Εκλογές

Π. Λουρίδας

Τμήμα Διοικητικής Επιστήμης και Τεχνολογίας, Οικονομικό Πανεπιστήμιο Αθηνών
louridas@aueb.gr

Επισκόπηση

Επισκόπηση

Υποψήφιοι

Η διεύθυνση προσωπικού μιας εταιρείας προτείνει στους εργαζόμενους να ψηφίσουν με ποια από τις παρακάτω τρεις εταιρείες catering θα ήθελαν να συνεργαστεί η εταιρεία για να προσφέρει μεσημεριανό φαγητό:

- "MeatLovers", ή *M*, προσφέρει γενικώς κρεατικά.
- "BitOfEverything", ή *E*, προσφέρει ένα γενικό μενού.
- "VegForLife", ή *V*, προσφέρει χορτοφαγικό μενού.

Αποτελέσματα Ψηφοφορίας

Έλαβον:

- M: 40%.
- E: 30%.
- V: 30%

Άρα όλοι θα τρώνε κρέας, θέλοντας και μη.

Οι Ιεραρχημένες Προτιμήσεις

Αν όμως λάβουμε υπόψη τις προτιμήσεις των υπαλλήλων σε ιεραρχημένη σειρά, από την περισσότερο προτιμητέα στη λιγότερο, έχουμε:

$$40\% : [M, E, V]$$
$$30\% : [V, E, M]$$
$$30\% : [E, V, M]$$

Προτιμήσεις ανά Ζεύγη: M και E

$$40\% : [M, E, V]$$
$$30\% : [V, E, M]$$
$$30\% : [E, V, M]$$

- Το 40% των ψηφοφόρων προτιμούν την M από την E.
- Το 30% των ψηφορόρων προτιμούν την E από την M.
- Το 30% των ψηφοφόρων προτιμούν την E από την M.
- Συνεπώς μεταξύ των δύο, το 60% των ψηφοφόρων προτιμούν την E και το 40% την M.

Προτιμήσεις ανά Ζεύγη: M και V

$$40\% : [M, E, V]$$
$$30\% : [V, E, M]$$
$$30\% : [E, V, M]$$

- Το 40% των ψηφοφόρων προτιμούν την M από την V.
- Το 30% των ψηφοφόρων προτιμούν την V από την M.
- Το 30% των ψηφοφόρων προτιμούν την V από την M.
- Συνεπώς μεταξύ των δύο, το 60% των ψηφοφόρων προτιμούν την V και το 40% την M.

Προτιμήσεις ανά Ζεύγη: E και V

$$40\% : [M, E, V]$$
$$30\% : [V, E, M]$$
$$30\% : [E, V, M]$$

- Το 40% των ψηφοφόρων προτιμούν την E από την V.
- Το 30% των ψηφοφόρων προτιμούν την V από την E.
- Το 30% των φηφορόρων προτιμούν την E από την V.
- Συνεπώς μεταξύ των δύο, το 70% των ψηφοφόρων προτιμούν την E και το 30% την V.

Προτιμήσεις ανά Ζεύγη: Αποτέλεσμα

- E εναντίον M: 60% προτιμούν E, 40% προτιμούν M.
- V εναντίον M: 60% προτιμούν V, 40% προτιμούν M.
- E εναντίον V: 70% προτιμούν E, 30% προτιμούν M.

Η E κερδίζει σε 2 επιμέρους συγκρίσεις, η V σε 1 επιμέρους σύγκριση, ενώ η M σε καμμία.
Άρα η E είναι η νικήτρια των εκλογών.

Επισκόπηση

Ορισμός

Ορισμός

Στο πλειοψηφικό εκλογικό σύστημα οι ψηφοφόροι μπορούν να κάνουν μόνο μία επιλογή στο ψηφοδέλτιό τους.

Το Πρόβλημα με το Πλειοψηφικό Σύστημα

- Το πρόβλημα του πλειοψηφικού εκλογικού συστήματος είναι ότι οι ψηφοφόροι δεν μπορούν να δηλώσουν το σύνολο των προτιμήσεών τους.
- Έτσι, ένας υποψήφιος που είναι αρεστός σε πολλούς ψηφοφόρους αλλά δεν είναι η κορυφαία επιλογή της σχετικής πλειοψηφίας θα χάσει έναντι ενός υποψηφίου που αποτελεί την κορυφαία επιλογή της.

Προεδρικές Εκλογές ΗΠΑ 2000 (1)

- Αντίπαλοι ήταν οι George W. Bush, Al Gore, και Ralph Nader.
- Ο πρόεδρος εκλέγεται από το κολλέγιο των αντιπροσώπων.
- Το κολλέγιο των αντιπροσώπων προκύπτει από τα αποτελέσματα των εκλογών στις επιμέρους πολιτείες.
- Το αποτελέσμα κρίθηκε στην πολιτεία της Florida.

Προεδρικές Εκλογές ΗΠΑ 2000 (2)

- Ο George W. Bush έλαβε 2.912.790 ψήφους, ήτοι το 48.847% του συνόλου.
- Ο Al Gore έλαβε 2.912.253 ψήφους, ήτοι το 48.838% του συνόλου.
- Ο Ralph Nader έλαβε 97.421 votes, ήτοι το 1.634% του συνόλου.

Προεδρικές Εκλογές ΗΠΑ 2000 (3)

- ο George W. Bush κέρδισε με διαφορά 537 ψήφων, ή 0.009% του συνόλου στη Florida.
- Εκτιμάται ότι οι περισσότεροι από τους ψηφοφόρους του Ralph Nader προτιμούσαν τον Al Gore από τον George W. Bush.
- Αν ισχύει αυτό, και οι Προεδρικές Εκλογές των ΗΠΑ χρησιμοποιούσαν ένα εκλογικό σύστημα που επέτρεπε στους ψηφοφόρους να δηλώνουν και τη δεύτερη επιλογή τους, νικητής θα ήταν ο Al Gore.

Προεδρικές Εκλογές Γαλλίας 2002 (1)

- Αντίπαλοι ήταν οι Jacques Chirac, Lionel Jospin, και δεκατέσσερεις άλλοι πολιτικοί.
- Πρόεδρος ανακυρύσσεται αυτός που θα λάβει πάνω από το 50% των ψήφων.
- Αν δεν συμβεί αυτό, οι εκλογές επαναλαμβάνονται με τους δύο πρώτους υποψήφιους.
- Με τα αποτελέσματα του πρώτου γύρου, αντί για τον Lionel Jospin, αντίπαλος του Jacques Chirac στον δεύτερο γύρο θα ήταν ο ακροδεξιός Jean-Marie Le Pen.

Προεδρικές Εκλογές Γαλλίας 2002 (2)

Αυτό δεν σημαίνει ότι οι περισσότεροι Γάλλοι ήταν ακροδεξιοί. Το αποτέλεσμα της καταμέτρησης του πρώτου γύρου ήταν:

- Ο Jacques Chirac έλαβε 5.666.440 ψήφους, ήτοι 19.88% του συνόλου.
- Ο Jean-Marie Le Pen έλαβε 4.805.307 ψήφους, ήτοι 16.86% του συνόλου.
- Ο Lionel Jospin έλαβε 4.610.749 votes, ήτοι 16.18% του συνόλου.

