{4k-1}$ και v_{4k} δεν εκλεγούν σαν $leaders$, τότε η εισφορά τους στο άθροισμα θα είναι είτε μονός αριθμός (3 για $< 1,1,1,0 >$ και 1 για $< 0,0,0,1 >$) στις μισές περιπτώσεις, είτε θα είναι ζυγός αριθμός (4 για $< 1,1,1,1 >$ και 0 για $< 0,0,0,0 >$) στις υπόλοιπες μισές. Και από την υπόθεση επαγωγής για τον ισχυρισμό 2, ξέρουμε ότι αν ένας από τους κόμβους $< v_1, v_2, \dots, v_{4k-4} >$ εκλεγεί σαν $leader$, τότε το άθροισμα στις μισές περιπτώσεις θα είναι ζυγός αριθμός (4, 4, 2, 0) και στις άλλες μισές μόνος αριθμός (3, 5, 1, 1). Επομένως, ο ισχυρισμός 2 ισχύει για $n = 4k$.

- Εάν τα $v_{4k-3}, v_{4k-2}, v_{4k-1}$ και v_{4k} εκλεγούν σαν $leaders$, μπορούμε να χρησιμοποιήσουμε την επαγωγική υπόθεση του ισχυρισμού 1 στα $< v_1, v_2, \dots, v_{4k-4} >$ και την επαγωγική υπόθεση του ισχυρισμού 2 στα $v_{4k-3}, v_{4k-2}, v_{4k-1}$ και v_{4k} και έτσι ο ισχυρισμός 2 ισχύει για $n = 4k$. Επομένως, από τον ισχυρισμό 2 είναι προφανές ότι $P(ring\ decides\ 0) = P(ring\ decides\ 1) = \frac{1}{2}$. ■

Χρησιμοποιώντας το λήμμα που ακολουθεί, θέλουμε να αποδείξουμε ότι οποιοσδήποτε πράκτορας στον δακτύλιο έχει την ίδια πιθανότητα παίξει καθοριστικό ρόλο στην επιλογή της τιμής συμφωνίας, δηλαδή να εκλεγεί ως $leader$, ανεξάρτητα από τις ενέργειες των πρακτόρων που προσπαθούν να παρεκκλίνουν από το πρωτόκολλο για προσωπικό συμφέρον.

Λήμμα 1.2 Έστω $cheaters\ C$ και $u \notin C$, τότε κάθε id στο $u.ids\_array$ έχει την ίδια πιθανότητα του $\frac{1}{n}$ να εκλεγεί ως $leader$ από τον u, άσχετα με τις ενέργειες που λαμβάνονται από το C.

Απόδειξη. Φαίνεται από τον αλγόριθμο $do\_consensus\ _i^{uni}(input, id)$. Αρχικά, το να εκλεγεί κάποιος $leader$ δεν εξαρτάται από το $input\_sum$, αλλά από το $rand\_sum$. Το my_rand κάθε κόμβου παίρνει τιμές $\{0, .., n-1\}$ και σύμφωνα με τον αλγόριθμο είναι ομοιόμορφα κατανεμημένο. Επομένως, και το $RandomT(u)$ είναι ομοιόμορφα κατανεμημένο με αποτέλεσμα το $rand\_sum$ mod n να είναι επίσης ομοιόμορφα κατανεμημένο. Επιπλέον, αν όλοι οι κόμβοι $u \notin C$ ακολουθούν τον αλγόριθμο, σημαίνει ότι όλοι οι κόμβοι μετά τη γραμμή 69 θα έχουν την ίδια τιμή $rand\_sum$. Επομένως, $leader := ids\_array[rand\_sum\%n]$ και κάθε id στο $u.ids\_array$ έχει την ίδια πιθανότητα του $\frac{1}{n}$ να εκλεγεί ως $leader$ από τον u άσχετα με τις ενέργειες που λαμβάνονται από το C. ■

Χρησιμοποιώντας το λήμμα που ακολουθεί, θέλουμε να δείξουμε ότι εάν ένας πράκτορας ο οποίος δεν ανήκει σε ένα συνασπισμό C εκλέξει σαν $leader$ ένα άλλο πράκτορα (εκτός του εαυτού του), τότε η πιθανότητα να πάρουμε σαν τελική τιμή συμφωνίας το 1 να ισούται με την πιθανότητα να πάρουμε σαν τελική τιμή συμφωνίας το 0.

Λήμμα 1.3 Έστω $cheaters\ C$ και $u \notin C$. Αν ένας $leader$ εκλεγεί από τον u και είναι ο $p \neq u$, τότε $P\ (u.consensus = 1) = P\ (u.consensus = 0) = \frac{1}{2}$.

Απόδειξη. Η τιμή της συμφωνίας εξαρτάται από το $input\_sum$. Από τον αλγόριθμο $do\_consensus\ {}_{i}^{uni}(input, id)$ έχουμε μετά τη γραμμή 73:

$$u.input\_sum\ =\ InputT\ (u)\ +\ InputL\ (u)\ +\ Trip\ (u.leader, u).input$$

Αφού τώρα δεν έχουμε ομοιόμορφη κατανομή στις εισόδους των πρακτόρων, υποθέτουμε ότι η πιθανότητα να έχουμε το 0 σαν είσοδο ισούται με x και η πιθανότητα να πάρουμε το 1 σαν είσοδο ισούται με $1 - x$.

Περίπτωση 1: Εάν η πιθανότητα να πάρουμε το 0 σαν είσοδο (x) είναι μεγαλύτερη από την πιθανότητα να πάρουμε το 1 σαν είσοδο $(1 - x)$:

$$P(u.input = 0, \forall u \in Ring) > P(u.input = 1, \forall u \in Ring)$$

Περίπτωση 2**:** Εάν η πιθανότητα να πάρουμε το 0 σαν είσοδο (x) είναι μικρότερη από την πιθανότητα να πάρουμε το 1 σαν είσοδο $(1 - x)$:

$$P(u.input = 0, \forall u \in Ring) < P(u.input = 1, \forall u \in Ring)$$

Περίπτωση 3: Εάν η πιθανότητα να πάρουμε το 0 σαν είσοδο (x) ισούται με την πιθανότητα να πάρουμε το 1 σαν είσοδο $(1 - x)$, τη οποία περίπτωση έχουμε ήδη ελέγξει και αποδείξει.

Όπως έχουμε αναφέρει πιο πάνω, η τιμή της συμφωνίας εξαρτάται από το $input\_sum$, το οποίο με τη σειρά του εξαρτάται από το $InputT(u)$.

Επομένως, θα ελέγξουμε για τις περιπτώσεις 1 και 2 τις τιμές που παίρνει το $InputT(u)$.

- $InputT(u)$ στη Περίπτωση 1:

  Σε αυτή την περίπτωση, η πιθανότητα να πάρουμε το 0 σαν είσοδο είναι μεγαλύτερη από την πιθανότητα να πάρουμε το 1 σαν είσοδο.

  Ο αριθμός των πρακτόρων μας n είναι ένας ζυγός αριθμός, έστω $n = 4$.

  Έστω οι κόμβοι $< v_1, v_2, \dots, v_4 >$.

  Οι πιθανοί συνδυασμοί εισόδων είναι οι πιο κάτω:

  (α) $< 0,0,0,0 >$: Άθροισμα=0$\Rightarrow$ζυγός αριθμός$\Rightarrow u.consensus = 0$

  (β) $< 0,0,0,1 >$: Άθροισμα=1$\Rightarrow$μονός αριθμός$\Rightarrow u.consensus = 1$

- $\underline{InputT(u)}$ στη Περίπτωση 2:

  Σε αυτή την περίπτωση, η πιθανότητα να πάρουμε το 1 σαν είσοδο είναι μεγαλύτερη από την πιθανότητα να πάρουμε το 0 σαν είσοδο.

  Ο αριθμός των πρακτόρων μας n είναι ένας ζυγός αριθμός, έστω $n = 4$.

  Έστω οι κόμβοι $< v_1, v_2, \dots, v_4 >$.

  Οι πιθανοί συνδυασμοί εισόδων είναι οι πιο κάτω:

  (α) $< 1,1,1,1 >$: Άθροισμα=4$\Rightarrow$ζυγός αριθμός$\Rightarrow u.consensus = 0$

  (β) $< 1,1,1,0 >$: Άθροισμα=3$\Rightarrow$μονός αριθμός$\Rightarrow u.consensus = 1$

Σύμφωνα με την ανάλυση των πιο πάνω περιπτώσεων, παρατηρούμε ότι η πιθανότητα το $InputT(u)$ να είναι μονός αριθμός ισούται με την πιθανότητα να είναι ζυγός αριθμός ($P = \frac{1}{2}$).

Επομένως, και το $input\_sum$ έχει τις ίδιες πιθανότητες ανεξάρτητα από το $InputL(u) + Trip(u.leader, u).input$.

Απόδειξη Περίπτωσης 1 με επαγωγή στο n.

1. Βάση Επαγωγής: Έστω $n = 4$ (ζυγός) και η πιθανότητα να πάρουμε 0 σαν είσοδο είναι μεγαλύτερη από την πιθανότητα να πάρουμε 1 σαν είσοδο.
   - $< 0,0,0,0 >$
   - $< 0,0,0,1 >$

   $\Rightarrow P\ [InputT\ (u)\ = μονός]\ = P\ [InputT\ (u)\ = ζυγός]\ = \frac{1}{2}.$

   $\Rightarrow P\ [input\_sum = μονός]\ = P\ [input\_sum = ζυγός]\ = \frac{1}{2}.$

   $\Rightarrow P\ [u.consensus = 1]\ = P\ [u.consensus = 0]\ = \frac{1}{2}.$
2. Επαγωγική υπόθεση: Έστω ότι η υπόθεση μας ισχύει για $n = k$. Θα αποδείξουμε ότι ισχύει και για $n = k + 4$.
3. Επαγωγικό βήμα: Σύμφωνα με την επαγωγική υπόθεση, ξέρουμε ότι $P\ (u.consensus = 1) = P(u.consensus = 0) = \frac{1}{2}$ για $n = k$. Σύμφωνα με τη βάση της επαγωγής, αυτό ισχύει και για $n = 4$.

Επομένως, η υπόθεση μας είναι σωστή για $n = k + 4$.

Απόδειξη Περίπτωσης 2 με επαγωγή στο n.
Αποδεικνύεται όπως και στη Περίπτωση 1.

Το Λήμμα έχει αποδειχθεί για μη ομοιόμορφη κατανομή στις εισόδους. ■

Χρησιμοποιώντας το πιο κάτω λήμμα, θέλουμε να αποδείξουμε ότι εάν εκλεγεί κάποιος πράκτορας u ως $leader$, ο οποίος ανήκει στο συνασπισμό C, τότε ο συνασπισμός C έχει τον πλήρη έλεγχο πάνω στην τελική απόφαση του u.

Λήμμα 1.4 Έστω cheaters C και $u \notin C$, όπου $u.input\_interface \in C$. Εάν ο $leader$ που έχει εκλεγεί από τον u είναι ο u, τότε ο C έχει τον πλήρη έλεγχο πάνω στην απόφαση του u.

Απόδειξη. Αφού $u.interface \in C$, σημαίνει ότι ο u είναι $cheater$ και $InputT(u) = u.input$.
Μετά τη γραμμή 73 του αλγορίθμου:

$$u.input\_sum = u.input + InputL(u) + Trip(u.leader, u).input$$
$$= 2u.input + InputL(u)$$

Επομένως, το αποτέλεσμα συμφωνίας του αλγορίθμου είναι:

$u.consensus = u.input\_sum \bmod 2 = (2u.input + InputL(u)) \bmod 2 = InputL(u) \bmod 2$, που σημαίνει ότι η απόφαση του u εξαρτάται μόνο από το $InputL(u)$, δηλαδή μόνο από τις τιμές που έχουν σταλεί στον u από τον $u.input\_interface$, το οποίο ανήκει στο C, επομένως το C έχει τον πλήρη έλεγχο πάνω στην απόφαση του u. ■

Χρησιμοποιώντας το πιο κάτω λήμμα , θέλουμε να δείξουμε ότι εάν οι πράκτορες που θα αποτελέσουν την Ισορροπία $(n-2)$ ανήκουν στο C και οι άλλοι δύο πράκτορες που απομένουν δεν είναι γείτονες στο δακτύλιο, τότε σε αυτή την περίπτωση το C δεν θα έχει κανένα όφελος να κλέψει.

Λήμμα 1.5 Έστω cheaters C με $|C| = n-2$. Επίσης, έστω $u1, u2 \notin C$ δεν είναι γείτονες στο δακτύλιο. Σε αυτή την περίπτωση το C δεν έχει λόγο να κλέψει.

Απόδειξη. Έστω, χωρίς απώλεια της γενικότητας, ότι το C προτιμά η συμφωνία να είναι 1. Για ευκολία, θα μοιράσουμε την ανάλυση μας στις ακόλουθες περιπτώσεις:

- Περίπτωση 1: Το u_1 και u_2 έχουν λάβει διαφορετικές my_rand ή id τιμές κατά τη διάρκεια εκτέλεσης του αλγορίθμου. Σε αυτή τη περίπτωση μπορούμε στην ουσία να θεωρήσουμε ότι τα u_1 και u_2 ήταν σε διαφορετικούς δακτύλιους, επομένως οι αποφάσεις τους είναι ανεξάρτητες.

  Από το Λήμμα 1.2, η πιθανότητα ο $leader$ που εκλέγηκε από τον u_1 να είναι ο ίδιος ο u_1 είναι $\frac{1}{n}$.

  $\Rightarrow P\ (u_1.leader = u1) = \frac{1}{n}$.

  $\Rightarrow P\ (u_1.leader \neq u1) = 1 - P\ (u_1.leader = u_1) = 1 - \frac{1}{n} = \frac{(n-1)}{n}$.

  $\Rightarrow$Η πιθανότητα ο $leader$ που εκλέγηκε από τον u_1 να μην είναι ο u_1 είναι $\frac{(n-1)}{n}$

  $\Rightarrow P\ (u_1.leader \neq u_1) = \frac{(n-1)}{n}$.

  (1) $P\ (u_1.leader = u_1) = \frac{1}{n}$

  (2) $P\ (u_1.leader \neq u_1) = \frac{(n-1)}{n}$

  Από το Λήμμα 1.3 μπορούμε να εξαγάγουμε ότι¨

  (3) $P\ (u_1.consensus = 1 | u_1.leader \neq u_1) = \frac{1}{2}$

  Από το Λήμμα 1.4 ξέρουμε ότι δεδομένου ότι ο $leader$ που εκλέγηκε από τον u είναι ο ίδιος ο u, αυτό σημαίνει ότι το C έχει τον πλήρη έλεγχο πάνω στην απόφαση του u και αφού για την απόδειξη αυτού του λήμματος (1.5) υποθέσαμε ότι το C προτιμά οι πράκτορες να συμφωνήσουν στο 1, τότε στην τελική σίγουρα το $u_1.consensous$ θα ισούται με 1.

  (4) $P\ (u_1.consensus = 1 | u_1.leader = u1) = 1$

  Συνδυάζοντας τα πιο πάνω (1), (2), (3) και (4):

  Από (1) και (4):

  (α) $P\ (u_1.consensus = 1)$ =

  $P\ (u_1.leader = u_1).P\ (u_1.consensus = 1 | u_1.leader = u_1) = \frac{1}{n}.1 = \frac{1}{n}$

  Από (2) και (3):

  (β) $P\ (u_1.consensus = 1)$ =

$P\ (u_1.leader{\neq}u1).P\ (u_1.consensus = 1|u_1.leader{\neq}u1)\ = \frac{(n\text{-}1)}{n}.\frac{1}{2}$

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

$\Rightarrow$Από (α) και (β):

(5) $P\ (u_1.consensus = 1) = \frac{1}{n} + \frac{(n\text{-}1)}{2n}$

Το ίδιο αποτέλεσμα ισχύει και για το u_2.

