ΟΙΚΟΝΟΜΙΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ - ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ
Θεωρία Παιγνίων και Αποφάσεων
Διδάσκων: Ε. Μαρκάκης, Εαρινό εξάμηνο 2020

Προγραμματιστική εργασία **Προθεσμία: 15/6 (θα επανοριστεί)**

Η εργασία είναι προαιρετική, με αξία μισή μονάδα, και μπορείτε να δουλέψετε είτε ατομικά, είτε σε ομάδες των 2. Αφορά την υλοποίηση ενός απλοϊκού αλγορίθμου μάθησης (*learning algorithms*) για την εύρεση σημείων ισορροπίας κατά Nash σε κάποιες κατηγορίες παιγνίων ψηφοφορίας.

Voting games. Τα παίγνια αυτά μοντελοποιούν εκλογικές διαδικασίες (συνήθως όχι βουλευτικές εκλογές, αλλά περισσότερο εκλογές για την επιλογή συμβουλίων, επιτροπών, καθώς και εκλογές για πολλά άλλα προβλήματα λήψης αποφάσεων, που πλέον διενεργούνται και ηλεκτρονικά, π.χ. Doodle polls, SurveyMonkey, etc).

Πιο συγκεκριμένα, έστω ότι έχουμε ένα σύνολο από n ψηφοφόρους, ας τους ονομάσουμε $1, 2, ..., n$, και ένα σύνολο από m υποψηφίους $C = \{c_1, c_2, ..., c_m\}$. Οι υποψήφιοι μπορεί να αντιστοιχούν σε υποψήφια μέλη για μια επιτροπή ή σε υποψήφια εστιατόρια που σκέφτεται μια παρέα πριν πάει για φαγητό ή υποψήφιες ταινίες για τις οποίες θέλουμε να βγάλουμε μια κατάταξη, κτλ. Οι προτιμήσεις κάθε ψηφοφόρου εκφράζονται από μια ολική διάταξη, π.χ. με τη διάταξη $c_2 \succ_i c_3 \succ_i c_5 \succ_i ... \succ_i c_4$, εννοούμε ότι για τον ψηφοφόρο i, η πρώτη του προτίμηση είναι ο υποψήφιος c_2, η δεύτερη προτίμηση είναι ο c_3, μετά ακολουθεί ο c_5, και η τελευταία του προτίμηση είναι ο c_4. Θα επικεντρωθούμε στους εξής 2 αρκετά δημοφιλείς εκλογικούς κανόνες:

1. **Plurality.** Στον κανόνα αυτό, κάθε ψηφοφόρος υποβάλλει ως ψήφο έναν υποψήφιο. Αν οι παίκτες δεν σκέφτονται στρατηγικά, θα υποβάλουν την 1η τους προτίμηση. Ενδέχεται βέβαια ένας παίκτης να πει ψέμματα, αν βλέπει ότι δεν μπορεί να εκλεγεί η πρώτη του προτίμηση, ώστε να ενισχύσει π.χ. την 2η του προτίμηση. Ο αλγόριθμος απλά θεωρεί ότι αυτό που υπέβαλε ο κάθε παίκτης είναι όντως η πρώτη του προτίμηση, αφού δεν μπορεί να ξέρει τις πραγματικές προτιμήσεις. Το σκορ κάθε υποψηφίου είναι ο αριθμός των ψήφων που πήρε και νικητής στις εκλογές είναι ο υποψήφιος με τις περισσότερες ψήφους.

2. **Borda.** Στον κανονα αυτό, ζητείται από κάθε ψηφοφόρο να υποβάλει μια διάταξη με τις προτιμήσεις του για όλους τους υποψηφίους. Αν το πλήθος των υποψηφίων είναι m, τότε ενας υποψήφιος λαμβάνει

   - $m-1$ μονάδες από κάθε ψηφοφόρο που τον έχει στην ψήφο του ως 1η προτίμηση.
   - $m-2$ μονάδες από κάθε ψηφοφόρο που τον έχει στην ψήφο του ως 2η προτίμηση.
   - κ.ο.κ....
   - 0 μονάδες από κάθε ψηφοφόρο που τον έχει στην τελευταία θέση.

Το τελικό σκορ κάθε υποψηφίου είναι το άθροισμα από τις μονάδες που παίρνει από κάθε ψηφοφόρο, και νικητής είναι ο υποψήφιος με το μεγαλύτερο τελικό σκορ.