Προεδρικές Εκλογές Γαλλίας 2002 (3)

- Στο δεύτερο γύρο των εκλογών, δύο εβδομάδες αργότερα, ο Jacques Chirac έλαβε περισσότερο από το 82% των ψήφων.
- Ο Jean-Marie Le Pen έλαβε λιγότερο από το 18% των ψήφων.
- Ενώ ο Chirac έλαβε σχεδόν όλους τους ψήφους από τους υποψήφιους που αποκλείστηκαν στον πρώτο γύρο, ο Le Pen παρέμεινε στα ποσοστά του πρώτου.

Προεδρικές Εκλογές Γαλλίας 2002 (4)

- Το πρόβλημα προέκυψε από τον μεγάλο αριθμό των υποψηφίων: 16.
- Αφού μοιράστηκαν οι ψήφοι στους μετριοπαθείς υποψήφιους, μπόρεσε να αναδειχθεί στον πρώτο γύρο ο ακραίος υποψήφιος.

Επισκόπηση

Το Κριτήριο Condorcet

Ορισμός

Στις εκλογές πρέπει να κερδίζει ο υποψήφιος που συγκρινόμενος με κάθε άλλο υποψήφιο συγκεντρώνει τις προτιμήσεις των περισσότερων ψηφοφόρων. Ο νικητής ονομάζεται νικητής Condorcet.

Ονομάστηκε από τον Marie Jean Antoine Nicolas Caritat, the Marquis de Condorcet, έναν Γάλλο μαθηματικό και φιλόσοφο του 18ου αιώνα που το περιέγραψε στο βιβλίο του (1785) *Essai sur l'application de l'analyse á la probabilité des décisions rendues à la pluralité des voix* (*Essay on the Application of Analysis to the Probability of Majority Decisions*).

Ορισμός Συστημάτων Condorcet

Ορισμός

Ένα εκλογικό σύστημα που αναδεικνύει ως νικητή των εκλογών τον υποψήφιο που πληροί το κριτήριο Condorcet, αν είναι δυνατόν, ονομάζεται σύστημα Condorcet.

Σύστημα Αποδοχής (Approval Voting)

- Στο σύστημα αποδοχής, οι ψηφοφόροι μπορούν να επιλέξουν όσους υποψήφιους θέλουν στο ψηφοδέλτιο.
- Νικητής είναι ο υποψήφιος με τους περισσότερους ψήφους.

Κάληψη Κριτηρίου Condorcet

- Έστω ότι έχουμε τρεις υποψήφιους A, B, C και τα παρακάτω ψηφοδέλτια:

$$60\%: A, B$$
$$40\%: C, B$$

- Αφού το 100% των ψηφοφόρων επέλεξε τον B, αυτός είναι ο νικητής.
- Έστω ότι το 60% των ψηφοφόρων προτιμούν τον A από τον B και αυτόν από τον C: $[A, B, C]$.
- Έστω επίσης ότι το 40% των ψηφοφόρων προτιμούν τον C από τον B και αυτόν από τον A: $[C, B, A]$.
- Τότε στη μεταξύ τους σύγκριση, ο A επικρατεί του B με 60% έναντι 40%.
- Παρότι οι περισσότεροι ψηφοφόροι προτιμούν τον A από τον B, δεν εκλέγεται ο A.

Συνεπώς δεν πληροί το κριτήριο Condorcet.

Σύστημα Borda (Borda Count)

Ονομάστηκε από έναν άλλο Γάλλο του 18ου αιώνα, τον μαθηματικό και πολιτικό επιστήμονα Jean-Charles de Borda, που το περιέγραφε το 1770.

- Οι ψηφοφόροι δίνουν πόντους στους υποψήφιους.
- Αν υπάρχουν n υποψήφιοι, η πρώτη επιλογή στο ψηφοδέλτιο παίρνει $n-1$ πόντους, η δεύτερη επιλογή $n-2$ πόντους, κ.λπ., μέχρι την τελευταία επιλογή που παίρνει 0 πόντους.

Κάλυψη Κριτηρίου Condorcet

- Έστω ότι έχουμε τρεις υποψήφιους, A, B, και C και ψηφοδέλτια ως εξής:

$$60\% : [A, B, C]$$
$$40\% : [B, C, A]$$

- Αν έχουμε n ψηφοφόρους, ο υποψήφιος A παίρνει $(60 \times 2)n = 120n$ πόντους, ο υποψήφιος B παίρνει $(60 + 2 \times 40)n = 140n$ πόντους και ο υποψήφιος C παίρνει $40n$ πόντους.
- Νικητής είναι ο υποψήφιος B, αν και οι περισσότεροι ψηφοφόροι προτιμούν τον A από τον B.

Συνεπώς δεν πληροί το κριτήριο Condorcet.

Αδυναμία Ανάδειξης Νικητή Condorcet

Δεν είναι απαραίτητο ότι υπάρχει νικητής Condorcet.

- Για παράδειγμα, έστω ότι έχουμε τρεις υποψήφιους A, B και C, με τα εξής ψηφοδέλτια:

$$30 : [A, B, C]$$
$$30 : [B, C, A]$$
$$30 : [C, A, B]$$

- Αν κάνουμε τις συγκρίσεις ανά ζεύγη, βρίσκουμε ότι ο A επικρατεί του B με 60 έναντι 40, ο B επικρατεί του C με 60 έναντι 40 και ο C επικρατεί του A με 60 έναντι 40.
- Αφού κανείς υποψήφιος δεν επικρατεί έναντι περισσότερων υποψήφιων, δεν υπάρχει νικητής Condorcet.

Επισκόπηση

Το Πρόβλημα (1)

- Η μέθοδος που χρησιμοποιήσαμε για την εύρεση του νικητή μεταξύ των εταιρειών catering οδηγεί αρκετές φορές σε ισοπαλίες.
- Για παράδειγμα, έστω ότι τα ψηφοδέλτια είναι ως εξής:

$$10 \times [A, B, C]$$
$$5 \times [B, C, A]$$
$$5 \times [C, A, B]$$

- Μεταξύ των A και B κερδίζει στις προτιμήσεις ο A με 15 προς 5.
- Μεταξύ των A και C υπάρχει ισοπαλία 10 προς 10.
- Μεταξύ των και C κερδίζει στις προτιμήσεις ο B με 15 προς 5.
- Άρα αφού και ο A και ο B έχουν από μία επικράτηση, δεν προκύπτει νικητής των εκλογών.

Μέθοδος Schulze

- Μία μέθοδος η οποία βρίσκει πάντα τον νικητή Condorcet, αν αυτός υπάρχει, είναι η μέθοδος Schulze.
- Προτάθηκε από τον Markus Schulze το 1997.
- Είναι πολύ δημοφιλής σε τεχνολογικούς οργανισμούς.
- Η μέθοδος Schulze δεν οδηγεί συχνά σε ισοπαλίες.