Όπως έχουμε εξηγήσει και πιο πάνω, σε αυτή την περίπτωση το $u_1.consensus$ είναι ανεξάρτητο από το $u_2.consensus$, επομένως, η πιθανότητα για τη συμφωνία σε 1 εξαρτάται το (4):

(6) $P(u_1.consensus = 1 \wedge u_2.consensus = 1)$

$=\left(\frac{1}{n} + \frac{n-1}{2n}\right)^2 = \left(\frac{2+n-1}{2n}\right)^2 = \left(\frac{n+1}{2n}\right)^2 = \frac{(n+1)^2}{4n^2} = \frac{1}{4}\left[\frac{(n+1)^2}{n^2}\right] = \frac{1}{4}\left(\frac{n+1}{n}\right)^2 =$
$\frac{1}{4}\left(1+\frac{1}{n}\right)^2 \leq_{1*} \frac{1}{4}\left(1+\frac{1}{4}\right)^2 = \frac{25}{64} \sim 0.39 < \frac{1}{2}$

1*: Το $n \geq 4$, καθώς υπάρχουν 2 $non - cheaters$ $(u_1, u_2 \notin C)$ και ένας μη μηδενικός ζυγός αριθμός των $cheaters$.

Επομένως, σύμφωνα με τα πιο πάνω, το C δεν έχει κίνητρο να κλέψει στην Περίπτωση 1.

- <u>Περίπτωση 2</u>: Οι τιμές my_rand και id που έχουν λάβει τα u_1 και u_2 κατά τη διάρκεια του αλγορίθμου είναι οι ίδιες, αλλά οι τιμές εισόδου είναι διαφορετικές. Σε αυτή τη περίπτωση καθώς το $rand\_sum$ είναι το ίδιο και για το u_1 και για το u_2, ο $leader$ που εκλέγεται από τον u_1 είναι ο ίδιος που εκλέγεται από τον u_2.
  Από το Λήμμα 1.2:

  $\rightarrow\ P\ (leader = u_1) = \frac{1}{n}$

  $\rightarrow\ P\ (leader = u_2) = \frac{1}{n}$

  $\rightarrow\ P\ (leader{\neq}u_1, u_2)\ = 1 - P\ (leader = u_1) -\ P(leader = u_2)$

  $= 1 - \frac{1}{n} - \frac{1}{n} = 1 - \frac{2}{n} = \frac{n-2}{n}$

  (7) $P(leader \neq u_1, u_2) = \frac{n-2}{2}, P(leader = u_1) = \frac{1}{n}, P(leader = u_2) = \frac{1}{n}$

  Από το Λήμμα 1.3 έχουμε:

(8) $P(u_1.consensus = u_2.consensus = 1|leader \neq u_1, u_2) = \frac{1}{2} * \frac{1}{2} = \frac{1}{4}$

(9) $P(u_1.consensus = u_2.consensus = 1|leader = u_1) = 1 -$

$P(u_1.consensus = u_2.consensus = 1|leader \neq u_1) = 1 - \frac{1}{2} = \frac{1}{2}$

(10) $P(u_1.consensus = u_2.consensus = 1|leader = u_2) = 1 -$

$P(u_1.consensus = u_2.consensus = 1|leader \neq u_2)$

$= 1 - \frac{1}{2} = \frac{1}{2}$

Από τα (7) - (10) μπορούμε να συμπεράνουμε ότι:

(11) $P(u_1.consensus = 1\ and\ u_2.consensus = 1) = \frac{1}{4}\frac{(n-2)}{n} + \frac{1}{2} * \frac{1}{n} + \frac{1}{2} * \frac{1}{n} = \frac{1}{4}\left(1 - \frac{2}{n}\right) + 2 * \frac{1}{2n} = \frac{1}{4} - \frac{2}{4n} + 2 * \frac{1}{2n} = \frac{1}{4} - \frac{1}{2n} + 2 * \frac{1}{2n} = \frac{1}{4} + \frac{1}{2n} \leq_{1*} \frac{1}{4} + \frac{1}{8} = \frac{3}{8} \sim 0.375 < \frac{1}{2}$

Επομένως, σύμφωνα με τα πιο πάνω το C δεν έχει κανένα κίνητρο να κλέψει ούτε και στη Περίπτωση 2.

- Περίπτωση 3: Οι κόμβοι u_1 και u_2 λαμβάνουν ακριβώς τα ίδια triplets $< id, my\_rand, input >$ κατά τη διάρκεια εκτέλεσης του αλγορίθμου. Σε αυτή την περίπτωση, τουλάχιστον ένας από τους πράκτορες δεν θα επιλέξει τον εαυτό του ως $leader$. Από το Λήμμα 1.3 η πιθανότητα αυτού του κόμβου να αποφασίσει το 1 είναι $\frac{1}{2}$. Επομένως, το C δεν έχει κανένα κίνητρο να κλέψει στη Περίπτωση 3.

Σύμφωνα με τις περιπτώσεις 1-3, το Λήμμα 1.5 έχει αποδειχθεί. ■

Μια πιο γενική απόδειξη του Λήμματος 1.5. Θέλουμε να δείξουμε ότι καμία ομάδα C από n πράκτορες με $|C| < n$ δεν θα έχει κίνητρο να αποκλίνει από το πρωτόκολλο. Εάν ένας πράκτορας $i \in C$ αποκλίνει από το πρωτόκολλο όταν δεν στέλνει ένα μήνυμα με το id στο γείτονα του όταν παίρνει ένα σήμα για να ξεκινήσει η εκλογή του $leader$ και η διαδικασία επίτευξης συμφωνίας, τότε είτε δεν έχει κανένα αντίκτυπο (εάν οι άλλοι πράκτορες παίρνουν επίσης ένα σήμα) ή κανένας $leader$ δεν εκλέγεται ενώ κάποιος θα

μπορούσε να είχε εκλεγεί (επομένως ο πράκτορας i δεν κερδίζει τίποτα από το να αποκλίνει από το πρωτόκολλο.

Εάν ο i δεν διαβιβάσει κάποιο μήνυμα, το οποίο έπρεπε να είχε διαβιβάσει ή καθυστερήσει στη προώθηση ενός μηνύματος, και πάλι, είτε δεν θα έχει καμία επίδραση (εάν ένας πράκτορας με χαμηλότερο id παίρνει επίσης ένα σήμα) ή κανένας $leader$ δεν εκλέγεται όταν κάποιος θα μπορούσε να είχε εκλεγεί, επομένως ο i δεν κερδίζει τίποτα. Εάν ο i τροποποιήσει ένα μήνυμα, το οποίο πρέπει να προωθήσει, μπορεί να είναι η περίπτωση που διαφορετικοί πράκτορες που δεν ανήκουν στο C καταλήγουν με διαφορετικά ids ή διαφορετικά my_rands. Στη τελευταία περίπτωση, εάν το $rand\_sum$ για κάθε πράκτορα είναι το ίδιο, τότε η αλλαγή δεν θα έχει καμία επίδραση. Διαφορετικά, διαφορετικοί πράκτορες θα έχουν διαφορετικές απόψεις για το ποιος θα έπρεπε να είναι ο $leader$, δηλαδή δεν υπάρχει κανένας εκλεγμένος $leader$.

Και στις 2 περιπτώσεις, ο πράκτορας που αποκλίνει από το πρωτόκολλο δεν κερδίζει κάτι. Επιπλέον, εάν ένας πράκτορας $i \in C$ δεν στέλνει ένα αριθμό $2n$ γύρους αφότου ο «νικητής» αποστολέας έχει στείλει το μήνυμα του σε ένα πράκτορα που δεν ανήκει στο C, τότε δεν εκλέγεται κανένας $leader$ και ο i δεν έχει κερδίσει τίποτα. Επομένως, κάθε πράκτορας $i \in C$ θα στείλει κάποιο id εάν πάρει ένα σήμα στην αρχή και θα στείλει κάποιο random αριθμό N_i $[0 \dots n-1]$ σε κάποια κατάλληλη στιγμή. Ο πράκτορας i δεν κερδίζει τίποτα από την απόκλιση του από το πρωτόκολλο όταν στέλνει ένα id, το οποίο δεν είναι το πραγματικό του id, ούτε κερδίζει κάτι εάν επιλέξει ένα N_i αριθμό, ο οποίος δεν είναι τυχαίος. ■

Χρησιμοποιώντας το πιο κάτω λήμμα, θέλουμε να δείξουμε ότι εάν έχουμε ένα συνασπισμό από $cheaters$ C και δύο πράκτορες που δεν ανήκουν στο συνασπισμό αυτό και έχουν την ίδια διεπαφή εισόδου, τότε η πιθανότητα η απόφαση του ενός από τους δύο πράκτορες να ισούται με 0 ισούται με την πιθανότητα η απόφαση του να ισούται με 1.

Λήμμα 1.6 Έστω $cheaters$ C και δύο κόμβοι u_1 , $u_2 \notin C$ έτσι ώστε u_1=$u\_2.input\_interface$. Τότε $P\ (u_2.consensus = 1) = P\ (u_2.consensus = 0) = \frac{1}{2}$

Απόδειξη. Ελέγχουμε τη περίπτωση του u_2 για να φτάσουμε στην απόδειξη.
Από το Λήμμα 1.3 ξέρουμε ότι εάν το u_2 επιλέξει ένα $leader$ που δεν είναι ο εαυτός του, δηλαδή ένα $leader\ p \neq u_2$, τότε:

$P\ (u_2.consensus = 1)\ =\ P\ (u_2.consensus = 0)\ = \frac{1}{2}$.

Εάν το u_2 επιλέξει τον εαυτό του σαν $leader$, τότε:

$u_2.input\_sum = 2u_2.input + u_1.input +\ InputL\ (u)$.

Αφού $P(u_1.input\ = 1) = P\ (u_2.input = 1) = \frac{1}{2}$, σύμφωνα με το λήμμα 1.3 έχουμε

$P\ (u_2.consensus = 1) = P\ (u_2.consensus = 0) = \frac{1}{2}$. ■

Χρησιμοποιώντας το πιο κάτω λήμμα , θέλουμε να δείξουμε ότι εάν οι πράκτορες που θα αποτελέσουν την Ισορροπία $(n-2)$ ανήκουν στο C και οι άλλοι δύο πράκτορες που απομένουν δεν ανήκουν σε αυτό τον συνασπισμό και έχουν την ίδια διεπαφή εισόδου, τότε σε αυτή την περίπτωση το C δεν θα έχει κανένα όφελος να κλέψει αποκλίνοντας από τον αλγόριθμο.

Λήμμα 1.7 Έστω $cheaters\ C$ με $|C|\ =\ n-2$ και δύο κόμβοι $u_1, u_2 \notin C$ έτσι ώστε $u\_1 = u\_2.input\_interface$. Τότε δεν υπάρχει κανένα κίνητρο για το C να αποκλίνει από τον αλγόριθμο.

Απόδειξη. Είναι επακόλουθη του λήμματος 1.6 ■

Τα πιο πάνω λήμματα οδηγούν στο πιο κάτω κύριο θεώρημα, το Θεώρημα 1:

Θεώρημα 1 Ο αλγόριθμος $do\_consensus_i^{uni}$ είναι ένας $(n-2)$ - Ισχυρός Αλγόριθμος Ισορροπίας με μη ομοιόμορφη κατανομή στις εισόδους των πρακτόρων:

Απόδειξη. Από το Λήμμα 1.1 ξέρουμε ότι ο αλγόριθμος διατηρεί τόσο την εγκυρότητα όσο και την ύπαρξη συμφωνίας όταν κάθε κόμβος του δακτυλίου τον ακολουθεί και επίσης ξέρουμε ότι $P\ (ring\ decides\ 0) = P\ (ring\ decides\ 1) = \frac{1}{2}$. Από τα Λήμματα 1.5 και 1.7 ξέρουμε ότι ανεξάρτητα από τη θέση των 2 $non-cheaters$ $((n-2)$ -

Ισχυρή - Ισορροπία) είτε είναι γείτονες είτε όχι, δεν υπάρχει κανένα κίνητρο για τους $cheaters$ να κλέψουν.

Επομένως, ο αλγόριθμος που έχουμε προτείνει παρέχει όντως μια $(n-2)$ - Ισχυρή - Ισορροπία. ■

Κεφάλαιο 5

Μια $(n-2)$ - Ισχυρή - Ισορροπία σε Ασύγχρονο Δακτύλιο

5.1 Εισαγωγή 84

5.2 Αλγόριθμος $(n-2)$ - Ισχυρής - Ισορροπίας με Ομοιόμορφες Εισόδους 85

5.3 Ανάλυση αλγορίθμου $(n-2)$ - Ισχυρής - Ισορροπίας με Ομοιόμορφες Εισόδους 92

5.4 Αλγόριθμος $(n-2)$ - Ισχυρής - Ισορροπίας με Μη Ομοιόμορφες Εισόδους 101

5.5 Ανάλυση αλγορίθμου $(n-2)$ - Ισχυρής - Ισορροπίας με Μη Ομοιόμορφες Εισόδους 106

5.1 Εισαγωγή

Στο Κεφάλαιο 4 έχουμε βρει ένα αλγόριθμο που να παρέχει μια $(n-2)$ - Ισχυρή - Ισορροπία σε ένα ασύγχρονο δακτύλιο μονής κατεύθυνσης (unidirectional) με ομοιόμορφη κατανομή στις εισόδους και με την υπόθεση ότι ο αριθμός των πρακτόρων n είναι ένας ζυγός αριθμός. Έχουμε αποδείξει ότι και για τους τρείς αλγορίθμους που παρέχονται σε αυτή τη διπλωματική εργασία ένας δακτύλιος μονού μεγέθους δεν μπορεί να δώσει λύση για το πρόβλημα της $(n-2)$ - Ισχυρής - Ισορροπίας, καθώς παραβιάζεται η συνθήκη εγκυρότητας των αλγορίθμων. Επομένως, και για τους δύο αλγορίθμους που θα δούμε σε αυτό το κεφάλαιο, ο αριθμός των πρακτόρων που βρίσκονται στο δακτύλιο θεωρούμε ότι είναι ένας ζυγός αριθμός.

Ο πρώτος αλγόριθμος που θα δούμε στο κεφάλαιο αυτό, παρέχει μια $(n-2)$ - Ισχυρή - Ισορροπία σε ένα ασύγχρονο δακτύλιο μονής κατεύθυνσης με την προϋπόθεση ότι οι εισόδοι που παρέχονται στους πράκτορες ακολουθούν την ομοιόμορφη κατανομή. Ο

αλγόριθμος αυτός βασίζεται στον αλγόριθμο που έχουν προτείνουν στο [1] οι Afek et al., ο οποίος παρουσιάζεται στο Παράρτημα Α της εργασίας αυτής. Στο Κεφάλαιο 3.8 βρίσκεται και η επεξήγηση του συγκεκριμένου αλγορίθμου.

Για την κατασκευή της απόδειξης ότι ο αλγόριθμος που προτείνω παρακάτω παρέχει όντως μια $(n-2)$ - Ισχυρή - Ισορροπία σε ένα σύγχρονο δακτύλιο, χρησιμοποιώ τους ορισμούς που χρησιμοποιούν και οι Afek et al. στο [1] και που αναγράφονται στο υποκεφάλαιο «Εισαγωγή» του Κεφαλαίου 3, καθώς και τους ορισμούς που παρατίθενται στο τέλος του υποκεφαλαίου «Θεωρήματα που έχουν αποδειχθεί» επίσης του Κεφαλαίου 3.

Θα ασχοληθούμε όπως έχω αναφέρει και πιο πάνω με την ασύγχρονη περίπτωση δακτυλίου, όπου όταν ένας πράκτορας κάνει μια ερώτηση δεν χρειάζεται να πάρει πρώτα την απάντηση για να συνεχίσει τη λειτουργία του, αλλά συνεχίζει κανονικά να εκτελεί τη λειτουργία του και μπορεί να λάβει την απάντηση που θέλει οποιαδήποτε στιγμή κατά τη διάρκεια εκτέλεσης του αλγορίθμου επικοινωνίας.

Για ένα δακτυλίδι μονής κατεύθυνσης επικοινωνίας, μπορούμε να βρούμε ένα αλγόριθμο που να παρέχει μια $(n-2)$ - Ισχυρή - Ισορροπία, όπου υπάρχουν τουλάχιστον 3 πράκτορες για τη n Ισορροπία, δηλαδή 5 για την $(n-2)$ - Ισχυρή - Ισορροπία. Ο αριθμός των πρακτόρων θεωρούμε ότι είναι ζυγός αριθμός, επομένως ο ελάχιστος αριθμός πρακτόρων που πρέπει να υπάρχουν στο δακτύλιο είναι έξι πράκτορες. Αυτό θα το δούμε και παρακάτω στην ανάλυση του αλγορίθμου.

