Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Ηλεκτρολόγων Μηχανικών
και Μηχανικών Υπολογιστών
Τομέας Τεχνολογίας Πληροφορικής και Υπολογιστών

Ταχύτητα Σύγκλισης εναντίον Μνήμης στη Διαμόρφωση Απόψεων

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

ΒΑΣΙΛΙΚΗ ΚΟΝΤΟΥΡΑ

Επιβλέπων : Δημήτριος Φωτάκης
Επικ. Καθηγητής Ε.Μ.Π.

Αθήνα, Νοέμβριος 2017

Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Ηλεκτρολόγων Μηχανικών
και Μηχανικών Υπολογιστών
Τομέας Τεχνολογίας Πληροφορικής και Υπολογιστών

Ταχύτητα Σύγκλισης εναντίον Μνήμης στη Διαμόρφωση Απόψεων

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

ΒΑΣΙΛΙΚΗ ΚΟΝΤΟΥΡΑ

Επιβλέπων : Δημήτριος Φωτάκης
Επικ. Καθηγητής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 10η Νοεμβρίου 2017.

.....	.....	.....
Δημήτριος Φωτάκης Επίκ. Καθηγητής Ε.Μ.Π.	Νικόλαος Σ. Παπασπύρου Αν. Καθηγητής Ε.Μ.Π.	Αριστείδης Παγουρτζής Αν. Καθηγητής Ε.Μ.Π.

Αθήνα, Νοέμβριος 2017

Βασιλική Κοντουρά

Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

Copyright © Βασιλική Κοντουρά, 2017.
Με επιφύλαξη παντός δικαιώματος. All rights reserved.

Απαγορεύεται η αντιγραφή, αποθήκευση και διανομή της παρούσας εργασίας, εξ ολοκλήρου ή τμήματος αυτής, για εμπορικό σκοπό. Επιτρέπεται η ανατύπωση, αποθήκευση και διανομή για σκοπό μη κερδοσκοπικό, εκπαιδευτικής ή ερευνητικής φύσης, υπό την προϋπόθεση να αναφέρεται η πηγή προέλευσης και να διατηρείται το παρόν μήνυμα. Ερωτήματα που αφορούν τη χρήση της εργασίας για κερδοσκοπικό σκοπό πρέπει να απευθύνονται προς τον συγγραφέα.

Οι απόψεις και τα συμπεράσματα που περιέχονται σε αυτό το έγγραφο εκφράζουν τον συγγραφέα και δεν πρέπει να ερμηνευθεί ότι αντιπροσωπεύουν τις επίσημες θέσεις του Εθνικού Μετσόβιου Πολυτεχνείου.

Περίληψη

Αντικείμενο της κυρτής βελτιστοποίησης είναι η ελαχιστοποίηση μίας κυρτής αντικειμενικής συνάρτησης, δεδομένων κάποιων περιορισμών. Συχνά, η επίλυση τέτοιων προβλημάτων στηρίζεται σε επαναληπτικούς αλγορίθμους, ενώ χρήσιμη είναι και η μελέτη κατανεμημένων παραλλαγών τους, που μπορούν να εφαρμοστούν για την επίλυση προβλημάτων βελτιστοποίησης σε δίκτυα.

Ωστόσο, η επίλυση προβλημάτων σε δίκτυα στην πράξη περιέχει διάφορους περιορισμούς και απαιτεί ιδιαίτερο χειρισμό. Δύο από τους βασικότερους τέτοιους περιορισμούς αποτελούν η ποσότητα μνήμης που είναι διαθέσιμη στους κόμβους ενός δικτύου και ο όγκος της επικοινωνίας που είναι εφικτός μεταξύ αυτών.

Στη διπλωματική αυτή εργασία, επικεντρωνόμαστε σε ένα συγκεκριμένο επαναληπτικό αλγόριθμο βελτιστοποίησης, τη μέθοδο gradient descent, και μελετάμε πειραματικά την επίδραση της περιορισμένης μνήμης και επικοινωνίας σε ένα δίκτυο σε μία παραλλαγή της κατανεμημένης της εκδοχής, μέσω του τρόπου που φυσικά αυτή εφαρμόζεται στο μοντέλο διαμόρφωσης απόψεων των Friedkin και Johnsen.

Για τη διεξαγωγή των πειραμάτων μας, υλοποιήσαμε έναν προσομοιωτή σε Python.

Λέξεις κλειδιά

Διαμόρφωση απόψεων, κυρτή βελτιστοποίηση, gradient descent, το μοντέλο των Friedkin-Johnsen, προσομοίωση, Python.

Abstract

The goal of convex optimization is to minimize a convex function given some constraints. Usually, problems like these are solved by iterative algorithms. Alongside these algorithms, it is useful to study their distributed variations, that can be used to solve optimization problems in networks.

However, in networks, computations like these are more restrictive. Two of the main constraints when dealing with networks are the capacity of the memory that is available in each of their nodes and the amount of communication that can be performed between them.

In this diploma dissertation, we focus on a specific iterative optimization algorithm, the gradient descent method, and we study experimentally the effect that limited available memory and communication in a network have on a distributed variation of it, by studying one of its applications which appears naturally in the model that Friedkin and Johnsen proposed as part of their research in opinion dynamics.

For our experiments, we implemented a simulator in Python.

Key words

Opinion dynamics, convex optimization, gradient descent, the Friedkin-Johnsen model, simulation, Python.

Ευχαριστίες

Ευχαριστώ θερμά τον επιβλέποντα καθηγητή μου, κ. Δημήτριο Φωτάκη, για την πολύχρονη καθοδήγηση, υποστήριξη και, κυρίως, εμπισοσύνη κατά τη διάρκεια τόσο αυτής της διατριβής, όσο και των σπουδών μου γενικότερα. Ευχαριστώ, επίσης, θερμά τα μέλη της συμβουλευτικής μου επιτροπής, κ.κ. Νικόλαο Σ. Παπασπύρου και Αριστείδη Παγουρτζή, καθώς και τον κ. Ευστάθιο Ζάχο, για όλη τη βοήθεια και στήριξη που μου παρείχαν ποικιλοτρόπως καθόλη τη διάρκεια των σπουδών μου, συνειδητά ή μη. Θέλω να ευχαριστήσω ακόμα τους λίγους σημαντικούς φίλους μου που αποτελέσανε ο καθένας με το δικό του τρόπο αναπόσπαστα κομμάτια των σπουδών που ολοκληρώνονται με αυτή τη διατριβή. Τέλος, θα ήθελα να ευχαριστήσω ιδιαίτερα τους γονείς μου και τη μικρή μου αδερφή για την όμορφη οικογένεια στην οποία μεγάλωσα και συνεχίζω να μεγαλώνω.

Βασιλική Κοντουρά,

Αθήνα, 10η Νοεμβρίου 2017

Περιεχόμενα

Περίληψη 5

Abstract 7

Ευχαριστίες 9

Περιεχόμενα 11

1. Εισαγωγή 13

2. Κυρτή βελτιστοποίηση 15
2.1 Προβλήματα μαθηματικής βελτιστοποίησης 15
2.2 Κατηγοριοποίηση προβλημάτων μαθηματικής βελτιστοποίησης 15
2.3 Προβλήματα κυρτής βελτιστοποίησης 16

3. Η μέθοδος gradient descent 17
3.1 Μέθοδοι descent 17
3.1.1 Περιγραφή του προβλήματος 17
3.1.2 Επαναληπτικοί αλγόριθμοι 17
3.1.3 Μέθοδοι descent 17
3.2 Η μέθοδος gradient descent 19
3.3 Ανάλυση σύγκλισης για τη μέθοδο gradient descent 19
3.3.1 Αρχικό σημείο 19
3.3.2 Αυστηρά κυρτές συναρτήσεις 19
3.3.3 Απόσταση από τη βέλτιστη τιμή 20
3.3.4 Απόσταση από το βέλτιστο διάνυσμα 21
3.4 Στοχαστική gradient descent 21

4. Το μοντέλο των Friedkin-Johnsen 23
4.1 Εισαγωγή 23
4.2 Το μοντέλο των Friedkin-Johnsen 23
4.3 Αναπαράσταση με μορφή πινάκων 24
4.4 Συνάρτηση κόστους 24
4.5 Σύγκλιση 25
4.6 Ισορροπία Nash 26
4.7 Συνάρτηση δυναμικού 26
4.8 Συνάρτηση δυναμικού και ισορροπία Nash 27
4.9 Σύγκλιση στην ισορροπία Nash 27

5. Παραλλαγή του μοντέλου των Friedkin-Johnsen με μνήμη 31
5.1 Gradient descent στο μοντέλο Friedkin-Johnsen 31
5.2 Κατανεμημένη gradient descent 31
5.3 Περιορισμοί της κατανεμημένης gradient descent 32

5.4 Παραλλαγές της κατανεμημένης gradient descent με μνήμη 32
5.5 Παραλλαγές του μοντέλου των Friedkin-Johnsen με μνήμη 33
5.6 Τυπική περιγραφή του μοντέλου των προσομοιώσεων 33

6. Ο Προσομοιωτής 35
6.1 Εισαγωγή 35
6.2 Τα αρχεία εισόδου 35
6.3 Η αρχιτεκτονική του προσομοιωτή 40
6.3.1 Το κυρίως πρόγραμμα 40
6.3.2 Η κλάση `Configuration` 40
6.3.3 Το πακέτο `memory` 41
6.3.4 Το πακέτο `agent` 41
6.3.5 Η κλάση `Scheduler` 42
6.3.6 Η συνάρτηση `computeStableStabe` 42

7. Τα αποτελέσματα των προσομοιώσεων 43
7.1 Εισαγωγή 43
7.2 Μεταβλητού μεγέθους πλήρης γράφος - Πλήρης `exclusive` μνήμη 44
7.3 Σταθερού μεγέθους πλήρης γράφος - Μεταβλητού μεγέθους μνήμη `window` 47
7.4 Σταθερού μεγέθους κανονικός γράφος - Μεταβλητού μεγέθους μνήμη `window` 51
7.5 Συνδυασμός μνήμης `window` και `exclusive` 55
7.6 Συμπεράσματα 59

Βιβλιογραφία 61

Κεφάλαιο 1

Εισαγωγή

Συχνά, καλούμαστε να επιλύσουμε προβλήματα βελτιστοποίησης στα οποία προσπαθούμε να ελαχιστοποιήσουμε την τιμή μίας αντικειμενικής συνάρτησης, δεδομένων κάποιων περιορισμών. Προβλήματα αυτής της μορφής καλούνται προβλήματα μαθηματικής βελτιστοποίησης.

Μία χρήσιμη υποκλάση αυτών των προβλημάτων αποτελούν τα προβλήματα κυρτής βελτιστοποίησης, όπου καλούμαστε να ελαχιστοποιήσουμε μία κυρτή συνάρτηση. Μία συνήθης προσέγγιση για την επίλυση τέτοιων προβλημάτων είναι αυτή της εφαρμογής επαναληπτικών αλγορίθμων.

Η μέθοδος gradient descent είναι ένας τέτοιος επαναληπτικός αλγόριθμος ο οποίος επιλύει προβλήματα κυρτής βελτιστοποίησης ελλείψει περιορισμών, ξεκινώντας από ένα σημείο και κάνοντας βήματα κάθε φορά όπως ορίζει η κλίση της αντικειμενικής συνάρτησης. Στην επαναληπτική σχέση που ορίζει τη μέθοδο gradient descent, ο υπολογισμός κάθε συντεταγμένης σε κάθε βήμα στηρίζεται μόνο σε τιμές του προηγούμενου βήματος και μπορεί να πραγματοποιηθεί ανεξάρτητα από τους υπόλοιπους. Προκύπτει έτσι, φυσικά, μία κατανεμημένη παραλλαγή της μεθόδου, η οποία μπορεί να χρησιμοποιηθεί για την επίλυση προβλημάτων βελτιστοποίησης σε δίκτυα κόμβων, όπου κάθε κόμβος αναλαμβάνει τον υπολογισμό μίας συντεταγμένης της επαναληπτικής διαδικασίας.

Ωστόσο, η επίλυση προβλημάτων σε δίκτυα στην πράξη περιέχει διάφορους περιορισμούς. Δύο βασικοί από αυτούς είναι η ποσότητα μνήμης που είναι διαθέσιμη στους κόμβους του δικτύου και ο όγκος επικοινωνίας που χρειάζεται μεταξύ αυτών. Για δίκτυα με πολλούς κόμβους περιορισμένων δυνατοτήτων και με μικρή δυνατότητα επικοινωνίας, οι απαιτήσεις σε μνήμη και επικοινωνία της κατανεμημένης gradient descent μπορεί να είναι αποτρεπτικές. Μας ενδιαφέρει να μελετήσουμε παραλλαγές της κατανεμημένης gradient descent που μειώνουν τις απαιτήσεις αυτές και συνεχίζουν να επιτυγχάνουν σύγκλιση.

Για τη μελέτη αυτή, στρεφόμαστε σε ένα μοντέλο που εισήγαγαν οι Friedkin και Johnsen στα πλαίσια της μελέτης τους στον τομέα της διαμόρφωσης απόψεων σε κοινωνικά δίκτυα. Στο μοντέλο αυτό, κάθε κόμβος του κοινωνικού δικτύου έχει μία γνώμη αρχικά, την οποία διαμορφώνει καθώς αλληλεπιδρά με τους γείτονές τους, στην προσπάθειά του να ελαχιστοποιήσει το πόσο απέχει από τις γνώμες των γειτόνων του, χωρίς, ωστόσο, να απομακρυνθεί τελείως κι από την αρχική δική του πεποίθηση. Αυτή η προσπάθεια των κόμβων, τους κάνει να επαναδιαμορφώνουν διαρκώς την άποψή τους, όσο οι απόψεις των γειτόνων τους μεταβάλλονται, μέχρι που το δίκτυο φτάνει σε μία κατάσταση ισορροπίας στην οποία οι κόμβοι δεν μπορούν να ελαχιστοποιήσουν άλλο τις αποστάσεις τους. Η επαναληπτική διαδικασία που ακολουθούν οι κόμβοι του κοινωνικού δικτύου, στην πραγματικότητα, αποτελεί μία εφαρμογή της κατανεμημένης gradient descent για μία κατάλληλα ορισμένη συνάρτηση δυναμικού.

Σε αυτή τη διπλωματική εργασία ορίζουμε μία παραλλαγή του μοντέλου των Friedkin και Johnsen, στην οποία οι κόμβοι του κοινωνικού δικτύου έχουν πρόσβαση σε περιορισμένη μνήμη και μειωμένη δυνατότητα επικοινωνίας μεταξύ τους, με σκοπό να μελετήσουμε πειραματικά εάν το κοινωνικό δίκτυο συνεχίζει να επιτυγχάνει σύγκλιση στην κατάσταση ισορροπίας. Αν τα δίκτυα που μελετάμε συνεχίζουν να συγκλίνουν, αποκτούμε κάποιες πρώτες ενδείξεις ότι ίσως μπορούν να ορισθούν παραλλαγές της κατανεμημένης gradient descent που να επιτρέπουν σε μεγάλης κλίμακας δίκτυα με περιορισμένους πόρους να επιλύουν μία χρήσιμη υποκλάση προβλημάτων κυρτής βελτιστοποίησης.

Κεφάλαιο 2

Κυρτή βελτιστοποίηση

2.1 Προβλήματα μαθηματικής βελτιστοποίησης

Συχνά μας ενδιαφέρει να ελαχιστοποιήσουμε την τιμή μίας συνάρτησης πολλών μεταβλητών δεδομένων κάποιων περιορισμών πάνω στις μεταβλητές αυτές. Σε ένα τέτοιο πρόβλημα, η συνάρτηση $f_0 : \mathbb{R}^n \to \mathbb{R}$ που θέλουμε να ελαχιστοποιήσουμε ονομάζεται *αντικειμενική συνάρτηση*, ενώ το διάνυσμα $\vec{x} = (x_1, \ldots, x_n)$ των μεταβλητών της ονομάζεται *μεταβλητή βελτιστοποίησης*. Οι περιορισμοί μπορούν να εκφραστούν μέσω ανισοτικών σχέσεων της μορφής $f_i(\vec{x}) \leq b_i,\ i = 1, \ldots, m$, όπου οι συναρτήσεις $f_i : \mathbb{R}^n \to \mathbb{R},\ i = 1, \ldots, m$ παίρνουν τιμές στο ίδιο πεδίο ορισμού με την αντικειμενική συνάρτηση και περιορίζονται από τις σταθερές $b_1, \ldots, b_m$.

Ένα τέτοιο πρόβλημα περιγράφεται τυπικά ως εξής

$$\begin{array}{ll} \text{minimize} & f_0(\vec{x}) \\ \text{subject to} & f_i(\vec{x}) \leq b_i,\ i = 1, \ldots, m. \end{array}$$

και ονομάζεται *πρόβλημα μαθηματικής βελτιστοποίησης*, ή απλά *πρόβλημα βελτιστοποίησης*.

Για ένα πρόβλημα βελτιστοποίησης λέμε ότι ένα διάνυσμα $\vec{x}^*$ είναι *βέλτιστο*, εάν, ανάμεσα σε όλα τα διανύσματα που ικανοποιούν τους περιορισμούς του προβλήματος, είναι αυτό με τη μικρότερη αντικειμενική τιμή. Δηλαδή για κάθε $\vec{x}$ με $f_i(\vec{x}) \leq b_i,\ i = 1, \ldots, m$, ισχύει $f_0(\vec{x}) \geq f_0(\vec{x}^*)$. Ένα τέτοιο βέλτιστο διάνυσμα $\vec{x}^*$ το λέμε και *λύση* του προβλήματος.

Ο στόχος μας σε ένα τέτοιο πρόβλημα είναι ανάμεσα σε ένα σύνολο διανυσμάτων στο $\mathbb{R}^n$ που αποτελούν τις δυνατές επιλογές μας και που περιορίζονται από τους περιορισμούς του προβλήματος $f_i(\vec{x}) \leq b_i,\ i = 1, \ldots, m$, να πραγματοποιήσουμε την καλύτερη δυνατή επιλογή. Εάν θεωρήσουμε ότι η επιλογή ενός διανύσματος $\vec{x}$ μας επιφέρει κόστος $f_0(\vec{x})$ που ορίζεται από την αντικειμενική συνάρτηση, μία λύση του προβλήματος βελτιστοποίησης αντιστοιχεί σε μία επιλογή ελάχιστου κόστους μεταξύ των επιλογών που ικανοποιούν τους περιορισμούς του προβλήματος. Εναλλακτικά, μπορούμε να θεωρήσουμε ότι η επιλογή ενός διανύσματος $\vec{x}$ μας αποδίδει ώφελος (κέρδος) $-f_0(\vec{x})$ και η λύση του προβληματός αντιστοιχεί σε μία επιλογή μεγίστου κέρδους.

2.2 Κατηγοριοποίηση προβλημάτων μαθηματικής βελτιστοποίησης

Με βάση το είδος της αντικειμενικής συνάρτησης και των περιορισμών τους, μπορούμε να χωρίσουμε τα προβλήματα βελτιστοποίησης σε οικογένειες ή κλάσεις.

Στην περίπτωση που τόσο η αντικειμενική συνάρτηση f_0 όσο και οι συναρτήσεις των περιορισμών $f_1, \ldots, f_m$ είναι γραμμικές, το πρόβλημα βελτιστοποίησης καλείται *γραμμικό πρόγραμμα*. Μία συνάρτηση είναι γραμμική, εάν ικανοποιεί τη σχέση

$$f(\alpha\vec{x} + \beta\vec{y}) = \alpha f(\vec{x}) + \beta f(\vec{y})$$

για κάθε $\vec{x}, \vec{y} \in \mathbb{R}^n$ και κάθε $\alpha, \beta \in \mathbb{R}$.

Ένα πρόβλημα βελτιστοποίησης που δεν είναι γραμμικό, καλείται *μη γραμμικό πρόγραμμα*.

Κατά αναλογία, στην περίπτωση που τόσο η αντικειμενική συνάρτηση f_0 όσο και οι συναρτήσεις των περιορισμών $f_1, \ldots, f_m$ είναι κυρτές, το πρόβλημα βελτιστοποίησης καλείται *κυρτό πρόβλημα βελτιστοποίησης*. Μία συνάρτηση είναι κυρτή, εάν ικανοποιεί τη σχέση

$$f(\alpha\vec{x} + \beta\vec{y}) \leq \alpha f(\vec{x}) + \beta f(\vec{y})$$

για κάθε $\vec{x}, \vec{y} \in \mathbb{R}^n$ και κάθε $\alpha, \beta \in \mathbb{R}$ όπου $\alpha + \beta = 1$, $\alpha \geq 0$, $\beta \geq 0$.

Εφόσον η κυρτότητα είναι πιο γενική ιδιότητα από τη γραμμικότητα, κάθε γραμμικό πρόγραμμα είναι κυρτό πρόβλημα βελτιστοποίησης. Με άλλα λόγια, τα κυρτά προβλήματα βελτιστοποίησης είναι μία γενίκευση του γραμμικού προγραμματισμού.

2.3 Προβλήματα κυρτής βελτιστοποίησης

Συνολικά, ονομάζουμε *πρόβλημα κυρτής βελτιστοποίησης* ένα πρόβλημα της μορφής

$$\begin{array}{ll} \text{minimize} & f_0(\vec{x}) \\ \text{subject to} & f_i(\vec{x}) \leq b_i, \ i = 1, \ldots, m \end{array}$$

όπου τόσο η αντικειμενική συνάρτηση f_0 όσο και οι συναρτήσεις των περιορισμών $f_1, \ldots, f_m$ είναι κυρτές, δηλαδή ικανοποιούν τη σχέση

$$f_i(\alpha\vec{x} + \beta\vec{y}) \leq \alpha f_i(\vec{x}) + \beta f_i(\vec{y})$$

για κάθε $\vec{x}, \vec{y} \in \mathbb{R}^n$ και κάθε $\alpha, \beta \in \mathbb{R}$ όπου $\alpha + \beta = 1$, $\alpha \geq 0$, $\beta \geq 0$.

