Μπορούν οι υπολογιστές να μας λύσουν όλα τα προβλήματα;

Αρχοντία Γιαννοπούλου

Τμήμα Πληροφορικής και Τηλεπικοινωνιών,
Εθνικό και Καποδιστριακό Πανεπιστήμιο Αθηνών

Προβλήματα που ξέρουμε να τα λύσουμε

- Εύρεση μέγιστου κοινού διαιρέτη
- Εύρεση λύσεων εξισώσεων της μορφής $ax^2 + bx + c = 0$
- Να αποφασίσουμε αν μία λέξη είναι παλίνδρομη
- Να βρούμε το συντομότερο δρόμο από μία πόλη A σε μία πόλη B

Τα προβλήματα του Hilbert



Το 1900 ο David Hilbert, διατύπωσε μία λίστα από 23 προβλήματα.

Τα προβλήματα του Hilbert



Το 1900 ο David Hilbert, διατύπωσε μία λίστα από 23 προβλήματα.

Πρόβλημα 10: *Δοσμένης μίας πολυωνυμικής εξίσωσης πολλών μεταβλητών με ακέραιους συντελεστές να σχεδιαστεί μία διαδικασία σύμφωνα με την οποία μετά από πεπερασμένο πλήθος πράξεων μπορεί να διαπιστωθεί αν η εξίσωση έχει ακέραιες λύσεις.*

Τα προβλήματα του Hilbert



Το 1900 ο David Hilbert, διατύπωσε μία λίστα από 23 προβλήματα.

Πρόβλημα 10: *Δοσμένης μίας πολυωνυμικής εξίσωσης πολλών μεταβλητών με ακέραιους συντελεστές να σχεδιαστεί μία διαδικασία σύμφωνα με την οποία μετά από πεπερασμένο πλήθος πράξεων μπορεί να διαπιστωθεί αν η εξίσωση έχει ακέραιες λύσεις.*

Με άλλα λόγια...

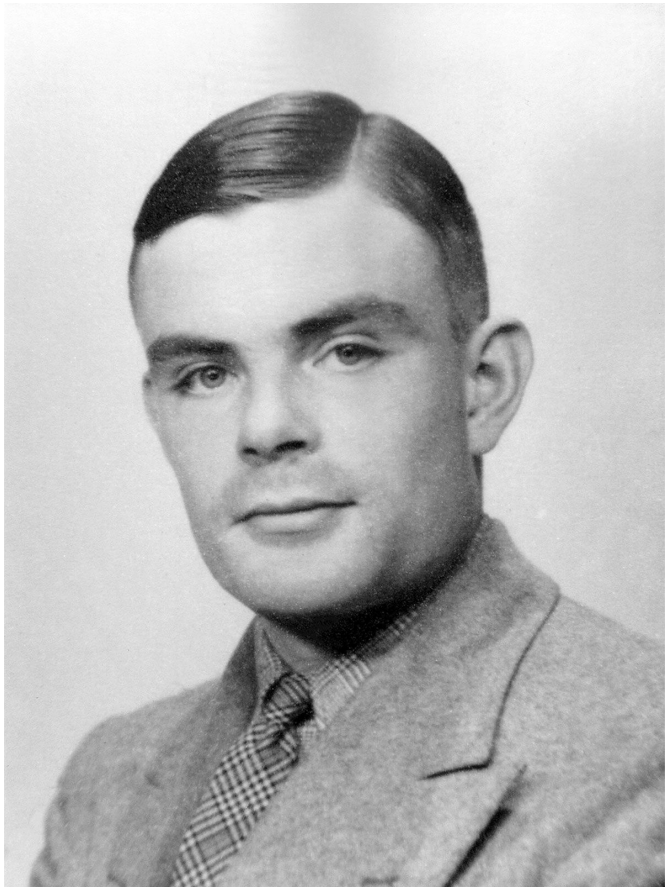
Ο Hilbert ρώτησε αν υπάρχει αλγόριθμος που να αποφασίζει αν αυτές οι εξισώσεις έχουν ακέραιες λύσεις.

Με άλλα λόγια...