Επίσης, σε αυτό το κεφάλαιο, θα δούμε ένα παρόμοιο αλγόριθμο με αυτό που έχουμε αναφέρει πιο πάνω, με τη διαφορά ότι οι εισόδοι που θα λαμβάνουν οι πράκτορες δεν θα ακολουθούν την ομοιόμορφη κατανομή. Ο αλγόριθμος αυτός προκύπτει με ελάχιστες αλλαγές του προηγούμενου αλγορίθμου.

5.2 Αλγόριθμος $(n-2)$ − Ισχυρής – Ισορροπίας με Ομοιόμορφες Εισόδους

Με βάση τον αλγόριθμο που προτείνουν οι Afek et al. στο [1], θέλουμε να δείξουμε ότι μπορεί να κατασκευαστεί ένας νέος αλγόριθμος που να λύνει το πρόβλημα της

Κατανεμημένης συμφωνίας παρέχοντας μια $(n-2)$ - Ισχυρή - Ισορροπία σε ένα ασύγχρονο δακτύλιο ζυγού μεγέθους και με την προϋπόθεση ότι οι εισόδοι των πρακτόρων ακολουθούν την ομοιόμορφη κατανομή. Στο κεφάλαιο αυτό, θα παρουσιάσουμε τον αλγόριθμο αυτό και θα εξηγήσουμε αναλυτικά τη λειτουργία του. Στο επόμενο κεφάλαιο (5.3), θα αποδείξουμε ότι ο αλγόριθμος αυτός είναι έγκυρος και δουλεύει σωστά παρέχοντας όντως μια $(n-2)$ - Ισχυρή Ισορροπία. Συνεπώς, παρέχοντας μια $(n-2)$ - Ισχυρή - Ισορροπία, ο αλγόριθμος αυτός θα είναι σε θέση να λύσει το πρόβλημα της Κατανεμημένης Συμφωνίας, παρέχοντας μια τιμή συμφωνίας στην οποία και θα συμφωνήσουν όλοι οι πράκτορες του δακτυλίου.

Στη συνέχεια, στο Κεφάλαιο 5.4 θα δούμε μια παραλλαγή του πιο πάνω αλγορίθμου για να παρέχει μια $(n-2)$ - Ισχυρή - Ισορροπία σε ένα ασύγχρονο δακτύλιο αλλά με την προϋπόθεση ότι οι εισόδοι των πρακτόρων δεν ακολουθούν την ομοιόμορφη κατανομή. Ουσιαστικά θα δούμε απλά ποιες είναι οι σημαντικές αλλαγές που πρέπει να γίνουν στον αλγόριθμο ομοιόμορφης κατανομής και στις αποδείξεις του για να υποστηριχθεί και η μη ομοιόμορφη περίπτωση.

Θεωρώντας ότι υπάρχει ομοιόμορφη κατανομή πάνω στις εισόδους των πρακτόρων και χρησιμοποιώντας ένα ασύγχρονο δακτύλιο ζυγού μεγέθους, θα κατασκευάσουμε ένα αλγόριθμο με βάση τον αλγόριθμο που προτείνεται στο [1], ο οποίος δουλεύει και παρέχει μια $(n-2)$ - Ισχυρή - Ισορροπία και είναι ο αλγόριθμος που ακολουθεί:

Αλγόριθμος 2: Πρωτόκολλο $Asynchronous\_do\_consensus_i^{uni}(input, id)$ που εκτελείται από κάθε πράκτορα i σύμφωνα με τον αλγόριθμο που προτείνουν οι Afek et al. στο [1]:

Asynchronous_do_consensus $_i^{uni}$**(*input*, *id*):**

```
1  /*Block αφύπνισης πρακτόρων*/
2  type_i:=1 //τι είδους μήνυμα περιμένουμε
3  next_i:=id_i //από ποιόν περιμένουμε το μήνυμα
4  my_rand:=rand() //τυχαίος ακέραιος αριθμός από 0…n-1
5  ids_array.append(<id, input, my_rand>)
6  input_sum=input
```

```
Send (<input,id, my_rand>,out_interface) //το out_interface
δηλώνει τους απογόνους του i στον δακτύλιο
for i=1,…,k: //k φάσεις
     tmp:=Recv(<input, id, my_rand>, in_interface)
     if (tmp=null or not(tmp.input∈{0,1})){
         REPORT CHEATER
         System.exit(0)
     }
  input_sum+=tmp.input
  /*Μετά την παραλαβή του μηνύματος(<id,ids_array>) από τον
  πρόγονο*/
  tmp=Recv(<id, ids_array>, in_interface)
     if tmp.id>next_i and type_i==1 then{
          next_i:=id_q
          ids_array.append(<tmp.id,tmp.input>)
          Send (<id, ids_array>,out_interface)
     }
      if tmp.id=next_i then
          if id≠ id_i or type_i ≠ 1 or |ids_array|≠n then{
              leader_i :=⊥
              System.exit(0)
          }
          else{
              ids_array_i = ids_array
              type_i:=2
              Send (<id, ids_array>,out_interface)
          }
  /*Με την παραλαβή του μηνύματος (<id,ids_array>) από τον
  πρόγονο*/
  tmp:= Recv(<id, ids_array>, in_interface)
     if tmp.id=next_i and type_i=1 and |ids_array|=n then
          ids_array_i:=ids_array
```

```
36            type_i=2
37            Send (<id, ids_array>,out_interface)
38       else if type_i=1 then{
39            leader_i:=⊥
40            System.exit(0)
41       }
42    if ids_array=ids_array_i and id=id_i and type_i=2 then{
43            my_rand=rand();
44            Send (<ids_array_i, my_rand>,out_interface)
45            type_i=3
46       }
47    /*Με τη παραλαβή του μηνύματος  (<ids_array,my_rand'>) από
      τον  πρόγονο*/
48    tmp:= Recv(<ids_array, my_rand’>, in_interface)
49       N_pre:= my_rand'
50       if type_i=2 then
51            my_rand=rand();
52            Send (<ids_array, my_rand>,out_interface)
53            type_i=3
54       else if type_i=3 then
55            Send (<ids_array_i,<my_rand>>,out_interface)
56            type_i=4
57       /*Με την παραλαβή του μηνύματος
         (<ids_array,my_rand_list>) από τον προκάτοχο*/
58    tmp:= Recv(<ids_array,my_rand_list>, in_interface)
59       if type_i=3 and N_pre είναι το τελευταίο στοιχείο στη
         λίστα my_rand_list then
60            my_rand_list.append(<my_rand_i>)
61            Send (<ids_array, my_rand_list>, out_interface)
62            type_i=4
63       else if type_i=4 and N_pre είναι το τελευταίο στοιχείο
```

στη λίστα my_rand_{list} then

64 Send Choose_msg(<ids_array_i, my_rand_{list}>, out_interface)

65 end for

66 /*Με την παραλαβή του μηνύματος (<ids_array, my_rand_{list}>) από τον προκάτοχο*/

67 tmp:= Recv(<ids_array, my_rand_{list}>, in_interface)

68 if tmp.input≠input or tmp. my_rand≠my_rand or tmp.id≠id or my_rand_{list} δεν συμφωνεί με την προηγούμενη λίστα που έχει δειχθεί προηγουμένως then{

69 REPORT CHEATER

70 System.exit(0)

71 }

72 sort ids_array by id

73 if ∃id ∈$ids_{array}$with ids_{array}.count(id)≠1then{

74 REPORT CHEATER

75 System.exit(0)

76 }

77 $leader_i$:=ids_array$\left[\left(\sum_{my_rand' \in my_rand_{list}} my_rand'\right)\%n\right]$

78 /*Συμφωνία*/

79 input_sum+=$leader_i$.input

80 if input_sum%2==1 then return 1

81 else return 0

end $\boldsymbol{Asynchronous\_do\_consensus}_{i}^{uni}$

Συνασπισμοί των $Cheaters\ C$:

Σύμφωνα με το πρωτόκολλο που έχω παρουσιάσει πιο πάνω, ένας συνασπισμός από $cheaters$ με $|C| = 2$ έχει κίνητρο να αποκλίνει από τον αλγόριθμο.

Έστω ότι A είναι ο δημιουργός ενός μηνύματος και ο A είναι ο γείτονας του B έτσι ώστε ο B να είναι ο τελευταίος πράκτορας στη λίστα που προέρχεται από τον A. Εάν ο A και

ο Β σχηματίσουν ένα συνασπισμό, τότε ο Β δεν χρειάζεται να στείλει στον Α ένα τυχαίο αριθμό τη δεύτερη φορά γύρω από το δακτυλίδι. Αφού λάβει τις τυχαίες επιλογές του καθενός, ο Β μπορεί να επιλέξει ένα my_rand_B, έτσι ώστε αυτός ή ο Α να είναι ηγέτης και να αποφασίσει σε μια τιμή συμφωνίας σύμφωνα με το input του ηγέτη. Αυτή η τεχνική μπορεί να είναι καλύτερη για τον Α και τον Β παρά μια τυχαία επιλογή ηγέτη. Επομένως, ο αλγόριθμος μας παρέχει μια $(n-2)$ - Ισχυρή - Ισορροπία για
$n > 2k \Rightarrow n > 4 \Rightarrow$ Αφού το n είναι ζυγός αριθμός, τότε $n_{min} = 6$.

Επεξήγηση Αλγορίθμου $Asynchronous\_do\_consensus\ {}_{i}^{uni}$:

Έχουμε κατασκευάσει ένα αλγόριθμο που να παρέχει μια $(n-2)$ - Ισχυρή - Ισορροπία αν $n > 2k$. Κάθε πράκτορας i στέλνει τη τυχαία επιλογή του k βήματα γύρω από το δακτυλίδι και όχι μόνο στο γείτονα του. Αυτό σημαίνει ότι ο πράκτορας i στέλνει τη τυχαία επιλογή του my_rand_i σε k άλλους πράκτορες. Με περισσότερη λεπτομέρεια, κάθε πράκτορας που παίρνει ένα σήμα για να ξεκινήσει το πρωτόκολλο, στέλνει ένα μήνυμα με το id, το $input$ και το my_rand του στο γείτονα του. Τα μηνύματα τότε περνούν γύρω από τον δακτύλιο με κάθε πράκτορα να αποθηκεύει το id και το $input$ του στο ids_array.

Εάν ένας πράκτορας λάβει ένα δεύτερο διπλότυπο μήνυμα που είχε δημιουργηθεί από ένα διαφορετικό πράκτορα, το μήνυμα αγνοείται εάν ο δημιουργός έχει μικρότερο id, διαφορετικά, το μήνυμα περνάει παρακάτω. Τελικά, ο δημιουργός του μηνύματος με το μεγαλύτερο id παίρνει πίσω το μήνυμα. Ο δημιουργός ελέγχει για να βεβαιωθεί ότι το μήνυμα έχει n ids για να διασφαλίσει ότι δεν έχουν προστεθεί άλλα ψεύτικα ids. Το μήνυμα τότε στέλνεται γύρω από τον δακτύλιο για δεύτερη φορά και μαζί με το μήνυμα κάθε πράκτορας i συμπεριλαμβανομένου του αποστολέα στέλνει ένα τυχαίο αριθμό my_rand_i. Ο γείτονας του πράκτορα i δεν μεταβιβάζει το my_rand_i, απλά το κρατάει μόνο, ενώ διαβιβάζει τη λίστα των ids. Όταν ο δημιουργός του μηνύματος παίρνει το μήνυμα για τρίτη φορά, προωθεί στο γείτονα του το τυχαίο αριθμό που έχει λάβει στον προηγούμενο γύρο (ο αριθμός που δημιουργήθηκε από τον προκάτοχο του i στον δακτύλιο). Ο γείτονας του i, ξανά δεν προωθεί το μήνυμα. Αντίθετα, στέλνει στο διάδοχό του τον τυχαίο αριθμό που έχει λάβει από τον δημιουργό στον προηγούμενο γύρο. Στο τέλος αυτής της φάσης, κάθε πράκτορας ξέρει το τυχαίο αριθμό του δημιουργού του

μηνύματος και του προκατόχου του. Συνεχίζουμε αυτή τη διαδικασία για τις k φάσεις συνολικά. Δηλαδή, όταν ο δημιουργός του μηνύματος παίρνει το μήνυμα για τρίτη φορά , στέλνει αυτό το μήνυμα (που είναι ο τυχαίος αριθμός που επέλεξε ο προκάτοχος του προκατόχου του) στο διάδοχο του. Κάθε φορά που ένας πράκτορας λαμβάνει ένα μήνυμα, προωθεί το μήνυμα που έλαβε στις προηγούμενες φάσεις. Στο τέλος τη q φάσης για $q \leq k$, κάθε πράκτορας γνωρίζει τους τυχαίους αριθμούς των q πιο κοντινών προκατόχων του. Αφού ολοκληρωθούν αυτές οι k φάσεις, ο αποστολέας στέλνει τον τυχαίο αριθμό στο γείτονά του. Κάθε πράκτορας τότε αποθηκεύει το id του στη λίστα και το id πηγαίνει γύρο από το δακτυλίδι 2 φορές. Κάθε πράκτορας ελέγχει ότι οι τυχαίοι αριθμοί των k προκατόχων του συμφωνούν με αυτούς που του έχουν πει προηγουμένως. Στο τέλος αυτής της διαδικασίας, κάθε πράκτορας γνωρίζει όλους τους τυχαίους αριθμούς. Κάθε πράκτορας τότε εκλέγει ένα $leader$ και με βάση τον $leader$ και το $input\_sum$ αποφασίζει μια τιμή.

Κάθε φορά που ένας πράκτορας i παίρνει ένα μήνυμα, ελέγχει ότι αυτό είναι συμβατό με προηγούμενα μηνύματα που έχει δει. Δηλαδή, τη δεύτερη φορά που παίρνει το μήνυμα, όλα τα ids μεταξύ του δημιουργού και του id του πρέπει να είναι τα ίδια. Τη τρίτη φορά που παίρνει το μήνυμα, όλα τα ids στη λίστα πρέπει να είναι τα ίδια όπως στη δεύτερη φορά που είδε το μήνυμα. Τη τέταρτη φορά που λαμβάνει το μήνυμα, όχι μόνο η λίστα με τα ids πρέπει να είναι η ίδια, αλλά όλες οι τυχαίες επιλογές που έχει δει πριν δεν πρέπει να έχουν αλλάξει. Επίσης, ο πράκτορας i ελέγχει ότι έχει όλες τις τυχαίες επιλογές των k προκατόχων του στο δακτύλιο. Εάν το μήνυμα δεν περάσει με επιτυχία όλους αυτούς τους ελέγχους, τότε ο πράκτορας i θέτει τον $leader$ σε $\perp$ και ο αλγόριθμος τερματίζεται με την συμφωνία να αποτυγχάνει.

Η προσέγγιση που έχω χρησιμοποιήσει δεν μπορεί να χρησιμοποιηθεί όταν το $n = 4, (n - 2 = 2$, μεταξύ 2 πρακτόρων$)$, γιατί ο γείτονας του δημιουργού του μηνύματος θα πάρει τον τυχαίο αριθμό (τυχαία επιλογή = my_rand_i) του δημιουργού πριν να στείλει το δικό του τυχαίο αριθμό και έτσι τότε θα μπορεί να επιλέξει το δικό του τυχαίο αριθμό με τέτοιο τρόπο ώστε να εξασφαλίσει ότι θα γίνει ηγέτης.

Έτσι, ο πιο πάνω αλγόριθμος μας δίνει μια ισχυρή $(n-2)$ – Ισχυρή - Ισορροπία με την προϋπόθεση ότι υπάρχουν τουλάχιστον 6 πράκτορες ($n-2 > 2 \Rightarrow n > 4 \Rightarrow$ αφού το n είναι ζυγός αριθμός$\Rightarrow n_{min}$=6).