Τα προβλήματα κυρτής βελτιστοποίησης είναι μία κλάση προβλημάτων μαθηματικής βελτιστοποίησης για τα οποία, αν και δεν υπάρχει αναλυτική φόρμουλα, υπάρχουν πολύ αποδοτικές μέθοδοι επίλυσης.

Κεφάλαιο 3

Η μέθοδος gradient descent

3.1 Μέθοδοι descent

3.1.1 Περιγραφή του προβλήματος

Θα μελετήσουμε, στη συνέχεια, μία σημαντική υποκλάση προβλημάτων κυρτής βελτιστοποίησης, αυτά στα οποία μας ενδιαφέρει να ελαχιστοποιήσουμε μία δύο φορές συνεχώς παραγωγίσιμη συνάρτηση f ελλείψει περιορισμών. Τυπικά, ένα τέτοιο πρόβλημα παίρνει τη μορφή

$$\text{minimize} f(\vec{x}) \tag{3.1}$$

Για ένα πρόβλημα της μορφής αυτής, κι επειδή η f είναι μία παραγωγίσιμη και κυρτή συνάρτηση, ένα διάνυσμα $\vec{x}^*$ είναι η βέλτιστη λύση του προβλήματος βελτιστοποίησης, αν και μόνον αν η συνάρτηση f έχει στη θέση αυτή μηδενική κλίση. Αποκτούμε, λοιπόν, μία αναγκαία και ικανή συνθήκη για την ύπαρξη βέλτισης λύσης, την

$$\nabla f(\vec{x}^*) = 0 \tag{3.2}$$

Βλέπουμε έτσι ότι η επίλυση του προβλήματος (3.1) γίνεται ισοδύναμη με την επίλυση της (3.2), δηλαδή ενός συστήματος n εξισώσεων με n αγνώστους.

3.1.2 Επαναληπτικοί αλγόριθμοι

Υπάρχουν περιπτώσεις στις οποίες η αναλυτική επίλυση του συστήματος εξισώσεων (3.2) μπορεί να μας οδηγήσει σε επίλυση του προβλήματος βελτιστοποίησης (3.1). Συνήθως, ωστόσο, για την επίλυση τέτοιας μορφής προβλημάτων στρεφόμαστε σε επαναληπτικούς αλγορίθμους.

Τέτοιοι επαναληπτικοί αλγόριθμοι υπολογίζουν μία ακολουθία από σημεία εντός του πεδίου ορισμού της f

$$\vec{x}^{(0)}, \vec{x}^{(1)}, \ldots \in \mathbf{Dom} f$$

προσπαθώντας να συγκλίνουν στη βέλτιση λύση

$$\lim_{k\to\infty} f(\vec{x}^{(k)}) = f(\vec{x}^*)$$

Μία τέτοια ακολουθία σημείων ονομάζεται *ακολουθία ελαχιστοποίησης* (για το πρόβλημα (3.1)).

Ο αλγόριθμος τερματίζει, όταν προσεγγίσει αρκετά τη βέλτιστη τιμή της αντικειμενικής συνάρτησης. Η ανοχή του αλγορίθμου προς το σφάλμα από τη βέλτιστη τιμή καθορίζεται μέσω μίας σταθερής θετικής παραμέτρου ϵ. Η συνθήκη τερματισμού των επαναληπτικών αυτών αλγορίθμων παίρνει τυπικά τη μορφή

$$f(\vec{x}^{(k)}) - f(\vec{x}^*) \leq \epsilon, \epsilon > 0$$

3.1.3 Μέθοδοι descent

Διαισθητικά, είναι πολύ εύκολο να φανταστούμε επαναληπτικούς αλγορίθμους που προσπαθούν να επιλύσουν προβλήματα βελτιστοποίησης επιλέγοντας σε κάθε επανάληψη μία κατεύθυνση πάνω

στο χώρο που είναι ορισμένα και κάνοντας ένα "βήμα" προς την κατεύθυνση αυτή. Η πιο πάνω περιγραφή παίρνει μαθηματικά τη μορφή μιας ακολουθίας ελαχιστοποίησης

$$\vec{x}^{(k+1)} = \vec{x}^{(k)} + t^{(k)}\vec{\Delta x}^{(k)},\ k = 1, \ldots$$

όπου $\vec{\Delta x}^{(k)} \in \mathbb{R}^n$ είναι η *κατεύθυνση αναζήτησης* (εναλλακτικά *βήμα*) και $t^{(k)}$ είναι το *μέγεθος βήματος* στην k-οστή επανάληψη. Το μέγεθος βήματος θεωρείται θετικό, $t^{(k)} > 0$, εκτός αν ο επαναληπτικός αλγόριθμος έχει συγκλίνει.

Φυσικά, μας ενδιαφέρουν επαναληπτικοί αλγόριθμοι που δεν περιφέρονται άσκοπα στο χώρο, αλλά καταφέρνουν να συγκλίνουν στη βέλτιστη λύση επιλέγοντας κατάλληλα την κατεύθυνση αναζήτησης και το μέγεθος βήματος.

Οι *μέθοδοι descent* είναι επαναληπτικοί αλγόριθμοι της πιο πάνω μορφής που διασφαλίζουν ότι σε κάθε (μη βέλτιστο) βήμα της ακολουθίας ελαχιστοποίησής τους η τιμή της αντικειμενικής συνάρτησης μειώνεται, δηλαδή

$$f(\vec{x}^{(k+1)}) < f(\vec{x}^{(k)}) \tag{3.3}$$

Η ανισότητα (3.3) των μεθόδων descent, για μία παραγωγίσιμη και κυρτή αντικειμενική συνάρτηση, μπορεί να μεταφραστεί στην απαίτηση το εσωτερικό γινόμενο της κατεύθυνσης της αναζήτησης με την κλίση της αντικειμενικής συνάρτησης στο σημείο $\vec{x}^{(k)}$ να είναι αρνητικό.

$$\nabla f(\vec{x}^{(k)})^T \vec{\Delta x}^{(k)} < 0 \tag{3.4}$$

Με άλλα λόγια, μας ενδιαφέρει η κατεύθυνση της αναζήτησης να σχηματίζει οξεία γωνία με την αρνητική κλίση της αντικειμενικής συνάρτησης στο σημείο $\vec{x}^{(k)}$. Την κατεύθυνση αυτή ονομάζουμε *κατεύθυνση descent*.

Μαθηματική ανάλυση

Εφόσον η αντικειμενική συνάρτηση f είναι παραγωγίσιμη και κυρτή, ισχύει η ανισότητα

$$f(\vec{y}) \geq f(\vec{x}) + \nabla f(\vec{x})^T(\vec{y} - \vec{x})$$

για όλα τα σημεία $\vec{x}, \vec{y} \in \mathbf{Dom} f$.

Η παραπάνω ανισότητα, λοιπόν, θα ισχύει και για δύο οποιαδήποτε διαδοχικά σημεία της ακολουθίας ελαχιστοποίησης ενός επαναληπτικού αλγορίθμου, δηλαδή

$$f(\vec{x}^{(k+1)}) \geq f(\vec{x}^{(k)}) + \nabla f(\vec{x}^{(k)})^T(\vec{x}^{(k+1)} - \vec{x}^{(k)})$$

Αν περιοριστούμε εδώ στις μεθόδους descent που απαιτούν να ικανοποιείται η ανισότητα (3.3)

$$f(\vec{x}^{(k+1)}) < f(\vec{x}^{(k)})$$

τότε παρατηρούμε ότι πρέπει

$$\nabla f(\vec{x}^{(k)})^T(\vec{x}^{(k+1)} - \vec{x}^{(k)}) \leq f(\vec{x}^{(k+1)}) - f(\vec{x}^{(k)}) < 0$$

δηλαδή

$$\nabla f(\vec{x}^{(k)})^T(t^{(k)}\vec{\Delta x}^{(k)}) < 0$$

κι, επειδή το μέγεθος του βήματος είναι θετικό (εκτός αν έχουμε ήδη συγκλίνει στη βέλτιστη λύση), χρειάζεται για τις μεθόδους descent να ισχύει η ανισότητα

$$\nabla f(\vec{x}^{(k)})^T \vec{\Delta x}^{(k)} < 0$$

3.2 Η μέθοδος gradient descent

Στις μεθόδους descent περιοριζόμαστε σε κατευθύνσεις αναζήτησης που σχηματίζουν οξεία γωνία με την αρνητική κλίση της αντικειμενικής συνάρτησης, δηλαδή ικανοποιούν τη σχέση (3.4).

$$\nabla f(\vec{x}^{(k)})^T \vec{\Delta x}^{(k)} < 0$$

Μία φυσική επιλογή για την κατεύθυνση αναζήτησης είναι η αρνητική κλίση της αντικειμενικής συνάρτησης.

$$\vec{\Delta x} = -\nabla f(\vec{x}) \tag{3.5}$$

Ο αλγόριθμος που προκύπτει καλείται *αλγόριθμος gradient* ή *μέθοδος gradient descent* και η ακολουθία ελαχιστοποίησης που παράγει είναι η

$$\vec{x}^{(k+1)} = \vec{x}^{(k)} - t^{(k)} \nabla f(\vec{x}^{(k)}),\ k = 1, \ldots \tag{3.6}$$

Το κριτήριο τερματισμού παίρνει συνήθως τη μορφή

$$\|\nabla f(\vec{x})\|_2 \leq \epsilon \tag{3.7}$$

όπου το ϵ είναι μία μικρή και θετική σταθερά.

3.3 Ανάλυση σύγκλισης για τη μέθοδο gradient descent

Πριν προχωρήσουμε, θα παρουσιάσουμε μία χρήσιμη έννοια. Δοθείσας μίας συνάρτησης f και μίας σταθεράς c, ορίζεται ως *sublevel set* το σύνολο των σημείων εντός του πεδίου ορισμού της συνάρτησης, για τα οποία αυτή παίρνει τιμή μικρότερη ή ίση της σταθεράς c.

$$S = \{\vec{x} \in \mathbf{Dom} f \mid f(\vec{x}) \leq c\}$$

3.3.1 Αρχικό σημείο

Το πρώτο πράγμα που θα μας απασχολήσει είναι να βρούμε ένα κατάλληλο αρχικό σημείο, $\vec{x}^{(0)}$, για να ξεκινήσουμε την αναζήτησή μας.

Φυσικά, το αρχικό σημείο αυτό χρειάζεται να βρίσκεται εντός του πεδίου ορισμού της f.

$$\vec{x}^{(0)} \in \mathbf{Dom} f$$

Επιπλέον, χρειάζεται το sublevel set που ορίζει

$$S = \{\vec{x} \in \mathbf{Dom} f \mid f(\vec{x}) \leq \vec{x}^{(0)}\}$$

να είναι κλειστό.

Για μία κλειστή συνάρτηση, η δεύτερη συνθήκη ισχύει για όλα τα πιθανά αρχικά σημεία εντός του πεδίου ορισμού της.

3.3.2 Αυστηρά κυρτές συναρτήσεις

Για τη συνέχεια της ενότητας θα περιοριστούμε στη μελέτη αντικειμενικών συναρτήσεων που είναι αυστηρά κυρτές στο σύνολο S.

Για τέτοιες αυστηρά κυρτές συναρτήσεις, υπάρχει μία θετική σταθερά $m > 0$, τέτοια ώστε

$$\nabla^2 f(\vec{x}) \geq mI \tag{3.8}$$

για κάθε $\vec{x} \in S$.

Γνωρίζουμε ότι για κάθε $\vec{x}, \vec{y} \in S$ και $\vec{z}$ που ανήκει στο γραμμικό τμήμα $[\vec{x}, \vec{y}]$ ισχύει

$$f(\vec{y}) = f(\vec{x}) + \nabla f(\vec{x})^T(\vec{y} - \vec{x}) + \frac{1}{2}(\vec{y} - \vec{x})^T \nabla^2 f(\vec{z})(\vec{y} - \vec{x})$$

Μπορούμε να φράξουμε την ποσότητα $\nabla^2 f(\vec{z})$ χρησιμοποιώντας τη σχέση (3.8), με αποτέλεσμα την ανισοτική σχέση

$$f(\vec{y}) \geq f(\vec{x}) + \nabla f(\vec{x})^T(\vec{y} - \vec{x}) + \frac{m}{2}\|\vec{y} - \vec{x}\|_2^2 \tag{3.9}$$

που ισχύει για όλα τα $\vec{x}, \vec{y} \in S$.

3.3.3 Απόσταση από τη βέλτιστη τιμή

Αρχικά, θα χρησιμοποιήσουμε την ανισότητα (3.9) για να φράξουμε την απόσταση μίας τιμής της αντικειμενικής συνάρτησης από τη βέλτιστη τιμή, $f(\vec{x}) - p^*$, συναρτήσει του όρου $\|\nabla f(\vec{x})\|_2$.

Αν θεωρήσουμε την τιμή του $\vec{x}$ σταθερή, το δεξί μέρος της ανισότητας (3.9)

$$h(\vec{y}) = f(\vec{x}) + \nabla f(\vec{x})^T(\vec{y} - \vec{x}) + \frac{m}{2}\|\vec{y} - \vec{x}\|_2^2$$

είναι μία κυρτή τετραγωνική συνάρτηση του $\vec{y}$, η οποία ελαχιστοποιείται στο σημείο

$$\tilde{\vec{y}} = \vec{x} - \frac{1}{m}\nabla f(\vec{x})$$

Έχουμε, λοιπόν, από τη σχέση (3.9)

$$\begin{aligned}
f(\vec{y}) &\geq f(\vec{x}) + \nabla f(\vec{x})^T(\vec{y} - \vec{x}) + \frac{m}{2}\|\vec{y} - \vec{x}\|_2^2 \\
&\geq f(\vec{x}) + \nabla f(\vec{x})^T(\tilde{\vec{y}} - \vec{x}) + \frac{m}{2}\|\tilde{\vec{y}} - \vec{x}\|_2^2 \\
&= f(\vec{x}) + \nabla f(\vec{x})^T(-\frac{1}{m}\nabla f(\vec{x})) + \frac{m}{2}\| - \frac{1}{m}\nabla f(\vec{x})\|_2^2 \\
&= f(\vec{x}) - \frac{1}{m}\|\nabla f(\vec{x})\|_2^2 + \frac{1}{2m}\|\nabla f(\vec{x})\|_2^2 \\
&= f(\vec{x}) - \frac{1}{2m}\|\nabla f(\vec{x})\|_2^2
\end{aligned}$$

Επειδή η ανισότητα αυτή ισχύει για κάθε $\vec{y} \in S$, θα ισχύει και για το βέλτιστο διάνυσμα.

$$p^* = f(\vec{x}^*) \geq f(\vec{x}) - \frac{1}{2m}\|\nabla f(\vec{x})\|_2^2 \tag{3.10}$$

Εναλλακτικά, μπορούμε να γράψουμε την ανισότητα αυτή και ως

$$f(\vec{x}) - p^* \leq \frac{1}{2m}\|\nabla f(\vec{x})\|_2^2 \tag{3.11}$$

Η ανισότητα (3.11) δηλώνει ότι, αν η κλίση σε ένα σημείο $\vec{x}$ είναι μικρή, τότε το σημείο αυτό είναι σχεδόν βέλτιστο.

Αν, επιπλέον, θυμηθούμε τον ορισμό που είχαμε δώσει για την ανοχή ϵ ενός επαναληπτικού αλγορίθμου από τη βέλτιστη τιμή,

$$f(\vec{x}) - p^* \leq \epsilon$$

μπορούμε να συνδέσουμε αυτήν την ανοχή με την κλίση ενός σημείου για να θεωρηθεί αυτό σχεδόν βέλτιστο.

$$\|\nabla f(\vec{x})\|_2 \leq (2m\epsilon)^{1/2} \Rightarrow f(\vec{x}) - p^* \leq \epsilon \tag{3.12}$$

3.3.4 Απόσταση από το βέλτιστο διάνυσμα

Στη συνέχεια, θα χρησιμοποιήσουμε την ανισότητα (3.9) για να φράξουμε την απόσταση ενός διανύσματος $\vec{x}$ από το βέλτιστο διάνυσμα, $\|\vec{x} - \vec{x}^*\|_2$.

Θέτοντας $\vec{y} = \vec{x}^*$ στην (3.9) και, στη συνέχεια, χρησιμοποιώντας την ανισότητα Cauchy-Schwarz, έχουμε

$$\begin{aligned} p^* = f(\vec{x}^*) &\geq f(\vec{x}) + \nabla f(\vec{x})^T(\vec{x}^* - \vec{x}) + \frac{m}{2}\|\vec{x}^* - \vec{x}\|_2^2 \\ &\geq f(\vec{x}) - \|\nabla f(\vec{x})\|_2\|\vec{x}^* - \vec{x}\|_2 + \frac{m}{2}\|\vec{x}^* - \vec{x}\|_2^2 \end{aligned}$$

Κι, επειδή $p^* \leq f(\vec{x})$,

$$0 \geq p^* - f(\vec{x}) \geq -\|\nabla f(\vec{x})\|_2\|\vec{x}^* - \vec{x}\|_2 + \frac{m}{2}\|\vec{x}^* - \vec{x}\|_2^2$$

δηλαδή

$$0 \geq -\|\nabla f(\vec{x})\|_2\|\vec{x}^* - \vec{x}\|_2 + \frac{m}{2}\|\vec{x}^* - \vec{x}\|_2^2$$

και, επομένως,

$$\begin{aligned} 0 &\geq \|\vec{x}^* - \vec{x}\|_2(\frac{m}{2}\|\vec{x}^* - \vec{x}\|_2 - \|\nabla f(\vec{x})\|_2) \\ 0 &\geq \frac{m}{2}\|\vec{x}^* - \vec{x}\|_2 - \|\nabla f(\vec{x})\|_2 \\ \|\vec{x}^* - \vec{x}\|_2 &\leq \frac{2}{m}\|\nabla f(\vec{x})\|_2 \end{aligned}$$

Αποκτούμε έτσι κι ένα άνω φράγμα της απόστασης ενός διανύσματος από το βέλτιστο, συναρτήσει του όρου $\|\nabla f(\vec{x})\|_2$.

$$\|\vec{x} - \vec{x}^*\|_2 \leq \frac{2}{m}\|\nabla f(\vec{x})\|_2 \tag{3.13}$$

3.4 Στοχαστική gradient descent

Παρατηρώντας τη σχέση που ορίζει την ακολουθία ελαχιστοποίησης της μεθόδου gradient descent (3.6), βλέπουμε ότι για κάθε επανάληψη της μεθόδου απαιτείται ο υπολογισμός της κλίσης $\nabla f(\vec{x}^{(k)})$.

$$\vec{x}^{(k+1)} = \vec{x}^{(k)} - t^{(k)}\nabla f(\vec{x}^{(k)}),\ k = 1, \ldots$$

Με άλλα λόγια, απαιτείται ο υπολογισμός μίας μερικής παραγώγου για κάθε συντεταγμένη του διανύσματος $\vec{x}$.

Μπορούμε να οδηγηθούμε σε μία δραστικά απλοποιημένη, στοχαστική, προσεγγιστική παραλλαγή της μεθόδου gradient descent, αν σε κάθε βήμα της επαναληπτικής διαδικασίας μετακινούμε το διάνυσμα $\vec{x}$ λαμβάνοντας υπόψη μόνο μία από αυτές τις μερικές παραγώγους, την οποία επιλέγουμε τυχαία. Η παραλλαγή που προκύπτει ονομάζεται *στοχαστική gradient descent*.

Για κάθε επανάληψη, η στοχαστική gradient descent απαιτεί γνώση μόνο μίας από τις μερικές παραγώγους που συναποτελούν την κλίση $\nabla f(\vec{x}^{(k)})$ και, μάλιστα, η μερική παράγωγος αυτή επιλέγεται τυχαία, ανεξάρτητα από το ιστορικό προηγούμενων επιλογών. Αυτό σημαίνει ότι η μέθοδος αυτή, όχι μόνο δε χρειάζεται πλήρη γνώση όλων των μερικών παραγώγων της $\nabla f(\vec{x}^{(k)})$ για κάθε της βήμα, αλλά μπορεί να προχωράει κάθε φορά που η πληροφορία οποιασδήποτε από αυτές της γίνει γνωστή.

Ο προσεγγιστικός χαρακτήρας της παραλλαγής αυτής, ωστόσο, η οποία κάθε φορά κάνει πρόοδο με βάση μόνο ένα μέρος της όλης πληροφορίας, καθιστά τη σύγκλισή της στο ακριβές ελάχιστο πιο περίπλοκη. Την ανεπιθύμητη, όσο πλησιάζουμε προς τη βέλτιστη λύση, συμπεριφορά αυτή μπορούμε να αντισταθμίσουμε μειώνοντας το μέγεθος του βήματος. Στην περίπτωση αυτή, τόσο η gradient descent όσο και η στοχαστική παραλλαγή της συγκλίνουν σχεδόν πάντα στη βέλτιστη λύση.

Κεφάλαιο 4

Το μοντέλο των Friedkin-Johnsen

4.1 Εισαγωγή

Στα πλαίσια της έρευνας του φαινομένου της διαμόρφωσης απόψεων σε ένα κοινωνικό δίκτυο, οι οικονομολόγοι μελετήσανε μοντέλα στα οποία ένα σύνολο ανθρώπων συνδεδεμένων μέσω ενός κοινωνικού δικτύου συγκλίνουν σε μία κοινή άποψη. Η κοινή αυτή άποψη εγκαθιδρύεται μέσω μίας επαναλαμβανόμενης διαδικασίας, όπου κάθε κόμβος (άνθρωπος) στο κοινωνικό δίκτυο ανανεώνει τη γνώμη του με βάση κάποιο γραμμικό συνδυασμό των απόψεων των γειτόνων του, με πολλά από αυτά τα μοντέλα να έχουν ως βάση το μοντέλο του DeGroot.