Βήμα 1: Υπολογισμός Επιμέρους Προτιμήσεων

- Το πρώτο βήμα στη μέθοδο Schulze είναι ο υπολογισμός των επιμέρους προτιμήσεων ανά ζεύγη ψηφοφόρων.
- Έστω ότι έχουμε n ψηφοδέλτια και m ψηφοφόρους.
- Τα ψηφοδέλτια είναι τα $b_1, b_2, \ldots, b_n$ και οι ψηφοφόροι οι $c_1, c_2, \ldots, c_m$.
- Δημιουργούμε έναν πίνακα p μεγέθους $m \times m$. Κάθε στοιχείο του p, $p_{i,j}$ δείχνει πόσοι ψηφοφόροι προτιμούν τον υποψήφιο c_i από τον υποψήφιο c_j.
- Παίρνουμε κάθε ψηφοδέλτιο με τη σειρά. Για κάθε ζευγάρι c_i και c_j ψηφοφόρων όπου ο c_i προηγείται του c_j στο ψηφοδέλτιο, προσθέτουμε ένα στο στοιχείο $p_{i,j}$ του πίνακα p.

Αλγόριθμος Υπολογισμού Επιμέρους Προτιμήσεων

Algorithm 1: Calculate pairwise preferences.

Input: *ballots*, n, m: *ballots* is an array of ballots, of size n. Each ballot is an array of candidates. m is the number of candidates.

Output: p: an array of size $m \times m$ with the pairwise preferences for candidates; $p[i, j]$ is the number of voters that prefer candidate i to candidate j.

```
1  for i = 0 to m do
2      for j = 0 to m do
3          p[i][j] ← 0

4  for i = 0 to n do
5      b ← ballots[i]
6      for j = 0 to Size(b) do
7          f ← b[j]
8          for k = j + 1 to Size(b) do
9              s ← b[k]
10             p[f, s] ← p[f, s] + 1
11 return p
```

Παράδειγμα Υπολογισμού Επιμέρους Προτιμήσεων (1)

Έστω ότι σε κάποιες εκλογές είχαμε τέσσερεις ψηφοφόρους, A, B, C, and D. Συμμετείχαν 21 ψηφοφόροι με τα παρακάτω ψηφοδέλτια:

$$6 \times [A, C, D, B]$$
$$4 \times [B, A, D, C]$$
$$3 \times [C, D, B, A]$$
$$4 \times [D, B, A, C]$$
$$4 \times [D, C, B, A]$$

Παράδειγμα Υπολογισμού Επιμέρους Προτιμήσεων (2)

Ο πίνακας των επιμέρους προτιμήσεων προκύπτει ως εξής:

$$\begin{array}{c} \\ A \\ B \\ C \\ D \end{array}\!\!\begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 6 & (6+4+4) & (6+4) \\ (4+3+4+4) & 0 & (4+4) & 4 \\ (3+4) & (6+3+4) & 0 & (6+3) \\ (3+4+4) & (6+3+4+5) & (4+4+4) & 0 \end{bmatrix} \end{array}$$

$$= \begin{array}{c} \\ A \\ B \\ C \\ D \end{array}\!\!\begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 6 & 14 & 10 \\ 15 & 0 & 8 & 4 \\ 7 & 13 & 0 & 9 \\ 11 & 17 & 12 & 0 \end{bmatrix} \end{array}$$

Βήμα 2: Κατασκευή Γράφου Προτιμήσεων

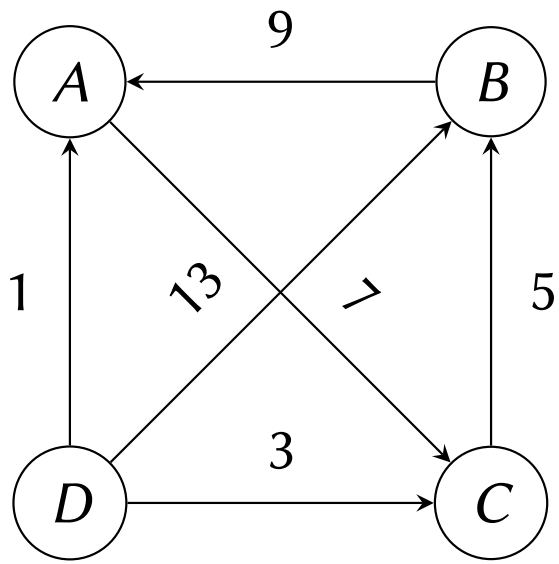
- Στο δεύτερο βήμα κατασκευάζουμε έναν γράφο όπου οι υποψήφιοι είναι οι κόμβοι και οι θετικές διαφορές μεταξύ των προτιμήσεων των υποψηφίων είναι τα βάρη των συνδέσμων.
- Αν για δύο υποψήφιους c_i and c_j ο αριθμός $p_{i,j}$ των ψηφοφόρων που προτιμούν τον υποψήφιο c_i από τον υποψήφιο c_j είναι μεγαλύτερος από τον αριθμό $p_{j,i}$ των υποψήφιων που προτιμούν τον c_j από τον c_i, προσθέτουμε στον γράφο τον σύνδεσμο $c_i \to c_j$ με βάρος $p_{i,j} - p_{j,i}$.
- Για τις άλλες περιπτώσεις χρησιμοποιούμε το $-\infty$ για να δείξουμε ότι δεν υπάρχει σύνδεσμος.

Παράδειγμα Κατασκευής Γράφου (1)

$$\begin{array}{c} \\ A \\ B \\ C \\ D \end{array}\begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & (6<15) & (14-7) & (10<11) \\ (15-6)=9 & 0 & (8<13) & (4<17) \\ (7<14) & (13-8) & 0 & (9<12) \\ (11-10)1 & (17-4) & (12-9) & -\infty \end{bmatrix} \end{array}$$

$$= \begin{array}{c} \\ A \\ B \\ C \\ D \end{array}\begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} -\infty & -\infty & 7 & -\infty \\ 9 & -\infty & -\infty & -\infty \\ -\infty & 5 & -\infty & -\infty \\ 1 & 13 & 3 & -\infty \end{bmatrix} \end{array}$$

Παράδειγμα Κατασκευής Γράφου (2)



Βήμα 3: Εύρεση Ισχυρότερων Μονοπατιών

Ορισμός

Η *ισχύς* (strength) ή το *πλάτος* (width) ενός μονοπατιού μεταξύ δύο κόμβων σε έναν γράφο είναι το βάρος του πιο αδύναμου συνδέσμου μεταξύ των δύο κόμβων.