Επειδή πρόκειται για ένα ασύγχρονο δακτύλιο, οι πράκτορες έχουν περισσότερη ελευθερία στις κινήσεις τους. Τώρα, το αποτέλεσμα μπορεί να εξαρτάται όχι μόνο από το ποιοι πράκτορες παίρνουν ένα αρχικό μήνυμα, αλλά και από την σειρά με την οποία προγραμματίζονται οι πράκτορες και από τους χρόνους παράδοσης των μηνυμάτων. Το πρωτόκολλο που έχω παρουσιάσει έχει την ιδιότητα ότι αν το ακολουθήσουν όλοι οι πράκτορες, η κατανομή των αποτελεσμάτων είναι ανεξάρτητη από τις επιλογές του αντιπάλου. Επομένως, ανεξάρτητα από τις επιλογές που κάνει ο αντίπαλος, έχουμε μια $(n-2)$ - Ισχυρή - Ισορροπία.

5.3 Ανάλυση Αλγορίθμου $(n-2)$ - Ισχυρής – Ισορροπίας με Ομοιόμορφες Εισόδους

Στο κεφάλαιο αυτό παρέχουμε την ανάλυση του αλγορίθμου του Κεφαλαίου 5.2 μέσα από μια σειρά αποδείξεων λημμάτων, με σκοπό να φτάσουμε στην απόδειξη του κύριου θεωρήματος που αφορά τον αλγόριθμο που έχουμε κατασκευάσει. Το κύριο αυτό θεώρημα θα το δούμε στο τέλος του κεφαλαίου αυτού.

Αρχικά θα αποδείξουμε μέσα από το λήμμα που ακολουθεί, ότι ο αλγόριθμος που έχουμε προτείνει είναι έγκυρος και παρέχει μια αξιόπιστη τιμή συμφωνίας όπου η πιθανότητα η τιμή συμφωνίας να είναι το 0 ισούται με την πιθανότητα η τιμή συμφωνίας να είναι το 1.

Λήμμα 2.1 Εάν $n > 2k$ και όλοι οι κόμβοι στον δακτύλιο ακολουθούν τον αλγόριθμο, τότε ο αλγόριθμος $Asynchronous\_do\_consensus_i^{uni}$ διατηρεί τόσο την εγκυρότητα όσο και την συμφωνία και $P\ (ring\ decides\ 0)\ = P\ (ring\ decides\ 1)\ = \frac{1}{2}$.

Απόδειξη.

1. Απόδειξη εγκυρότητας:

- Εάν κάθε κόμβος έπερνε το 0 σαν είσοδο, σημαίνει ότι το τελικό $input\_sum$ θα ήταν 0, με αποτέλεσμα όλοι οι κόμβοι να συμφωνούσαν στο 0, καθώς $input\_sum\%2 = 0$.

- Εάν κάθε κόμβος έπερνε το 1 σαν είσοδο, σημαίνει ότι το τελικό $input\_sum$ θα ήταν $n + 1$ (+1= είσοδος του $leader$), με αποτέλεσμα όλοι οι κόμβοι να συμφωνούσαν στο 0 εάν το n είναι μονός αριθμός ή να συμφωνούσαν στο 1 εάν το n είναι ζυγός αριθμός με αντίστοιχα $input\_sum\%2 = 0$ και $input\_sum\%2 = 1$.
- Εάν κάποιοι κόμβοι λάμβαναν το 0 σαν είσοδο και άλλοι λάμβαναν το 1 σαν είσοδο, τότε κάθε αποτέλεσμα θα ήταν αποδεκτό όσο αφορά την εγκυρότητα.

2. Απόδειξη συμφωνίας: Εάν όλοι οι κόμβοι ακολουθούν τον αλγόριθμο, τότε κάθε κόμβος μετά τη γραμμή 71 θα έχει ακριβώς την ίδια my_rand_{list} , το ίδιο $input\_sum$ και ids_array (σύμφωνα με μια απλή επαγωγή), με αποτέλεσμα όλοι οι κόμβοι να συμφωνούν στην ίδια τιμή.
3. Για να αποδείξουμε ότι $P\ (ring\ decides\ 0) = P\ (ring\ decides\ 1) = \frac{1}{2}$, θα πρέπει να αποδείξουμε τους πιο κάτω ισχυρισμούς με επαγωγή στον αριθμό των κόμβων, δηλαδή των πρακτόρων n.

Ισχυρισμοί:

1. Έστω οι κόμβοι $v_1, v_2, \dots, v_{8k}$ (Χρησιμοποιούμε το $8k$, για να δείξουμε στην απόδειξη ότι ο αριθμός των κόμβων είναι πάντα ένας ζυγός αριθμός και χρησιμοποιούμε ένα αριθμό που είναι μεγαλύτερος από τον ελάχιστο αριθμό των πρακτόρων που είναι $n = 6$):

   $P\big(\big[\big(\sum_{i=1}^{8k} v_i.input\big)\big] mod\ 2 = 0\big) = P\big(\big[\big(\sum_{i=1}^{8k} v_i.input\big)\big] mod\ 2 = 1\big) = \frac{1}{2}$.

   - Η πιθανότητα το $input\_sum$ των εισόδων να είναι μονός ή ζυγός αριθμός είναι $\frac{1}{2}$.
   - $P\ (input\_sum = 0) = P\ (input\_sum = 1) = \frac{1}{2}$.

2. Έστω οι κόμβοι $v_1, v_2, \dots, v_{8k}$:

   $P\big(\big[\big(\sum_{i=1}^{8k} v_i.input\big) + leader.input\big] mod\ 2 = 0\big) =$

   $P\big(\big[\big(\sum_{i=1}^{8k} v_i.input\big) + leader.input\big] mod\ 2 = 1\big) = \frac{1}{2}$.

- Η πιθανότητα το $input\_sum$ των εισόδων συν την είσοδο του $leader$ να είναι μονός ή ζυγός αριθμός είναι $\frac{1}{2}$.
- $P\ (input\_sum\ +\ leader.input = 0)\ =\ P\ (input\_sum\ +\ leader.input = 1)\ =\ \frac{1}{2}$.

Απόδειξη των πιο πάνω ισχυρισμών με επαγωγή στον αριθμό των κόμβων n:

Βάση επαγωγής: Έστω $n = 8 \Rightarrow (v1, v2, v3, v4, v5, v6, v7, v8)$:

1ος Ισχυρισμός. Υπάρχουν 8 συνδυασμοί εισόδων:

(α) $P\ (input = 1) > P\ (input = 0)$

- $< 1,1,1,1,1,1,1,1 >$: Άθροισμα=8$\Rightarrow \left[\left(\sum_{i=1}^{8k} v_i.input\right)\right] mod\ 2 = 0$
- $< 1,1,1,1,1,1,1,0 >$: Άθροισμα=7$\Rightarrow \left[\left(\sum_{i=1}^{8k} v_i.input\right)\right] mod\ 2 = 1$
- $< 1,1,1,1,1,1,0,0 >$: Άθροισμα=6$\Rightarrow \left[\left(\sum_{i=1}^{8k} v_i.input\right)\right] mod\ 2 = 0$
- $< 1,1,1,1,1,0,0,0 >$: Άθροισμα=5$\Rightarrow \left[\left(\sum_{i=1}^{8k} v_i.input\right)\right] mod\ 2 = 1$

(β) $P\ (input = 1) < P\ (input = 0)$

- $< 0,0,0,0,0,0,0,0 >$: Άθροισμα=0$\Rightarrow \left[\left(\sum_{i=1}^{8k} v_i.input\right)\right] mod\ 2 = 0$
- $< 0,0,0,0,0,0,0,1 >$: Άθροισμα=1$\Rightarrow \left[\left(\sum_{i=1}^{8k} v_i.input\right)\right] mod\ 2 = 1$
- $< 0,0,0,0,0,0,1,1 >$: Άθροισμα=2$\Rightarrow \left[\left(\sum_{i=1}^{8k} v_i.input\right)\right] mod\ 2 = 0$
- $< 0,0,0,0,0,1,1,1 >$: Άθροισμα=3$\Rightarrow \left[\left(\sum_{i=1}^{8k} v_i.input\right)\right] mod\ 2 = 1$

Πίνακας 5.1 Συνδυασμοί Εισόδων Πρακτόρων και Άθροισμα

$<\left(\sum_{i=1}^{8k} vi.input\right), leader.input>$	Άθροισμα
<8,0>	$8\Rightarrow\left[\left(\sum_{i=1}^{8k} vi.input\right) + leader.input\right] mod\ 2 = 0$
<8,1>	$9\Rightarrow\left[\left(\sum_{i=1}^{8k} vi.input\right) + leader.input\right] mod\ 2 = 1$
<7,0>	$7\Rightarrow\left[\left(\sum_{i=1}^{8k} vi.input\right) + leader.input\right] mod\ 2 = 1$
<7,1>	$8\Rightarrow\left[\left(\sum_{i=1}^{8k} vi.input\right) + leader.input\right] mod\ 2 = 0$
<6,0>	$6\Rightarrow\left[\left(\sum_{i=1}^{8k} vi.input\right) + leader.input\right] mod\ 2 = 0$
<6,1>	$7\Rightarrow\left[\left(\sum_{i=1}^{8k} vi.input\right) + leader.input\right] mod\ 2 = 1$

<5,0>	$5 \Rightarrow \left[\left(\sum_{i=1}^{8k} vi.input\right) + leader.input\right] mod\ 2 = 1$
<5,1>	$6 \Rightarrow \left[\left(\sum_{i=1}^{8k} vi.input\right) + leader.input\right] mod\ 2 = 0$
<0,0>	$0 \Rightarrow \left[\left(\sum_{i=1}^{8k} vi.input\right) + leader.input\right] mod\ 2 = 0$
<0,1>	$1 \Rightarrow \left[\left(\sum_{i=1}^{8k} vi.input\right) + leader.input\right] mod\ 2 = 1$
<1,0>	$1 \Rightarrow \left[\left(\sum_{i=1}^{8k} vi.input\right) + leader.input\right] mod\ 2 = 1$
<1,1>	$2 \Rightarrow \left[\left(\sum_{i=1}^{8k} vi.input\right) + leader.input\right] mod\ 2 = 0$
<2,0>	$2 \Rightarrow \left[\left(\sum_{i=1}^{8k} vi.input\right) + leader.input\right] mod\ 2 = 0$
<2,1>	$3 \Rightarrow \left[\left(\sum_{i=1}^{8k} vi.input\right) + leader.input\right] mod\ 2 = 1$
<3,0>	$3 \Rightarrow \left[\left(\sum_{i=1}^{8k} vi.input\right) + leader.input\right] mod\ 2 = 1$
<3,1>	$4 \Rightarrow \left[\left(\sum_{i=1}^{8k} vi.input\right) + leader.input\right] mod\ 2 = 0$

Παρατηρούμε ότι στις μισές περιπτώσεις (4) το $\left[\left(\sum_{i=1}^{8k} v_i.input\right)\right] mod\ 2 = 0$ και στις άλλες μισές (4) το $\left[\left(\sum_{i=1}^{8k} v_i.input\right)\right] mod\ 2 = 1$. Επομένως, ο πρώτος ισχυρισμός ισχύει με $P\left(\left[\left(\sum_{i=1}^{8k} v_i.input\right)\right] mod\ 2 = 0\right) = P\left(\left[\left(\sum_{i=1}^{8k} v_i.input\right)\right] mod\ 2 = 0\right) = \frac{1}{2}$ για $n = 8$.

2ος Ισχυρισμός. Υπάρχουν 16 συνδυασμοί εισόδων:

Παρατηρούμε ότι στις μισές περιπτώσεις (8), το $\left[\left(\sum_{i=1}^{8k} v_i.input\right) + leader.input\right] mod\ 2 = 0$ και στις άλλες μισές (8) το $\left[\left(\sum_{i=1}^{8k} v_i.input\right) + leader.input\right] mod\ 2 = 1$.

Επομένως, ο δεύτερος ισχυρισμός ισχύει με
$P\left(\left[\left(\sum_{i=1}^{8k} v_i.input\right) + leader.input\right] mod\ 2 = 0\right) = P\left(\left[\left(\sum_{i=1}^{8k} v_i.input\right) + leader.input\right] mod\ 2 = 1\right) = \frac{1}{2}$ για $n = 8$.

4. Επαγωγική υπόθεση: Υποθέτουμε ότι οι 2 ισχυρισμοί ισχύουν για $n = 8\ (k - 1)\ = 8k - 8$. Θα αποδείξουμε ότι ισχύουν και για $n\ = 8k$.
5. Επαγωγικό βήμα:

   1ος Ισχυρισμός. $< v_1, v_2, v_3 \ldots,$
   $v_{8k-8}, v_{8k-7}, v_{8k-6}, v_{8k-5}, v_{8k-4}, v_{8k-3}, v_{8k-2}, v_{8k-1}, v_{8k} >$

Χρησιμοποιώντας την ίδια ανάλυση με τη βάση επαγωγής και την επαγωγική υπόθεση για $< v_1, v_2,.., v_{8k-8} >$, ο ισχυρισμός αποδεικνύεται και ισχύει για $n = 8k$.

2ος Ισχυρισμός. $< v_1, v_2, \ldots, v_{8k-1}, v_{8k} >$

- Εάν τα $v_{8k-7}, v_{8k-6}, v_{8k-5}, v_{8k-4}, v_{8k-3}, v_{8k-2}, v_{8k-1}$ και v_{8k} δεν εκλεγούν σαν leaders, τότε η εισφορά τους στο άθροισμα θα είναι είτε μονός αριθμός (7 για $< 1,1,1,1,1,1,1,0 >$, 5 για $< 1,1,1,1,1,0,0,0 >$, 1 για $< 0,0,0,0,0,0,0,1 >$ και 3 για $< 0,0,0,0,0,1,1,1 >$) στις μισές περιπτώσεις, είτε θα είναι ζυγός αριθμός (8 για $< 1,1,1,1,1,1,1,1 >$, 6 για $< 1,1,1,1,1,1,0,0 >$, 0 για $< 0,0,0,0,0,0,0,0 >$ και 2 για $< 0,0,0,0,0,0,1,1 >$) στις υπόλοιπες μισές. Και από την υπόθεση επαγωγής για τον ισχυρισμό 2, ξέρουμε ότι αν ένας από τους κόμβους $< v_1, v_2, \ldots, v_{8k-8} >$ εκλεγεί σαν $leader$, τότε το άθροισμα στις μισές περιπτώσεις θα είναι ζυγός αριθμός (8, 6, 0, 2, 8, 6, 2, 4) και στις άλλες μισές μόνος αριθμός (7, 5, 1, 3, 9, 7, 1, 3). Επομένως, ο ισχυρισμός 2 ισχύει για $n = 8k$.
- Εάν τα $v_{8k-7}, v_{8k-6}, v_{8k-5}, v_{8k-4}, v_{8k-3}, v_{8k-2}, v_{8k-1}$ και v_{8k} εκλεγούν σαν leaders, μπορούμε να χρησιμοποιήσουμε την επαγωγική υπόθεση του ισχυρισμού 1 στα $< v_1, v_2, \ldots, v_{8k-8} >$ και την επαγωγική υπόθεση του ισχυρισμού 2 στα $v_{8k-7}, v_{8k-6}, v_{8k-5}, v_{8k-4}, v_{8k-3}, v_{8k-2}, v_{8k-1}$ και v_{8k} και έτσι ο ισχυρισμός 2 ισχύει για $n = 8k$.

  Επομένως, από τον ισχυρισμό 2 είναι προφανές ότι $P(ring\ decides\ 0) = P(ring\ decides\ 1) = \frac{1}{2}$. ■

Χρησιμοποιώντας το πιο κάτω λήμμα, θα δείξουμε ότι εάν κανένας κόμβος δεν αποκλίνει από τον αλγόριθμο, τότε ο αλγόριθμος αυτός παρέχει μια δίκαιη $(n-2)$ - Ισχυρή - Ισορροπία σε ένα ασύγχρονο δακτύλιο μονής κατεύθυνσης μεταβίβασης μηνυμάτων (unidirectional ring).

Λήμμα 2.2 Εάν $n > 2k$ και όλοι οι κόμβοι στο δακτύλιο ακολουθούν τον αλγόριθμο, τότε ο αλγόριθμος $Asynchronous\_do\_consensus_i^{uni}$ είναι ένας αλγόριθμος που παρέχει μια δίκαιη $(n-2)$ - Ισχυρή - Ισορροπία σε ένα ασύγχρονο δακτύλιο μονής κατεύθυνσης μεταβίβασης μηνυμάτων (unidirectional ring).