Τα μοντέλα αυτής της κατηγορίας αφορούν περιπτώσεις όπου το κοινωνικό δίκτυο συγκλίνει σε ομοφωνία, με όλους τους κόμβους να έχουν τελικά την ίδια γνώμη. Ωστόσο, στην πράξη οι άνθρωποι είναι λιγότερο ή περισσότερο ισχυρογνώμονες και η ομοφωνία μία κατάσταση που σπάνια εμφανίζεται. Το 1990 οι Friedkin και Johnsen εισήγαγαν μία παραλλαγή του μοντέλου του DeGroot, όπου κάθε κόμβος του κοινωνικού δικτύου έχει, εκτός της γνώμης του, και μία αρχική εσωτερική πεποίθηση. Η εσωτερική αυτή πεποίθηση διατηρείται σταθερή με την πάροδο του χρόνου και συνυπολογίζεται με τις γνώμες των γειτόνων ενός κόμβου, όποτε αυτός ανανεώνει τη γνώμη του. Αυτή η μερική επιμονή των κόμβων στην αρχική τους εσωτερική πεποίθηση, δεν τους επιτρέπει, στη γενική περίπτωση, να αποκλίνουν από αυτή αρκετά ώστε η κοινωνία να οδηγηθεί σε ομοφωνία.

4.2 Το μοντέλο των Friedkin-Johnsen

Πιο συγκεκριμένα, στο μοντέλο Friedkin-Johnsen, το κοινωνικό δίκτυο περιγράφεται από έναν κατευθυνόμενο γράφο $G(V, E)$, όπου V το σύνολο των κόμβων του κοινωνικού δικτύου και E το σύνολο των ακμών μεταξύ αυτών. Κάθε κόμβος $i \in V$ έχει αρχικά μία εσωτερική πεποίθηση s_i. Όντας ισχυρογνώμων, ο κόμβος θα ανανεώνει την άποψή του, καθώς αλληλεπιδρά με τους γείτονές του μέσα στο κοινωνικό δίκτυο, αλλά θα επιμένει στην αρχική του αυτή πεποίθηση με ένα βάρος w_i. Κάθε ακμή $(i, j) \in E$ του κοινωνικού δικτύου αναπαριστά μία ενεργή εξάρτηση του κόμβου i από τον κόμβο j και χαρακτηρίζεται από ένα βάρος w_{ij}, που αντικατοπτρίζει το βαθμό με τον οποίο ο κόμβος i εμπιστεύεται τον κόμβο j.

Η ανανέωση των απόψεων πραγματοποιείται σε διακριτές χρονικές στιγμές t, σε κάθε μία από τις οποίες ο κόμβος i λαμβάνει υπόψη την εσωτερική του πεποίθηση s_i και τις απόψεις των γειτόνων του $x_j(t), j \in N(i)$ και αποκτά την άποψη $x_i(t+1)$. Δηλαδή, η άποψη του κόμβου i τη χρονική στιγμή $t+1$ διαμορφώνεται ως εξής:

$$x_i(t+1) = \frac{\sum\limits_{j \in N(i)} w_{ij} x_j(t) + w_i s_i}{\sum\limits_{j \in N(i)} w_{ij} + w_i} \tag{4.1}$$

Μπορούμε να γενικεύσουμε τον ορισμό του w_{ij} σε όλα τα ζεύγη (i, j) με $i \neq j$, θεωρώντας μηδενικό το βάρος της εξάρτησης μεταξύ κόμβων που δε συνδέονται άμεσα με κάποια ακμή, $(i, j) \notin E$. Με άλλα λόγια, θεωρούμε μηδενική την εξάρτηση του κόμβου i από κόμβους που δεν

ανήκουν στη γειτονιά του, $N(i)$.

$$w_{ij} = \begin{cases} 0 & \text{, if } i = j \\ w_{ij} & \text{, if } i \neq j \text{ and } j \in N(i) \\ 0 & \text{, if } i \neq j \text{ and } j \notin N(i) \end{cases} \tag{4.2}$$

Χρησιμοποιώντας τη γενίκευση (4.2) για την εξάρτηση μεταξύ κόμβων, μπορούμε να εκφράσουμε τη σχέση (4.1) και ως εξής:

$$x_i(t+1) = \frac{\sum\limits_{j \in V} w_{ij} x_j(t) + w_i s_i}{\sum\limits_{j \in V} w_{ij} + w_i} \tag{4.3}$$

4.3 Αναπαράσταση με μορφή πινάκων

Έστω $\vec{s}$ το διάνυσμα των αρχικών πεποιθήσεων των κόμβων του δικτύου και $\vec{x}(t)$ το διάνυσμα των απόψεων τους τη χρονική στιγμή t.

$$\vec{s} = [s_1, s_2, \ldots, s_n]^T$$

$$\vec{x}(t) = [x_1(t), x_2(t), \ldots, x_n(t)]^T$$

Αν ορίσουμε τους βοηθητικούς πίνακες $A = [A_{ij}]$ και $B = [B_{ij}]$, όπου

$$A_{ij} = \begin{cases} \frac{w_{ij}}{\sum\limits_{k \in V} w_{ik} + w_i} & \text{, if } i \neq j \\ 0 & \text{, if } i = j \end{cases}$$

και

$$B_{ij} = \begin{cases} 0 & \text{, if } i \neq j \\ \frac{w_i}{\sum\limits_{k \in V} w_{ik} + w_i} & \text{, if } i = j \end{cases}$$

μπορούμε να εκφράσουμε τη σχέση (4.3) συνολικά για όλους τους κόμβους του δικτύου ως εξής

$$\vec{x}(t+1) = A\vec{x}(t) + B\vec{s} \tag{4.4}$$

Ο πίνακας A εκφράζει, για κάθε ζευγάρι κόμβων (i, j), τη σχετική εμπιστοσύνη που δείχνει ο κόμβος i στη γνώμη του κόμβου j, αναλογικά της εμπιστοσύνης που δείχνει αθροιστικά στις γνώμες όλων των γειτόνων του και στη δική του αρχική πεποίθηση, $\sum\limits_{j \in V} w_{ij} + w_i$. Εδώ, όπως φαίνεται από τη σχέση (4.2), όλα τα διαγώνια στοιχεία του πίνακα A είναι μηδενικά.

Ο διαγώνιος πίνακας B δρα συμπληρωματικά του πίνακα A και εκφράζει, για κάθε κόμβο i, τη σχετική εμπιστοσύνη που δείχνει ο κόμβος i στην αρχική του πεποίθηση, αναλογικά του ίδιου αθροίσματος $\sum\limits_{j \in V} w_{ij} + w_i$.

4.4 Συνάρτηση κόστους

Όπως προείπαμε, το μοντέλο Friedkin-Johnsen αφορά κοινωνικά δίκτυα που δε συγκλίνουν σε ομοφωνία. Αυτή η συνύπαρξη διαφορετικών απόψεων επιφέρει ένα κόστος σε κάθε κόμβο του δικτύου.

Επειδή θεωρούμε ότι οι κόμβοι του δικτύου δρουν λογικά, αυτό που οδηγεί τη συμπεριφορά τους είναι η προσπάθειά τους να μειώσουν, στο βαθμό που αυτό είναι εφικτό, το προσωπικό τους κόστος. Επομένως, με το δίκτυο να αλλάζει συνεχώς, οι κόμβοι αποφασίζουν σε κάθε βήμα μυωπικά ποια γνώμη τους συμφέρει να υιοθετήσουν, προκειμένου να ελαχιστοποιήσουν το κόστος που αυτή τους επιφέρει.

Επειδή έχουμε ήδη καθορίσει τη συμπεριφορά του δικτύου μας, μέσω του κανόνα ανανέωσης (4.1) που ακολουθούν οι κόμβοι σε κάθε βήμα, χρειαζόμαστε έναν ορισμό για τη συνάρτηση κόστους μας που να είναι συμβατός με τη συμπεριφορά αυτή. Όπως γνωρίζουμε, μία συνάρτηση μίας μεταβλητής μπορεί να ελαχιστοποιηθεί μόνο σε σημεία που μηδενίζουν την πρώτη της παράγωγο. Οπότε η γραμμική σχέση (4.1) μπορεί να ελαχιστοποιεί μόνο μία συνάρτηση κόστους που έχει τετραγωνική μορφή.

Συγκεκριμένα, θεωρούμε ότι το κόστος που επιφέρει η άποψη x_i στον κόμβο i, δεδομένων των απόψεων των γειτόνων του x_j και της δικής του αρχικής πεποίθησης s_i, είναι:

$$C_i(\vec{x}) = \sum_{j \in N(i)} w_{ij}(x_i - x_j)^2 + w_i(x_i - s_i)^2 \tag{4.5}$$

Τη σχέση (4.5) μπορούμε να εκφράσουμε, χρησιμοποιώντας την (4.2), και ως:

$$C_i(\vec{x}) = \sum_{j \in V} w_{ij}(x_i - x_j)^2 + w_i(x_i - s_i)^2 \tag{4.6}$$

Η σχέση (4.5) αποτελεί μία πολύ λογική επιλογή για τον ορισμό του κόστους στο μοντέλο που περιγράφουμε, καθώς το κόστος που προκύπτει είναι πάντα θετικό και αυξάνεται όσο η γνώμη x_i ενός κόμβου i αποκλείνει από τη γνώμη x_j οποιουδήποτε γείτονά του ή όσο αυτός απομακρύνεται από την αρχική του πεποίθηση s_i. Επιπλέον, το κόστος αυξάνεται αναλογικά με το βάρος που ο κόμβος δίνει στις γνώμες αυτές.

Αυτό που επιτυγχάνεται με την εισαγωγή της έννοιας του κόστους εδώ, είναι η μοντελοποίηση της διαδικασίας διαμόρφωσης απόψεων του μοντέλου Friedkin-Johnsen ως ένα παιχνίδι ελαχιστοποίησης κόστους. Πιο συγκεκριμένα, μπορούμε να φανταστούμε τους κόμβους του κοινωνικού δικτύου ως παίκτες ενός παιχνιδιού, όπου το κέρδος κάθε παίκτη προσδιορίζεται από το πόσο μικρό είναι το κόστος του $C_i(\vec{x})$, ή, ισοδύναμα, από το πόσο μεγάλο είναι το αρνητικό κόστος του, $-C_i(\vec{x})$. Σε ένα τέτοιο παιχνίδι, η κίνηση κάθε παίκτη σε κάθε χρονική στιγμή είναι η γνώμη που αυτός εξωτερικεύει, $x_i(t)$.

Κοινωνικό κόστος

Αν αθροίσουμε τα κόστη (4.5) για όλους τους κόμβους του κοινωνικού δικτύου, θα πάρουμε το συνολικό κόστος $SC(\vec{x})$ που επιφέρει ένα διάνυσμα απόψεων $\vec{x}$ στην κοινωνία.

$$\begin{aligned} SC(\vec{x}) &= \sum_{i \in V} C_i(\vec{x}) \\ &= \sum_{i \in V} (\sum_{j \in V} w_{ij}(x_i - x_j)^2 + w_i(x_i - s_i)^2) \end{aligned} \tag{4.7}$$

4.5 Σύγκλιση

Εφόσον εξετάζουμε ένα δυναμικό μοντέλο, είναι φυσικό να μας ενδιαφέρει η συμπεριφορά που θα έχει αυτό στην πάροδο του χρόνου.

Καθώς το μοντέλο μας πραγματεύεται την αλληλεπίδραση απόψεων στην κοινωνία, μία βασική ερώτηση είναι κατά πόσο το κοινωνικό δίκτυο θα συγκλίνει σε μία κατάσταση ομοφωνίας. Στο μοντέλο των Friedkin-Johnsen, που οι κόμβοι είναι ισχυρογνώμονες και επιμένουν στην αρχική τους γνώμη με κάποιο βάρος, το κοινωνικό δίκτυο δε συγκλίνει στη γενική περίπτωση σε ομοφωνία.

Εφόσον η ομοφωνία δεν επιτυγχάνεται πάντα, μία επόμενη σημαντική ερώτηση είναι εάν το δίκτυο μπορεί να συγκλίνει σε μία κατάσταση ισορροπίας, στην οποία κανένας κόμβος δε θα θέλει να αλλάξει την εξωτερική του γνώμη, ή, αντιθέτως, οι κόμβοι θα χρειάζεται συνέχεια να αναπροσαρμόζουν την εξωτερική τους γνώμη στην προσπάθειά τους να ελαχιστοποιήσουν το κόστος τους. Όπως θα δείξουμε, όχι μόνο υπάρχει μία τέτοια κατάσταση ισορροπίας, αλλά το μοντέλο Friedkin-Johnsen πάντα συγκλίνει σε αυτή.

4.6 Ισορροπία Nash

Όπως μόλις αναφέραμε, μία ενδιαφέρουσα ερώτηση για τη δυναμική συμπεριφορά του μοντέλου Friedkin-Johnsen είναι εάν υπάρχει κάποια κατάσταση ισορροπίας, στην οποία κανένας κόμβος δε θα θέλει να αλλάξει τη γνώμη του.

Έχουμε εκφράσει το μοντέλο Friedkin-Johnsen ως ένα παιχνίδι ελαχιστοποίησης κόστους, στο οποίο οι κινήσεις των παικτών (κόμβων) του παιχνιδιού είναι αυτές ακριβώς οι γνώμες. Υπό αυτό το παιγνιοθεωρητικό πρίσμα, μας ενδιαφέρει να μελετήσουμε κατά πόσο υπάρχει μία κατάσταση, ένα διάνυσμα γνωμών, στην οποία, αν βρεθεί το κοινωνικό δίκτυο, κανένας κόμβος δε θα μπορεί να οδηγηθεί σε περαιτέρω μείωση του κόστους του αλλάζοντας μόνο τη δική του γνώμη. Μία τέτοια κατάσταση ονομάζεται στη θεωρία παιγνίων *ισορροπία Nash* και τη συμβολίζουμε με $\vec{x}^*$.

Για να διερευνήσουμε αν υπάρχει κάποια ισορροπία Nash στο παίγνιό μας, ξεκινάμε από το βασικό χαρακτηριστικό της, ότι αν το κοινωνικό μας δίκτυο βρεθεί σε ισορροπία Nash, δε θα φύγει από αυτή. Με άλλα λόγια

$$\vec{x}^*(t+1) = \vec{x}^*(t) \tag{4.8}$$

Χρησιμοποιώντας τη σχέση (4.4), η σχέση (4.8) παίρνει τη μορφή του γραμμικού συστήματος

$$A\vec{x}^*(t) + B\vec{s} = \vec{x}^*(t)$$

Για το μοντέλο Friedkin-Johnsen το σύστημα αυτό έχει πάντοτε μία ακριβώς λύση και η λύση αυτή δίνεται από τη σχέση

$$\vec{x}^* = (I - A)^{-1}B\vec{s} \tag{4.9}$$

Επομένως, για το μοντέλο Friedkin-Johnsen υπάρχει ακριβώς μία ισορροπία Nash, η οποία δίνεται από τη σχέση (4.9).

4.7 Συνάρτηση δυναμικού

Έχουμε πει ότι μπορούμε να βλέπουμε τη δυναμική συμπεριφορά του κοινωνικού δικτύου ως ένα παιχνίδι ελαχιστοποίησης κόστους, δηλαδή το κίνητρο που διέπει τις κινήσεις των παικτών είναι η προσπάθεια αυτών να ελαχιστοποιήσουν σε κάθε στιγμή το προσωπικό τους κόστος (4.6).

Ωστόσο, αυτή η μυωπική προσπάθεια των παικτών να μειώσουν ο καθένας το δικό του κόστος είναι δύσκολο να μας βοηθήσει να κατανοήσουμε τη γενική συμπεριφορά του συστήματος. Για το λόγο αυτό, θα ορίσουμε τη συνάρτηση

$$\Phi(\vec{x}) = \sum_{(i,j)\in E} w_{ij}(x_i - x_j)^2 + \sum_{i\in V} w_i(x_i - s_i)^2 \tag{4.10}$$

Η χρησιμότητα της συνάρτησης (4.10) προκύπτει από τον τρόπο με τον οποίο αυτή συνδέεται με τις συναρτήσεις κόστους όλων των παικτών.

Αν παρατηρήσουμε προσεκτικά, η συνάρτηση $\Phi(\vec{x})$ μπορεί να αποτυπώσει οποιαδήποτε μεταβολή του κόστους ενός παίκτη που προκύπτει από τη μεμονωμένη αλλαγή της δικής του γνώμης.

$$\Phi(x_i, \vec{x}_{-i}) - \Phi(x_i', \vec{x}_{-i}) = C_i(x_i, \vec{x}_{-i}) - C_i(x_i', \vec{x}_{-i}) \tag{4.11}$$

Στην πιο πάνω σχέση ο συμβολισμός $\Phi(x_i, \vec{x}_{-i})$ δηλώνει την τιμή της συνάρτησης Φ στην περίπτωση που η τιμή του παίκτη i είναι x_i, ενώ οι τιμές όλων των υπολοίπων παικτών είναι $\vec{x}_{-i}$. Κατ' αναλογία, με $\Phi(x_i', \vec{x}_{-i})$ συμβολίζουμε την τιμή της συνάρτησης Φ στην περίπτωση που η τιμή του παίκτη i έχει αλλάξει σε x_i', ενώ οι τιμές όλων των υπολοίπων παικτών παραμένουν $\vec{x}_{-i}$.

Η ιδιότητα (4.11) προκύπτει αναλυτικά ως εξής:

$$\begin{aligned}
\Phi(x_k, \vec{x}_{-k}) - \Phi(x'_k, \vec{x}_{-k}) &= \sum_{(i,j)\in E} w_{ij}(x_i - x_j)^2 + \sum_{i\in V} w_i(x_i - s_i)^2 \\
&\quad - \sum_{(i,j)\in E} w_{ij}(x'_i - x'_j)^2 - \sum_{i\in V} w_i(x'_i - s_i)^2 \\
&= \sum_{(k,j)\in E} w_{kj}(x_k - x_j)^2 + w_k(x_k - s_k)^2 \\
&\quad - \sum_{(k,j)\in E} w_{kj}(x'_k - x_k)^2 - w_k(x'_k - s_k)^2 \\
&= C_k(x_k, \vec{x}_{-k}) - C_k(x'_k, \vec{x}_{-k})
\end{aligned}$$

Η χρήσιμη αυτή συνάρτηση Φ (4.10) ονομάζεται *συνάρτηση δυναμικού*. Ο ορισμός συναρτήσεων δυναμικού για τη μοντελοποίηση ποσοτήτων που ελαττώνονται είναι αρκετά διαδεδομένος, ενώ, στη θεωρία παιγνίων, τα παιγνία που έχουν μία συνάρτηση δυναμικού ονομάζονται *παίγνια δυναμικού*.

4.8 Συνάρτηση δυναμικού και ισορροπία Nash

Έχουμε πει ότι στο παίγνιό μας το κίνητρο των παικτών είναι να ελαχιστοποιήσουν το κόστος τους και ότι οι παίκτες βρίσκονται σε μια κατάσταση ισορροπίας Nash, εάν κανένας από αυτούς δε θέλει να μεταβάλει την άποψή του, γιατί καμία τέτοια μονομερής μεταβολή της γνώμης ενός παίκτη δεν μπορεί να οδηγήσει σε πτώση του προσωπικού του κόστους. Έχουμε, επίσης, πει ότι η συνάρτηση δυναμικού κωδικοποιεί την πτώση του προσωπικού κόστους σε ακριβώς αυτές τις μονομερείς μεταβολές της γνώμης κάποιου παίκτη. Συνδυάζοντας τα πιο πάνω, προκύπτει η εξής παρατήρηση.

Παρατήρηση. *Εάν $\vec{x}^*$ είναι ένα τοπικό ελάχιστο της $\Phi(\vec{x})$, τότε το $\vec{x}^*$ είναι μία ισορροπία Nash.*

Από τον ορισμό (4.10) της συνάρτησης δυναμικού Φ, παρατηρούμε ότι είναι μία συνεχής συνάρτηση που είναι πάντα μη αρνητική και, άρα, κάτω-φραγμένη. Όπως γνωρίζουμε, μία συνεχής, κάτω-φραγμένη συνάρτηση παρουσιάζει ολικό ελάχιστο. Επιπλέον, παρατηρούμε ότι η συνάρτηση Φ είναι αυστηρά κυρτή και, συνεπώς, δεν υπάρχουν τοπικά ελάχιστα και το ολικό ελάχιστο είναι μοναδικό.

Συνολικά, η $\Phi(\vec{x})$ έχει μοναδικό ολικό ελάχιστο $\vec{x}^*$ και αυτό το ολικό ελάχιστο είναι ακριβώς η ισορροπία Nash του μοντέλου Friedkin-Johnsen.

4.9 Σύγκλιση στην ισορροπία Nash

Για το μοντέλο Friedkin-Johnsen, η ισορροπία Nash είναι μοναδική και ταυτίζεται με το μοναδικό ολικό ελάχιστο της συνάρτησης δυναμικού Φ. Μας ενδιαφέρει να υπολογίσουμε αυτό το μοναδικό ολικό ελάχιστο και για το σκοπό αυτό μπορούμε να χρησιμοποιήσουμε έναν αλγόριθμο βελτιστοποίησης.

Εφόσον ενδιαφερόμαστε για την ελαχιστοποίηση μίας δύο φορές παραγωγίσιμης συνάρτησης, της συνάρτησης δυναμικού Φ, ελλείψει περιορισμών, μπορούμε να επιλέξουμε ως αλγόριθμο βελτιστοποίησης τη μέθοδο gradient descent.