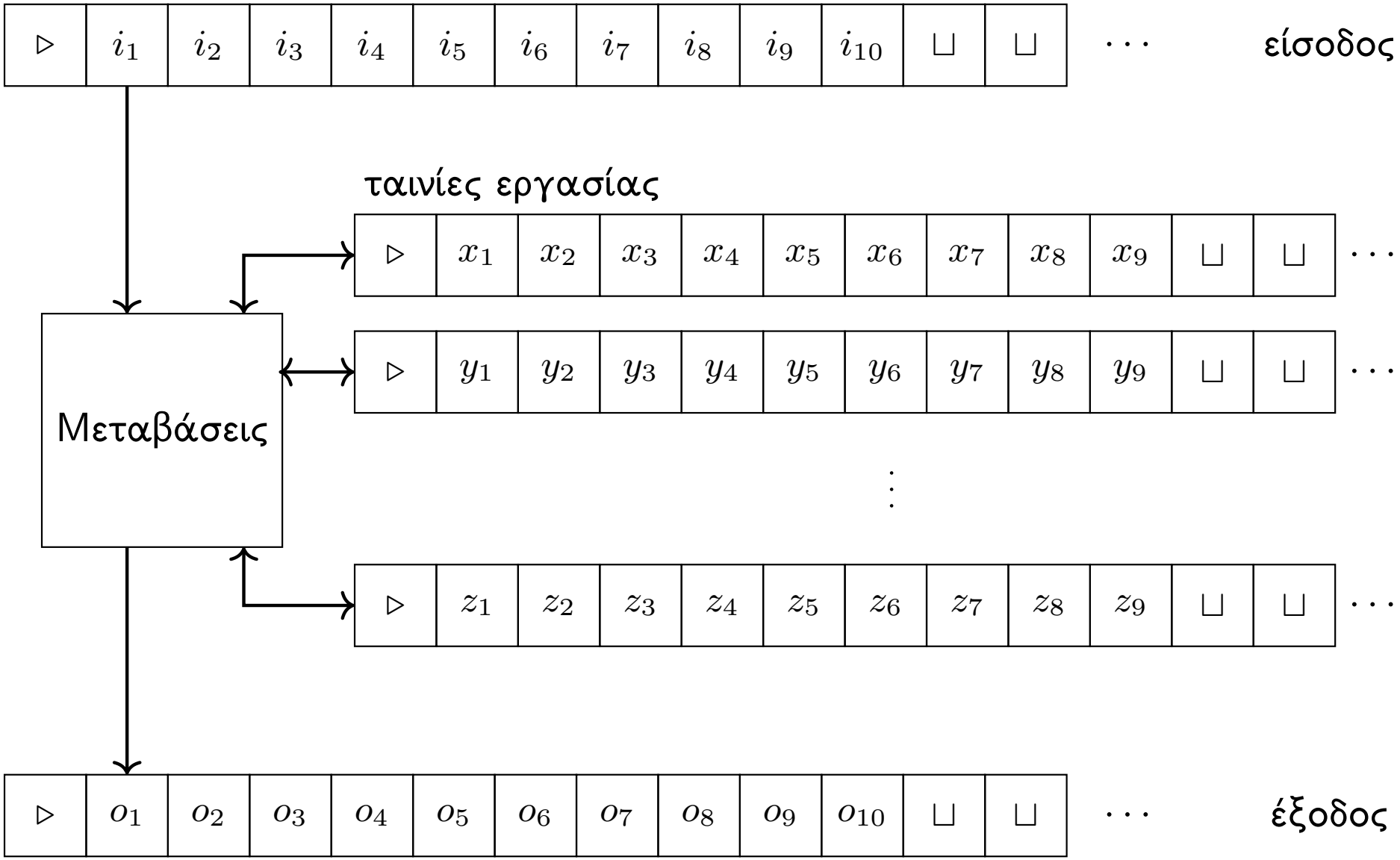
Ο Hilbert ρώτησε αν υπάρχει αλγόριθμος που να αποφασίζει αν αυτές οι εξισώσεις έχουν ακέραιες λύσεις.

Τι είναι όμως ένας αλγόριθμος;

Alan Turing



Μηχανές Turing



Δόγμα Church-Turing

Υπάρχει αλγόριθμος που επιλύει ένα πρόβλημα αν και μόνο αν υπάρχει Μηχανή Turing που να επιλύει το πρόβλημα.