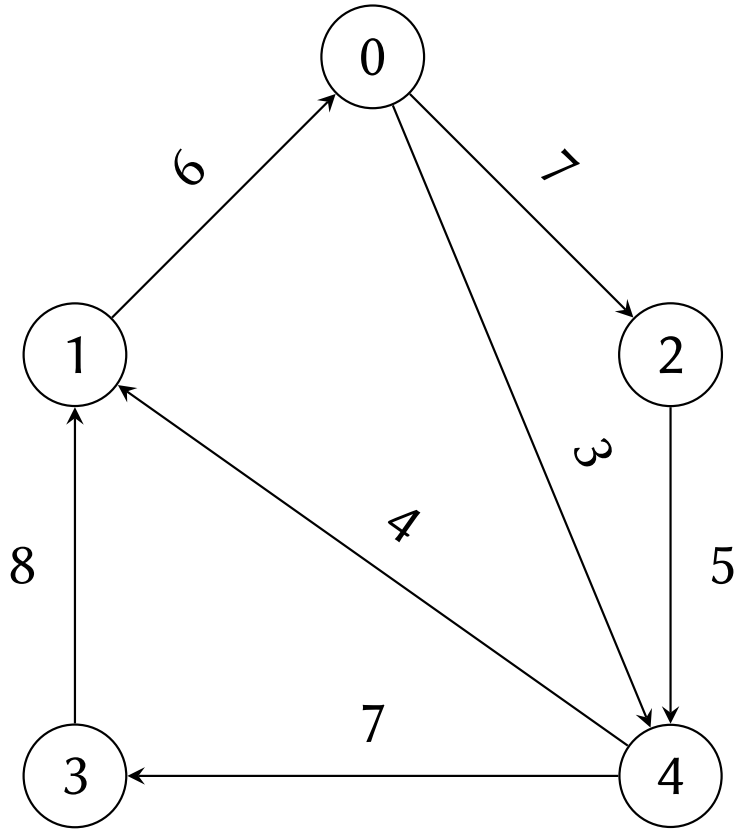
Η έννοια είναι αντίστοιχη με τη χωρητικότητα μιας γέφυρας, ή μιας σύνδεσης.

Ισχυρότερο Μονοπάτι

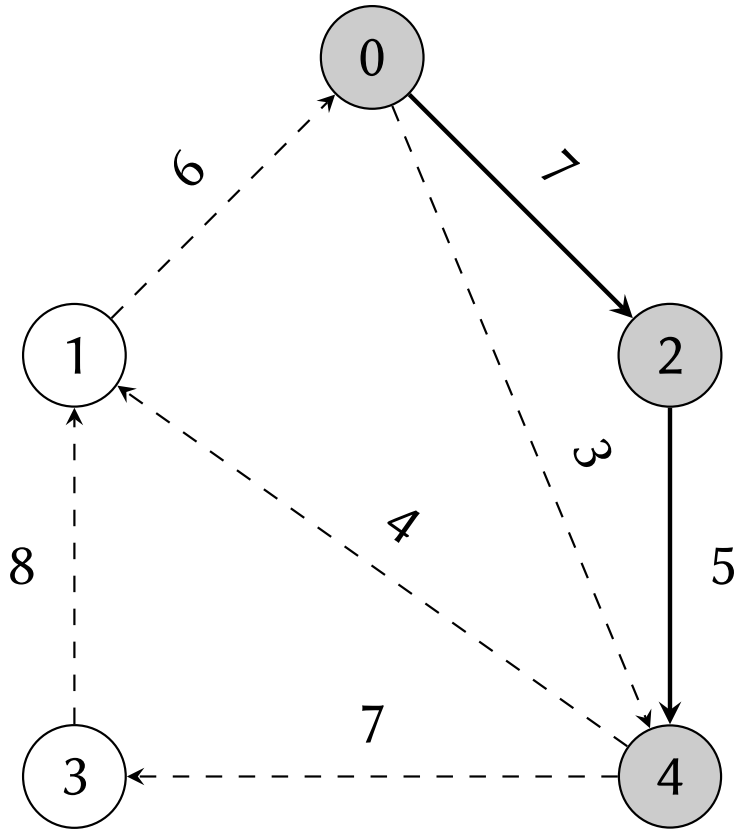
Ορισμός

Το *ισχυρότερο μονοπάτι* (strongest path) ή πλατύτερο μονοπάτι (widest path) μεταξύ δύο κόμβων είναι το μονοπάτι με τη μέγιστη δύναμη ανάμεσα σε όλα τα μονοπάτια μεταξύ των δύο κόμβων.

Παράδειγμα Ισχυρότερων Μονοπατιών (1)

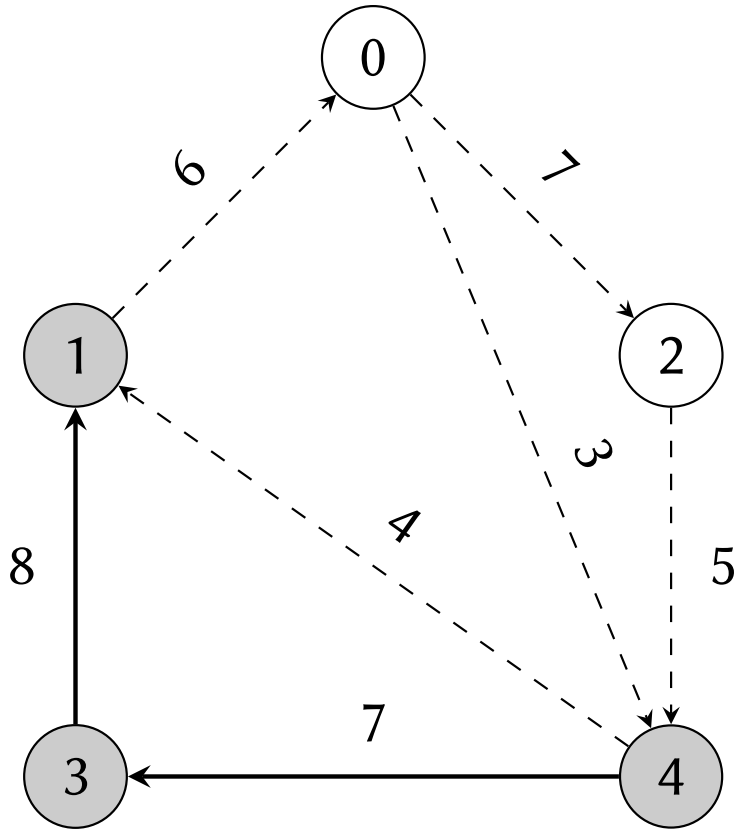


Παράδειγμα Ισχυρότερων Μονοπατιών (2)



Το ισχυρότερο μονοπάτι μεταξύ των κόμβων 0 και 4 είναι το $0 \to 2 \to 4$ και έχει ισχύ 5.

Παράδειγμα Ισχυρότερων Μονοπατιών (3)



Το ισχυρότερο μονοπάτι μεταξύ των κόμβων 4 και 1 είναι το $4 \to 3 \to 1$ και έχει ισχύ 7.