Και στους 2 κανόνες, αν υπάρξει ισοβαθμία, θεωρούμε για απλότητα ότι νικητής είναι ο υποψήφιος με το χαμηλότερο δείκτη (δηλαδή σε ισοβαθμία μεταξύ των υποψηφίων c_3 και c_5, νικητής θα είναι ο c_3).

Παράδειγμα. Έστω ότι έχουμε 4 ψηφοφόρους και 4 υποψηφίους, και οι πραγματικές προτιμήσεις είναι ως εξής:

- Ψηφοφόρος 1: $c_1 \succ_1 c_3 \succ_1 c_2 \succ_1 c_4$
- Ψηφοφόρος 2: $c_2 \succ_2 c_1 \succ_2 c_3 \succ_2 c_4$
- Ψηφοφόρος 3: $c_2 \succ_3 c_1 \succ_3 c_4 \succ_3 c_3$
- Ψηφοφόρος 4: $c_3 \succ_4 c_1 \succ_4 c_4 \succ_4 c_2$

Αν οι παίκτες είναι ειλικρινείς, τότε σύμφωνα με τον κανόνα Plurality, νικητής θα είναι ο υποψήφιος c_2. Αν όμως χρησιμοποιήσουμε τον κανόνα Borda, τότε ο υποψήφιος c_1 θα έχει τελικό σκορ 9 (3 μονάδες από τον ψηφοφόρο 1, και 2 μονάδες από τους υπόλοιπους ψηφοφόρους), ενώ ο c_2 θα έχει σκορ 7. Άρα νικητής θα είναι ο c_1.

Best Response Dynamics

Και οι 2 κανόνες παραπάνω χρησιμοποιούνται ευρύτατα σε πολλές εφαρμογές. Οι ψηφοφόροι όμως δεν έχουν πάντα κίνητρο να δηλώσουν απλά τις πραγματικές τους προτιμήσεις. Σε περιπτώσεις όπου δεν μπορεί να εκλεγεί η πρώτη τους προτίμηση, ενδέχεται να υπάρχουν κίνητρα να δηλώσουν π.χ. την 2η προτίμηση τους ως πρώτη για να εκλεγεί τουλάχιστον η 2η προτίμηση. Στο παραπάνω παράδειγμα, με τον κανόνα Plurality, ο ψηφοφόρος 4 έχει κίνητρο π.χ. να πει ψέμματα και να δηλώσει τον c_1 ως 1η του προτίμηση, καθώς βλέπει ότι ο c_3 δεν μπορεί να εκλεγεί, και ο c_2 που εκλέγεται αν είναι ειλικρινής, είναι η τελευταία του προτίμηση. Τέτοια φαινόμενα παρατηρούνται ειδικά σε σενάρια όπου η ψηφοφορία μπορεί να είναι σε εξέλιξη για μεγάλο χρονικό διάστημα και οι ψηφοφόροι μπορούν να αλλάζουν την ψήφο τους όποτε θελήσουν (π.χ. σε ηλεκτρονικές ψηφοφορίες). Μπορείτε για περισσότερες πληροφορίες σχετικά με την στρατηγική συμπεριφορά των ψηφοφόρων, να δείτε στο Internet για το θεώρημα των Gibbard-Satterthwaite.

Με βάση τις παραπάνω παρατηρήσεις, μπορούμε να μοντελοποιήσουμε τέτοιες ψηφοφορίες ως παίγνια όπου οι παίκτες είναι οι ψηφοφόροι και οι διαθέσιμες στρατηγικές του κάθε παίκτη είναι όλες οι δυνατές ψήφοι (στον κανόνα Plurality επομένως υπάρχουν m διαθέσιμες στρατηγικές, ενώ στον κανόνα Borda υπάρχουν $m!$ πιθανές στρατηγικές αφού μια ψηφος είναι μια διάταξη των υποψηφίων).

Σκοπός της εργασίας είναι να υλοποιήσετε και να αξιολογήσετε την παρακάτω διαδικασία, η οποία αναφέρεται ως *best response dynamics* και προσομοιώνει το πώς θα μπορούσε ένα τέτοιο παίγνιο να οδηγηθεί σε σημείο ισορροπίας, μέσω της έυρεσης βέλτιστης απόκρισης σε κάθε βήμα του αλγορίθμου. Τέτοιοι αλγόριθμοι έχουν μελετηθεί αρκετά στη βιβλιογραφία των τελευταίων ετών για πολλές κατηγορίες παιγνίων.