Η μέθοδος gradient descent είναι ένας επαναληπτικός αλγόριθμος βελτιστοποίησης που παράγει την ακολουθία ελαχιστοποίησης (3.6),

$$\vec{x}^{(k+1)} = \vec{x}^{(k)} - t^{(k)} \nabla f(\vec{x}^{(k)}),\ k = 1, \ldots$$

η οποία συγκλίνει, καθώς το πλήθος των επαναλήψεων τείνει στο άπειρο, στη βέλτιστη λύση $\vec{x}^*$.

$$\lim_{k\to\infty} \vec{x}^{(k)} = \vec{x}^*$$

Εφαρμόζοντας τη μέθοδο gradient descent στη συνάρτηση δυναμικού Φ, παράγεται η ακολουθία ελαχιστοποίησης

$$\vec{x}^{(k+1)} = \vec{x}^{(k)} - t^{(k)} \nabla \Phi(\vec{x}^{(k)}),\ k = 1, \ldots \tag{4.12}$$

η οποία συγκλίνει στο ολικό ελάχιστο της συνάρτησης δυναμικού, δηλαδή την ισορροπία Nash του κοινωνικού δικτύου.

Από τη σχέση (4.12), κι επειδή

$$\nabla f(\vec{x}) = (\frac{\partial f(\vec{x})}{\partial x_1}, \frac{\partial f(\vec{x})}{\partial x_2}, \ldots, \frac{\partial f(\vec{x})}{\partial x_n})$$

βλέπουμε ότι κάθε μία από τις συνιστώσες του $\vec{x}$, έστω η i-οστή, ανανεώνεται όπως ορίζει η ακολουθία

$$x_i^{(k+1)} = x_i^{(k)} - t^{(k)} \frac{\partial \Phi}{\partial x_i}(\vec{x}^{(k)}),\ k = 1, \ldots \tag{4.13}$$

Αν, επιπλέον, θεωρήσουμε το βήμα $t^{(k)}$ σταθερό και ίσο με

$$t^{(k)} = \frac{1}{2} \frac{1}{\sum\limits_{j \in N(i)} w_{ij} + w_i} \tag{4.14}$$

(κανονικοποιώντας τα βάρη αν αυτό απαιτείται), έχουμε

$$\begin{aligned}
x_i^{(k+1)} &= x_i^{(k)} - t^{(k)} \frac{\partial \Phi}{\partial x_i}(\vec{x}^{(k)}) \\
&= x_i^{(k)} - t^{(k)}(2 \sum_{j \in N(i)} w_{ij}(x_i^{(k)} - x_j^{(k)}) + 2w_i(x_i^{(k)} - s_i)) \\
&= x_i^{(k)} - 2t^{(k)}(\sum_{j \in N(i)} w_{ij}(x_i^{(k)} - x_j^{(k)}) + w_i(x_i^{(k)} - s_i)) \\
&= x_i^{(k)} - \frac{1}{\sum\limits_{j \in N(i)} w_{ij} + w_i}(\sum_{j \in N(i)} w_{ij}(x_i^{(k)} - x_j^{(k)}) + w_i(x_i^{(k)} - s_i)) \\
&= x_i^{(k)} - \frac{\sum\limits_{j \in N(i)} w_{ij} x_i^{(k)} + w_i x_i^{(k)}}{\sum\limits_{j \in N(i)} w_{ij} + w_i} + \frac{\sum\limits_{j \in N(i)} w_{ij} x_j^{(k)} + w_i s_i}{\sum\limits_{j \in N(i)} w_{ij} + w_i} \\
&= \frac{\sum\limits_{j \in N(i)} w_{ij} x_j^{(k)} + w_i s_i}{\sum\limits_{j \in N(i)} w_{ij} + w_i}
\end{aligned}$$

Άρα, κάθε συνιστώσα ανανεώνεται όπως ορίζει ο κανόνας

$$x_i^{(k+1)} = \frac{\sum\limits_{j \in N(i)} w_{ij} x_j^{(k)} + w_i s_i}{\sum\limits_{j \in N(i)} w_{ij} + w_i} \tag{4.15}$$

Αν στη σχέση (4.15) αντικαταστήσουμε την έννοια των επαναλήψεων με την έννοια του χρόνου, αυτή θα πάρει τη μορφή

$$x_i(t+1) = \frac{\sum\limits_{j \in N(i)} w_{ij} x_j(t) + w_i s_i}{\sum\limits_{j \in N(i)} w_{ij} + w_i}$$

δηλαδή ταυτίζεται με τον κανόνα ανανέωσης του μοντέλου Friedkin-Johnsen.

Με άλλα λόγια, ο κανόνας ανανέωσης του μοντέλου Friedkin-Johnsen ακολουθεί φυσικά τα ίδια ακριβώς βήματα που ακολουθεί κι η μέθοδος gradient descent, αν την εφαρμόσουμε για να υπολογίσουμε το ολικό ελάχιστο της συνάρτησης δυναμικού Φ. Άρα, οι κόμβοι του μοντέλου Friedkin-Johnsen ανανεώνουν τις γνώμες τους με τέτοιο τρόπο που τελικά συγκλίνει στην κατάσταση ολικού ελαχίστου της συνάρτησης δυναμικού Φ, δηλαδή ακριβώς στην κατάσταση ισορροπίας Nash.

Κεφάλαιο 5

Παραλλαγή του μοντέλου των Friedkin-Johnsen με μνήμη

5.1 Gradient descent στο μοντέλο Friedkin-Johnsen

Στο προηγούμενο κεφάλαιο παρουσιάσαμε ένα μοντέλο που εισήγαγαν οι Friedkin και Johnsen στα πλαίσια της μελέτης τους για το πως διαμορφώνονται οι απόψεις σε κοινωνικά δίκτυα. Στο μοντέλο αυτό, οι κόμβοι ενός κοινωνικού δικτύου, ξεκινώντας ο καθένας από μία αρχική εσωτερική πεποίθηση, διαμορφώνουν με την πάροδο του χρόνου την άποψή τους, συνυπολογίζοντας κάθε φορά αυτήν την αρχική πεποίθηση με τις γνώμες που υιοθετούν δυναμικά οι γείτονές τους. Το μοντέλο των Friedkin-Johnsen αντικατοπτρίζει δύο βασικά χαρακτηριστικά της διαμόρφωσης απόψεων στην κοινωνία, αφενός ότι οι απόψεις των ανθρώπων αλλάζουν όπως αυτοί δρουν και αλληλεπιδρούν στα πλαίσια της κοινωνίας, αφετέρου ότι οι άνθρωποι είναι σχετικά "ισχυρογνώμονες" και δεν είναι διατεθειμένοι να απομακρυνθούν τελείως από την αυθόρμητη, αρχική, εσωτερική τους πεποίθηση.

Σε ένα τέτοιο δίκτυο, που οι κόμβοι δεσμεύονται από την αρχική τους πεποίθηση και δεν μπορούν να απομακρυνθούν πολύ από αυτή, στη γενική περίπτωση οι κόμβοι δε συγκλίνουν σε ομοφωνία. Ωστόσο, δείξαμε ότι το κοινωνικό δίκτυο του μοντέλου Friedkin-Johnsen συγκλίνει με την πάροδο του χρόνου σε μία ισορροπία Nash, δηλαδή μία κατάσταση στην οποία οι κόμβοι έχουν σταθεροποιήσει ο καθένας την άποψή του και κανένας από αυτούς δε θέλει να αλλάξει μονόπλευρα τη δική του. Δείξαμε, επίσης, ότι αυτή η ισορροπία Nash είναι μοναδική και ταυτίζεται με το σημείο ολικού ελαχίστου μίας κατάλληλα ορισμένης συνάρτησης δυναμικού Φ.

Επιπλέον, κατά τη μελέτη της δυναμικής συμπεριφοράς του μοντέλου Friedkin-Johnsen, είδαμε ότι ο τρόπος που οι κόμβοι του κοινωνικού δικτύου ανανεώνουν τις απόψεις τους μέχρι να συγκλίνουν στην κατάσταση ισορροπίας Nash ταυτίζεται με την ακολουθία ελαχιστοποίησης που προκύπτει αν εφαρμόσουμε τη μέθοδο gradient descent για τον υπολογισμό αυτού του ολικού ελαχίστου της συνάρτησης δυναμικού Φ. Δηλαδή, οι κόμβοι του μοντέλου Friedkin-Johnsen, ενός αρκετά διαισθητικού μοντέλου διαμόρφωσης απόψεων σε ένα κοινωνικό δίκτυο, στην προσπάθειά τους να υιοθετήσουν μία άποψη που να απέχει όσο το δυνατόν λιγότερο από την αρχική τους πεποίθηση και τις απόψεις των γειτόνων τους τρέχουν την επαναληπτική μέθοδο βελτιστοποίησης gradient descent.

Μάλιστα, οι κόμβοι του μοντέλου Friedkin-Johnsen τρέχουν μία κατανεμημένη παραλλαγή της μεθόδου gradient descent, αφού ο ολικός υπολογισμός κάθε επανάληψης (4.12)

$$\vec{x}^{(k+1)} = \vec{x}^{(k)} - t^{(k)} \nabla \Phi(\vec{x}^{(k)}),\ k = 1, \ldots$$

διαμοιράζεται στους κόμβους, με κάθε έναν από αυτούς να υπολογίζει μία συντεταγμένη του (4.13).

$$x_i^{(k+1)} = x_i^{(k)} - t^{(k)} \frac{\partial \Phi}{\partial x_i}(\vec{x}^{(k)}),\ k = 1, \ldots$$

5.2 Κατανεμημένη gradient descent

Όπως είδαμε, η μέθοδος gradient descent είναι ένας επαναληπτικός αλγόριθμος που μας επιτρέπει να επιλύσουμε αποδοτικά μία σημαντική υποκλάση προβλημάτων κυρτής βελτιστοποίησης, τα προβλήματα ελαχιστοποίησης μίας δύο φορές συνεχώς παραγωγίσιμης συνάρτησης $f : \mathbb{R}^n \to \mathbb{R}$. Η

επίλυση αυτή στηρίζεται στον υπολογισμό, σε κάθε βήμα, ενός διανύσματος $\vec{x}$ μέσω μίας σχέσης της μορφής (3.6).

$$\vec{x}^{(k+1)} = \vec{x}^{(k)} - t^{(k)} \nabla f(\vec{x}^{(k)}),\ k = 1, \ldots$$

Αν διαμοιράσουμε τον υπολογισμό κάθε συντεταγμένης του διανύσματος $\vec{x}^{(k+1)}$

$$x_i^{(k+1)} = x_i^{(k)} - t^{(k)} \frac{\partial f}{\partial x_i}(\vec{x}^{(k)}),\ k = 1, \ldots$$

σε έναν κόμβο ενός δικτύου, όπως διαμοιράζεται φυσικά ο υπολογισμός στο κοινωνικό δίκτυο του μοντέλου Friedkin-Johnsen, προκύπτει μία κατανεμημένη παραλλαγή της μεθόδου gradient descent, η *κατανεμημένη gradient descent*.

5.3 Περιορισμοί της κατανεμημένης gradient descent

Η κατανεμημένη παραλλαγή της gradient descent μας επιτρέπει να διαιρέσουμε τον ολικό υπολογισμό που πραγματοποιείται σε κάθε επανάληψη σε n ανεξάρτητους υπολογισμούς που μπορούν να εκτελεστούν παράλληλα και να αναθέσουμε καθέναν από αυτούς τους υπολογισμούς σε έναν κόμβο ενός δικτύου, όπως στην περίπτωση του κοινωνικού δικτύου του μοντέλου Friedkin-Johnsen.

Ωστόσο, ο διαμοιρασμός αυτός απαιτεί κάθε κόμβος του δικτύου να γνωρίζει σε κάθε επανάληψη ολόκληρο το διάνυσμα $\vec{x}^{(k)}$. Με άλλα λόγια, για να προχωρήσουμε από τη μία επανάληψη στην επόμενη, χρειάζεται να ολοκληρωθούν όλοι οι επιμέρους υπολογισμοί του προηγούμενου βήματος και τα αποτελέσματά τους να διαδοθούν από όλους τους κόμβους σε όλους τους κόμβους του δικτύου. Επιπλέον, χρειάζεται όλοι οι κόμβοι να έχουν αρκετή μνήμη για να αποθηκεύσουν ολόκληρο το διάνυσμα $\vec{x}^{(k)}$ που λαμβάνουν.

Οι απαιτήσεις αυτές είναι αρκετά περιοριστικές για την εφαρμογή της κατανεμημένης gradient descent σε συστήματα πολλών κόμβων, όπου η επικοινωνιακή πολυπλοκότητα και οι απαιτήσεις για μνήμη σε κάθε κόμβο αυξάνονται.

Θα ήταν χρήσιμο, για τέτοιες περιπτώσεις, να μελετήσουμε παραλλαγές της κατανεμημένης gradient descent που χαλαρώνουν τους παραπάνω περιορισμούς, ενώ συνεχίζουν να επιτρέπουν στη μέθοδο που προκύπτει να συγκλίνει. Μας ενδιαφέρουν, δηλαδή, παραλλαγές στις οποίες η δυνατότητα ανταλλαγής τιμών στο δίκτυο και η διαθέσιμη μνήμη σε κάθε κόμβο είναι περιορισμένες.

5.4 Παραλλαγές της κατανεμημένης gradient descent με μνήμη

Όπως είπαμε, μας ενδιαφέρει να μελετήσουμε τη συμπεριφορά παραλλαγών της κατανεμημένης gradient descent μεθόδου οι οποίες απαιτούν μικρότερη επικοινωνιακή πολυπλοκότητα και λιγότερη διαθέσιμη μνήμη σε κάθε κόμβο.

Θα περιορίσουμε την ποσότητα της πληροφορίας που βρίσκεται διαθέσιμη σε κάθε κόμβο σε κάθε χρονική στιγμή θεωρώντας ότι κάθε κόμβος έχει πρόσβαση σε μία περιορισμένη ποσότητα μνήμης M θέσεων.

$$Memory_i : 1 \times M$$

Επιπλέον, προκειμένου να περιορίσουμε την ποσότητα της πληροφορίας που διαδίδεται μεταξύ των κόμβων, θεωρούμε ότι σε κάθε επανάληψη μόνο ένα υποσύνολο των συντεταγμένων του διανύσματος $\vec{x}^{(k)}$ γίνονται διαθέσιμες σε κάθε κόμβο. Με άλλα λόγια, θεωρούμε ότι σε κάθε επανάληψη κάθε κόμβος λαμβάνει πληροφορία από ένα υποσύνολο των γειτόνων του.

$$N_{avail.}^{(k)}(i) \subseteq N(i)$$

5.5 Παραλλαγές του μοντέλου των Friedkin-Johnsen με μνήμη

Για να μελετήσουμε τις παραλλαγές της κατανεμημένης gradient descent με μνήμη που περιγράψαμε, θα στρέψουμε την προσοχή μας σε μία συγκεκριμένη εφαρμογή της, αυτή που προκύπτει με φυσικό τρόπο από τη δυναμική συμπεριφορά του δικτύου του μοντέλου των Friedkin και Johnsen. Ο στόχος μας είναι, τροποποιώντας και μελετώντας τη δυναμική συμπεριφορά του μοντέλου των Friedkin-Johnsen στην πάροδο του χρόνου, να αποκτήσουμε κάποιες πρώτες ενδείξεις για την επίδραση που έχουν πιθανές αλλαγές στην επαναληπτική διαδικασία της κατανεμημένης gradient descent στη συμπεριφορά του επαναληπτικού αλγορίθμου.

Επομένως, θεωρούμε μία παραλλαγή του μοντέλου των Friedkin-Johnsen όπου κάθε κόμβος του κοινωνικού δικτύου έχει πρόσβαση σε μία περιορισμένη ποσότητα μνήμης M θέσεων.

$$Memory_i : 1 \times M$$

Οι κόμβοι μπορούν να αξιοποιήσουν διαφορετικά τμήματα της μνήμης τους με διαφορετικό τρόπο, οπότε μπορούμε να φανταστούμε τη συνολική μνήμη κάθε κόμβου ως ένα σύνολο m επιμέρους μνημών.

$$Memory_i = [mem_i^1|mem_i^2|...|mem_i^m] : 1 \times M$$

$$mem_i^k : 1 \times M_k, k = 1, 2, ..., m \text{ και } M_1 + M_2 + ... + M_m = M$$

Επιπλέον, προκειμένου να περιορίσουμε την ποσότητα της πληροφορίας που διαδίδεται μεταξύ των κόμβων, θεωρούμε ότι μεταξύ δύο επαναλήψεων κάθε κόμβος λαμβάνει πληροφορία από ένα υποσύνολο των γειτόνων του.

$$N_{avail.}^{(k)}(i) \subseteq N(i)$$

Όταν κάποιος κόμβος λάβει μία καινούρια τιμή από κάποιον από τους γείτονές του, μπορεί να επιλέξει να τη χρησιμοποιήσει για να αλλάξει την κατάστασή του, δηλαδή να ανανεώσει τη μνήμη του με βάση την καινούργια πληροφορία που έλαβε.

Η παραλλαγή του μοντέλου των Friedkin-Johnsen που περιγράψαμε λαμβάνει υπόψη τους περιορισμούς σε μνήμη που μπορεί να υπάρχουν στους κόμβους ενός δικτύου, καθώς και τους περιορισμούς σε επικοινωνιακή πολυπλοκότητα ανάμεσα σε αυτούς. Ως αποτέλεσμα προκύπτει ένα αρκετά πιο χαλαρωμένο μοντέλο, στο οποίο η μόνη πληροφορία που είναι διαθέσιμη στους κόμβους σε κάθε χρονική στιγμή είναι η πληροφορία που αυτοί έχουν αποφασίσει να αποθηκεύσουν με κάποια μορφή στη μνήμη που διαθέτουν. Η πληροφορία αυτή μπορεί να αφορά μόνο ένα υποσύνολο των γειτόνων τους, είτε επειδή οι κόμβοι δεν έχουν αρκετή μνήμη για να αποθηκεύσουν πληροφορία για όλους τους γείτονές τους, είτε επειδή δεν κατέφθασε στους κόμβους πληροφορία από όλους αυτούς. Επίσης, σε ένα τέτοιο μοντέλο, η πληροφορία που είναι αποθηκευμένη στη μνήμη των κόμβων μπορεί να μην ανανεώνεται πλήρως σε κάθε επανάληψη, αλλά να αφορά απόψεις που γνωστοποιήθηκαν σε έναν κόμβο σε προηγούμενες επαναλήψεις.

5.6 Τυπική περιγραφή του μοντέλου των προσομοιώσεων

Μας ενδιαφέρει, λοιπόν, η μελέτη της συμπεριφοράς της παραλλαγής του μοντέλου των Friedkin-Johnsen με μνήμη που μόλις περιγράψαμε. Η μελέτη μας θα πάρει τη μορφή προσομοιώσεων. Στην ενότητα αυτή, θα περιγράψουμε το ακριβές μοντέλο που χρησιμοποιήθηκε στις προσομοιώσεις αυτές.

Όπως και στο μοντέλο Friedkin-Johnsen, ξεκινάμε με ένα κοινωνικό δίκτυο που περιγράφεται από ένα γράφο $G(V, E)$, όπου V το σύνολο των κόμβων του κοινωνικού δικτύου και E το σύνολο των ακμών μεταξύ αυτών.

Κάθε κόμβος $i \in V$ έχει αρχικά μία εσωτερική πεποίθηση s_i. Την πεποίθηση αυτή θεωρούμε μη αρνητική και, χωρίς βλάβη της γενικότητας, φραγμένη από τη μονάδα, $s_i \in [0, 1]$. Όντας ισχυρογνώμων, ο κόμβος θα ανανεώνει την άποψή του καθώς αλληλεπιδρά με τους γείτονές του μέσα στο κοινωνικό δίκτυο, αλλά θα επιμένει στην αρχική του αυτή πεποίθηση με ένα βάρος w_i. Το βάρος

αυτό θεωρούμε ότι είναι μη αρνητικό και, χωρίς βλάβη της γενικότητας, φραγμένο από τη μονάδα, $w_i \in [0, 1]$. Επιπλέον, για τη διεξαγωγή των προσομοιώσεών μας, θεωρούμε ότι το βάρος αυτό είναι κοινό για όλους τους κόμβους, $w_i = w$.

Κάθε ακμή $(i, j) \in E$ του γράφου μας αναπαριστά μία ενεργή εξάρτηση του κόμβου i από τον κόμβο j και χαρακτηρίζεται από ένα βάρος w_{ij}, που αντικατοπτρίζει το βαθμό με τον οποίο ο κόμβος i εμπιστεύεται τον κόμβο j. Για τη διεξαγωγή των προσομοιώσεών μας, θα περιοριστούμε σε γράφους όπου το βάρος w_{ij} είναι μοναδιαίο για ακμές μεταξύ γειτονικών κόμβων και μπορεί να θεωρηθεί μηδενικό για ακμές μεταξύ κόμβων που δε γειτνιάζουν.

$$w_{ij} = \begin{cases} 0 & \text{, if } i = j \\ 1 & \text{, if } i \neq j \text{ and } j \in N(i) \\ 0 & \text{, if } i \neq j \text{ and } j \notin N(i) \end{cases}$$

Επιπλέον, κάθε κόμβος έχει μία μνήμη M θέσεων συνολικά, η οποία μπορεί να είναι διαχωρισμένη εικονικά σε m επιμέρους τμήματα.

$$Memory_i = [mem_i^1 | mem_i^2 | ... | mem_i^m] : 1 \times M$$

$$mem_i^k : 1 \times M_k, k = 1, 2, ..., m \text{ και } M_1 + M_2 + ... + M_m = M$$

Θεωρούμε ότι σε κάθε χρονική στιγμή ο κάθε κόμβος επιτρέπει στο k-οστό από αυτά τα τμήματα να επηρεάσει την άποψή του όπως ορίζει η συνάρτηση βάρους $w_{ik}(t)$. Για τις προσομοιώσεις μας, θεωρούμε αυτή τη συνάρτηση βάρους μοναδιαία.

$$w_{ik}(t) = 1$$

Τέλος, προκειμένου να ποσοτικοποιήσουμε την έννοια της σύγκλισης, θεωρούμε ότι ένα κοινωνικό δίκτυο έχει συγκλίνει στην κατάσταση ισορροπίας του, εάν απέχει από αυτή απόσταση $e = 0.01$.