Απόδειξη. Για να αποδείξουμε ότι ο αλγόριθμος μας παρέχει μια $(n-2)$ - Ισχυρή - Ισορροπία αρκεί να δείξουμε ότι εάν $n > 2(n-2)$, τότε κανένας συνασπισμός από cheaters C με $|C| \leq (n-2)$ δεν έχει κάποιο κίνητρο να αποκλίνει από τον αλγόριθμο. Θέλουμε να υποστηρίξουμε ότι κάποιος πράκτορας $i \in C$ δεν έχει κάποιο πλεονέκτημα να αποκλίνει από το πρωτόκολλο μέχρι και συμπεριλαμβανομένου του σημείου όπου ο πράκτορας i στέλνει τον τυχαίο του αριθμό my_rand_i, γιατί η απόκλιση είτε δεν έχει καμία επίδραση είτε έχει ως αποτέλεσμα να μην εκλεγεί κανένας $leader$ και ως εκ τούτου το πρωτόκολλο μα να μην παραθέτει μια τιμή συμφωνίας για κάθε πράκτορα.

Εάν ένας πράκτορας $i \in C$ αποκλίνει από τον αλγόριθμο με το να μη στείλει ένα υπογεγραμμένο μήνυμα (με το id του) στο γείτονα του όταν πάρει ένα σήμα για να ξεκινήσει η διαδικασία εκλογής του $leader$ και έπαρσης της τελικής απόφασης κάποιου πράκτορα, τότε είτε δεν θα υπάρξει καμία επίδραση εάν υπάρχουν και άλλοι πράκτορες που λαμβάνουν σήμα είτε δεν θα εκλεγεί κανένας $leader$. Επομένως σε αυτή την περίπτωση ο πράκτορας $i \in C$ δεν θα έχει κανένα απολύτως κίνητρο να αποκλίνει από τον αλγόριθμο.

Εάν ένας πράκτορας $i \in C$ δεν προωθήσει κάποιο μήνυμα, το οποίο ο i θα έπρεπε να προωθήσει κατά τη διάρκεια των πρώτων 2 φάσεων όπου μόνο η λίστα με τα ids περνάει γύρω από τον δακτύλιο, τότε είτε δεν θα υπάρξει καμία επίδραση είτε δεν θα εκλεγεί κανένας $leader$, επομένως και σε αυτή τη περίπτωση ο πράκτορας $i \in C$ δεν θα έχει κανένα απολύτως κίνητρο να αποκλίνει από τον αλγόριθμο.

Εάν ένας πράκτορας $i \in C$ τροποποιήσει ένα μήνυμα το οποίο πρέπει να προωθήσει, τότε διαφορετικοί πράκτορες που δεν ανήκουν στο C καταλήγουν με διαφορετικά ids, αλλά από τη στιγμή που όλοι οι πράκτορες συμφωνούν για το ποιο id είναι το μικρότερο, αυτό δεν πρόκειται να κάνει διαφορά. Εάν οι πράκτορες διαφωνούν για αυτό, τότε δεν θα έχουμε μόνο ένα ηγέτη. Όπως και πριν, αυτό δείχνει ότι όλοι οι πράκτορες θα στείλουν ένα id, αλλά όχι κατά ανάγκη το δικό τους. Αλλά σαφώς, δεν αποκομίζουν κάποιο όφελος από το να μην στείλουν το δικό τους πραγματικό id.

Παρόμοια, όλοι οι πράκτορες που ανήκουν στο C στέλνουν μηνύματα σε κάθε μία από τις $n-2$ φάσεις μετά που ο δημιουργός του μηνύματος έχει λάβει τη λίστα με τα ids για

δεύτερη φορά, αλλιώς δεν θα μπορούσε να εκλεγεί κάποιος ηγέτης. Οι πράκτορες που ανήκουν στο C θα ήθελαν να έχουν το άθροισμα $\sum_{i=1}^{n} my\_rand_i$, να είναι τέτοιο ώστε το μεγαλύτερο id να βρίσκεται στο C με αποτέλεσμα ο $leader$ που θα εκλεγεί να ανήκει στο C. Εάν κάποιος πράκτορας $i \in C$ μπορούσε να μάθει τους τυχαίους αριθμούς όλων των πρακτόρων που δεν ανήκουν στο C πριν να στείλει ένα τυχαίο αριθμό σε ένα πράκτορα $\notin C$, τότε ο i μπορούσε να επιλέξει ένα my_rand_i, έτσι ώστε να εκλεγεί ένας $leader \in C$.

Μια εύκολη επαγωγή, δείχνει ότι στο τέλος της rth φάσης μετά που ο δημιουργός του μηνύματος έχει λάβει τη λίστα με τα ids, ο πράκτορας $i \in C$ γνωρίζει τις τιμές των περισσότερων r πρακτόρων στο C που προηγούνται του i στο δακτύλιο.

Επομένως, στο τέλος της $n-2$ φάσης, ο πράκτορας i μπορεί να μάθει τις τυχαίες επιλογές των περισσότερων $n-2$ πρακτόρων $\notin C$ που προηγούνται του i στο δακτύλιο, όπως μπορεί να μάθει και τις τυχαίες επιλογές των άλλων πρακτόρων στο C. Επομένως, ο i γνωρίζει το πολύ $2(n-1) - 1$ τυχαίες επιλογές, αλλά αυτό δεν είναι αρκετό για τον i για να εμποδίσει ένα τυχαίο αποτέλεσμα. Επίσης, μέχρι το τέλος της $n-1$ φάσης, για κάθε $i \in C$, θα υπάρχει ένα $j \notin C$ που γνωρίζει την τυχαία επιλογή του i, αφού πρέπει να σταλεί στους $n-2$ πράκτορες που ακολουθούν τον i στο δακτύλιο και ένας από αυτούς δεν μπορεί να είναι στο C. Ο πράκτορας i πρέπει να στείλει μια τιμή, αλλιώς δεν θα μπορέσει να γίνει η εκλογή ενός $leader$ και μόλις ένας πράκτορας $j \notin C$ μάθει τη τιμή που έχει στείλει ο i, δεν υπάρχει πλεονέκτημα, άρα δεν υπάρχει κίνητρο για τον i να αποκλίνει από τον αλγόριθμο και να στείλει μια διαφορετική τιμή. Αυτό που βοηθά την κατάσταση αυτή, είναι ότι κάθε πράκτορας $i \in C$ πρέπει να δεσμευτεί με μια τιμή χωρίς να ξέρει αρκετές άλλες τιμές για να μπορέσει να επηρεάσει τη τελική τιμή του my_rand. Επομένως, δεδομένου ότι ο i δεσμεύεται με μια τιμή, ο i δεν κερδίζει τίποτα με το να δεσμευτεί με μια τιμή η οποία δεν είναι τυχαία. ■

Χρησιμοποιώντας τη πιο κάτω Πρόταση, θα δείξουμε ότι εάν ο αριθμός των πρακτόρων στο δακτυλίδι είναι μικρότερος ή ισούται με το διπλάσιο της k τιμής Ισχυρής Ισορροπίας, τότε δεν μπορεί να υπάρξει μια k - Ισχυρή - Ισορροπία σε ένα ασύγχρονο δακτύλιο μονής κατεύθυνσης (unidirectional) αποστολής μηνυμάτων.

Πρόταση 2.3 Εάν $n \leq 2k$, τότε δεν μπορεί να υπάρξει μια k - Ισχυρή - Ισορροπία σε ένα ασύγχρονο δακτύλιο μονής κατεύθυνσης (unidirectional) αποστολής μηνυμάτων.

Απόδειξη. Θα αποδείξουμε το πιο πάνω θεώρημα με τη μέθοδο της αντίφασης. Έστω $n = 2$ και $k = 1$ με πράκτορες p_0 και p_1 με $\mathrm{p}_0 .input = 0$ και $\mathrm{p}_1 .input = 1$. Έστω με τη μέθοδο της αντίφασης ότι το σύνολο των στρατηγικών που χρησιμοποιεί κάθε πράκτορας (s_0, s_1) αποτελεί μια Ισχυρή Ισορροπία. Όταν αναφερόμαστε σε στρατηγικές πρακτόρων εννοούμε το ιστορικό των μηνυμάτων που στέλνει και λαμβάνει κάποιος πράκτορας.

Μπορούμε να θεωρήσουμε την εξής στρατηγική για ευκολία στην απόδειξη: Έστω 2 πράκτορες, ο p_0 και p_1. Εάν ο πράκτορας p_0 στείλει ένα μήνυμα στο γύρο k, το μήνυμα στέλνεται τη χρονική στιγμή $4k + 1$ και λαμβάνεται από τον πράκτορα p_1 τη χρονική στιγμή $4k + 2$. Με την ίδια λογική, εάν ο πράκτορας p_1 στείλει ένα μήνυμα, το μήνυμα στέλνεται τη χρονική στιγμή $4k + 3$ και παραλαμβάνεται από τον πράκτορα p_0 τη χρονική στιγμή $4k + 4 = 4(k + 1)$.

Σύμφωνα με τα πιο πάνω, μπορούμε να καταλάβουμε ότι πριν να στείλει ένα μήνυμα στο γύρο k, ο πράκτορας p_1 έχει ήδη λάβει όλα τα μηνύματα που ο άλλος ο πράκτορας p_0 είχε στείλει νωρίτερα. Επομένως, για να αποδείξουμε το θεώρημα, θα δείξουμε ότι δεν μπορεί η συγκεκριμένη στρατηγική να παράξει μια k - Ισχυρή - Ισορροπία. Θεωρούμε ένα γύρο εκτέλεσης r του αλγορίθμου από κάποιο πράκτορα i. Οι πράκτορες χρησιμοποιούν τις στρατηγικές (s_0, s_1). Έστω ότι ο $leader$ που εκλέγεται στο τέλος του γύρου r είναι ο p_0. Έστω ότι τη χρονική στιγμή t ένας από τους δύο πράκτορες μαθαίνει για πρώτη φορά ότι ο ηγέτης θα είναι ο p_0. Η κατάσταση όπου ο p_0 θα είναι ο ηγέτης αποτελεί από μόνη της ένα σύνολο από γύρους εκτέλεσης του αλγορίθμου όπου ο p_0 είναι ο $leader$.

- Έστω h_i είναι το ιστορικό των μηνυμάτων του πράκτορα i και έστω ότι ο πράκτορας p_1 είναι αυτός που ξέρει ότι ο $leader$ είναι ο p_0 τη χρονική στιγμή t. Αυτό σημαίνει ότι τη χρονική στιγμή t, ο πράκτορας p_1 είτε έλαβε είτε έστειλε κάποιο μήνυμα, ενώ ο p_0 δεν έχει κάνει τίποτα. Αυτό σημαίνει ότι ο p_0 δεν ξέρει ότι έχει εκλεγεί $leader$ τη χρονική στιγμή t. Επομένως, πρέπει να υπάρχει ένα ${h_1}'$ για

τον πράκτορα p_1 και ένας γύρος r' έτσι ώστε ο γύρος r' να συμβαίνει με μια θετική πιθανότητα σύμφωνα με την στρατηγική της απόδειξης μας που έχουμε αναφέρει πιο πάνω και οι πράκτορες να χρησιμοποιούν τις στρατηγικές (s_0, s_1). Επίσης, ο p_1 εκλέγεται σαν $leader$ στο γύρο r' και (h_0, h_1) είναι το ιστορικό των μηνυμάτων των p_0 και p_1 σε κάποια χρονική στιγμή t' στο γύρο r. Εάν $t = 4k + 2$ που σημαίνει ότι ο πράκτορας p_1 έχει μόλις λάβει ένα μήνυμα από τον πράκτορα p_0, τότε το h_1 πρέπει λογικά να προηγείται του $h_1{}'$, γεγονός που έρχεται σε αντίθεση με την υπόθεση ότι ο πράκτορας p_1 ξέρει ότι ο πράκτορας p_0 θα είναι ο $leader$ δεομένης της ιστορίας του p_1, δηλαδή της ιστορίας h_1.

Έτσι, μπορούμε να υποθέσουμε ότι $t = t' = 4k + 3$, όπου οι ιστορίες του p_1 τη χρονική στιγμή $t - 1$ είναι πανομοιότυπες στους γύρους r και r' και ο p_1 έχει μόλις στείλει ένα μήνυμα τη χρονική στιγμή t σε τουλάχιστον ένα από τους γύρους r και r'. Έστω $h_1{}''$ να είναι το ιστορικό του p_1 τη χρονική στιγμή $t - 1$ στους γύρους r και r'. Αφού ο πράκτορας p_1 ενεργεί διαφορετικά τη χρονική στιγμή t στους γύρους r και r' χρησιμοποιώντας τη στρατηγική s_1, παρόλο που ο p_1 έχει το ίδιο ιστορικό τη χρονική στιγμή $t - 1$, πρέπει να είναι η περίπτωση που το s_1 τυχαιοποιείται. Αλλά σε αυτή την περίπτωση, τον p_1 τον συμφέρει να αποκλίνει από το πρωτόκολλο, για να αυξηθεί η πιθανότητα του να εκλεγεί ως ηγέτης, αφού πήρε μια απόφαση όπως στον γύρο r και μπορεί να προσποιηθεί ότι πήρε μια απόφαση την ίδια με τον γύρο r', καθώς προηγουμένως στο γύρο r', ο πράκτορας p_1 είχε εκλεγεί ως $leader$, όπως αναφέρουμε και πιο πάνω.

- Έστω h_i είναι το ιστορικό των μηνυμάτων του πράκτορα i και τώρα υποθέτουμε ότι ο πράκτορας p_0 είναι αυτός που ξέρει ότι ο $leader$ είναι ο p_0 τη χρονική στιγμή t. Επομένως, αυτή τη φορά, ο p_1 δεν ξέρει ότι ο p_0 είναι ο $leader$ τη χρονική στιγμή t. Επομένως, πρέπει να υπάρχει ένας γύρος r'' έτσι ώστε ο γύρος r'' να συμβαίνει με μια θετική πιθανότητα σύμφωνα με τη στρατηγική της απόφασης μας και οι πράκτορες να χρησιμοποιούν τις στρατηγικές (s_0, s_1). Επίσης, ο p_1 εκλέγεται ως $leader$ στο γύρο r'' και (h_0', h_1) είναι το ιστορικό των μηνυμάτων των p_0 και p_1 σε κάποια χρονική στιγμή t' στο γύρο r. Όπως και πιο πάνω, $t = 4k + 1$, επομένως, ο p_0 θα έχει ήδη στείλει ένα μήνυμα. Οι ιστορίες του p_0 τη χρονική στιγμή $t - 1$

στους γύρους r και r' είναι πανομοιότυπες και οι διαφορές ανάμεσα στα h_0' και h_0 εξαρτώνται από τον πράκτορα p_0 που λαμβάνει διαφορετικά τυχαία αποτελέσματα στους γύρους r και r''. Αλλά, τον p_0 τώρα τον συμφέρει να αποκλίνει από το πρωτόκολλο όταν η απόφαση που πήρε ήταν η ίδια με τον γύρο r'' και παριστάνοντας ότι η απόφαση που πήρε ήταν η ίδια με τον γύρο r. Επομένως, είναι σίγουρος ότι θα εκλεγεί ως $leader$ στο γύρο r''.

Και στις δύο πιο πάνω περιπτώσεις, φαίνεται ότι η στρατηγική (s_0, s_1) δεν αποτελεί μια Ισχυρή Ισορροπία, γιατί οι πράκτορες όπως φαίνεται και πιο πάνω, έχουν κίνητρο να αποκλίνουν από τον αλγόριθμο. ■

Από τα πιο πάνω Λήμματα και την Πρόταση 2.3, συμπεραίνουμε το πιο κάτω κύριο θεώρημα, το Θεώρημα 2:

Θεώρημα 2 Ο αλγόριθμος $Asynchronous\_do\_consensus_i^{uni}$ είναι ένας $(n-2)$ - Ισχυρός Αλγόριθμος Ισορροπίας με ομοιόμορφη κατανομή στις εισόδους των πρακτόρων:

Απόδειξη. Από το Λήμμα 2.1 ξέρουμε ότι ο αλγόριθμος διατηρεί τόσο την εγκυρότητα όσο και την ύπαρξη συμφωνίας όταν κάθε κόμβος του δακτυλίου τον ακολουθεί και επίσης ξέρουμε ότι $P\ (ring\ decides\ 0) = P\ (ring\ decides\ 1) = \frac{1}{2}$. Από το Λήμμα 2.2 ξέρουμε ότι δεν υπάρχει κανένα κίνητρο για τους πράκτορες που ανήκουν στον συνασπισμό C να κλέψουν.