Η καθολική Μηχανή Turing

Κάθε Μηχανή Turing μπορούμε να την κωδικοποιήσουμε/αναπαραστήσουμε με μία πεπερασμένη ακολουθία από bits.

Υπάρχει μία Μ.Τ. (που την καλούμε καθολική) και μπορεί να προσομοιώνει μία δοσμένη Μ.Τ. M με είσοδο x.

Η καθολική Μηχανή Turing

Κάθε Μηχανή Turing μπορούμε να την κωδικοποιήσουμε/αναπαραστήσουμε με μία πεπερασμένη ακολουθία από bits.

Υπάρχει μία Μ.Τ. (που την καλούμε καθολική) και μπορεί να προσομοιώνει μία δοσμένη Μ.Τ. M με είσοδο x.

Ένα παράδειγμα που ήδη ξέρετε

Ο διερμηνέας της Python στον υπολογιστή σας: παίρνει ως είσοδο ένα πρόγραμμα (ένα αρχείο .py) μαζί με τα δεδομένα του, και το εκτελεί.

Άρα ένα πρόγραμμα μπορεί να είναι είσοδος σε ένα άλλο πρόγραμμα.

Το πρόβλημα του Τερματισμού

Υπάρχει μηχανή Turing που να αποφασίζει αν μία μηχανή M με είσοδο x θα τερματίσει;

Το πρόβλημα του Τερματισμού

Υπάρχει μηχανή Turing που να αποφασίζει αν μία μηχανή M με είσοδο x θα τερματίσει;

Όχι! (Turing, 1936)

Γιατί όχι; (1/2)

Ας υποθέσουμε ότι υπάρχει πρόγραμμα `halts(P, x)` το οποίο, για *κάθε* πρόγραμμα P και *κάθε* είσοδο x, πάντα τερματίζει και απαντά

- `YES`, αν το P με είσοδο x τερματίζει,
- `NO`, αν το P με είσοδο x τρέχει για πάντα.

Γιατί όχι; (1/2)

Ας υποθέσουμε ότι υπάρχει πρόγραμμα `halts(P, x)` το οποίο, για *κάθε* πρόγραμμα P και *κάθε* είσοδο x, πάντα τερματίζει και απαντά

- `YES`, αν το P με είσοδο x τερματίζει,
- `NO`, αν το P με είσοδο x τρέχει για πάντα.

Τότε μπορούμε να γράψουμε και το εξής πρόγραμμα (το P δίνεται ως είσοδος, όπως στην καθολική Μ.Τ.):

```
trouble(P):
    if halts(P, P) = YES:
        loop forever
    else:
        stop
```

Δηλαδή το `trouble` κάνει το αντίθετο από αυτό που προβλέπει το `halts`.

Γιατί όχι; (2/2)

Τι κάνει το `trouble(trouble);`

Γιατί όχι; (2/2)

Τι κάνει το `trouble(trouble)`;

- Αν **τερματίζει**, τότε `halts(trouble, trouble) = YES`, άρα το `trouble(trouble)` μπαίνει σε ατέρμονο βρόχο. Άτοπο.

Γιατί όχι; (2/2)

Τι κάνει το `trouble(trouble)`;

- Αν **τερματίζει**, τότε `halts(trouble, trouble) = YES`, άρα το `trouble(trouble)` μπαίνει σε ατέρμονο βρόχο. Άτοπο.

- Αν **δεν τερματίζει**, τότε `halts(trouble, trouble) = NO`, άρα το `trouble(trouble)` σταματά. Άτοπο.

Γιατί όχι; (2/2)

Τι κάνει το `trouble(trouble)`;

- Αν **τερματίζει**, τότε `halts(trouble, trouble) = YES`, άρα το `trouble(trouble)` μπαίνει σε ατέρμονο βρόχο. Άτοπο.
- Αν **δεν τερματίζει**, τότε `halts(trouble, trouble) = NO`, άρα το `trouble(trouble)` σταματά. Άτοπο.