Περιγραφή αλγορίθμου. Ο αλγόριθμος εξελίσσεται σε γύρους και στον πρώτο γύρο, όλοι οι παίκτες δηλώνουν τις πραγματικές τους προτιμήσεις. Με βάση τις ψήφους αυτές, υπάρχει ένας τρέχων νικητής (που καθορίζεται απο τον κανόνα που έχουμε επιλέξει να χρησιμοποιήσουμε, π.χ. Plurality ή Borda). Σε καθένα από τους επόμενους γύρους, ελέγχουμε αν υπάρχει κάποιος παίκτης ο οποίος έχει κίνητρο να αλλάξει την ψηφο του. Ουσιαστικά δηλαδή κάθε ψηφοφόρος βλέπει τον τρέχοντα νικητή, και σκέφτεται αν υπάρχει περίπτωση αλλάζοντας την ψήφο του (και θεωρώντας ότι οι υπόλοιποι δεν θα αλλάξουν), να εκλεγεί κάποιος ο οποιος να βρίσκεται υψηλότερα στη διάταξη που αντιστοιχεί στις *πραγματικές* του προτιμήσεις. Άρα πρέπει να υπολογίσει την βέλτιστη απόκριση του με βάση το τι έχουν ψηφίσει οι υπόλοιποι παίκτες και να συγκρίνει με το τρέχον αποτέλεσμα. Αν δεν υπάρχει κανένας παίκτης που να ευνοείται από μια αλλαγή ψήφου, τότε ο αλγόριθμος σταματάει και νικητής των εκλογών είναι ο τρέχων νικητής (και ταυτόχρονα έχουμε φτάσει σε σημείο ισορροπίας κατά Nash, αφού κανένας ψηφοφόρος δεν θέλει να αλλάξει στρατηγική). Αν υπάρχει έστω και ένας ψηφοφόρος που θέλει να αλλάξει, τότε επιλέγουμε έναν από αυτούς, ο ψηφοφόρος που επιλέχθηκε αλλάζει την ψήφο του, με βάση το ποια είναι η βέλτιστη απόκριση, και συνεχίζουμε στον επόμενο γύρο, όπου επαναλαμβάνουμε την ίδια διαδικασία. Δεν έχει σημασία ποιον επιλέγουμε σε κάθε γύρο, όταν υπάρχουν πολλοί που θέλουν να αλλάξουν. Μπορείτε να επιλέγετε τυχαία, ή απλά να επιλέγετε τον πρώτο που θα βρείτε ότι θέλει να αλλάξει μέσα στο loop που θα κάνετε τον έλεγχο.

Στην εργασία σας θα πρέπει να υλοποιήσετε τα εξής:

1. Αρχικά, υλοποιήστε τον αλγόριθμο Best Response Dynamics, για τον κανόνα Plurality και για τον κανόνα Borda. Το πρόγραμμά σας θα πρέπει να δέχεται ως είσοδο από αρχείο (καθορίστε εσείς σε τι μορφή ακριβώς θα είναι η είσοδος) τον αριθμό των ψηφοφόρων, τον αριθμό των υποψηφίων, τον εκλογικό κανόνα (που θα είναι είτε Plurality είτε Borda), και την διάταξη του κάθε ψηφοφόρου που αντιστοιχεί στις πραγματικές του προτιμήσεις. Σχετικά με την υλοποίηση του κάθε γύρου και τον τερματισμό του προγράμματος:

    - Hint 1: Για τον τερματισμό του αλγορίθμου, ορίστε στο πρόγραμμά σας μία παράμετρο T_{max}, η οποία θα υποδηλώνει το μέγιστο αριθμό επιτρεπτών επαναλήψεων. Ο αλγόριθμος θα τερματίζει είτε όταν φτάσουμε σε σημείο ισορροπίας και δεν υπάρχει παίκτης που να θελει να αλλάξει την ψήφο του (σε τέτοια περίπτωση λέμε ότι ο αλγόριθμος συγκλίνει), είτε όταν ολοκληρωθούν T_{max} γύροι, χωρίς να έχουμε επιτύχει σύγκλιση. Μπορείτε να θέσετε $T_{max} = 1000$ για αρχή ή μπορείτε να το προσαρμόζετε με βάση τον αριθμό των υποψηφίων και των ψηφοφόρων. Μην το θέσετε κάτω από 500 πάντως.
    - Hint 2: Σε κάθε γύρο θα πρέπει να καλείτε καποια μέθοδο που θα υπολογίζει τη βέλτιστη απόκριση ενός παίκτη, δεδομένων των στρατηγικών που έχουν επιλέξει οι άλλοι παίκτες. Για τον κανόνα Borda, ίσως σας φανεί αρχικά ότι το πρόβλημα αυτό είναι δύσκολο λόγω του μεγάλου αριθμού πιθανών στρατηγικών (που είναι ίσος με $m!$). Όμως, σκεφτείτε ότι το μόνο που θέλετε ουσιαστικά είναι να ελέγξετε αν ένας ψηφοφόρος, αλλάζοντας την ψήφο του, μπορεί να αλλάξει το εκλογικό αποτέλεσμα και να έχουμε ενα νέο νικητή που είναι πιο πάνω στις

πραγματικές του προτιμήσεις από τον τρέχοντα νικητή. Με βάση αυτό σκεφτείτε πώς θα υλοποιήσετε τον αλγόριθμο για το Borda. Αν σας δυσκολεύει, υλοποιήστε τον πρώτα μόνο για τον κανόνα Plurality.

2. Αξιολογήστε τους 2 εκλογικούς κανόνες ως προς τη σύγκλιση του best response dynamics. Για να το κάνετε αυτό θα πρέπει να παράγετε πολλά τυχαία παίγνια. Χρησιμοποιήστε τουλάχιστον 4 διαφορετικές τιμές για το n και τουλάχιστον 3 διαφορετικές τιμές για το m (με $n \geq 5$, και $m \geq 5$). Για κάθε ζευγος τιμών n, m που θα χρησιμοποιήσετε, παράγετε 20-30 τυχαία παίγνια (για κάθε παίγνιο θα πρέπει να παράξετε τυχαία permutations που θα αντιστοιχούν στις προτιμήσεις των παικτών). Σχολιάστε την επίδοση για τον κάθε κανόνα, π.χ. πόσο συχνά έχουμε σύγκλιση σε σημείο ισορροπίας, και πόσες επαναλήψεις χρειάστηκαν κατά μέσο όρο στις περιπτώσεις όπου είχαμε σύγκλιση. Παρατηρήσατε καποια διαφορά ως προς τη συχνότητα σύγκλισης ανάμεσα στους 2 εκλογικούς κανόνες;

3. Στα παίγνια όπου υπήρξε σύγκλιση σχολιάστε το εκλογικό αποτέλεσμα. Ένας τρόπος αξιολόγησης είναι ο εξής: έστω c ο νικητής στο σημείο ισορροπίας στο οποίο συνέκλινε ο αλγόριθμος, και έστω c' ο νικητής που θα προέκυπτε αν όλοι δήλωναν τις πραγματικές τους προτιμήσεις. Δείτε καταρχήν αν διαφέρουν ο c από τον c' και υπολογίστε το σκορ του c και του c' με βάση το πραγματικό προφίλ. Αν το σκορ του c' στο πραγματικό προφίλ είναι αρκετά μικρότερο από το σκορ του c, αυτό σημαίνει ότι το εκλογικό αποτέλεσμα του best response dynamics δεν έχει "καλό" κοινωνικό όφελος (η στρατηγική συμπεριφορά των ψηφοφόρων δηλαδή τους οδήγησε σε ενα σημείο ισορροπίας που στο σύνολο των ψηφοφόρων δεν είναι τόσο αποδεκτό, παρ'όλα αυτά είναι σημείο ισορροπίας αφού κανένας από μόνος του δεν μπορεί να επηρεάσει το αποτέλεσμα).

Μπορείτε να κάνετε την υλοποίηση σε Java, C, C++, Python, Matlab. Παραδοτέα για την εργασία είναι ο κώδικάς σας καθώς και μία τελική αναφορά όπου θα εξηγείτε αναλυτικά την υλοποίηση που κάνατε (1-2 σελίδες), τα πειράματα, και τα συμπεράσματά σας από τα αποτελέσματα (1-2 σελίδες). Μπορείτε αν θέλετε στην αναφορά σας να παραθέσετε και γραφικές παραστάσεις για να επεξηγήσετε περαιτέρω τα αποτελέσματα που βρήκατε.