Αλγόριθμος Ισχυρότερων Μονοπατιών (1)

- Βάζουμε όλους τους κόμβους του γράφου σε μια σειρά $c_1, c_2, \ldots, c_n$.
- Βρίσκουμε το ισχυρότερο μονοπάτι μεταξύ κάθε ζεύγους κόμβων c_i και c_j χρησιμοποιώντας μηδέν ενδιάμεσους κόμβους από τη σειρά $c_1, c_2, \ldots, c_n$.
- Το ισχυρότερο μονοπάτι μεταξύ δύο κόμβων χωρίς τη χρήση ενδιάμεσων κόμβων είναι ο απευθείας σύνδεσμος μεταξύ των δύο κόμβων, αν υπάρχει.

Αλγόριθμος Ισχυρότερων Μονοπατιών (2)

- Αναζητούμε το ισχυρότερο μονοπάτι μεταξύ κάθε ζεύγους κόμβων c_i και c_j χρησιμοποιώντας τον πρώτο κόμβο στη σειρά, c_1, ως ενδιάμεσο κόμβο.
- Αν στο προηγούμενο βήμα βρήκαμε ήδη ένα μονοπάτι μεταξύ των c_i and c_j, τότε αν υπάρχουν μονοπάτια $c_i \to c_1$ και $c_1 \to c_j$ συγκρίνουμε την ισχύ του μονοπατιού $c_i \to c_j$ με την ισχύ του μονοπατιού $c_i \to c_1 \to c_j$, και παίρνουμε το ισχυρότερο των δύο ως το νέο ισχυρότερο μονοπάτι μεταξύ των c_i και c_j.

Αλγόριθμος Ισχυρότερων Μονοπατιών (3)

- Έστω ότι έχουμε βρει το ισχυρότερο μονοπάτιου μεταξύ κάθε ζεύγους c_i και c_j χρησιμοποιώντας τους πρώτους k κόμβους ως ενδιάμεσους.
- Αναζητούμε το ισχυρότερο μονοπάτι μεταξύ των c_i και c_j χρησιμοποιώντας τους πρώτους $(k+1)$ κόμβους.
- Αν βρούμε ένα τέτοιο μονοπάτι, θα αποτελείται από δύο μέρη. Το πρώτο θα είναι από το c_i στον c_{k+1} χρησιμοποιώντας τους πρώτους k κόμβους ως ενδιάμεσους και το δεύτερο θα είναι από τον c_{k+1} στον c_j πάλι χρησιμοποιώντας τους πρώτους k κόμβους ως ενδιάμεσους.
- Έχουμε βρει την ισχύ αυτών των μονοπατιών προηγουμένως, άρα θα έχουμε, αν $s_{i,j}(k)$ η ισχύς με τους k ως ενδιάμεσους κόμβους:

$$s_{i,j}(k+1) = max\Big(s_{i,j}(k), min\big(s_{i,k+1}(k), s_{k+1,j}(k)\big)\Big)$$

Αλγόριθμος Ισχυρότερων Μονοπατιών (4)

Algorithm 2: Calculate strongest paths.

Input: w, n: w is an array of size $n \times n$ representing the adjacency matrix of a graph; $w[i, j]$ is the weight of the edge between nodes i and j.

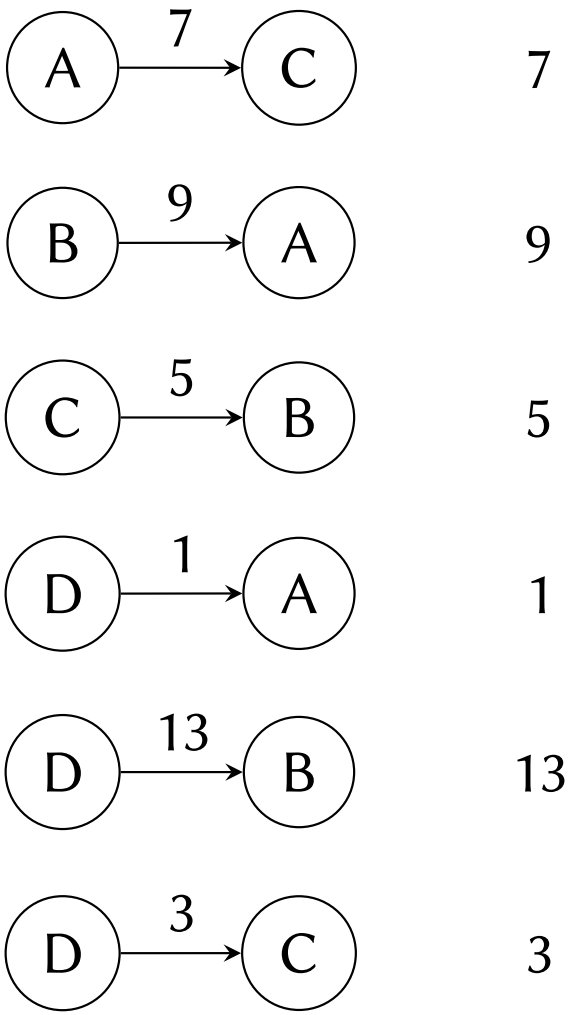
Output: $(s, pred)$: s is an array of size $n \times n$ such that $s[i, j]$ is the strongest path between nodes i and j. $pred$ is an array of size $n \times n$ such that $pred[i, j]$ is the predecessor of node i in the strongest path to node j.

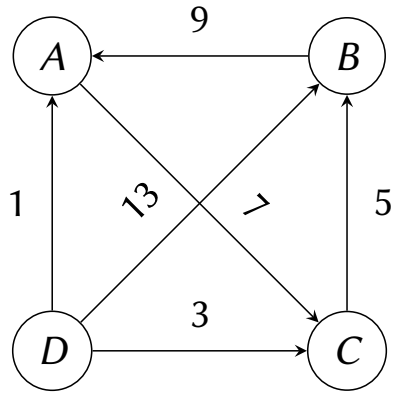
```
1  for i = 0 to n do
2      for j = 0 to n do
3          if w[i, j] > w[j, i] then
4              s[i][j] ← w[i, j] − w[j, i]
5              pred[i, j] ← i
6          else
7              s[i][j] ← −∞
8              pred[i, j] ← −1
9  for k = 0 to n do
10     for i = 0 to n do
11         if i ≠ k then
12             for j = 0 to n do
13                 if j ≠ i then
14                     if s[i, j] < min(s[i, k], s[k, j]) then
15                         s[i, j] ← min(s[i, k], s[k, j])
16                         pred[i, j] ← pred[k, j];
17 return (s, pred)
```

Αλγόριθμος Ισχυρότερων Μονοπατιών (5)

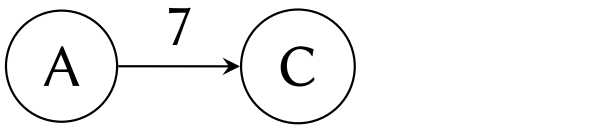
- Η επανάληψη στις γραμμές 1–8 εκτελείται n^2 φορές.
- Η επανάληψη στις γραμμές 9–16 εκτελείται n^3 φορές.
- Αφού ο γράφος αναπαρίσταται με τον πίνακα γειτνίασης, οι προσβάσεις στα στοιχεία του απαιτούν σταθερό χρόνο, και συνεπώς απαιτείται χρόνος $\Theta(n^3)$.

