Θεωρία Παιγνίων και Αποφάσεων

Αναφορά Προγραμματιστικής Εργασίας

Χρήστος Ρούσας - 3190180

Αυτή είναι η αναφορά παράδοσης της προγραμματιστικής εργασίας. Η εργασία αναπτύχθηκε σε Python, και αποτελείται από 2 αρχεία, το plurality.py και το borda.py.

Το πρόγραμμα τρέχει από τη γραμμή εντολών, και παίρνει σαν ορίσματα:

1. Το όνομα του αρχείου (plurality.py ή borda.py).
2. Τον τρόπο εκτέλεσης (με είσοδο από αρχείο csv που περιλαμβάνει τις προτιμήσεις, με τυχαίο τρόπο δίνοντας μόνο διαστάσεις παιγνίου, ή το testing του προγράμματος).
   Π.χ. για τον κανόνα plurality:
   - Για να τρέξει με είσοδο από csv, δίνουμε ως είσοδο την παράμετρο csv και το όνομα του αρχείου (εδώ game1.csv), ως εξής:
     `python plurality.py csv game1.csv`
   - Για να τρέξει με τυχαίο παίγνιο μεγέθους nxm (εδώ 6x4) δίνουμε ως είσοδο την παράμετρο random και τις διαστάσεις του παιγνίου, ως εξής:
     `python plurality.py random 6 4`
   - Για να τρέξουμε τις δοκιμές (testing) για την αξιολόγηση των εκλογικών κανόνων και την εξαγωγή csv αρχείου αποτελεσμάτων, δίνουμε ως είσοδο την παράμετρο test, ως εξής:
     `python plurality.py test`

   Και αντίστοιχα για το borda.

Στο πρόγραμμα, αφού είναι γραμμένο σε Python, οι δείκτες των πινάκων ξεκινάνε από το 0, οπότε π.χ. το παίγνιο που βρίσκεται στη δεύτερη σελίδα της εκφώνησης, θα αναπαρασταθεί σε αρχείο csv ως:

```
0,2,1,3
1,0,2,3
1,0,3,2
2,0,3,1
```

Όταν αναφερόμαστε σε παίγνια nxm, εννοούμε παίγνια n ψηφοφόρων με m υποψηφίους.

Υλοποίηση:

Στο πρόγραμμα υλοποιήθηκε ο αλγόριθμος Best Response Dynamics, ο οποίος παίρνει ως είσοδο έναν πίνακα (list of lists) με τις προτιμήσεις των παικτών.

Αρχικά, έχουμε ένα loop στο οποίο αρχικοποιούνται οι πίνακες players και choices, οι οποίοι περιέχουν τους παίκτες και τις αρχικές επιλογές τους αντίστοιχα. Μετά τρέχουμε τον αλγόριθμο BRD για μέγιστο αριθμό επαναλήψεων t_{max} (αρχικοποιημένη στο 1000). Σε κάθε επανάληψη, ελέγχουμε για κάθε παίκτη αν μπορεί, αλλάζοντας τις προτιμήσεις του, να αυξήσει την ωφέλειά του, και αν βρεθεί, αλλάζουμε την επιλογή του και τρέχουμε τον επόμενο γύρο, με τα αλλαγμένα δεδομένα (την αλλαγμένη επιλογή του παίκτη). Αν δε βρεθεί τέτοιος παίκτης, τότε έχουμε βρει σημείο ισορροπίας του παιγνίου και άρα επιστρέφουμε τις τελικές επιλογές των παικτών.

Υλοποίηση άλλων μεθόδων:

Η μέθοδος canChange ελέγχει αν ένας παίκτης (του οποίου η $1^η$ προτίμηση δεν νικάει ήδη) μπορεί, αλλάζοντας τις προτιμήσεις του, να αυξήσει την ωφέλειά του, δηλαδή να βγει νικητής που είναι πιο ψηλά στις προτιμήσεις του, και επιστρέφει την επιλογή του μετά από τον έλεγχο (μπορεί να είναι διαφορετική ή ίδια με την αρχική).

Στον κανόνα plurality, ελέγχουμε αν η επιλογή άλλου υποψηφίου, χαμηλότερου στις προτιμήσεις του παίκτη, μπορεί να αυξήσει την ωφέλειά του.

Στον κανόνα borda, κρατάμε την $1^η$ προτίμηση του παίκτη στην $1^η$ θέση της δήλωσής του, και βάζουμε τους υπόλοιπους υποψηφίους στη διάταξη με αύξουσα σειρά ως προς το σκορ, για να "τιμωρηθούν" οι υποψήφιοι με πιο ψηλό σκορ, που δεν θέλει να βγουν (και που είναι κοντά στο να βγουν). Ο παίκτης, και εδώ, κρατάει αυτή την επιλογή αν αυτή αυξάνει την ωφέλειά του.

Η μέθοδος calculateResult παράγει ένα διάνυσμα αποτελεσμάτων της ψηφοφορίας. Στον κανόνα plurality, το σκορ του κάθε υποψηφίου αυξάνεται κατά 1 για κάθε παίκτη που τον έχει επιλέξει. Στον κανόνα borda, το σκορ του κάθε υποψηφίου αυξάνεται κατά m-j για κάθε παίκτη που τον έχει στην j θέση της διάταξης προτιμήσεων που δηλώνει.