Κεφάλαιο 6

Ο Προσομοιωτής

6.1 Εισαγωγή

Για τη μελέτη της συμπεριφοράς του θεωρητικού μοντέλου που παρουσιάσαμε στο τέλος του προηγούμενου κεφαλαίου, υλοποιήσαμε έναν προσομοιωτή στη γλώσσα προγραμματισμού `Python`, και συγκεκριμένα στην `Python 2.7`.

Ο λόγος που στραφήκαμε σε αυτήν την προσέγγιση ήταν να χρησιμοποιήσουμε ένα πρόγραμμα για να αποτυπώσουμε το γράφο του κοινωνικού μας δικτύου και να προσομοιώσουμε τις αλληλεπιδράσεις μεταξύ των κόμβων του, προκειμένου να μελετήσουμε τη δυναμική συμπεριφορά του στην πάροδο του χρόνου και να αποκτήσουμε κάποιες πρώτες ενδείξεις για τον τρόπο με τον οποίο η ύπαρξη περιορισμένης μνήμης στους κόμβους ενός κοινωνικού δικτύου και επικοινωνίας μεταξύ αυτών επηρεάζει τη σύγκλιση σε αυτήν την παραλλαγή του μοντέλου των Friedkin-Johnsen.

Απώτερος σκοπός μας είναι, μέσω της πραγματοποίησης των προσομοιώσεων αυτών, να αποκτήσουμε μία διαίσθηση για το αν κατανεμημένες παραλλαγές επαναληπτικών αλγορίθμων βελτιστοποίησης, όπως η gradient descent, μπόρουν να εφαρμοστούν επιτυχώς σε μεγαλύτερης κλίμακας δίκτυα με περιορισμούς στη μνήμη που είναι διαθέσιμη στους κόμβους τους και στη δυνατότητα επικοινωνίας μεταξύ αυτών.

6.2 Τα αρχεία εισόδου

Η παραμετροποίηση του προσομοιωτή μας καθίσταται εφικτή μέσω της παροχής σε αυτόν κατάλληλων αρχείων εισόδου. Τα αρχεία αυτά είναι απλά αρχεία κειμένου, που περιέχουν τα ονόματα των παραμέτρων που δέχεται ο προσομοιωτής και τις επιθυμητές τιμές που θέλει ο χρήστης να δώσει σε αυτές.

Μέσω των αρχείων εισόδου, ο χρήστης μπορεί να προσδιορίσει το γράφο του κοινωνικού δικτύου, τη μνήμη που είναι διαθέσιμη σε καθέναν από τους κόμβους του, καθώς και τον τρόπο που οι κόμβοι χειρίζονται τη μνήμη αυτή. Μπορεί, επίσης, να προσδιορίσει τις αρχικές πεποιθήσεις των κόμβων και την εμπιστοσύνη που οι κόμβοι έχουν σε αυτές. Επιπλέον, ο χρήστης έχει τη δυνατότητα να επιλέξει εάν σε κάθε βήμα αλληλεπιδρά ένα τυχαίο ζεύγος κόμβων ή εάν κάθε κόμβος αλληλεπιδρά με έναν τυχαίο γείτονά του. Τέλος, ο χρήστης μπορεί να παραμετροποιήσει τον τρόπο με τον οποίο ο προσομοιωτής ελέγχει εάν το δίκτυο έχει συγκλίνει στην κατάσταση ισορροπίας.

Πιο συγκεκριμένα, τα αρχεία εισόδου του προσομοιωτή έχουν την παρακάτω μορφή.

```
[population]

    PopulationSize:{positive integer}

    InteractionGraphType:<included_edges/excluded_edges/networkx>

    InteractionGraphEdgesType:<undirected/directed>

    IncludeEdges:{list of to—be—included edges}
    ——option available when [population]InteractionGraphType:included_edges

    ExcludeEdges:{list of to—be—excluded edges}
    ——option available when [population]InteractionGraphType:excluded_edges

    NetworkX:<complete_graph/
              cycle_graph/
              path_graph/
              star_graph/
              wheel_graph/
              barbell_graph {size of the two complete graphs} {size of the in—between path graph}/
              lollipop_graph {size of the complete graph} {size of the path graph}/
              complete_multipartite_graph {sizes of the independent sets}/
              gnm_random_graph {number of edges to be added}/
              random_regular_graph {degree}>
    ——option available when [population]InteractionGraphType:networkx

    NetworkXTries:{positive integer}
    ——option available when [population]InteractionGraphType:networkx
    ——and [population]NetworkX:<gnm_random_graph/random_regular_graph>

    AgentType:<adaptive>

    AgentMemory:{list of memory items to be allocated to each agent}
        Memory items are (MemoryType, MemorySize, Coefficient(optional)) tuples, where:
        MemoryType:<'window'/'exclusive_memory'>
        MemorySize:{positive integer}
        Coefficient:<{number}/{coefficient function}>

    AgentConfidence:<{number}/'equal'>

[input]

    InitialValues:<random/{list of real values}>

[convergence]

    PerRoundUpdate:<random_edge_per_node/single_random_edge>

    Norm:<{non—zero integer}/inf/—inf/fro/nuc>

    Epsilon:{positive number}

    IntervalBetweenChecks:{positive integer}

    VerificationChecks:<{positive integer}/log>

[output]

    FileSafeMode:<on/off>

    PrintReport:<yes/no>

    OutputPlots:<yes/no>

    SavePlots:<yes/no>
```

[population]PopulationSize

Η παράμετρος αυτή προσδιορίζει το πλήθος των κόμβων του κοινωνικού δικτύου και παίρνει την τιμή οποιουδήποτε θετικού ακεραίου.

[population]InteractionGraphType

Η παράμετρος αυτή προσδιορίζει τον τρόπο με τον οποίο θα ορισθεί ο γράφος.

- `included_edges`: Η επιλογή αυτή επιτρέπει στο χρήστη να προσδιορίσει το σύνολο των ακμών του γράφου.
- `excluded_edges`: Η επιλογή αυτή επιτρέπει στο χρήστη να προσδιορίσει το σύνολο των ακμών που δεν υπάρχουν στο γράφο, δηλαδή τις ακμές που θέλει να αφαιρεθούν ξεκινώντας από έναν πλήρη γράφο.
- `networkx`: Η επιλογή αυτή επιτρέπει στο χρήστη να επιλέξει ανάμεσα σε κάποια συνηθισμένα είδη γράφων, όπως αυτά παρέχονται από το πακέτο `NetworkX` της `Python`.

[population]InteractionGraphEdgesType

Η παράμετρος αυτή προσδιορίζει εάν ο γράφος θα είναι κατευθυνόμενος ή μη.

- `undirected`: Η επιλογή αυτή αντιστοιχεί σε μη κατευθυνόμενο γράφο.
- `directed`: Η επιλογή αυτή αντιστοιχεί σε κατευθυνόμενο γράφο.

[population]NetworkX

Η επιλογή αυτή αφορά την περίπτωση που ο χρήστης θέλει να χρησιμοποιήσει τις δυνατότητες που προσφέρονται από το πακέτο `NetworkX` της `Python` για να ορίσει το γράφο του κοινωνικού δικτύου και του επιτρέπει να επιλέξει μέσα από ένα σύνολο συνηθισμένων γράφων.

- `complete_graph`: Η επιλογή αυτή αντιστοιχεί σε έναν πλήρη γράφο (κλίκα), όπου όλοι οι κόμβοι του κοινωνικού δικτύου συνδέονται με όλους τους υπόλοιπους κόμβους του.
- `cycle_graph`: Η επιλογή αυτή συνδέει τους κόμβους σε έναν (απλό) κύκλο.
- `path_graph`: Η επιλογή αυτή συνδέει τους κόμβους σε ένα (απλό) μονοπάτι.
- `star_graph`: Η επιλογή αυτή συνδέει τους κόμβους σε ένα γράφο αστέρα, όπου ένας κόμβος βρίσκεται στο κέντρο του αστέρα και όλοι οι υπόλοιποι κόμβοι είναι συνδεδεμένοι αποκλειστικά με αυτόν τον πρώτο κόμβο.
- `wheel_graph`: Η επιλογή αυτή συνδέει τους κόμβους σε ένα γράφο τροχό, όπου ένας κόμβος βρίσκεται στο κέντρο του τροχού και όλοι οι υπόλοιποι κόμβοι είναι συνδεδεμένοι με τον πρώτο αυτόν κόμβο και, επιπλέον, μεταξύ τους σε έναν (απλό) κύκλο.
- `barbell_graph <c> <p>`: Η επιλογή αυτή δημιουργεί ένα γράφο που αποτελείται από δύο ίδιου μεγέθους, `c`, πλήρεις γράφους συνδεδεμένους μέσω ενός μονοπατιού μεγέθους `p`.
- `lollipop_graph <c> <p>`: Η επιλογή αυτή δημιουργεί ένα γράφο που αποτελείται από έναν πλήρη γράφο μεγέθους `c` που συνδέεται με ένα μονοπάτι μεγέθους `p`.
- `complete_multipartite_graph <int1> <int2> ... <intk>`: Η επιλογή αυτή δημιουργεί έναν (πλήρη) πολυμερή γράφο, τα μεγέθη των συνόλων ανεξαρτησίας του οποίου καθορίζονται από τους ακεραίους που ακολουθούν, `int1`, `int2`, ..., `intk`.

- `gnm_random_graph <m>`: Η επιλογή αυτή δημιουργεί έναν τυχαίο γράφο πλήθους ακμών `m`.
- `random_regular_graph <d>`: Η επιλογή αυτή δημιουργεί έναν τυχαίο κανονικό γράφο βαθμού `d`.

`[population]NetworkXTries`

Η επιλογή αυτή αφορά την περίπτωση που ο χρήστης θέλει να χρησιμοποιήσει τις δυνατότητες που προσφέρονται από το πακέτο `NetworkX` της `Python` για τη δημιουργία ενός τυχαίου γράφου. Επειδή μας ενδιαφέρουν μόνο τυχαίοι γράφοι που είναι συνδεδεμένοι (ή αντίστοιχα ισχυρά συνδεδεμένοι), στην περίπτωση που ο τυχαίος γράφος που παράγεται από το πακέτο `NetworkX` δεν είναι κατάλληλος, ζητάμε από το πακέτο να μας παράξει έναν καινούριο. Η παράμετρος αυτή θέτει ένα άνω όριο στο πλήθος αυτών των επαναλήψεων και παίρνει την τιμή ενός ακεραίου.

`[population]AgentType`

Η παράμετρος αυτή προσδιορίζει το είδος των κόμβων του κοινωνικού δικτύου, ως προς τον τρόπο που αυτοί χρησιμοποιούν τη διαθέσιμή τους μνήμη.

- `adaptive`: Κάθε κόμβος έχει πρόσβαση σε μία συλλογή από μνήμες και ανανεώνει την άποψή του ως ένα γραμμικό συνδυασμό των βεβαρυμένων μέσων όρων των τιμών που είναι αποθηκευμένες στις μνήμες αυτές και της αρχικής του πεποίθησης. Οι συντελεστές του γραμμικού συνδυασμού είναι χρονομεταβλητοί.

`[population]AgentMemory`

Η παράμετρος αυτή προσδιορίζει τα είδη της μνήμης που είναι διαθέσιμα σε κάθε κόμβο και παίρνει την τιμή μίας λίστας από κομμάτια μνήμης. Καθένα από αυτά τα κομμάτια μνήμης προσδιορίζεται από μία τριπλέτα της μορφής **(`memType`, `memSize`, `cff`)**.

Η παράμετρος **`memType`** προσδιορίζει το είδος της μνήμης.

- `'window'`: Μνήμη που αποθηκεύει τις πιο πρόσφατες τιμές που της γνωστοποιούνται.
- `'exclusive'`: Μνήμη που τίθεται κατά τη δημιουργία της να παρακολουθεί ένα πλήθος κόμβων και να αποθηκεύει, για καθέναν από αυτούς, την πιο πρόσφατη τιμή που της γνωστοποιείται. Η μνήμη αυτή δεν αποθηκεύει πληροφορία για κόμβους που δεν παρακολουθεί.

Η παράμετρος **`memSize`** προσδιορίζει το μέγεθος της μνήμης.

Η παράμετρος **`cff`** προσδιορίζει το συντελεστή βάρους της μνήμης συναρτήσει του χρόνου (της τρέχουσας επανάληψης) και μπορεί να πάρει την τιμή είτε κάποιας προκαθορισμένης συνάρτησης μίας μεταβλητής, είτε μίας αριθμητικής σταθεράς. Η παράμετρος αυτή είναι προαιρετική και θεωρείται μοναδιαία σε περίπτωση που δεν καθορίζεται.

`[population]AgentConfidence`

Η παράμετρος αυτή προσδιορίζει την ισχυρογνωμοσύνη των κόμβων ως προς την αρχική τους πεποίθηση και μπορεί να πάρει οποιαδήποτε τιμή στο διάστημα $[0, 1]$ ή την τιμή `'equal'`, που δηλώνει ότι ο κόμβος εμπιστεύεται την αρχική του πεποίθηση όσο και τις απόψεις των γειτόνων του.

`[population]InitialValues`

Η παράμετρος αυτή προσδιορίζει τις αρχικές πεποιθήσεις των κόμβων του κοινωνικού δικτύου και παίρνει την τιμή μίας λίστας από αριθμούς ή την τιμή `random`, που οδηγεί στην παραγωγή μίας τυχαίας λίστας αρχικών πεποιθήσεων από αριθμούς στο διάστημα $[0, 1]$.

[convergence]PerRoundUpdate

Η παράμετρος αυτή προσδιορίζει τον τρόπο με τον οποίο επιλέγονται οι κόμβοι που θα αλληλεπιδράσουν σε κάθε γύρο.

- `single_random_edge`: Σε κάθε γύρο, επιλέγεται μία τυχαία ακμή του γράφου και αλληλεπιδρούν οι κόμβοι στα άκρα της. Εάν ο γράφος είναι μη κατευθυνόμενος, η αλληλεπίδραση είναι συμμετρική και οι δύο κόμβοι ανταλλάσουν τιμές. Εάν ο γράφος είναι κατευθυνόμενος, η αλληλεπίδραση θεωρείται μη συμμετρική και μόνο ο κόμβος αφετηρία της ακμής μαθαίνει την τιμή του κόμβου πέρατος της ακμής.

- `random_edge_per_node`: Σε κάθε γύρο, κάθε κόμβος αλληλεπιδρά με έναν τυχαίο γείτονά του. Η αλληλεπίδραση θεωρείται μη συμμετρική και η ροή πληροφορίας πραγματοποιείται από τον γείτονα προς τον εκάστοτε κόμβο.

[convergence]Norm

Η παράμετρος αυτή προσδιορίζει τη νόρμα που θα χρησιμοποιηθεί για τη μέτρηση των αποστάσεων όπου αυτές απαιτούνται και μπορεί να πάρει την τιμή οποιουδήποτε μη αρνητικού ακέραιου αριθμού, καθώς και τις τιμές `inf` ($+\infty$), `-inf` ($-\infty$), `fro` (Frobenius) και nuc (nuclear).

[convergence]Epsilon

Η παράμετρος αυτή προσδιορίζει το σφάλμα σύγκλισης και μπορεί να πάρει την τιμή οποιουδήποτε θετικού αριθμού.

[convergence]IntervalBetweenChecks

Η παράμετρος αυτή καθορίζει ανά πόσους γύρους πραγματοποιείται ο έλεγχος για σύγκλιση και μπορεί να πάρει την τιμή οποιουδήποτε θετικού ακεραίου. Ο σκοπός αυτής της παραμέτρου είναι να μειώσει τη διάρκεια των προσομοιώσεων.

[convergence]VerificationChecks

Η παράμετρος αυτή καθορίζει το πλήθος των συνεχόμενων γύρων που πρέπει η κατάσταση του δικτύου να απέχει από την κατάσταση ισορροπίας απόσταση μικρότερη από το σφάλμα σύγκλισης, ώστε να θεωρήσουμε ότι όντως το δίκτυό μας έχει συγκλίνει. Επιτρεπτές τιμές για αυτήν την παράμετρο είναι όλοι οι θετικοί ακέραιοι αριθμοί και η τιμή `log`, που ορίζει πλήθος συνεχόμενων γύρων λογαριθμικό ως προς το μέγεθος του πληθυσμού.

[output]FileSafeMode

Η παράμετρος αυτή καθορίζει εάν θα απαγορεύεται στον προσομοιωτή να αντικαθιστά υπάρχοντα αρχεία εξόδου και παίρνει τις τιμές `on`, για προστασία των υπαρχόντων αρχείων εξόδου, και `off`, για μη προστασία.

[output]PrintReport

Η παράμετρος αυτή καθορίζει εάν ο προσομοιωτής θα παράγει ένα επιπλέον αρχείο εξόδου που θα περιέχει μία αναλυτική καταγραφή της προσομοίωσης. Οι επιτρεπτές τιμές αυτής της παραμέτρου είναι `yes` και `no`.

[output]OutputPlots

Η παράμετρος αυτή καθορίζει εάν ο προσομοιωτής θα παρουσιάζει στο χρήστη επιπλέον ορισμένες γραφικές παραστάσεις, συγκεκριμένα τη γραφική παράσταση της απόστασης των καταστάσεων του δικτύου που ελέγχουμε από την κατάσταση ισορροπίας και τη γραφική παράσταση της απόστασης μίας κατάστασης του δικτύου που ελέγχουμε από την προηγούμενη κατάσταση του δικτύου που ελέγξαμε. Οι επιτρεπτές τιμές αυτής της παραμέτρου είναι `yes` και `no`.

[output]SavePlots

Η παράμετρος αυτή καθορίζει εάν ο προσομοιωτής θα αποθηκεύει τις πιο πάνω γραφικές παραστάσεις που παρουσίασε στο χρήστη. Οι επιτρεπτές τιμές αυτής της παραμέτρου είναι `yes` και `no`.

6.3 Η αρχιτεκτονική του προσομοιωτή

Ο προσομοιωτής έχει υλοποιηθεί στη γλώσσα προγραμματισμού `Python 2.7` και για την παραμετροποίησή του χρησιμοποιούμε ένα αρχείο εισόδου το οποίο δίνεται στο πρόγραμμα ως παράμετρος κατά την εκτέλεση.

Σε ένα περιβάλλον τύπου `Unix`, μπορούμε να εκτελέσουμε τον προσομοιωτή από τη γραμμή εντολών με μία εντολή της μορφής:

```
$ python main.py input—file.txt
```

όπου `input-file.txt` είναι το αρχείο εισόδου.

6.3.1 Το κυρίως πρόγραμμα

Το κυρίως μέρος του προσομοιωτή είναι αυτό που τελικά πραγματοποιεί την προσομοίωση συνδυάζοντας κατάλληλα τα επιμέρους τμήματά του.

Αρχικά, το κυρίως πρόγραμμα ελέγχει ότι έχει όντως κληθεί από τη γραμμή εντολών με κάποιο υπαρκτό και κατάλληλης μορφής αρχείο εισόδου και κάνει χρήση της κλάσης `Configuration` του πακέτου `configuration`, προκειμένου να αναγνωρίσει κάποιες βασικές παραμέτρους της προσομοίωσης, όπως το μέγεθος και το γράφο του κοινωνικού δικτύου, τον τρόπο που συμπεριφέρονται οι κόμβοι αυτού και τις αρχικές τους πεποιθήσεις.

Με τη χρήση των πιο πάνω πληροφοριών, το κυρίως πρόγραμμα καλεί τη συνάρτηση `computeStableState` του πακέτου `stableState`, για να μάθει την κατάσταση ισορροπίας του κοινωνικού δικτύου.

Πριν την εκκίνηση της προσομοίωσης, το κυρίως πρόγραμμα διαβάζει από το αρχείο εισόδου ορισμένες επιπλέον παραμέτρους σχετικές με τη δυναμική συμπεριφορά του δικτύου, τον έλεγχο για σύγκλιση και τον επιθυμητό τρόπο παρουσίαση των αποτελεσμάτων και προετοιμάζει το περιβάλλον για την πραγματοποίηση της προσομοίωσης, πραγματοποιώντας τις κατάλληλες αρχικοποιήσεις και προετοιμάζοντας τα απαιτούμενα αρχεία εξόδου.

Στη συνέχεια, το κυρίως πρόγραμμα τρέχει την προσομοίωση, όπως αυτή έχει περιγραφεί και παραμετροποιηθεί.

Τέλος, το κυρίως πρόγραμμα παρουσιάζει και καταγράφει τα αποτελέσματα της προσομοίωσης και τερματίζει.

6.3.2 Η κλάση `Configuration`

Οι βασικές παράμετροι που καθορίζουν τη μορφή του κοινωνικού δικτύου που θέλουμε να προσομοιώσουμε και τις αρχικές συνθήκες αυτού διαβάζονται μέσω της κλάσης `Configuration` του πακέτου `configuration`, που έχει σκοπό να αποφορτίσει το κυρίως πρόγραμμα από το μεγαλύτερο μέρος της διαχείρισης της εισόδου του προσομοιωτή.