Παράδειγμα Εύρεσης Ισχυρότερων Μονοπατιών (1)

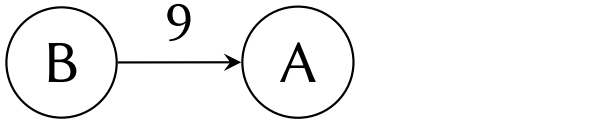




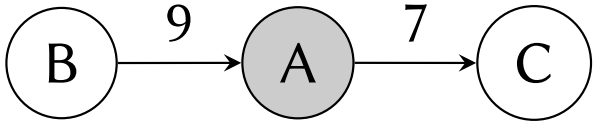
Παράδειγμα Εύρεσης Ισχυρότερων Μονοπατιών (2)



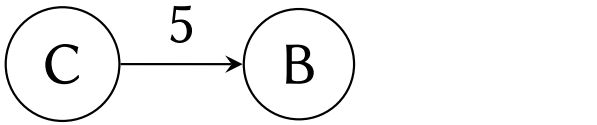
7



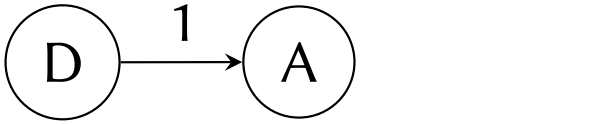
9



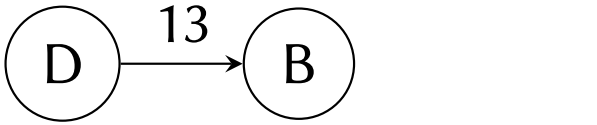
7



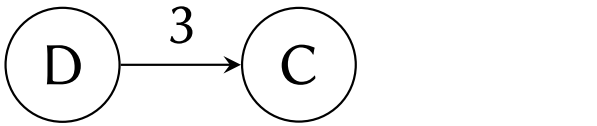
5



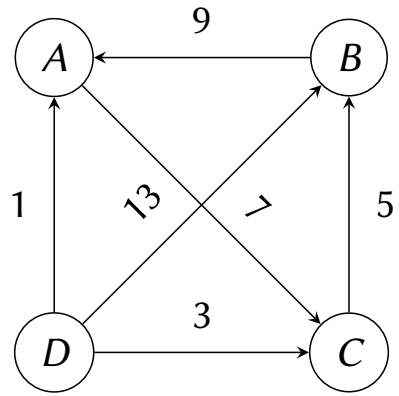
1



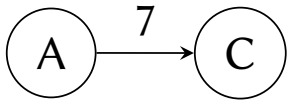
13



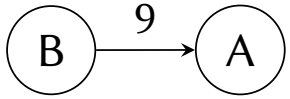
3



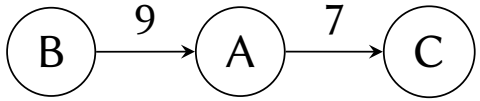
Παράδειγμα Εύρεσης Ισχυρότερων Μονοπατιών (3)



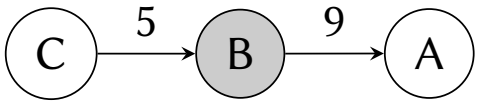
7



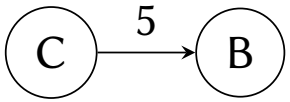
9



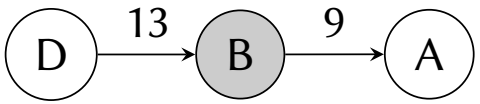
7



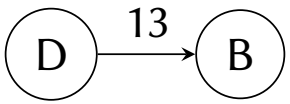
5



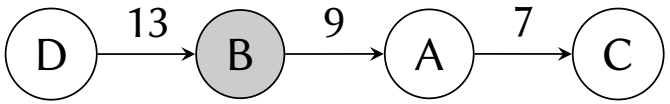
5



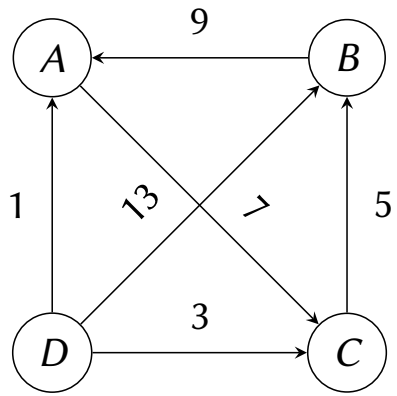
9



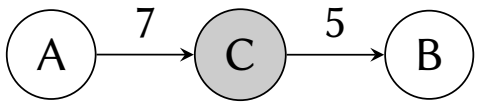
13



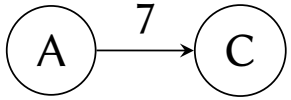
7



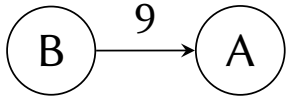
Παράδειγμα Εύρεσης Ισχυρότερων Μονοπατιών (4)



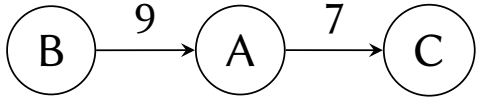
5



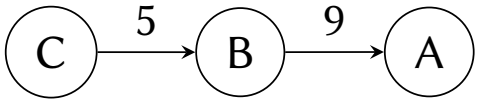
7



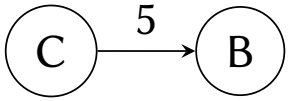
9



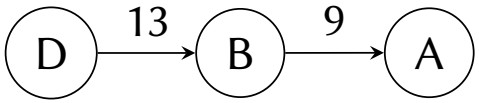
7



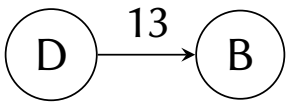
5



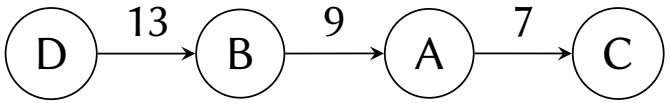
5



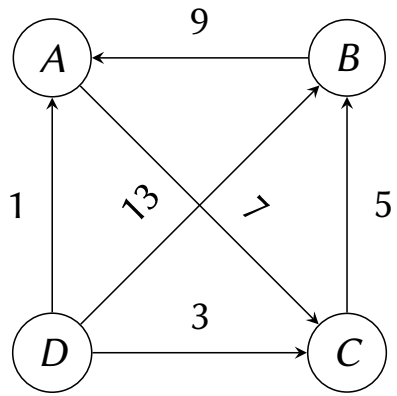
9



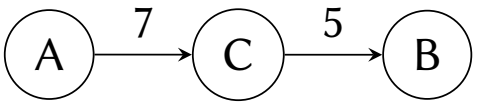
13



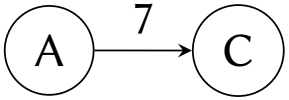
7



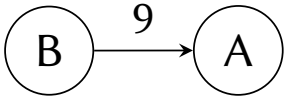
Παράδειγμα Εύρεσης Ισχυρότερων Μονοπατιών (5)