Επομένως, ο αλγόριθμος που έχουμε προτείνει παρέχει όντως μια $(n-2)$ - Ισχυρή - Ισορροπία. ■

5.4 Αλγόριθμος $(n-2)$ - Ισχυρής - Ισορροπίας με Μη Ομοιόμορφες Εισόδους

Θεωρώντας ότι δεν υπάρχει ομοιόμορφη κατανομή πάνω στις εισόδους των πρακτόρων και χρησιμοποιώντας ένα ασύγχρονο δακτύλιο ζυγού μεγέθους, υπάρχει ένας αλγόριθμος με βάση τον αλγόριθμο που προτείνεται στο [1], ο οποίος δουλεύει και παρέχει μια

$(n-2)$ - Ισχυρή - Ισορροπία. Ο αλγόριθμος αυτός είναι μια παραλλαγή του αλγορίθμου του Κεφαλαίου 5.2 με ελάχιστες αλλαγές:

- Το block αφύπνισης των πρακτόρων παραμένει το ίδιο με τη διαφορά ότι τα μηνύματα αφύπνισης περιλαμβάνουν και τις πιθανότητες των τιμών εισόδου 0 και 1 (pos_0, pos_1), δηλαδή τις πιθανότητες που έχουν καθορίσει ποια είσοδος (0 ή 1) δίνεται στους περισσότερους πράκτορες.
- Κάθε απόγονος ενός πράκτορα i που λαμβάνει το μήνυμα αυτού του πράκτορα, ελέγχει εάν αυτός ο πράκτορας πήρε την είσοδο του με βάση όντως τις πιθανότητες pos_0 και pos_1 με τις οποίες πήρε και ο απόγονος τη δική του είσοδο. Επίσης γίνεται έλεγχος αν αυτές οι πιθανότητες ισούνται μεταξύ τους με τη τιμή 0.5, γιατί αν ισχύει αυτό σημαίνει ακολουθείται ομοιόμορφη κατανομή τις εισόδους και θα προκύψει σφάλμα στο πρωτόκολλο με αποτέλεσμα τον τερματισμό του.
- Εφόσον η τιμή της τελικής συμφωνίας εξαρτάται από το άθροισμα των εισόδων των πρακτόρων, τότε δεν χρειάζεται κάποια αλλαγή στον τρόπο υπολογισμού της τιμής συμφωνίας με βάση τις πιθανότητες pos_0 και pos_1, καθώς αυτή η αλλαγή αντικατοπτρίζεται στο τελικό άθροισμα καθώς είτε θα έχουμε περισσότερα μηδενικά είτε περισσότερους άσσους.

Αλγόριθμος 3:

Πρωτόκολλο $Asynchronous\_non\_uni\_do\_consensus_{i}^{non\_uni}(input, id)$
που εκτελείται από κάθε πράκτορα *i* σύμφωνα με τον αλγόριθμο που προτείνουν οι Afek et al. στο [1]:

Asynchronous_non_uni_do_consensus $_{i}^{uni}$**(*input*, *id*):**

```
1  /*Block αφύπνισης πρακτόρων*/
2  type_i:=1 //τι είδους μήνυμα περιμένουμε
3  next_i:=id_i //από ποιόν περιμένουμε το μήνυμα
4  my_rand:=rand() //τυχαίος ακέραιος αριθμός από 0…n-1
5  ids_array.append(<id,input, my_rand, pos_0, pos_1>)//pos_0,
   pos_1: πιθανότητες για είσοδο 0 ή 1 αντίστοιχα σύμφωνα με
   τις οποίες αποφασίστηκε μια τιμή εισόδου για τον πράκτορα i
```

```
6  input_sum=input
7  Send (<input,id, my_rand, pos_0, pos_1>,out_interface) //το
   out_interface δηλώνει τους απογόνους του i στον δακτύλιο
8  for i=1,…,k: //k φάσεις
9       tmp:=Recv(<input, id, my_rand, pos_0, pos_1>,
        in_interface)
10      if (tmp=null or not(tmp.input∈{0,1}) or
        (tmp.pos_0==tmp.pos_1==0.5)){
11          REPORT CHEATER
12          System.exit(0)
13      }
14      input_sum+=tmp.input
15      /*Μετά την παραλαβή του μηνύματος(<id,ids_array>) από
        τον πρόγονο*/
16      tmp=Recv(<id, ids_array, pos_0, pos_1>, in_interface)
17      if tmp.id>next_i and type_i==1 then{
18           next_i≔id_q
19           ids_array.append(<tmp.id,tmp.input>)
20           Send (<id, ids_array>,out_interface)
21      }
22    if tmp.id=next_i then
23           if id≠ id_i or type_i ≠ 1 or |ids_array|≠n then{
24               leader_i ≔⊥
25               System.exit(0)
26           }
27           else{
28               ids_array_i = ids_array
29               type_i:=2
30               Send (<id, ids_array>,out_interface)
31           }
32   /*Με την παραλαβή του μηνύματος (<id,ids_array>) από τον
     πρόγονο*/
```

```
33      tmp:= Recv(<id, ids_array>, in_interface)
34      if tmp.id=next_i and type_i=1 and |ids_array|=n then{
35           ids_array_i:=ids_array
36           type_i=2
37           Send (<id, ids_array>,out_interface)
38      else if type_i=1 then{
39           leader_i:=⊥
40           System.exit(0)
41      }
42    if ids_array=ids_array_i and id=id_i and type_i=2 then{
43           my_rand=rand();
44           Send (<ids_array_i, my_rand>,out_interface)
45           type_i=3
46      }
47    /*Με τη παραλαβή του μηνύματος  (<ids_array,my_rand'>) από
      τον  πρόγονο*/
48      tmp:= Recv(<ids_array, my_rand'>, in_interface)
49      N_pre:= my_rand'
50      if type_i=2 then
51           my_rand=rand();
52           Send (<ids_array, my_rand>,out_interface)
53           type_i=3
54      else if type_i=3 then
55           Send (<ids_array_i,<my_rand>>,out_interface)
56           type_i=4
57    /*Με την παραλαβή του μηνύματος (<ids_array,my_rand_list>)
      από τον προκάτοχο*/
58    tmp:= Recv(<ids_array,my_rand_list>, in_interface)
59      if type_i=3 and N_pre είναι το τελευταίο στοιχείο στη
        λίστα my_rand_list then
60           my_rand_list.append(<my_rand_i>)
```

61 Send (<ids_array, my_rand_{list}>, out_interface)

62 $type_i$=4

63 else if $type_i$=4 and N_{pre} είναι το τελευταίο στοιχείο στη λίστα my_rand_{list} then

64 Send Choose_msg(<ids_array_i, my_rand_{list}>, out_interface)

65 end for

66 /*Με την παραλαβή του μηνύματος (<ids_array, my_rand_{list}>) από τον προκάτοχο*/

67 tmp:= Recv(<ids_array, my_rand_{list}>, in_interface)

68 if tmp.input≠input or tmp. my_rand≠ my_rand or tmp.id≠id or my_rand_{list} δεν συμφωνεί με την προηγούμενη λίστα που έχει δειχθεί προηγουμένως or **tmp.pos_0 ≠pos_0 or tmp.pos_1 ≠ pos_1** then{

69 REPORT CHEATER

70 System.exit(0)

71 }

72 sort ids_array by id

73 if ∃id ∈$ids_{array}$with ids_{array}.count(id)≠1 then{

74 REPORT CHEATER

75 System.exit(0)

76 }

77 $leader_i$:=$ids\_array\left[\left(\sum_{my\_rand' \in my\_rand_{list}} my\_rand'\right)\%n\right]$

78 /*Συμφωνία*/

79 input_sum+=$leader_i$.input

80 if input_sum%2==1 then return 1

81 else return 0

end $\boldsymbol{Asynchronous\_non\_uni\_do\_consensus_i^{non\_uni}}$

5.5 Ανάλυση Αλγορίθμου $(n-2)$ - Ισχυρής – Ισορροπίας με Μη Ομοιόμορφες Εισόδους

Στο κεφάλαιο αυτό παρέχουμε την ανάλυση του αλγορίθμου του Κεφαλαίου 5.4 μέσα από μια σειρά αποδείξεων λημμάτων, με σκοπό να φτάσουμε στην απόδειξη του κύριου θεωρήματος που αφορά τον αλγόριθμο που έχουμε κατασκευάσει. Το κύριο αυτό θεώρημα θα το δούμε στο τέλος του κεφαλαίου αυτού.

Αρχικά θα αποδείξουμε μέσα από το λήμμα που ακολουθεί, ότι ο αλγόριθμος που έχουμε προτείνει είναι έγκυρος και παρέχει μια αξιόπιστη τιμή συμφωνίας όπου η πιθανότητα η τιμή συμφωνίας να είναι το 0 ισούται με την πιθανότητα η τιμή συμφωνίας να είναι το 1.

Λήμμα 3.1 Αν όλοι οι κόμβοι στο δακτύλιο ακολουθούν τον πιο πάνω αλγόριθμο $Asynchronous\_non\_uni\_do\_consensus_i^{non\_uni}$, τότε ο αλγόριθμος διατηρεί τόσο την εγκυρότητα του όσο και την ύπαρξη συμφωνίας και $P\ (απόφαση\ δακτυλίου = 0) = P(απόφαση\ δακτυλίου = 1) = \frac{1}{2}$.

Απόδειξη. Απόδειξη εγκυρότητας- Η απόδειξη εγκυρότητας είναι ακριβώς η ίδια με αυτή της περίπτωσης της σύγχρονης περίπτωσης του Κεφαλαίου 4.3.

Απόδειξη συμφωνίας- Αν όλοι οι κόμβοι ακολουθούν τον αλγόριθμο $Asynchronous\_non\_uni\_do\_consensus\ _i^{non\_uni}$, τότε μετά τη γραμμή 72, κάθε κόμβος θα έχει τις ίδιες τιμές για $input\_sum, ids\_array$ και my_rand_{list} (σύμφωνα με μια απλή επαγωγή), που σημαίνει ότι όλοι οι κόμβοι θα συμφωνούν στην ίδια τιμή.

Για να αποδείξουμε ότι $P\ (ring\ decides\ 0) = P\ (ring\ decides\ 1) = \frac{1}{2}$, θα πρέπει να αποδείξουμε τους πιο κάτω ισχυρισμούς με επαγωγή στον αριθμό των κόμβων, δηλαδή των πρακτόρων n.

Ισχυρισμοί:

1. Έστω οι κόμβοι v_1 , v_2 ,…, v_{8k} : $P\left(\left[\sum_{i=1}^{8k} v_i.input\right] mod 2 = 0\right) = P\left(\left[\sum_{i=1}^{8k} v_i.input\right] mod 2 = 1\right) = \frac{1}{2}$.

2. Έστω οι κόμβοι v_1, v_2,…, v_{8k}: $P\left(\left[\left(\sum_{i=1}^{8k} v_i.input\right) + leader.input\right] mod 2 = 0\right) = P\left(\left[\left(\sum_{i=1}^{k} v_i.input\right) + leader.input\right] mod 2 = 1\right) = \frac{1}{2}$.

   $\Rightarrow$ Έστω $n = 8$, καθώς για να λειτουργήσει ο αλγόριθμος μας όπως είχαμε αναφέρει στην επεξήγησή του, το $n \geq 6$, επομένως για μεγαλύτερη ασφάλεια θα χρησιμοποιήσουμε το $n = 8$. Επομένως, έχουμε τους κόμβους $<v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8>$.

   Απόδειξη ισχυρισμών με επαγωγή στον αριθμό των κόμβων n.

   Η απόδειξη είναι ακριβώς η ίδια με την απόδειξη για ομοιόμορφη κατανομή εισόδων ασύγχρονου δακτυλίου στο Κεφάλαιο 5.3. ■

Χρησιμοποιώντας το λήμμα που ακολουθεί, θέλουμε να αποδείξουμε ότι οποιοσδήποτε πράκτορας στον δακτύλιο έχει την ίδια πιθανότητα παίξει καθοριστικό ρόλο στην επιλογή της τιμής συμφωνίας, δηλαδή να εκλεγεί ως $leader$, ανεξάρτητα από τις ενέργειες των πρακτόρων που προσπαθούν να παρεκκλίνουν από το πρωτόκολλο για προσωπικό συμφέρον.

Λήμμα 3.2 Έστω $cheaters\ C$ και $u \notin C$, τότε κάθε id στο $u.ids\_array$ έχει την ίδια πιθανότητα του $\frac{1}{n}$ να εκλεγεί ως $leader$ από τον u, άσχετα με τις ενέργειες που λαμβάνονται από το C.

Απόδειξη. Φαίνεται από τον αλγόριθμο $Asynchronous\_non\_uni\_do\_consensus_i^{non\_uni}(input, id)$. Αρχικά, το να εκλεγεί κάποιος $leader$ δεν εξαρτάται από το $input\_sum$, αλλά από το άθροισμα των στοιχείων της λίστας my_rand_{list}. Το my_rand' κάθε κόμβου παίρνει τιμές $\{0, .., n-1\}$ και σύμφωνα με τον αλγόριθμο είναι ομοιόμορφα κατανεμημένο. Επομένως, και το $RandomT(u)$ είναι ομοιόμορφα κατανεμημένο με αποτέλεσμα το $\sum_{my\_rand' \in my\_rand_{list}} my\_rand'\ mod\ n$ να είναι επίσης ομοιόμορφα κατανεμημένο. Επιπλέον, αν όλοι οι κόμβοι $u \notin C$ ακολουθούν τον αλγόριθμο, σημαίνει ότι όλοι οι κόμβοι μετά τη γραμμή 72 θα έχουν τις ίδιες τιμές στο my_rand_{list}. Επομένως, $leader := ids\_array\ [\ (\sum_{my\_rand' \in my\_rand_{list}} my\_rand')\ \%n\ ]$ και κάθε id στο

$u.ids\_array$ έχει την ίδια πιθανότητα του $\frac{1}{n}$ να εκλεγεί ως $leader$ από τον u άσχετα με τις ενέργειες που λαμβάνονται από το C. ■

Χρησιμοποιώντας το λήμμα που ακολουθεί, θέλουμε να δείξουμε ότι εάν ένας πράκτορας ο οποίος δεν ανήκει σε ένα συνασπισμό C εκλέξει σαν $leader$ ένα άλλο πράκτορα (εκτός του εαυτού του), τότε η πιθανότητα να πάρουμε σαν τελική τιμή συμφωνίας το 1 να ισούται με την πιθανότητα να πάρουμε σαν τελική τιμή συμφωνίας το 0.

Λήμμα 3.3 Έστω $cheaters\ C$ και $u \notin C$. Αν ένας $leader$ εκλεγεί από τον u και είναι ο $p \neq u$, τότε $P\ (u.consensus = 1) = P\ (u.consensus = 0) = \frac{1}{2}$.

Απόδειξη. Η τιμή της συμφωνίας εξαρτάται από το $input\_sum$. Από τον αλγόριθμο $Asynchronous\_non\_uni\_do\_consensus\ _i^{non\_uni}(input, id)$ έχουμε μετά τη γραμμή 77:

$$u.input\_sum\ =\ InputT\ (u)\ +\ InputL\ (u)\ +\ Trip\ (u.leader, u).input$$

Αφού δεν έχουμε ομοιόμορφη κατανομή στις εισόδους των πρακτόρων, υποθέτουμε ότι η πιθανότητα να έχουμε το 0 σαν είσοδο ισούται με x και η πιθανότητα να πάρουμε το 1 σαν είσοδο ισούται με $1 - x$.

Περίπτωση 1: Εάν η πιθανότητα να πάρουμε το 0 σαν είσοδο (x) είναι μεγαλύτερη από την πιθανότητα να πάρουμε το 1 σαν είσοδο $(1 - x)$:
$P(u.input = 0, \forall u \in Ring) > P(u.input = 1, \forall u \in Ring)$.