Συγκεκριμένα, η κλάση `Configuration` είναι υπεύθυνη για τη δημιουργία του ζητούμενου γράφου του κοινωνικού δικτύου, το διάβασμα των παραμέτρων που αφορούν τη διαθέσιμη μνήμη σε κάθε κόμβο του κοινωνικού δικτύου, καθώς και τον τρόπο που οι κόμβοι αυτοί χρησιμοποιούν αυτή τη μνήμη για τη διαμόρφωση της άποψής τους, και, τέλος, τις αρχικές πεποιθήσεις τους και το βαθμό που οι κόμβοι επιμένουν σε αυτές.

6.3.3 Το πακέτο `memory`

Σκοπός του πακέτου `memory` είναι να παρέχει τις απαιτούμενες κλάσεις για τις μνήμες που είναι διαθέσιμες στους κόμβους του κοινωνικού δικτύου.

Η κλάση `Memory`

Αρχικά, το πακέτο περιέχει τον ορισμό της κλάσης `Memory` που αποτελεί τη βασική υλοποίηση μίας μνήμης που δεν περιέχει και δε φορτώνεται με πληροφορία. Η κλάση αυτή ορίστηκε με σκοπό να χρησιμοποιηθεί ως κοινή βάση για πιο εξειδικευμένα και λειτουργικά είδη μνήμης, συγκεκριμένα τις κλάσεις `Window` και `ExclusiveMemory`.

Η κλάση `Window`

Η κλάση `Window` αναπαριστά μία μνήμη που αποθηκεύει τις πιο πρόσφατες τιμές που μαθαίνει. Για κάθε τιμή στη μνήμη αποθηκεύεται προαιρετικά η πηγή της και ένας συντελεστής βάρους, που προκαθορίζεται στη μονάδα.

Η κλάση `ExclusiveMemory`

Η κλάση `ExclusiveMemory` αναπαριστά μία μνήμη που αποθηκεύει την πιο πρόσφατη τιμή που μαθαίνει για ένα σύνολο πηγών που καθορίζονται κατά τη δημιουργία της. Με άλλα λόγια, κατά την αρχικοποίησή της, μία `ExclusiveMemory` μνήμη δεσμεύεται να παρακολουθεί ένα σύνολο πηγών, για κάθε μία από τις οποίες δεσμεύει μία θέση μνήμης, την οποία ανανεώνει κάθε φορά που της γίνεται διαθέσιμη μία πιο πρόσφατη τιμή από την πηγή αυτή. Για κάθε ζεύγος τιμής-πηγής αποθηκεύται και ένας συντελεστής βάρους, που προκαθορίζεται στη μονάδα.

6.3.4 Το πακέτο `agent`

Σκοπός του πακέτου `agent` είναι να παρέχει τις απαιτούμενες κλάσεις για την αναπαράσταση των κόμβων του δικτύου.

Η κλάση `Agent`

Αρχικά, το πακέτο περιέχει τον ορισμό της κλάσης `Agent` που αποτελεί τη βασική υλοποίηση ενός κόμβου και έχει οριστεί με σκοπό να χρησιμοποιηθεί ως βάση για πιο εξειδικευμένες υλοποιήσεις κόμβων, συγκεκριμένα εδώ την κλάση `AdaptiveAgent`. Θεωρούμε ότι ένας κόμβος χαρακτηρίζεται από την ταυτότητά (όνομά) του, την αρχική του πεποίθηση, την εμπιστοσύνη που δείχνει στην αρχική αυτή πεποίθηση, τη μνήμη που του είναι διαθέσιμη, την εκάστοτε άποψή του και το σύνολο των γειτόνων του στο κοινωνικό δίκτυο.

Η κλάση `AdaptiveAgent`

Η κλάση `AdaptiveAgent` αναπαριστά έναν κόμβο που έχει πρόσβαση σε μία συλλογή από μνήμες και ανανεώνει την άποψή του ως ένα γραμμικό συνδυασμό των βεβαρυμένων μέσων όρων των τιμών που είναι αποθηκευμένες στις μνήμες αυτές και της αρχικής του πεποίθησης. Οι συντελεστές του γραμμικού συνδυασμού είναι χρονομεταβλητοί.

Κατά τη δημιουργία ένας `AdaptiveAgent` κόμβου, αυτός αρχικοποιείται με βάση τα βασικά χρονοσταθερά μεγέθη του (την ταυτότητά του, την αρχική του πεποίθηση, την εμπιστοσύνη που έχει σε αυτή και το σύνολο των γειτόνων) και, επιπλέον, δημιουργούνται και αρχικοποιούνται για αυτόν όλα τα κομμάτια μνήμης που του είναι διαθέσιμα. Επιπλέον, θεωρούμε ότι ο κόμβος έχει μία άποψη κάθε χρονική στιγμή η οποία αρχικοποιείται, όπως είναι φυσικό, στην αρχική του πεποίθηση.

Η κατάσταση του κόμβου ενημώνεται, όταν καταφθάνει σε αυτόν μία καινούρια άποψη από κάποιον από τους γείτονές του. Μόλις στον κόμβο γνωστοποιηθεί μία τέτοια άποψη, αυτός τη χρησιμοποιεί για να ενημερώσει τις διαθέσιμες σε αυτόν μνήμες για την καινούργια τιμή. Ανάλογα με το είδος της μνήμης, η τιμή αποθηκεύεται κατάλληλα ή όχι.

Η άποψη του κόμβου κάθε χρονική στιγμή προκύπτει, όπως αναφέραμε πριν λίγο, ως ένας γραμμικός συνδυασμός των βεβαρυμένων μέσων όρων των τιμών που είναι αποθηκευμένες στις μνήμες του και της αρχικής του πεποίθησης.

6.3.5 Η κλάση `Scheduler`

Η κλάση `Scheduler` του πακέτου `scheduler` αποτελεί μία υλοποίηση ενός πολύ απλού τυχαίου δρομολογητή, που επιλέγει μία τιμή από ένα σύνολο τιμών ισοπίθανα.

6.3.6 Η συνάρτηση `computeStableStabe`

Ο υπολογισμός της κατάστασης ισορροπίας του κοινωνικού δικτύου πραγματοποιείται μέσω της συνάρτησης `computeStableState` που ορίζεται στο πακέτο `stableState`. Η συνάρτηση αυτή δέχεται ως ορίσματα τα βάρη με τα οποία οι κόμβοι εμπιστεύονται τους γείτονές τους, το βάρος με το οποίο επιστεύονται την αρχική τους πεποίθηση και τις αρχικές πεποιθήσεις τους και υπολογιζει την κατάσταση ισορροπίας του κοινωνικού δικτύου μέσω της σχέσης (4.9)

$$\vec{x}^* = (I - A)^{-1} B \vec{s}$$

όπου

$$A_{ij} = \begin{cases} 0 & \text{, if } i = j \\ \frac{w_{ij}}{\sum\limits_{k \in V} w_{ik} + w_i} & \text{, if } i \neq j \end{cases}$$

$$B_{ij} = \begin{cases} \frac{w_i}{\sum\limits_{k \in V} w_{ik} + w_i} & \text{, if } i = j \\ 0 & \text{, if } i \neq j \end{cases}$$

$$\vec{s} = [s_1, s_2, \ldots, s_n]$$

Κεφάλαιο 7

Τα αποτελέσματα των προσομοιώσεων

7.1 Εισαγωγή

Σκοπός για τη δημιουργία του προσομοιωτή είναι να μας επιτρέψει να αποκτήσουμε μία πρώτη πειραματική εικόνα για το πως συμπεριφέρεται το μοντέλο μας. Συγκεκριμένα, αλλάζοντας το μοντέλο των Friedkin-Johnsen ώστε να περιορίζεται η διαθέσιμη μνήμη σε κάθε κόμβο και η δυνατή ποσότητα επικοινωνίας μεταξύ των κόμβων αυτών, μας ενδιαφέρει να αποκτήσουμε ενδείξεις για το αν οι περιορισμοί αυτοί συνεχίζουν να επιτρέπουν στο δίκτυο να συγκλίνει στην κατάσταση ισορροπίας του, και, αν ναι, με τι ταχύτητα.

Προκειμένου να μην υπερφορτώσουμε την παρουσίαση των αποτελεσμάτων με περιττή πληροφορία, αλλά να επικεντρωθούμε σε αυτό που όντως μας ενδιαφέρει, εάν τα πειραματικά μας αποτελέσματα αποτελούν ενθαρρυντικές ενδείξεις για τη σύγκλιση του μοντέλου μας, θα θεωρήσουμε το μεγαλύτερο μέρος των παραμέτρων μας σταθερό και θα παρουσιάζουμε μόνο τις βασικές παραμέτρους που αλλάζουν μεταξύ των προσομοιώσεων.

Συγκεκριμένα, όλα τα δίκτυα δημιουργήθηκαν με αυτόματο τρόπο μέσω του πακέτου `NetworkX` της `Python`,

```
InteractionGraphType:networkx
NetworkXTries:5
```

όλοι οι κόμβοι είναι τύπου `adaptive` και εμπιστεύονται την αρχική τους πεποίθηση όσο και τις απόψεις των γειτόνων τους,

```
AgentType:adaptive
AgentConfidence:'equal'
```

οι αρχικές τιμές των κόμβων παράγονται με τυχαίο τρόπο,

```
InitialValues:random
```

ενώ σε κάθε βήμα κάθε κόμβος αλληλεπιδρά τυχαία με ένα γείτονά του.

```
PerRoundUpdate:random_edge_per_node
```

Τέλος, σχετικά με τη σύγκλιση, χρησιμοποιούμε την άπειρη νόρμα για τη μέτρηση αποστάσεων, θεωρούμε την τιμή σφάλματος ϵ ίση με 0.01, ελέγχουμε για σύγκλιση κάθε 5 βήματα και για την επιβεβαίωση της σύγκλισης απαιτούμε λογαριθμικό αριθμό βημάτων ως προς το μέγεθος του πληθυσμού.

```
Norm:inf
Epsilon:0.01
IntervalBetweenChecks:5
VerificationChecks:log
```

7.2 Μεταβλητού μεγέθους πλήρης γράφος - Πλήρης `exclusive` μνήμη

Περιγραφή

Στην υποενότητα αυτή θα παρουσιάσουμε τα αποτελέσματα των προσομοιώσεων σε κοινωνικά δίκτυα που έχουν μορφή πλήρη γράφου και κάθε κόμβος διατηρεί στη μνήμη του την τελευταία τιμή που έλαβε από κάθε γείτονά του.

```
PopulationSize:<n>
NetworkX:complete_graph
AgentMemory:[('exclusive_memory', <n>)]
```

Αποτελέσματα

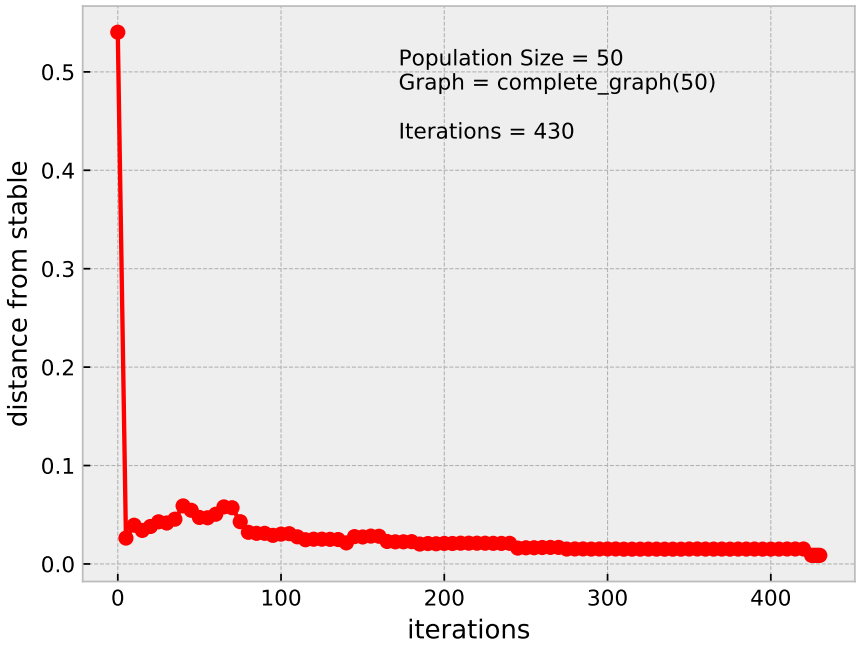
Οι γραφικές παραστάσεις 7.1, 7.2 και 7.3 αποτυπώνουν την απόσταση του δικτύου από την κατάσταση ισορροπίας ως προς τον τρέχοντα αριθμό της επανάληψης για μέγεθος πληθυσμού $n = 20$, $n = 50$ και $n = 150$ αντίστοιχα.

Η γραφική παράσταση 7.4 αποτυπώνει τη μεταβολή του πλήθους των επαναλήψεων που απαιτούνται για τη σύγκλιση στην κατάσταση ισορροπίας καθώς αυξάνεται το μέγεθος του πληθυσμού.

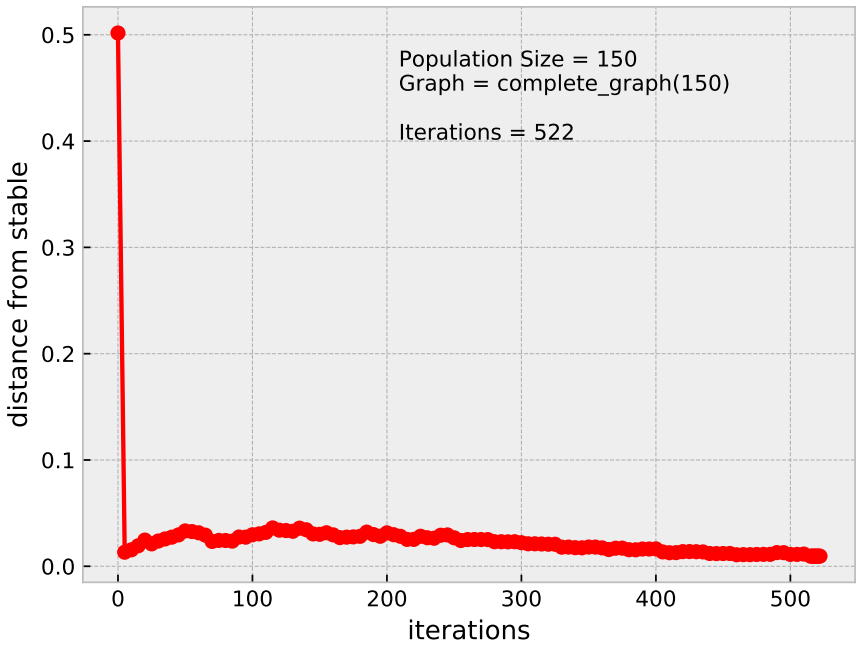
Συμπέρασματα

Αρχικά, παρατηρούμε ότι στην περίπτωση του πλήρους γράφου, εάν κάθε κόμβος διατηρεί στη μνήμη του την τελευταία τιμή που του έχει γνωστοποιηθεί από κάθε γείτονά του, οι προσομοιώσεις μας συγκλίνουν. Με άλλα λόγια, σε ένα πλήρες δίκτυο, ακόμη κι αν οι κόμβοι δε γνωρίζουν σε κάθε βήμα την τελευταία άποψη όλων των γειτόνων τους, αφού κάθε φορά γνωστοποιείται σε αυτούς μόνο η γνώμη ενός από αυτούς, αλλά διατηρούν αρκετή παλιά πληροφορία, το δίκτυο συνεχίζει να οδηγείται σε σύγκλιση. Επίσης, όπως ήταν αναμενόμενο, παρατηρούμε ότι αύξηση του μεγέθους του πληθυσμού οδηγεί σε αύξηση των απαιτούμενων επαναλήψεων μέχρι να επιτευχθεί η σύγκλιση αυτή.

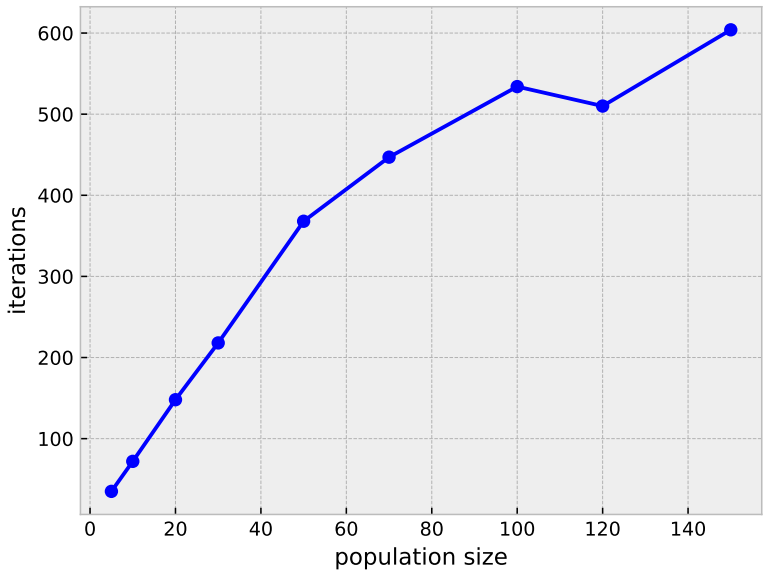
Σχήμα 7.1



Σχήμα 7.2



Σχήμα 7.3



Σχήμα 7.4

7.3 Σταθερού μεγέθους πλήρης γράφος - Μεταβλητού μεγέθους μνήμη `window`

Περιγραφή

Στην υποενότητα αυτή θα παρουσιάσουμε τα αποτελέσματα των προσομοιώσεων σε κοινωνικά δίκτυα που έχουν μορφή πλήρη γράφου σταθερού μεγέθους πληθυσμού $n = 1000$ και κάθε κόμβος διατηρεί στη μνήμη του μόνο τις τελευταίες m τιμές που λαμβάνει.

```
PopulationSize:1000
NetworkX:complete_graph
AgentMemory:[('window', <m>)]
```

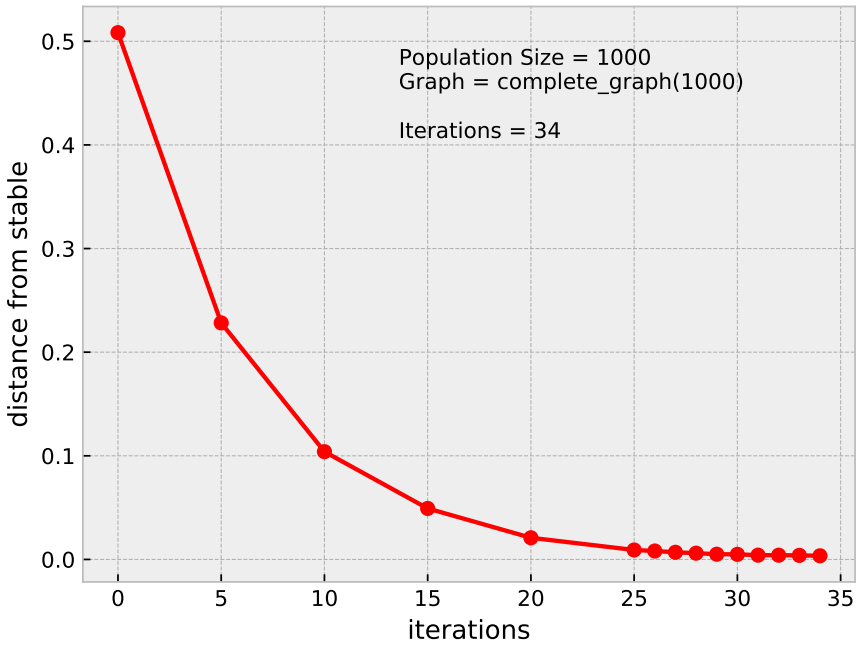
Αποτελέσματα

Οι γραφικές παραστάσεις 7.5, 7.6, 7.7 και 7.8 αποτυπώνουν την απόσταση του δικτύου από την κατάσταση ισορροπίας ως προς τον τρέχοντα αριθμό της επανάληψης για μέγεθος μνήμης `window` $m = 10$, $m = 50$, $m = 300$ και $m = 500$ αντίστοιχα.

Η γραφική παράσταση 7.9 αποτυπώνει τη μεταβολή του πλήθους των επαναλήψεων που απαιτούνται για τη σύγκλιση στην κατάσταση ισορροπίας καθώς αυξάνεται το μέγεθος της μνήμης `window`.