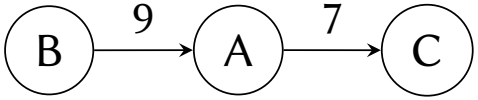
5



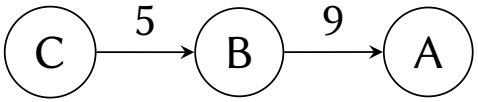
7



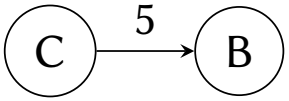
9



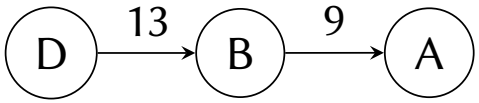
7



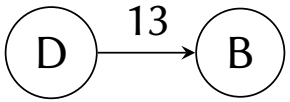
5



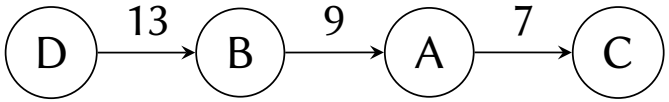
5



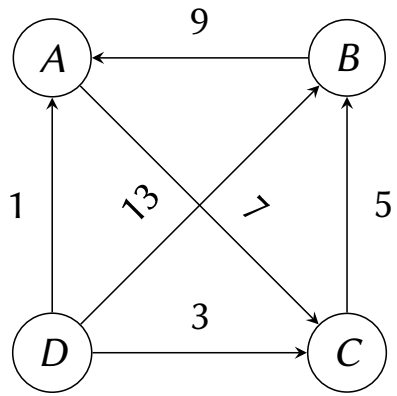
9



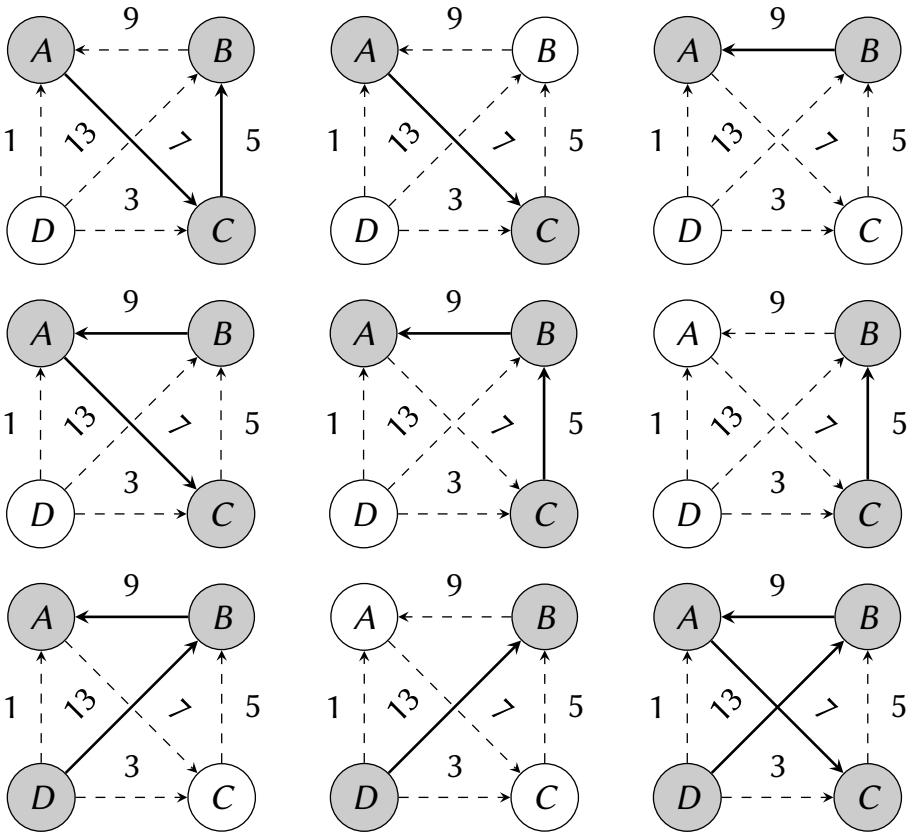
13



7



Τα Ισχυρότερα Μονοπάτια στο Γράφο



Βήμα 3: Σύγκριση Υποψηφίων

- Έχοντας βρει τα ισχυρότερα μονοπάτια, θέλουμε να βρούμε για κάθε ζεύγος υποψηφίων c_i και c_j πόση υποστήριξη υπάρχει για τον c_i έναντι του c_j και πόση υποστήριξη υπάρχει για τον c_j έναντι του c_i.
- Η υποστήριξη ενός υποψήφιου έναντι κάποιου άλλου είναι η ισχύς του μονοπατιού μεταξύ των δύο υποψηφίων.
- Αν η ισχύς του μονοπατιού από τον c_i το c_j είναι μεγαλύτερη από την ισχύ του μονοπατιού από τον c_j το c_i, λέμε ότι ο υποψήφιος c_i επικρατεί έναντι του υποψήφιου c_j.
- Άρα θέλουμε να βρούμε για κάθε υποψήφιο έναντι πόσων άλλων υποψήφιων επικρατεί.
- Αυτό το κάνουμε διαβάζοντας το αποτέλεσμα της εύρεσης των συντομότερων μονοπατιών και προσθέτοντας τις φορές που ο c_i επικρατεί των άλλων κόμβων.

Πίνακας Ισχυρότερων Μονοπατιών

$$\begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} -\infty & 5 & 7 & -\infty \\ 9 & -\infty & 7 & -\infty \\ 5 & 5 & -\infty & -\infty \\ 9 & 13 & 7 & -\infty \end{bmatrix} \end{array}$$

Αλγόριθμος Σύγκρισης Υποψηφίων

Algorithm 3: Calculate results.

Input: s, n: an array of size $n \times n$ with the strongest paths between nodes; $s[i, j]$ is the strongest path between nodes i and j.