Περίπτωση 2**:** Εάν η πιθανότητα να πάρουμε το 0 σαν είσοδο (x) είναι μικρότερη από την πιθανότητα να πάρουμε το 1 σαν είσοδο $(1 - x)$:
$P(u.input = 0, \forall u \in Ring) < P(u.input = 1, \forall u \in Ring)$.

Περίπτωση 3: Εάν η πιθανότητα να πάρουμε το 0 σαν είσοδο (x) ισούται με την πιθανότητα να πάρουμε το 1 σαν είσοδο $(1 - x)$, τη οποία περίπτωση έχουμε ήδη ελέγξει και αποδείξει.

Όπως έχουμε αναφέρει πιο πάνω, η τιμή της συμφωνίας εξαρτάται από το $input\_sum$, το οποίο με τη σειρά του εξαρτάται από το $InputT(u)$. Επομένως, θα ελέγξουμε για τις περιπτώσεις 1 και 2 τις τιμές που παίρνει το $InputT(u)$.

- $\underline{InputT(u)}$ στη Περίπτωση 1:

  Σε αυτή την περίπτωση, η πιθανότητα να πάρουμε το 0 σαν είσοδο είναι μεγαλύτερη από την πιθανότητα να πάρουμε το 1 σαν είσοδο.

  - Ο αριθμός των πρακτόρων μας n είναι ένας ζυγός αριθμός > 4, έστω $n = 8$.

    Έστω οι κόμβοι $< v_1, v_2, \dots, v_8 >$.

    Οι πιθανοί συνδυασμοί εισόδων είναι οι πιο κάτω:

    (α) $< 0,0,0,0,0,0,0,0 >$: Άθροισμα=0⇒ζυγός αριθμός⇒$u.consensus = 0$

    (β) $< 0,0,0,0,0,0,0,1 >$: Άθροισμα=1⇒μονός αριθμός⇒$u.consensus = 1$

    (γ) $< 0,0,0,0,0,0,1,1 >$: Άθροισμα=2⇒ζυγός αριθμός⇒$u.consensus = 0$

    (δ) $< 0,0,0,0,0,1,1,1 >$: Άθροισμα=3⇒μονός αριθμός⇒$u.consensus = 1$

- $\underline{InputT(u)}$ στη Περίπτωση 2:

  Σε αυτή την περίπτωση, η πιθανότητα να πάρουμε το 1 σαν είσοδο είναι μεγαλύτερη από την πιθανότητα να πάρουμε το 0 σαν είσοδο.

  - Ο αριθμός των πρακτόρων μας n είναι ένας ζυγός αριθμός > 4, έστω $n = 8$.

    Έστω οι κόμβοι $< v_1, v_2, \dots, v_8 >$.

    Οι πιθανοί συνδυασμοί εισόδων είναι οι πιο κάτω:

    (α) $< 1,1,1,1,1,1,1,1 >$: Άθροισμα=8⇒ζυγός αριθμός⇒$u.consensus = 0$

    (β) $< 1,1,1,1,1,1,1,0 >$: Άθροισμα=7⇒μονός αριθμός⇒$u.consensus = 1$

    (γ) $< 1,1,1,1,1,1,0,0 >$: Άθροισμα=6⇒ζυγός αριθμός⇒$u.consensus = 0$

    (δ) $< 1,1,1,1,1,0,0,0 >$: Άθροισμα=5⇒μονός αριθμός⇒$u.consensus = 1$

Σύμφωνα με την ανάλυση των πιο πάνω περιπτώσεων, παρατηρούμε ότι η πιθανότητα το $InputT(u)$ να είναι μονός αριθμός ισούται με την πιθανότητα να είναι ζυγός αριθμός ($P = \frac{1}{2}$).

Επομένως, και το $input\_sum$ έχει τις ίδιες πιθανότητες ανεξάρτητα από το $InputL(u) + Trip(u.leader, u).input$.

Απόδειξη Περίπτωσης 1 για n = ζυγός αριθμός με επαγωγή στο n.

1. Βάση Επαγωγής: Έστω $n = 8$. Οι συνδυασμοί των εισόδων ανάλογα με τις πιθανότητες κατανομής των εισόδων σε 0 ή 1είναι οι εξής:
   - $< 0,0,0,0,0,0,0,0 >$
   - $< 0,0,0,0,0,0,0,1 >$
   - $< 0,0,0,0,0,0,1,1 >$
   - $< 0,0,0,0,0,1,1,1 >$
   - $< 1,1,1,1,1,1,1,1 >$
   - $< 1,1,1,1,1,1,1,0 >$
   - $< 1,1,1,1,1,1,0,0 >$
   - $< 1,1,1,1,1,0,0,0 >$

   $\Rightarrow P\,[InputT(u) = \mu ονός] = P\,[InputT(u) = \zeta υγός] = \frac{1}{2}.$

   $\Rightarrow P\,[input\_sum = \mu ονός] = P\,[input\_sum = \zeta υγός] = \frac{1}{2}.$

   $\Rightarrow P\,[u.consensus = 1] = P\,[u.consensus = 0] = \frac{1}{2}.$

2. Επαγωγική υπόθεση: Έστω ότι η υπόθεση μας ισχύει για $n = k$. Θα αποδείξουμε ότι ισχύει και για $n = k + 8$.

3. Επαγωγικό βήμα: Σύμφωνα με την επαγωγική υπόθεση, ξέρουμε ότι:

   $P\,(u.consensus = 1) = P(u.consensus = 0) = \frac{1}{2}$ για $n = k$. Σύμφωνα με τη βάση της επαγωγής, αυτό ισχύει και για $n = 8$.

   Επομένως, η υπόθεση μας είναι σωστή για $n = k + 8$.

Απόδειξη Περίπτωσης 2 για n = ζυγός αριθμός με επαγωγή στο n.

Αποδεικνύεται όπως και στη Περίπτωση 1.

Το Λήμμα έχει αποδειχθεί για μη ομοιόμορφη κατανομή στις εισόδους. ■

Χρησιμοποιώντας το πιο κάτω λήμμα, θέλουμε να δείξουμε ότι εάν εκλεγεί κάποιος πράκτορας u ως $leader$, ο οποίος ανήκει στο συνασπισμό C, τότε ο συνασπισμός C έχει τον πλήρη έλεγχο πάνω στην απόφαση του u.

Λήμμα 3.4 Έστω $cheaters\ C$ και $u \notin C$, όπου $u.input\_interface \in C$. Εάν ο $leader$ που έχει εκλεγεί από τον u είναι ο u, τότε ο C έχει τον πλήρη έλεγχο πάνω στην απόφαση του u.

Απόδειξη. Αφού $u.interface \in C$, σημαίνει ότι ο u είναι $cheater$ και $InputT(u) = u.input$.
Μετά τη γραμμή 77 του αλγορίθμου:

$$u.input\_sum = u.input + InputL(u) + Trip(u.leader, u).input$$
$$= 2u.input + InputL(u)$$

Επομένως, το αποτέλεσμα συμφωνίας του αλγορίθμου είναι:
$u.consensus = u.input\_sum \bmod 2 = (2u.input + InputL(u)) mod 2 =$
$InputL(u) mod 2$, που σημαίνει ότι η απόφαση του u εξαρτάται μόνο από το $InputL(u)$, δηλαδή μόνο από τις τιμές που έχουν σταλεί στον u από τον $u.input\_interface$, το οποίο ανήκει στο C, επομένως το C έχει τον πλήρη έλεγχο πάνω στην απόφαση του u. ■

Χρησιμοποιώντας το πιο κάτω λήμμα, θέλουμε να δείξουμε ότι εάν οι πράκτορες που θα αποτελέσουν την Ισορροπία $(n-2)$ ανήκουν στο C, και οι άλλοι δύο πράκτορες που απομένουν δεν είναι γείτονες στο δακτύλιο, τότε σε αυτή τη περίπτωση το C δεν θα έχει κανένα όφελος να κλέψει.

Λήμμα 3.5 Έστω $cheaters\ C$ με $|C| = n - 2$. Επίσης, έστω $u_1, u_2 \notin C$ δεν είναι γείτονες στο δακτύλιο. Σε αυτή την περίπτωση το C δεν έχει λόγο να κλέψει.

Απόδειξη. Έστω, χωρίς απώλεια της γενικότητας, ότι το C προτιμά η συμφωνία να είναι 1. Για ευκολία, θα μοιράσουμε την ανάλυση μας στις ακόλουθες περιπτώσεις:

- Περίπτωση 1: Το u_1 και u_2 έχουν λάβει διαφορετικές my_rand_{list} ή id τιμές κατά τη διάρκεια εκτέλεσης του αλγορίθμου. Σε αυτή τη περίπτωση μπορούμε στην ουσία να θεωρήσουμε ότι τα u_1 και u_2 ήταν σε διαφορετικούς δακτύλιους, επομένως οι αποφάσεις τους είναι ανεξάρτητες.

Από το Λήμμα 3.2, η πιθανότητα ο $leader$ που εκλέγηκε από τον u_1 να είναι ο ίδιος ο u_1 είναι $\frac{1}{n}$.

$\Rightarrow P\ (u_1.leader = u_1) = \frac{1}{n}$.

$\Rightarrow P\ (u_1.leader \neq u_1)\ = 1 - P\ (u_1.leader = u_1) = 1 - \frac{1}{n} = \frac{(n-1)}{n}$.

$\Rightarrow$Η πιθανότητα ο $leader$ που εκλέγηκε από τον u_1 να μην είναι ο u_1 είναι $\frac{(n-1)}{n}$.

$\Rightarrow P\ (u_1.leader \neq u_1)\ = \frac{(n-1)}{n}$.

(1) $P\ (u_1.leader = u_1) = \frac{1}{n}$

(2) $P\ (u_1.leader \neq u_1) = \frac{(n-1)}{n}$

Από το Λήμμα 3.3 μπορούμε να εξαγάγουμε ότι¨

(3) $P\ (u_1.consensus = 1 | u_1.leader \neq u_1) = \frac{1}{2}$

Από το Λήμμα 3.4 ξέρουμε ότι δεδομένου ότι ο $leader$ που εκλέγηκε από τον u είναι ο ίδιος ο u, αυτό σημαίνει ότι το C έχει τον πλήρη έλεγχο πάνω στην απόφαση του u και αφού για την απόδειξη αυτού του λήμματος (3.5) υποθέσαμε ότι το C προτιμά οι πράκτορες να συμφωνήσουν στο 1, τότε στην τελική σίγουρα το $u_1.consensous$ θα ισούται με 1.

(4) $P\ (u_1.consensus = 1 | u_1.leader = u1) = 1$

Συνδυάζοντας τα πιο πάνω (1), (2), (3) και (4):

Από (1) και (4):

(α) $P\ (u_1.consensus = 1)$ =

$$P\ (u_1.leader = u_1).P\ (u_1.consensus = 1 | u_1.leader = u_1)\ = \frac{1}{n}.1 = \frac{1}{n}$$

Από (2) και (3):

(β) $P\ (u_1.consensus = 1)$ =

$$P\ (u_1.leader \neq u_1).P\ (u1.consensus = 1 | u1.leader \neq u1)\ = \frac{(n-1)}{n}.\frac{1}{2}$$

$$= \frac{(n-1)}{2n}$$

$\Rightarrow$Από (α) και (β):

(5) $P\ (u_1.consensus = 1) = \frac{1}{n} + \frac{(n-1)}{2n}$

Το ίδιο αποτέλεσμα ισχύει και για το u_2.

Όπως έχουμε εξηγήσει και πιο πάνω, σε αυτή την περίπτωση το $u_1.consensus$ είναι ανεξάρτητο από το $u_2.consensus$, επομένως, η πιθανότητα για τη συμφωνία σε 1 εξαρτάται το (4):

(6) $P(u_1.consensus = 1 \wedge u_2.consensus = 1)$

$=\left(\frac{1}{n}+\frac{n-1}{2n}\right)^2 = \left(\frac{2+n-1}{2n}\right)^2 = \left(\frac{n+1}{2n}\right)^2 = \frac{(n+1)^2}{4n^2} = \frac{1}{4}\left[\frac{(n+1)^2}{n^2}\right] = \frac{1}{4}\left(\frac{n+1}{n}\right)^2 = \frac{1}{4}\left(1+\frac{1}{n}\right)^2 \leq_{1*} \frac{1}{4}\left(1+\frac{1}{4}\right)^2 = \frac{25}{64} \sim 0.39 < \frac{1}{2}.$

1*: Το $n \geq 4$, καθώς υπάρχουν 2 $non-cheaters$ $(u_1, u_2 \notin C)$ και ένας μη μηδενικός ζυγός αριθμός των $cheaters$.

Επομένως, σύμφωνα με τα πιο πάνω, το C δεν έχει κίνητρο να κλέψει στην Περίπτωση 1.

- <u>Περίπτωση 2</u>: Οι τιμές my_rand_{list} και id που έχουν λάβει τα u_1 και u_2 κατά τη διάρκεια του αλγορίθμου είναι οι ίδιες, αλλά οι τιμές εισόδου είναι διαφορετικές. Σε αυτή τη περίπτωση καθώς το άθροισμα των στοιχείων της λίστας my_rand_{list} είναι το ίδιο και για το u_1 και για το u_2, ο $leader$ που εκλέγεται από τον u_1 είναι ο ίδιος που εκλέγεται από τον u_2.

  Από το Λήμμα 3.2:

  $\rightarrow P(leader = u_1) = \frac{1}{n}$

  $\rightarrow P(leader = u_2) = \frac{1}{n}$

  $\rightarrow P(leader \neq u_1, u_2) = 1 - P(leader = u_1) - P(leader = u_2)$

  $= 1 - \frac{1}{n} - \frac{1}{n} = 1 - \frac{2}{n} = \frac{n-2}{n}$

  (7) $P(leader \neq u_1, u_2) = \frac{n-2}{2}, P(leader = u_1) = \frac{1}{n}, P(leader = u_2) = \frac{1}{n}$

  Από το Λήμμα 3.3 έχουμε:

  (8) $P(u_1.consensus = u_2.consensus = 1 | leader \neq u_1, u_2) = \frac{1}{2} * \frac{1}{2} = \frac{1}{4}$

  (9) $P(u_1.consensus = u_2.consensus = 1 | leader = u_1) = 1 -$

  $P(u_1.consensus = u_2.consensus = 1 | leader \neq u_1) = 1 - \frac{1}{2} = \frac{1}{2}$

(10) $P(u_1.consensus = u_2.consensus = 1|leader = u_2) = 1 - P(u_1.consensus = u_2.consensus = 1|leader \neq u_2)$

$= 1 - \frac{1}{2} = \frac{1}{2}$

Από τα (7) - (10) μπορούμε να συμπεράνουμε ότι:

(11) $P(u_1.consensus = 1\ and\ u_2.consensus = 1) = \frac{1}{4}\frac{(n-2)}{n} + \frac{1}{2} * \frac{1}{n} + \frac{1}{2} * \frac{1}{n} = \frac{1}{4}\left(1 - \frac{2}{n}\right) + 2 * \frac{1}{2n} = \frac{1}{4} - \frac{2}{4n} + 2 * \frac{1}{2n} = \frac{1}{4} - \frac{1}{2n} + 2 * \frac{1}{2n} = \frac{1}{4} + \frac{1}{2n} \leq_{1*} \frac{1}{4} + \frac{1}{8} = \frac{3}{8} \sim 0.375 < \frac{1}{2}$

Επομένως, σύμφωνα με τα πιο πάνω το C δεν έχει κανένα κίνητρο να κλέψει ούτε και στη Περίπτωση 2.

- Περίπτωση 3: Οι κόμβοι u_1 και u_2 λαμβάνουν ακριβώς τα ίδια $triplets$ $< id, my\_rand_{list}, input >$ κατά τη διάρκεια εκτέλεσης του αλγορίθμου. Σε αυτή την περίπτωση, τουλάχιστον ένας από τους πράκτορες δεν θα επιλέξει τον εαυτό του ως $leader$. Από το Λήμμα 3.3 η πιθανότητα αυτού του κόμβου να αποφασίσει το 1 είναι $\frac{1}{2}$. Επομένως, το C δεν έχει κανένα κίνητρο να κλέψει στη Περίπτωση 3.