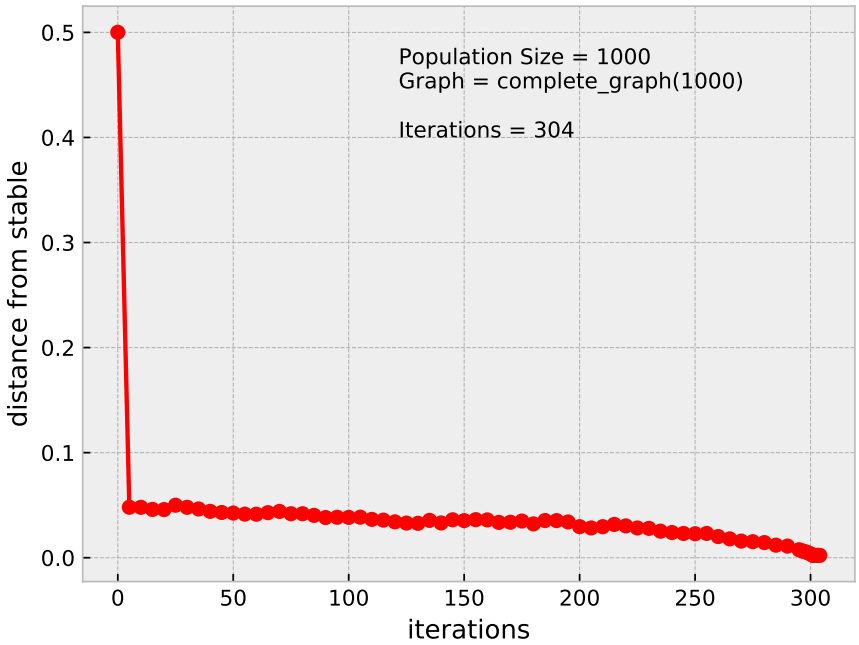
Συμπέρασματα

Μελετάμε εδώ την περίπτωση που οι κόμβοι του κοινωνικού δικτύου, που έχει τη μορφή πλήρους γράφου, χρησιμοποιούν τη διαθέσιμη σε αυτούς μνήμη για να αποθηκεύσουν τις τελευταίες απόψεις που τους έχουν γνωστοποιηθεί από τους γείτονές τους. Παρατηρούμε ότι, για ένα μέγεθος γράφου 1000 κόμβων, η σύγκλιση επιτυγχάνεται πολύ γρήγορα για την περίπτωση που οι κόμβοι διατηρούν τις τελευταίες 10 γνώμες που τους έχουν γνωστοποιηθεί. Αυξάνοντας το μέγεθος της μνήμης σε 50, και στη συνέχεια σε 300 και 500, παρατηρούμε ότι η σύγκλιση επιτυγχάνεται όλο και πιο αργά. Τα αποτελέσματα των προσομοιώσεών μας συνάδουν με την παρατήρηση ότι όσο μεγαλώνει το ιστορικό που διατηρείται από τους κόμβους, τόσο πιο δύσκολο είναι από αυτούς να απομακρυνθούν από τις αρχικές συνθήκες του δικτύου και να αρχίσουν να προσεγγίζουν την κατάσταση ισορροπίας.



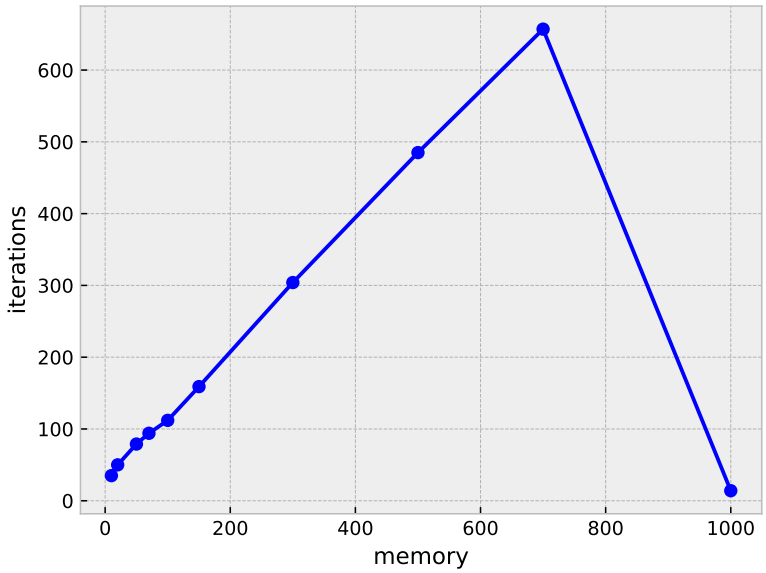
Σχήμα 7.5

Σχήμα 7.6



Σχήμα 7.7

Σχήμα 7.8


iterations
memory
0
100
200
300
400
500
600
0
200
400
600
800
1000

Σχήμα 7.9

7.4 Σταθερού μεγέθους κανονικός γράφος - Μεταβλητού μεγέθους μνήμη `window`

Περιγραφή

Στην υποενότητα αυτή θα παρουσιάσουμε τα αποτελέσματα των προσομοιώσεων σε κοινωνικά δίκτυα που έχουν μορφή 100-κανονικού γράφου σταθερού μεγέθους πληθυσμού $n = 1000$, όπου κάθε κόμβος διατηρεί στη μνήμη του μόνο τις τελευταίες m τιμές που λαμβάνει.

```
PopulationSize:1000
NetworkX:random_regular_graph 100
AgentMemory:[('window', <m>)]
```

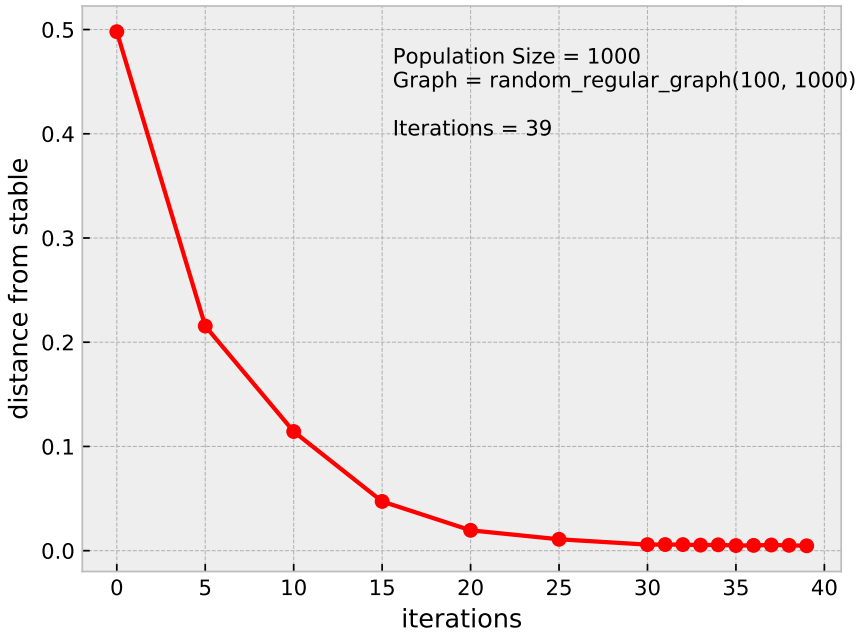
Αποτελέσματα

Οι γραφικές παραστάσεις 7.10, 7.11, 7.12 και 7.13 αποτυπώνουν την απόσταση του δικτύου από την κατάσταση ισορροπίας ως προς τον τρέχοντα αριθμό της επανάληψης για μέγεθος μνήμης `window` $m = 10$, $m = 50$, $m = 300$ και $m = 500$ αντίστοιχα.

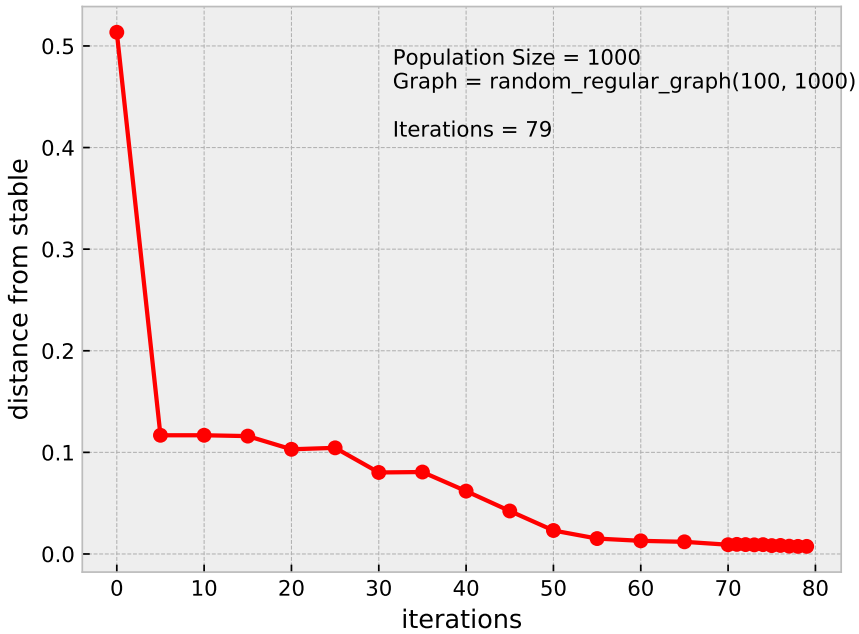
Η γραφική παράσταση 7.14 αποτυπώνει τη μεταβολή του πλήθους των επαναλήψεων που απαιτούνται για τη σύγκλιση στην κατάσταση ισορροπίας καθώς αυξάνεται το μέγεθος της μνήμης `window`.

Συμπέρασματα

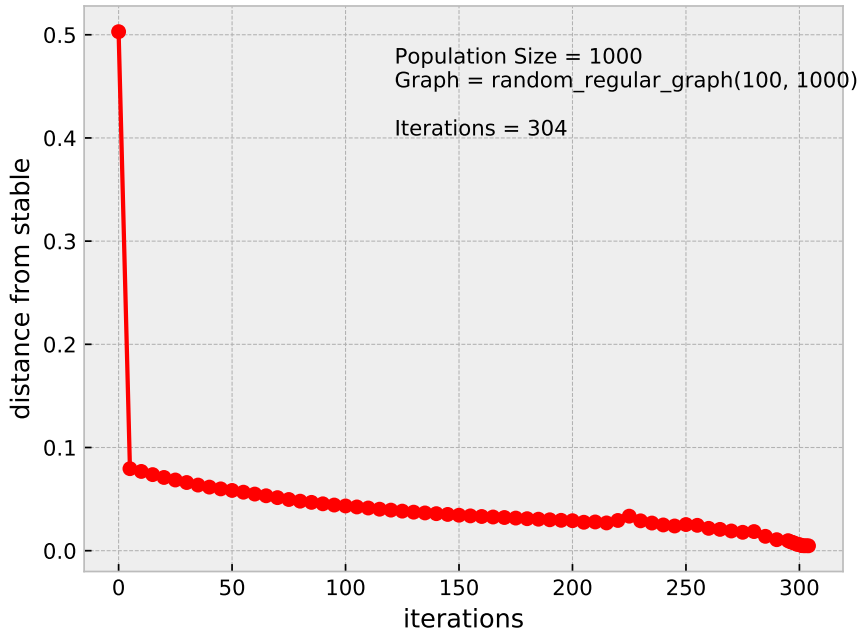
Όπως και στην προηγούμενη ενότητα, μελετάμε την περίπτωση που οι κόμβοι ενός κοινωνικού δικτύου χρησιμοποιούν τη διαθέσιμη σε αυτούς μνήμη για να αποθηκεύσουν τις τελευταίες απόψεις που τους έχουν γνωστοποιηθεί από τους γείτονές τους, ωστόσο εδώ εξετάζουμε κοινωνικά δίκτυα που έχουν τη μορφή σταθερού μεγέθους κανονικού γράφου. Για μέγεθος πληθυσμού 1000 όπου κάθε κόμβος έχει 100 γείτονες, ξεκινάμε από την περίπτωση που οι κόμβοι διατηρούν τις τελευταίες 10 γνώμες που τους έχουν γνωστοποιηθεί και παρατηρούμε ότι το δίκτυο συγκλίνει. Αυξάνοντας τη μνήμη σε 50, και στη συνέχεια σε 300 και 500, παρατηρούμε ότι η σύγκλιση επιτυγχάνεται όλο και πιο αργά. Κι εδώ, τα αποτελέσματα των προσομοιώσεών μας συνάδουν με την παρατήρηση ότι όσο μεγαλώνει το ιστορικό που διατηρείται από τους κόμβους, τόσο πιο δύσκολο είναι από αυτούς να απομακρυνθούν από τις αρχικές συνθήκες του δικτύου και να αρχίσουν να προσεγγίζουν την κατάσταση ισορροπίας.



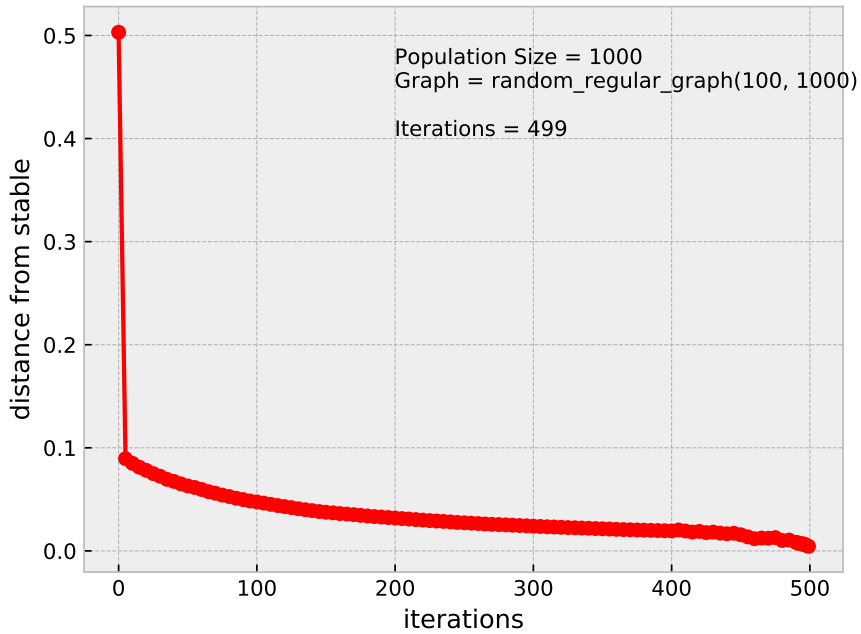
Σχήμα 7.10



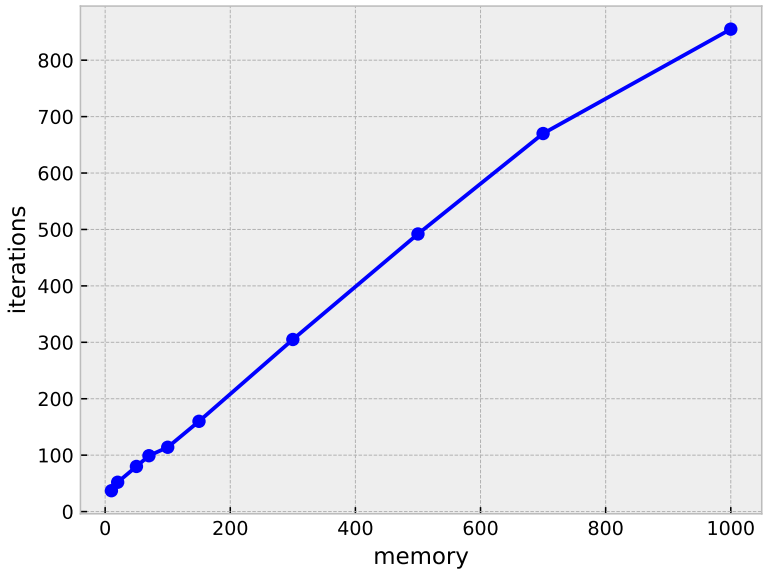
Σχήμα 7.11



Σχήμα 7.12



Σχήμα 7.13


iterations
memory
0
100
200
300
400
500
600
700
800
0
200
400
600
800
1000

Σχήμα 7.14

7.5 Συνδυασμός μνήμης `window` και `exclusive`

Περιγραφή

Στην υποενότητα αυτή θα παρουσιάσουμε, για διάφορα είδη κοινωνικών δικτύων μεγέθους πληθυσμού 100, τα αποτελέσματα των προσομοιώσεων εάν οι κόμβοι διατηρούν στη μνήμη τους μόνο τις 100 τελευταίες τιμές που έχουν λάβει (μόνο μνήμη `window` 100 θέσεων)

```
AgentMemory: [('window', 100)]
```

ή, επιπλέον, διατηρούν στη μνήμη τους την τελευταία τιμή που έχουν λάβει για κάθε έναν από το πολύ 10 προκαθορισμένους γείτονές τους (μνήμη `exclusive` 10 θέσεων).

```
AgentMemory: [('exclusive_memory', 10), ('window', 100)]
```

Αποτελέσματα

Αρχικά, προσομοιώσαμε και τις δύο περιπτώσεις σε ένα δίκτυο στο οποίο οι κόμβοι είναι συνδεδεμένοι σε κύκλο.

```
PopulationSize: 100
NetworkX: cycle_graph
```

Για την περίπτωση που οι κόμβοι έχουν μόνο μία μνήμη `window` 100 θέσεων, η προσομοίωση **συνέχιζε να τρέχει** για κάποιο μη ευκαταφρόνητο διάστημα. Μετά την προσθήκη μίας μνήμης `exclusive` 10 θέσεων προέκυψε η γραφική παράσταση 7.15.

Στη συνέχεια, προσομοιώσαμε τις ίδιες δύο περιπτώσεις σε ένα δίκτυο στο οποίο οι κόμβοι είναι συνδεδεμένοι σε μονοπάτι.

```
PopulationSize: 100
NetworkX: path_graph
```

Για την περίπτωση που οι κόμβοι έχουν μόνο μία μνήμη `window` 100 θέσεων, η προσομοίωση **συνέχιζε να τρέχει** για κάποιο μη ευκαταφρόνητο διάστημα. Μετά την προσθήκη μίας μνήμης `exclusive` 10 θέσεων προέκυψε η γραφική παράσταση 7.16.

Προσομοιώσαμε, επίσης, τις ίδιες δύο περιπτώσεις σε ένα 3-κανονικό δίκτυο.

```
PopulationSize: 100
NetworkX: random_regular_graph 3
```

Για την περίπτωση που οι κόμβοι έχουν μόνο μία μνήμη `window` 100 θέσεων, η προσομοίωση **συνέχιζε να τρέχει** για κάποιο μη ευκαταφρόνητο διάστημα. Μετά την προσθήκη μίας μνήμης `exclusive` 10 θέσεων προέκυψε η γραφική παράσταση 7.17.

Τέλος, προσομοιώσαμε τις ίδιες δύο περιπτώσεις σε έναν πλήρη γράφο.

```
PopulationSize: 100
NetworkX: complete_graph
```

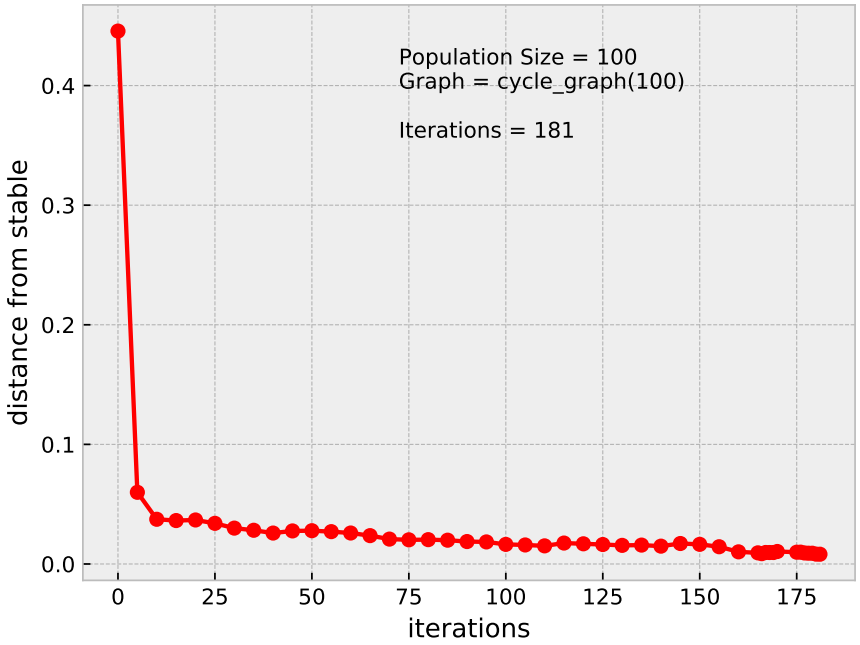
Για την περίπτωση που οι κόμβοι έχουν μόνο μία μνήμη `window` 100 θέσεων προέκυψε η γραφική παράσταση 7.18, ενώ με την προσθήκη μίας μνήμης `exclusive` 10 θέσεων προέκυψε η γραφική παράσταση 7.19.

Συμπέρασματα

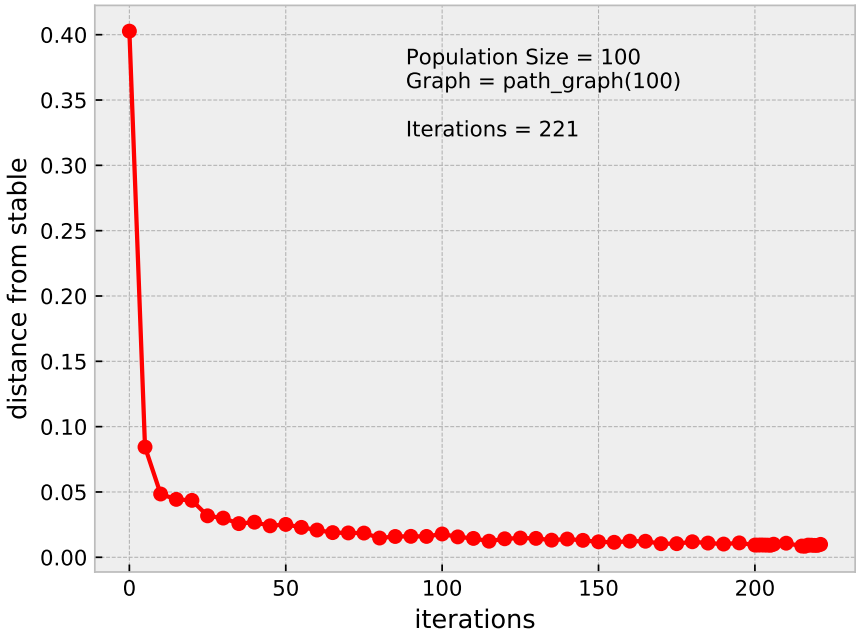
Αρχικά, εξετάζουμε την περίπτωση που οι κόμβοι είναι συνδεδεμένοι σε κύκλο μεγέθους 100 και παρατηρούμε ότι, στην περίπτωση που αυτοί έχουν μόνο μία μνήμη `window`, η προσομοίωση δεν τερματίζει για ένα λογικό διάστημα. Με την προσθήκη σε κάθε κόμβο μίας μνήμης `exclusive` όση και το πλήθος των γειτόνων του κόμβου, η κατάσταση αλλάζει και το δίκτυο συγκλίνει. Η συμπεριφορά αυτή μπορεί να εξηγηθεί από την παρατήρηση ότι στην πρώτη περίπτωση οι 100 θέσεις μνήμης για

τους 2 γείτονες κάθε κόμβου δεν επιτρέπουν στο δίκτυο να απομακρυνθεί αρκετά από τις αρχικές του συνθήκες, ενώ η προσθήκη της `exclusive` μνήμης αναγκάζει τους κόμβους να δώσουν μεγαλύτερο βάρος στις πιο πρόσφατες γνώμες των γειτόνων τους. Οι ίδιες ακριβώς παρατηρήσεις προκύπτουν και στην περίπτωση που οι κόμβοι είναι συνδεδεμένοι σε ένα μονοπάτι ίδιου μεγέθους ή σε ένα γράφο με μικρό βαθμό.