Output: *wins*: a list of size n. Item i of *wins* is a list containing m integer items $j_1, j_2, \ldots, j_m$ for which $s[i, j_k] > s[j_k, i]$.

```
1 for i = 0 to n do
2     listi ← []
3     Add(wins, listi)
4     for j = 0 to n do
5         if i ≠ j then
6             if s[i, j] > s[j, i] then
7                 Add(listi, j)
8 return wins
```

Αποτελέσματα Εκλογών Παραδείγματος (1)

Από τον:

$$\begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} -\infty & 5 & 7 & -\infty \\ 9 & -\infty & 7 & -\infty \\ 5 & 5 & -\infty & -\infty \\ 9 & 13 & 7 & -\infty \end{bmatrix} \end{array}$$

Βρίσκουμε:

$$wins = [[2], [2, 0], [], [0, 1, 2]]$$

Αποτελέσματα Εκλογών Παραδείγματος (2)

- Άρα ο A νικάει τον C, ο B νικάει τον A και τον C, ο C δεν νικάει κανέναν, και ο D νικάει τους A, B, C.
- Συνεπώς τα αποτελέσματα των εκλογών του παραδείγματος είναι: $A = 1$, $B = 2$, $C = 0$ και $D = 3$.
- Ο D είναι ο υποψήφιος που προτιμάται έναντι των περισσότερων άλλων υποψηφίων

Επιστροφή στο Αρχικό Παράδειγμα Ισοπαλίας (1)

Είχαμε τις παρακάτω εκλογές, όπου με απλές συγκρίσεις είχε προκύψει ισοπαλία:

$$\begin{aligned} 10 \times [A, B, C] \\ 5 \times [B, C, A] \\ 5 \times [C, A, B] \end{aligned}$$

- Ο A προτιμάται του B με 15 προς 5
- Ο B προτιμάται του C με 15 προς 5
- Οι C και A ισοπαλούν με 10 προτιμήσεις ο καθένας.
- Αφού οι A και B έχουν από μία επικράτηση, είχαμε ισοπαλία.

Επιστροφή στο Αρχικό Παράδειγμα Ισοπαλίας (2)

Ο πίνακας προτιμήσεων είναι:

$$\begin{array}{c} \\ A \\ B \\ C \end{array} \begin{array}{c} \begin{array}{ccc} A & B & C \end{array} \\ \begin{bmatrix} 0 & 15 & 10 \\ 5 & 0 & 15 \\ 10 & 5 & 0 \end{bmatrix} \end{array}$$

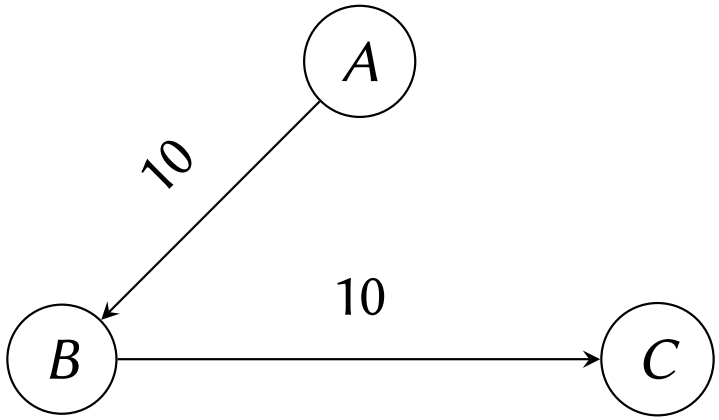
Επιστροφή στο Αρχικό Παράδειγμα Ισοπαλίας (3)

Ο πίνακας γειτνίασης για τον γράφο εκλογών είναι:

$$\begin{array}{c} \\ A \\ B \\ C \end{array} \begin{array}{c} \begin{array}{ccc} A & B & C \end{array} \\ \begin{bmatrix} -\infty & 10 & -\infty \\ -\infty & -\infty & 10 \\ -\infty & -\infty & -\infty \end{bmatrix} \end{array}$$

Επιστροφή στο Αρχικό Παράδειγμα Ισοπαλίας (4)

Ο γράφος εκλογών είναι:



Επιστροφή στο Αρχικό Παράδειγμα Ισοπαλίας (5)

Η μέθοδος Schulze θα δώσει ως αποτέλεσμα ότι

- Ο A επικρατεί των B και C.
- Ο B επικρατεί του C.
- Ο C δεν επικρατεί κανενός.

Άρα ο νικητής θα είναι ο A, και δεν έχουμε πλέον ισοπαλία.

Παρατηρήσεις για τη Μέθοδο Schulze

- Δεν δίνει μόνο τον πρώτο υποψήφιο, αλλά δημιουργεί μια ταξινόμηση των υποψηφίων, άρα μπορεί να χρησιμοποιηθεί και για την επιλογή των πρώτων k από n υποψήφιους.
- Αν δεν υπάρχει νικητής Condorcet, φυσικά δεν μπορεί να τον βρει.
- Μπορεί γενικώς να υπάρχουν υποψήφιοι στην ίδια θέση στην ταξινόμηση (αν είναι στην πρώτη θέση, δεν υπάρχει νικητής Condorcet), αλλά είναι σπάνιο να μην προκύψει νικητής.

Προσοχή!

- Η μέθοδος Schulze δεν είναι τέλεια.
- Στην πραγματικότητα έχει αποδειχθεί από το Νομπελίστα Kenneth Arrow ότι καμμία μέθοδος στην οποία οι ψηφοφόροι δίνουν τις προτιμήσεις τους δεν είναι τέλεια.

Επισκόπηση

Σχέση με τον Αλγόριθμο Floyd-Warshall

- Ο αλγόριθμος για τον υπολογισμό των ισχυρότερων μονοπατιών είναι μια παραλλαγή ενός αλγορίθμου για τον υπολογισμό των ζευγών συντομότερων μονοπατιών.
- Ο αλγόριθμος αυτός ονομάζεται αλγόριθμος Floyd-Warshall.
- Δημοσιεύθηκε από τον Robert Floyd το 1962, αλλά αντίστοιχοι αλγόριθμοι έχουν επίσης δημοσιευθεί από τον Bernard Roy το 1959 και από τον Stephen Warshall το 1962.
- Ο αλγόριθμος Floyd-Warshall απαιτεί χρόνο $\Theta(n^3)$. Είναι γενικώς πιο αργός από τον αλγόριθμο του Dijkstra, αλλά έχει καλή απόδοση. Επιπλέον, είναι απλός στην υλοποίηση και δουλεύει και με αρνητικά βάρη.

Ο Αλγόριθμος Floyd-Warshall

Algorithm 4: Floyd-Warshall all pairs shortest paths.

Input: w, n: w is an array of size $n \times n$ representing the adjacency matrix of a graph; $w[i, j]$ is the weight of the edge between nodes i and j.

Output: $(dist, pred)$: $dist$ is an array of size $n \times n$ such that $dist[i, j]$ is the shortest path between nodes i and j. $pred$ is an array of size $n \times n$ such that $pred[i, j]$ is the predecessor of node i in the shortest path to node j.

```
1  for i = 0 to n do
2      for j = 0 to n do
3          if w[i, j] ≠ −∞ then
4              dist[i, j] ← w[i, j]
5              pred[i, j] ← i
6          else
7              dist[i, j] ← +∞
8              pred[i, j] ← −1
9  for k = 0 to n do
10     for i = 0 to n do
11         if i ≠ k then
12             for j = 0 to n do
13                 if j ≠ i then
14                     if dist[i, j] > dist[i, k] + dist[k, j] then
15                         dist[i, j] ← dist[i, k] + dist[k, j]
16                         pred[i, j] ← pred[k, j];
17 return (dist, pred)
```