Η μέθοδος utilityCandidate(i) υπολογίζει τη χρησιμότητα που λαμβάνει ο παίκτης από τη νίκη του i υποψηφίου, και υπολογίζεται ως m-j, όπου m το πλήθος των υποψηφίων και j η θέση του i υποψηφίου στις προτιμήσεις του παίκτη.

Η μέθοδος csvToTable παίρνει ως είσοδο ένα αρχείο csv με τις προτιμήσεις των παικτών και παράγει έναν πίνακα προτιμήσεων, ο οποίος μπορεί να μπει ως είσοδος του αλγορίθμου.

Η μέθοδος createRandomGame(n,m,seed) παράγει ένα τυχαίο παίγνιο μεγέθους nxm, παίρνοντας ως είσοδο ένα seed παραγωγής τυχαίων αριθμών.

Πειράματα και Αξιολογήσεις:

Για την εκτέλεση των πειραμάτων και την παραγωγή των αποτελεσμάτων, υπάρχει η λειτουργία test (βλ. 1η σελίδα). Στο testing τρέχουμε δοκιμές για 5, 10, 20, 40 παίκτες, και για 5, 10, 15, 20 υποψηφίους. Για κάθε συνδυασμό αριθμού παικτών και υποψηφίων, δημιουργούμε 20 τυχαία παίγνια nxm, και εξάγουμε τα αποτελέσματα του κάθε παιγνίου, δηλαδή αριθμό παικτών, αριθμό υποψηφίων, αποτελέσματα αρχικών επιλογών, τελικές επιλογές, αρχικός νικητής, τελικός νικητής, αριθμός επαναλήψεων για εύρεση Σ.Ι.

Και για τους 2 κανόνες, σε όλα τα παίγνια που δοκιμάστηκαν, είχαμε σύγκλιση σε σημείο ισορροπίας.

Για τον κανόνα Plurality, χρειάστηκαν κατά μέσο όρο 6.5 επαναλήψεις για τη σύγκλιση σε σημείο ισορροπίας, και μεταξύ των παιγνίων στα οποία οι αρχικές επιλογές δεν ήταν σημεία ισορροπίας, ο αριθμός αυτός ήταν 9.86.

Για τον κανόνα Borda, χρειάστηκαν κατά μέσο όρο 2.81 επαναλήψεις για τη σύγκλιση σε σημείο ισορροπίας, και μεταξύ των παιγνίων στα οποία οι αρχικές επιλογές δεν ήταν σημεία ισορροπίας, ο αριθμός αυτός ήταν 5.08.

Όσον αφορά τα εκλογικά αποτελέσματα στον κανόνα Plurality, παρατηρούμε ότι στα περισσότερα παίγνια, ο νικητής που θα προέκυπτε αν όλοι δήλωναν τις πραγματικές τους προτιμήσεις, παραμένει νικητής και στο τελικό σημείο ισορροπίας (όχι μόνο όταν τα αρχικά προφίλ είναι σημεία ισορροπίας, αλλά και όταν δεν είναι).

Αυτό είναι αναμενόμενο, αφού αν σκεφτούμε το πώς δουλεύει ο αλγόριθμος Best Response Dynamics, σε κάθε γύρο ελέγχουμε για κάθε παίκτη αν μπορεί, αλλάζοντας τις προτιμήσεις του, να αυξήσει την ωφέλειά του, και αν βρεθεί, αλλάζουμε την επιλογή του και τρέχουμε τον επόμενο γύρο, με τα αλλαγμένα δεδομένα. Αυτό σημαίνει πως κάθε αλλαγή έχει μικρό αντίκτυπο στο τελικό αποτέλεσμα και άρα συνήθως θα παραμένει νικητής ο αρχικός νικητής.

Τα σκορ τους με βάση το πραγματικό προφίλ διαφέρουν το πολύ κατά 1 (ή είναι ίδια όταν ο δείκτης του αρχικού νικητή είναι μικρότερος από αυτόν του νικητή του τελικού σημείου ισορροπίας). Σε κανένα από τα παίγνια δεν αυξήθηκε το κοινωνικό όφελος.

Αυτό είναι επίσης λογικό, πρώτον γιατί οι αλλαγές που έχουν νόημα για αλλαγή του νικητή σε ένα προφίλ στρατηγικών θα είναι επιλογή μεταξύ των 2 ψηλότερων σε σκορ υποψηφίων, και δεύτερον επειδή κατά την εκτέλεση του αλγορίθμου, μόλις φτάσουμε σε σημείο ισορροπίας σταματάμε, οπότε είναι αναμενόμενο τα σκορ τους με βάση το πραγματικό προφίλ να διαφέρουν το πολύ κατά 1.

Για τον κανόνα Borda, δεν ισχύουν σε μεγάλο βαθμό τα παραπάνω, γιατί η αλλαγή του κάθε παίκτη μπορεί να έχει μεγάλο αντίκτυπο στο σκορ της ψηφοφορίας (της τάξης του αριθμού των υποψηφίων), Ο παίκτης μπορεί π.χ. να μετακινήσει τον $2^{ο}$ υποψήφιό του στην τελευταία θέση της δήλωσής του, στην οποία περίπτωση το σκορ του $2^{ου}$ θα μειωθεί κατά m-2, και ενώ είχαμε ισοπαλία μεταξύ $1^{ης}$ και $2^{ης}$ προτίμησης του παίκτη, να έχουμε πλέον νίκη της $1^{ης}$ με μεγάλη (m-2 στην προκειμένη) διαφορά.