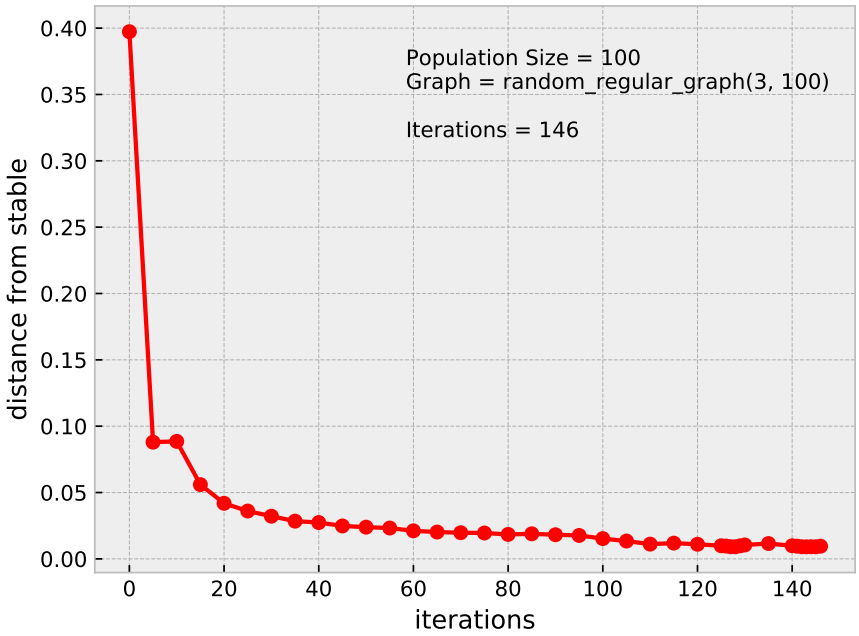
Στην περίπτωση που οι κόμβοι είναι συνδεδεμένοι σε έναν πλήρη γράφο, ωστόσο, το δίκτυο συγκλίνει ακόμη και αν οι κόμβοι έχουν μόνο μία μνήμη `window`. Η συμπεριφορά αυτή μπορεί να εξηγηθεί από την παρατήρηση ότι εδώ το μέγεθος της μνήμης `window` είναι ίσο με το μέγεθος του γράφου κι ο γράφος είναι πλήρης, επομένως κάθε κόμβος έχει περίπου τόση μνήμη όσους και γείτονες. Λόγω της τυχαιότητας με την οποία επιλέγεται ο γείτονας με τον οποίο θα αλληλεπιδράσει ένας κόμβος σε κάθε επανάληψη, βλέπουμε ότι πιθανότατα η μνήμη `window` δε θα αποθηκεύει πληροφορία για τους ίδιους λίγους κόμβους, αλλά θα προσεγγίζει τη συμπεριφορά μίας μνήμης `exclusive`. Στην περίπτωση του πλήρους γράφου, η προσθήκη μίας `exclusive` μνήμης επιτρέπει στο δίκτυο να συγκλίνει πιο γρήγορα. Η συμπεριφορά αυτή ίσως εξηγείται από το γεγονός ότι το δίκτυο επιμένει σε κάποιες πιο πρόσφατες τιμές ή ίσως ότι εδώ κάθε κόμβος επιλέγει να επιμένει λίγο περισσότερο σε ένα υποσύνολο των γειτόνων του κι αποκτά έτσι ένα καλύτερο σημείο αναφοράς.



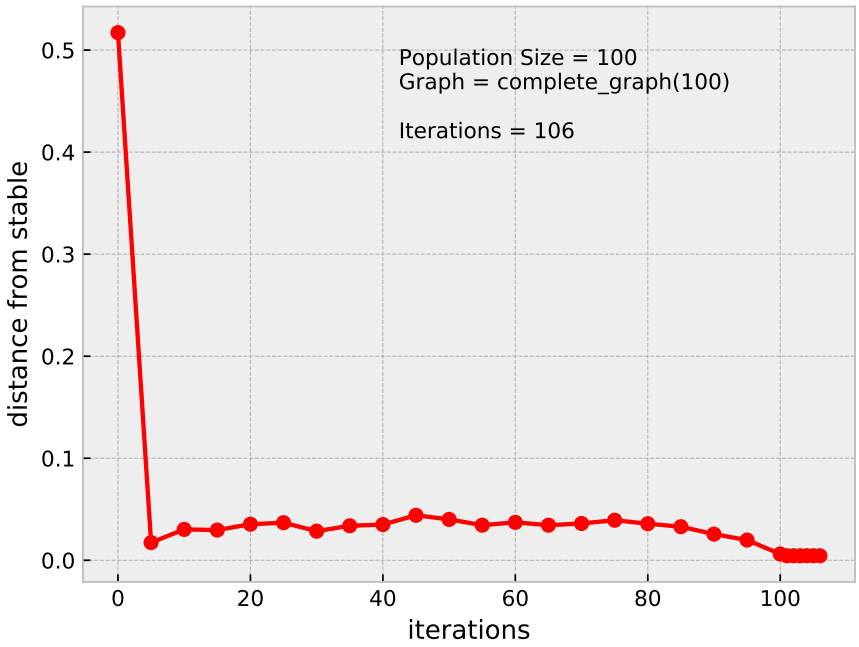
Σχήμα 7.15



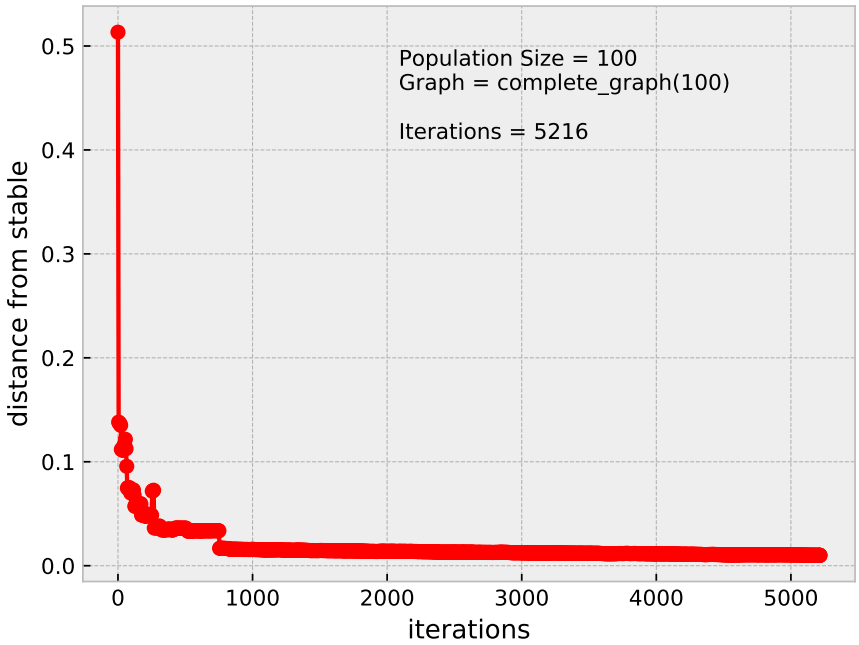
Σχήμα 7.16



Σχήμα 7.17



Σχήμα 7.18



Σχήμα 7.19

7.6 Συμπεράσματα

Συνολικά, παρατηρούμε ότι στις περισσότερες από τις προσομοιώσεις που παρουσιάσαμε το δίκτυο της παραλλαγής του μοντέλου των Friedkin-Johnsen που χρησιμοποιήσαμε, με περιορισμένη μνήμη στους κόμβους και επικοινωνία μεταξύ τους, συνεχίζει να συγκλίνει. Ωστόσο, όπως δείξαμε, μία τέτοια παραλλαγή αντιστοιχεί στην πραγματικότητα σε μία εφαρμογή μίας αντίστοιχης παραλλαγής της κατανεμημένης gradient descent μεθόδου, με περιορισμένη μνήμη στους κόμβους και επικοινωνία μεταξύ τους. Τα αποτελέσματα των προσομοιώσεών μας, λοιπόν, αποτελούν θετικές ενδείξεις ότι μπορούν να υπάρξουν τέτοιες παραλλαγές της κατανεμημένης gradient descent που να συνεχίζουν να επιτυγχάνουν σύγκλιση, ακόμη και σε μεγάλα δίκτυα με μικρές δυνατότητες αποθήκευσης πληροφορίας και περιορισμένη δυνατότητα επικοινωνίας ανάμεσα στο δίκτυο.

Βιβλιογραφία

[Abeb18] Rediet Abebe, Jon M. Kleinberg, David C. Parkes and Charalampos E. Tsourakakis, "Opinion Dynamics with Varying Susceptibility to Persuasion", *CoRR*, vol. abs/1801.07863, 2018.

[Alfo05] John R. Alford, Carolyn L. Funk, John R. Hibbing, John R. Alford and Carolyn L. Funk, "Are political orientations genetically transmitted", *American Political Science Review*, pp. 153–167, 2005.

[Bert97] Dimitri P. Bertsekas and John N. Tsitsiklis, *Parallel and Distributed Computation: Numerical Methods*, Athena Scientific, 1997.

[Bhat13] Arnab Bhattacharyya, Mark Braverman, Bernard Chazelle and Huy L. Nguyen, "On the convergence of the Hegselmann-Krause system", in *Innovations in Theoretical Computer Science, ITCS '13, Berkeley, CA, USA, January 9-12, 2013*, pp. 61–66, 2013.

[Bhaw13] Kshipra Bhawalkar, Sreenivas Gollapudi and Kamesh Munagala, "Coevolutionary opinion formation games", in *Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*, pp. 41–50, 2013.

[Bilo16] Vittorio Bilò, Angelo Fanelli and Luca Moscardelli, "Opinion Formation Games with Dynamic Social Influences", in Yang Cai and Adrian Vetta, editors, *Web and Internet Economics*, pp. 444–458, Berlin, Heidelberg, 2016, Springer Berlin Heidelberg.

[Bind11] David Bindel, Jon M. Kleinberg and Sigal Oren, "How Bad is Forming Your Own Opinion?", in *IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS 2011, Palm Springs, CA, USA, October 22-25, 2011*, pp. 57–66, 2011.

[Blum06] Avrim Blum, Eyal Even-Dar and Katrina Ligett, "Routing Without Regret: On Convergence to Nash Equilibria of Regret-minimizing Algorithms in Routing Games", in *Proceedings of the Twenty-fifth Annual ACM Symposium on Principles of Distributed Computing*, PODC '06, pp. 45–52, New York, NY, USA, 2006, ACM.

[Blum08] Avrim Blum, MohammadTaghi Hajiaghayi, Katrina Ligett and Aaron Roth, "Regret Minimization and the Price of Total Anarchy", in *Proceedings of the Fortieth Annual ACM Symposium on Theory of Computing*, STOC '08, pp. 373–382, New York, NY, USA, 2008, ACM.

[Boyd04] Stephen Boyd and Lieven Vandenberghe, *Convex Optimization*, Cambridge University Press, 2004.

[Brav17] Mario Bravo and Panayotis Mertikopoulos, "On the robustness of learning in games with stochastically perturbed payoff observations", *Games and Economic Behavior*, vol. 103, pp. 41–66, 2017.

[Brow51] George W. Brown, "Iterative Solution of Games by Fictitious Play", in T. C. Koopmans, editor, *Activity Analysis of Production and Allocation*, Wiley, New York, 1951.

[Cesa03] Nicolò Cesa-Bianchi and Gábor Lugosi, "Potential-Based Algorithms in On-Line Prediction and Game Theory", *Machine Learning*, vol. 51, no. 3, pp. 239–261, Jun 2003.

[Cesa06] Nicolo Cesa-Bianchi and Gabor Lugosi, *Prediction, Learning, and Games*, Cambridge University Press, New York, NY, USA, 2006.

[Chen16a] Po-An Chen, Yi-Le Chen and Chi-Jen Lu, "Bounds on the price of anarchy for a more general class of directed graphs in opinion formation games", *Oper. Res. Lett.*, vol. 44, no. 6, pp. 808–811, 2016.

[Chen16b] Jianqiang Cheng, Janny Leung and Abdel Lisser, "Random-payoff two-person zero-sum game with joint chance constraints", *European Journal of Operational Research*, vol. 252, no. 1, pp. 213 – 219, 2016.

[Cheu16] Yun Kuen Cheung and Richard Cole, "A Unified Approach to Analyzing Asynchronous Coordinate Descent and Tatonnement", *CoRR*, vol. abs/1612.09171, 2016.

[Chie13] Flavio Chierichetti, Jon M. Kleinberg and Sigal Oren, "On discrete preferences and coordination", in *ACM Conference on Electronic Commerce, EC '13, Philadelphia, PA, USA, June 16-20, 2013*, pp. 233–250, 2013.

[Cohe17] Johanne Cohen, Amélie Héliou and Panayotis Mertikopoulos, "Hedging Under Uncertainty: Regret Minimization Meets Exponentially Fast Convergence", in *Algorithmic Game Theory - 10th International Symposium, SAGT 2017, L'Aquila, Italy, September 12-14, 2017, Proceedings*, pp. 252–263, 2017.

[Deff00] Guillaume Deffuant, David Neau, Frédéric Amblard and Gérard Weisbuch, "Mixing beliefs among interacting agents", *Advances in Complex Systems*, vol. 3, no. 1-4, pp. 87–98, 2000.

[DeGr74] M.H. DeGroot, "Reaching a Consensus", *Journal of the American Statistical Association*, vol. 69, pp. 118–121, 1974.

[Dubh09] D.P. Dubhashi and A. Panconesi, *Concentration of Measure for the Analysis of Randomized Algorithms*, Cambridge University Press, 2009.

[Duch] John Duchi, "Stats311, Lecture Notes".

[Epit17] Markos Epitropou, Dimitris Fotakis, Martin Hoefer and Stratis Skoulakis, "Opinion Formation Games with Aggregation and Negative Influence", in *Algorithmic Game Theory - 10th International Symposium, SAGT 2017, L'Aquila, Italy, September 12-14, 2017, Proceedings*, pp. 173–185, 2017.

[Even09] Eyal Even-Dar, Yishay Mansour and Uri Nadav, "On the convergence of regret minimization dynamics in concave games", in *Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC 2009, Bethesda, MD, USA, May 31 - June 2, 2009*, pp. 523–532, 2009.

[Ferr16a] D. Ferraioli, P. Goldberg and C. Ventre, "Decentralized Dynamics for Finite Opinion Games", *Theoretical Computer Science*, vol. 648, pp. 96–115, 2016.

[Ferr16b] Diodato Ferraioli, Paul W. Goldberg and Carmine Ventre, "Decentralized Dynamics for Finite Opinion Games", *Theor. Comput. Sci.*, vol. 648, no. C, pp. 96–115, October 2016.

[Fost97] Dean P. Foster and Rakesh V. Vohra, "Calibrated Learning and Correlated Equilibrium", *Games and Economic Behavior*, vol. 21, no. 1, pp. 40 – 55, 1997.

[Fost16] Dylan J. Foster, Zhiyuan Li, Thodoris Lykouris, Karthik Sridharan and Éva Tardos, "Learning in Games: Robustness of Fast Convergence.", in Daniel D. Lee, Masashi Sugiyama, Ulrike von Luxburg, Isabelle Guyon and Roman Garnett, editors, *NIPS*, pp. 4727–4735, 2016.

[Fota16] Dimitris Fotakis, Dimitris Palyvos-Giannas and Stratis Skoulakis, "Opinion Dynamics with Local Interactions", in *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016*, pp. 279–285, 2016.

[Freu99] Yoav Freund and Robert E. Schapire, "Adaptive Game Playing Using Multiplicative Weights", *Games and Economic Behavior*, vol. 29, no. 1, pp. 79 – 103, 1999.

[Frie90] Noah E. Friedkin and Eugene C. Johnsen, "Social influence and opinions", *The Journal of Mathematical Sociology*, vol. 15, no. 3-4, pp. 193–206, 1990.

[Ghad14] Javad Ghaderi and R. Srikant, "Opinion dynamics in social networks with stubborn agents: Equilibrium and convergence rate", *Automatica*, vol. 50, no. 12, pp. 3209–3215, 2014.

[Gion13] Aristides Gionis, Evimaria Terzi and Panayiotis Tsaparas, "Opinion Maximization in Social Networks", in *Proceedings of the 13th SIAM International Conference on Data Mining, May 2-4, 2013. Austin, Texas, USA.*, pp. 387–395, 2013.

[Gras16] Markus Grasmair, "Basic Properties of Convex Functions", 2016. Available from `https://wiki.math.ntnu.no/_media/tma4180/2016v/note2.pdf`.

[Haza07] Elad Hazan, Amit Agarwal and Satyen Kale, "Logarithmic regret algorithms for online convex optimization", *Machine Learning*, vol. 69, no. 2, pp. 169–192, Dec 2007.

[Haza16] Elad Hazan, "Introduction to Online Convex Optimization", *Found. Trends Optim.*, vol. 2, no. 3-4, pp. 157–325, August 2016.

[Hegs02] R. Hegselmann and U. Krause, "Opinion Dynamics and Bounded Confidence Models, Analysis, and Simulation", *Journal Artificial Societies and Social Simulation*, vol. 5, 2002.

[Heli17] Amélie Héliou, Johanne Cohen and Panayotis Mertikopoulos, "Learning with Bandit Feedback in Potential Games", in *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pp. 6372–6381, 2017.

[Jack08] M.O. Jackson, *Social and Economic Networks*, Princeton University Press, 2008.

[Klei09] Robert Kleinberg, Georgios Piliouras and Eva Tardos, "Multiplicative Updates Outperform Generic No-regret Learning in Congestion Games: Extended Abstract", in *Proceedings of the Forty-first Annual ACM Symposium on Theory of Computing*, STOC '09, pp. 533–542, New York, NY, USA, 2009, ACM.

[Klei11] Robert Kleinberg, Georgios Piliouras and Éva Tardos, "Load balancing without regret in the bulletin board model", *Distributed Computing*, vol. 24, no. 1, pp. 21–29, Sep 2011.

[Krac09] David Krackhardt, "A Plunge into Networks", *Science*, vol. 326, no. 5949, pp. 47–48, 2009.

[Mert17] Panayotis Mertikopoulos and Mathias Staudigl, "Convergence to nash equilibrium in continuous games with noisy first-order feedback", in *56th IEEE Annual Conference on Decision and Control, CDC 2017, Melbourne, Australia, December 12-15, 2017*, pp. 5609–5614, 2017.

[Musc17] Cameron Musco, Christopher Musco and Charalampos E. Tsourakakis, "Minimizing Polarization and Disagreement in Social Networks", *CoRR*, vol. abs/1712.09948, 2017.

[Serg00] Hart Sergiu and Mas-Colell Andreu, "A Simple Adaptive Procedure Leading to Correlated Equilibrium", 2000.

[Syrg15] Vasilis Syrgkanis, Alekh Agarwal, Haipeng Luo and Robert E. Schapire, "Fast Convergence of Regularized Learning in Games", in *NIPS*, pp. 2989–2997, 2015.

[Tsyb08] Alexandre B. Tsybakov, *Introduction to Nonparametric Estimation*, Springer, New York ; London, 1 edition edition, November 2008.

[Wald39] Abraham Wald, "Contributions to the Theory of Statistical Estimation and Testing Hypotheses", *The Annals of Mathematical Statistics*, vol. 10, no. 4, pp. 299–326, December 1939.

[Yild13] Mehmet Ercan Yildiz, Asuman E. Ozdaglar, Daron Acemoglu, Amin Saberi and Anna Scaglione, "Binary Opinion Dynamics with Stubborn Agents", *ACM Trans. Economics and Comput.*, vol. 1, no. 4, pp. 19:1–19:30, 2013.

[Yu97] Bin Yu, "Assouad, Fano, and Le Cam", in *Festschrift for Lucien Le Cam*, pp. 423–435, Springer, New York, NY, 1997.

[Zhou17] Yichi Zhou, Jialian Li and Jun Zhu, "Identify the Nash Equilibrium in Static Games with Random Payoffs", in *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, pp. 4160–4169, 2017.

[Zink03] Martin Zinkevich, "Online Convex Programming and Generalized Infinitesimal Gradient Ascent", in *Proceedings of the Twentieth International Conference on International Conference on Machine Learning*, ICML'03, pp. 928–935, AAAI Press, 2003.

[Σκ14] Ευστράτιος-Παντελεήμων Χ. Σκουλάκης, "Παίγνια Διαμόρφωσης Άποψης σε Κοινωνικά Δίκτυα", διπλωματική εργασία, Εθνικό Μετσόβιο Πολυτεχνείο, Οκτώβριος 2014.