Σύμφωνα με τις περιπτώσεις 1-3, το Λήμμα 3.5 έχει αποδειχθεί. ■

Χρησιμοποιώντας το πιο κάτω λήμμα, θέλουμε να δείξουμε ότι εάν έχουμε ένα συνασπισμό από $cheaters\ C$ και δύο πράκτορες που δεν ανήκουν στο συνασπισμό αυτό και έχουν την ίδια διεπαφή εισόδου, τότε η πιθανότητα η απόφαση του ενός από τους δύο πράκτορες να ισούται με 0 ισούται με την πιθανότητα η απόφαση του να ισούται με 1.

Λήμμα 3.6 Έστω $cheaters\ \ C$ και δύο κόμβοι u_1 και $u_2 \notin C$ έτσι ώστε u_1=$u_2.input\_interface$. Τότε $P(u_2.consensus = 1) = P\ (u_2.consensus = 0) = \frac{1}{2}$.

Απόδειξη. Ελέγχουμε τη περίπτωση του u_2 για να φτάσουμε στην απόδειξη.

Από το Λήμμα 3.3 ξέρουμε ότι εάν το u_2 επιλέξει ένα $leader$ που δεν είναι ο εαυτός του, δηλαδή ένα $leader\ p \neq u_2$, τότε:

$P\ (u_2.consensus = 1)\ =\ P\ (u_2.consensus = 0)\ = \frac{1}{2}$.

Εάν το u_2 επιλέξει τον εαυτό του σαν $leader$, τότε:

$u_2.input\_sum =\ 2u_2.input\ +\ u\_1.input\ +\ InputL\ (u)$.

Αφού $P\ (u_1.input = 1) = P\ (u_2.input = 1) = \frac{1}{2}$, σύμφωνα με το λήμμα 3.3 έχουμε

$P\ (u_2.consensus = 1) = P\ (u_2.consensus = 0) = \frac{1}{2}$. ■

Χρησιμοποιώντας το πιο κάτω λήμμα , θέλουμε να δείξουμε ότι εάν οι πράκτορες που θα αποτελέσουν την Ισορροπία $(n-2)$ ανήκουν στο C και οι άλλοι δύο πράκτορες που απομένουν δεν ανήκουν σε αυτό τον συνασπισμό και έχουν την ίδια διεπαφή εισόδου, τότε σε αυτή την περίπτωση το C δεν θα έχει κανένα όφελος να κλέψει αποκλίνοντας από τον αλγόριθμο.

Λήμμα 3.7 Έστω $cheaters\ C$ με $|C| = n-2$ και δύο κόμβοι u_1 και $u_2 \notin C$ έτσι ώστε u_1=$u_2.input\_interface$. Τότε δεν υπάρχει κανένα κίνητρο για το C να αποκλίνει από τον αλγόριθμο.

Απόδειξη. Είναι επακόλουθη του λήμματος 3.6. ■

Τα πιο πάνω λήμματα οδηγούν στο πιο κάτω κύριο θεώρημα, το Θεώρημα 3:

Θεώρημα 3 Ο αλγόριθμος $Asynchronous\_non\_uni\_do\_consensus_i^{non\_uni}$ είναι ένας $(n-2)$ - Ισχυρός Αλγόριθμος Ισορροπίας με μη ομοιόμορφη κατανομή στις εισόδους των πρακτόρων:

Απόδειξη. Από το Λήμμα 3.1 ξέρουμε ότι ο αλγόριθμος διατηρεί τόσο την εγκυρότητα όσο και την ύπαρξη συμφωνίας όταν κάθε κόμβος του δακτυλίου τον ακολουθεί και επίσης

ξέρουμε ότι $P\ (ring\ decides\ 0) = P\ (ring\ decides\ 1) = \frac{1}{2}$. Από τα Λήμματα 3.5 και 3.7 ξέρουμε ότι ανεξάρτητα από τη θέση των 2 $non - cheaters$ ($(n-2)$ - Ισχυρή - Ισορροπία) είτε είναι γείτονες είτε όχι, δεν υπάρχει κανένα κίνητρο για τους cheaters να κλέψουν. Επομένως, ο αλγόριθμος που έχουμε προτείνει παρέχει όντως μια $(n-2)$ - Ισχυρή - Ισορροπία. ■

Κεφάλαιο 6

Επίλογος

6.1 Σύνοψη 117
6.2 Συμπεράσματα 118
6.3 Μελλοντική Εργασία 119

6.1 Σύνοψη

Σε αυτή την εργασία παρουσιάσαμε τα δομικά στοιχεία (blocks) που είναι εξαιρετικά χρήσιμα για πολλά προβλήματα κατανεμημένου υπολογισμού. Επίσης, παρουσιάσαμε ένα μοντέλο σύμφωνα με τη θεωρία των παιγνίων για ένα κατανεμημένο σύστημα πρακτόρων. Τα δομικά μας στοιχεία βασίζονται σε αυτό το μοντέλο θεωρίας παιγνίων κατανεμημένου υπολογισμού, έτσι ώστε τα πρωτόκολλα των στοιχείων αυτών να παρέχουν δηλαδή μια Ισχυρή Ισορροπία σε σχέση με τους ορθολογικούς πράκτορες και συγκεκριμένα τους ορθολογικούς πράκτορες που ανήκουν σε συνασπισμούς.

Έχουμε χρησιμοποιήσει τα πρωτόκολλα των δομικών αυτών στοιχείων έτσι ώστε να καταλήξουμε σε τρείς αλγορίθμους που να παρέχουν μια $(n-2)$ - Ισχυρή - Ισορροπία στο πρόβλημα της κατανεμημένης συμφωνίας έχοντας να αντιμετωπίσουμε εγωιστικούς πράκτορες με την προϋπόθεση ότι ο αριθμός των πρακτόρων n είναι ένας ζυγός αριθμός. Ο πρώτος αλγόριθμος, που παρουσιάζεται στο Κεφάλαιο 4, παρέχει μια $(n-2)$ - Ισχυρή - Ισορροπία σε ένα σύγχρονο δακτύλιο με την προϋπόθεση ότι οι εισόδοι που παρέχονται στους πράκτορες μας δεν ακολουθούν την ομοιόμορφη κατανομή. Ο δεύτερος αλγόριθμος, που παρουσιάζεται στο Κεφάλαιο 5.1, παρέχει μια $(n-2)$ - Ισχυρή - Ισορροπία σε ένα ασύγχρονο όμως αυτή τη φορά δακτύλιο με την προϋπόθεση

όμως ότι οι εισόδοι που παρέχονται στους πράκτορες μας ακολουθούν την ομοιόμορφη κατανομή. Ο τρίτος αλγόριθμος, που παρουσιάζεται στο Κεφάλαιο 5.3, παρέχει μια $(n-2)$ - Ισχυρή - Ισορροπία σε ένα ασύγχρονο δακτύλιο με την προϋπόθεση ότι οι εισόδοι που παρέχονται στους πράκτορες μας δεν ακολουθούν την ομοιόμορφη κατανομή.

6.2 Συμπεράσματα

Και οι τρείς αλγόριθμοι που έχουμε κατασκευάσει αντιμετωπίζουν το πρόβλημα των συνασπισμών C, έτσι ώστε οι πράκτορες που τείνουν να αποκλίνουν από ένα αλγόριθμο να μην έχουν κάποιο κίνητρο να το κάνουν, με αποτέλεσμα η τελική συμφωνία να είναι δίκαιη για όλους και να μην υπάρχουν πράκτορες που να επωφελούνται της τελικής τιμής συμφωνίας περισσότερο από άλλους. Ωστόσο, ο δεύτερος και ο τρίτος αλγόριθμος που έχουμε παρουσιάσει, οι οποίοι λειτουργούν για ένα ασύγχρονο περιβάλλον επικοινωνίας των πρακτόρων είναι πιο ευέλικτοι από τον πρώτο. Επειδή πρόκειται για ένα ασύγχρονο δακτύλιο, οι πράκτορες έχουν περισσότερη ελευθερία στις κινήσεις τους, καθώς δεν χρειάζεται να αναστέλλεται η λειτουργία των πρακτόρων οι οποίοι περιμένουν κάποιο μήνυμα. Τώρα, το αποτέλεσμα μπορεί να εξαρτάται όχι μόνο από το ποιοι πράκτορες παίρνουν ένα αρχικό μήνυμα, αλλά και από την σειρά με την οποία προγραμματίζονται οι πράκτορες και από τους χρόνους παράδοσης των μηνυμάτων.

Επομένως, και οι τρείς αλγόριθμοι παρέχουν μια $(n-2)$ - Ισχυρή - Ισορροπία και δίνουν μια εξαιρετική και βολική λύση στο πρόβλημα της κατανεμημένης συμφωνίας, με τη μόνη διαφορά ότι ο πρώτος αλγόριθμος είναι πιο αυστηρός και περιοριστικός όσο αφορά τις στρατηγικές ανταλλαγής μηνυμάτων των πρακτόρων στο δακτυλίδι. Παρόλα αυτά, η πολυπλοκότητα του ως προς τον αριθμό των γύρων που χρειάζεται να εκτελεστεί ο αλγόριθμος για να καταλήξουμε σε συμφωνία, είναι πιο μικρή από την πολυπλοκότητα του δεύτερου και τρίτου αλγορίθμου καθώς η μεγαλύτερη ανεξαρτησία των πρακτόρων στις κινήσεις τους προκαλεί περισσότερους συνδυασμούς υποπροβλημάτων προς επίλυση.

6.3 Μελλοντική Εργασία

Η προγραμματιστική υλοποίηση των αλγορίθμων που έχουμε αναφέρει για την πειραματική τους αξιολόγηση μέσα από ένα πρόγραμμα προσομοίωσης θα ήταν μια αρκετά ενδιαφέρουσα μελλοντική δουλειά ανάπτυξης της παρούσας διπλωματικής εργασίας για να δούμε στην πράξη πως λειτουργεί το πρόβλημα της Κατανεμημένης Συμφωνίας με Εγωιστικούς πράκτορες.

Επίσης, μια άλλη ενδιαφέρουσα μελλοντική δουλειά θα ήταν η επίλυση του προβλήματος κατανεμημένου υπολογισμού σε ένα μοντέλο όπου ένας πράκτορας έχει μια προτίμηση στο αποτέλεσμα, αλλά δεν ικανοποιείται η προτίμηση λύσης που έχουμε αναφέρει στον Ορισμό 3.2.1 του Κεφαλαίου 6 και έτσι ο πράκτορας προτιμά να αποτύχει το πρωτόκολλο εάν δεν επωφεληθεί αλλιώς.

Βιβλιογραφία

[1] Y. Afek, Y. Ginzberg, S. Landau Feibish, and M. Sulamy. “Distributed Computing Building Blocks for Rational Agents”. Available: http://mcg.cs.tau.ac.il/papers/podc091f-afek.pdf.

[2] I. Abraham, D. Dolev, and J. Y. Halpern. “Distributed protocols for leader election: A game-theoretic perspective”. In DISC, pages 61–75, 2013.

[3] Davies, and Wakerly. "Synchronization and Matching in Redundant Systems." IEEE Transactions on Computers C-27.6 (1978): 531-39

[4] Pease, M., R. Shostak, and L. Lamport. "Reaching Agreement in the Presence of Faults." Journal of the ACM 27.2 (1980): 228-34.

[5] Dolev, Danny, and H. Raymond Strong. "Polynomial algorithms for multiple processor agreement." Proceedings of the fourteenth annual ACM symposium on Theory of computing - STOC 82 (1982).

[6] Fischer, Michael J. "The consensus problem in unreliable distributed systems (a brief survey)." Foundations of Computation Theory Lecture Notes in Computer Science (1983): 127-40.

[7] Fischer, Michael J., Nancy A. Lynch, and Michael S. Paterson. "Impossibility of distributed consensus with one faulty process." Proceedings of the 2nd ACM SIGACT-SIGMOD symposium on Principles of database systems - PODS 83 (1983).

[8] I. Abraham, L. Alvisi, and J. Y. Halpern. Distributed computing meets game theory: combining insights from two fields. SIGACT News, 42(2):69–76, 2011.

[9] I. Abraham, D. Dolev, R. Gonen, and J. Y. Halpern. Distributed computing meets game theory: robust mechanisms for rational secret sharing and multiparty computation. In PODC, pages 53–62, 2006.

[10] A. S. Aiyer, L. Alvisi, A. Clement, M. Dahlin, J.-P. Martin, and C. Porth. Bar fault tolerance for cooperative services. In SOSP, pages 45–58, 2005.

[11] J. Y. Halpern and V. Teague. Rational secret sharing and multiparty computation: extended abstract. In STOC, pages 623–632, 2004.

[12] Impossibility of $n-1$-strong-equllibrium for Distributed Consensus with Rational Agents. Available: https://arxiv.org/ftp/arxiv/papers/1708/1708.02543.pdf.

[13] http//:eclass.uop.gr/modules/document/file.php/TST125/Ο1%202ΙΑΛΕΞΕΙΣ/ΘΠ%2001%20Εισαγωγή.pdf

[14] http//:www.uni-bonn.de/~sgeorgan/Paihnidia.ppt

[15] http//:el.wikipedia.org/wiki/θεωρία παιγνίων

[16] en.wikipedia.org/wiki/Game_theory

[17] Mansour Yishay (2003), Computational Learning Theory, Lecture 1:March 2

[18] http//:www.gametheory.net

[19] Mansour Yishay(2003), Computational Learning Theory, Lecture 1:March 2

[20] http://arielrubinstein.tau.ac.il/99/gt100.html#g1

[21] Hargreaves Heap P.Shaun and Varoufakis Yanis (1995), Game theory: A critical Introduction, London, Routledge

[22] Rasmusen Eric(2001), Games and Information:An introduction to Game Theory,fourth edition

[23] Siegfried Tom (2006), A beautiful Math: John Nash, game theory and the Modern quest for a code of nature, Washington,D.C.,Joseph Henry Press

[24] Fisher Len Ph.D.(2008), Rock, Paper, Scissors: Game Theory in every day life, New York, Basic Books

[25] http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.471.5463&rep=rep1&type=pdf

[26] https://pdfs.semanticscholar.org/3075/7ce8305060ba6111f06d446bfe299b4f8ca0.pdf

[27] http://index.lib.teithe.gr:8080/bitstream/handle/10184/4331/Tzivanakis_Christos.pdf?sequence=8

[28] https://www.isi.edu/~blythe/cs541/Readings/ACAI01.pdf

[29] http://www.eng.ucy.ac.cy/mandreou/Courses/ECE417/introduction.pdf

[30] http://www.cdam.lse.ac.uk/Reports/Files/cdam-2001-09.pdf

[31] http://krishikosh.egranth.ac.in/bitstream/1/2033620/1/IVRI%203205.pdf

[32] file:///C:/Users/User/Downloads/An_Introduction_to_Game_Theory_-_Eric_Rasmusen.pdf

[33] http://homepage.univie.ac.at/Mariya.Teteryatnikova/WS2011/FT.pd

Παράρτημα Α

Algorithm 1 (Afek *et al*. [1])**:**

```
do_consensus (input, id):
1       my_rand:= rand ()
2       ids_array.append (<id, input>)
3       rand_sum:= my_rand
4       input_sum:= input
5       Send (<input, id, my_rand>, out_interface)
6       for i = 1,…, n-1:
7           tmp: = Recv (in_interface)
8           if tmp = null or not (tmp.input ∈ {0, 1}):
9               REPORT CHEATER
10          ids_array.append (<tmp.id, tmp.input>)
11          rand_sum += tmp.rand
12          input_sum += tmp.input
13          Send (tmp, out_interface)
14      end for
15      tmp: = Recv (in_interface)
16      if tmp.input ≠ input or tmp.id ≠ id or tmp.rand
        ≠ my_rand:
17          REPORT CHEATER
18      sort ids_array by id
19      if ∃id ∈ ids_array with ids_array.count(id) ≠ 1:
20          REPORT CHEATER
21      leader: = ids_array [rand_sum % n]
22      input_sum += leader.input
23      if input_sum % 2 == 1:
24          return 1
25      else:
26          return 0
end do_consensus
```