Συμπέρασμα

Το `halts` δεν μπορεί να υπάρχει: κανένας αλγόριθμος δεν αποφασίζει το πρόβλημα του Τερματισμού.

Επιστρέφοντας στο 10ο Πρόβλημα του Hilbert

Δεν υπάρχει αλγόριθμος που να αποφασίζει αν μία πολυωνυμική εξίσωση με ακέραιους συντελεστές έχει ακέραιες λύσεις.

(Matiyasevich, 1970, με βάση τη δουλειά των Davis, Putnam και Robinson)

Επιστρέφοντας στο 10ο Πρόβλημα του Hilbert

Δεν υπάρχει αλγόριθμος που να αποφασίζει αν μία πολυωνυμική εξίσωση με ακέραιους συντελεστές έχει ακέραιες λύσεις.

(Matiyasevich, 1970, με βάση τη δουλειά των Davis, Putnam και Robinson)

Ιδέα: αν υπήρχε τέτοιος αλγόριθμος, θα μπορούσαμε να τον χρησιμοποιήσουμε για να λύσουμε το πρόβλημα του Τερματισμού.

Επιστρέφοντας στο 10ο Πρόβλημα του Hilbert

Δεν υπάρχει αλγόριθμος που να αποφασίζει αν μία πολυωνυμική εξίσωση με ακέραιους συντελεστές έχει ακέραιες λύσεις.

(Matiyasevich, 1970, με βάση τη δουλειά των Davis, Putnam και Robinson)

Ιδέα: αν υπήρχε τέτοιος αλγόριθμος, θα μπορούσαμε να τον χρησιμοποιήσουμε για να λύσουμε το πρόβλημα του Τερματισμού.

- Μία μεταβλητή, π.χ. $ax^2 + bx + c = 0$: ξέρουμε να αποφασίσουμε.
- Πολλές μεταβλητές, ακέραιες λύσεις: κανένας αλγόριθμος δεν μπορεί.

Ας επικεντρωθούμε σε προβλήματα που μπορούμε να τα λύσουμε με τη χρήση του υπολογιστή (υπάρχει αλγόριθμος)...

Υπολογιστική Πολυπλοκότητα

Η μελέτη του κατά πόσο ένα επιλύσιμο πρόβλημα μπορεί να λυθεί αποδοτικά.

Πώς κατατάσσουμε επιλύσιμα υπολογιστικά προβλήματα σύμφωνα με τη δυσκολία τους;

Πόσο γρήγορα;

Ένας υπολογιστής που κάνει 10^9 βήματα το δευτερόλεπτο. Πόσο χρόνο χρειάζεται ένας αλγόριθμος με:

βήματα	$n = 10$	$n = 20$	$n = 50$
n	10 ns	20 ns	50 ns
n^2	0.1 μs	0.4 μs	2.5 μs
2^n	1 μs	1 ms	
$n!$	3.6 ms		

Πόσο γρήγορα;

Ένας υπολογιστής που κάνει 10^9 βήματα το δευτερόλεπτο. Πόσο χρόνο χρειάζεται ένας αλγόριθμος με:

βήματα	$n = 10$	$n = 20$	$n = 50$
n	10 ns	20 ns	50 ns
n^2	0.1 μs	0.4 μs	2.5 μs
2^n	1 μs	1 ms	13 μέρες
$n!$	3.6 ms	77 χρόνια	10^{48} χρόνια

Πόσο γρήγορα;

Ένας υπολογιστής που κάνει 10^9 βήματα το δευτερόλεπτο. Πόσο χρόνο χρειάζεται ένας αλγόριθμος με:

βήματα	$n = 10$	$n = 20$	$n = 50$
n	10 ns	20 ns	50 ns
n^2	0.1 μs	0.4 μs	2.5 μs
2^n	1 μs	1 ms	13 μέρες
$n!$	3.6 ms	77 χρόνια	10^{48} χρόνια

Για σύγκριση: η ηλικία του σύμπαντος είναι περίπου $1.4 \cdot 10^{10}$ χρόνια.

Πόσο γρήγορα;

Ένας υπολογιστής που κάνει 10^9 βήματα το δευτερόλεπτο. Πόσο χρόνο χρειάζεται ένας αλγόριθμος με:

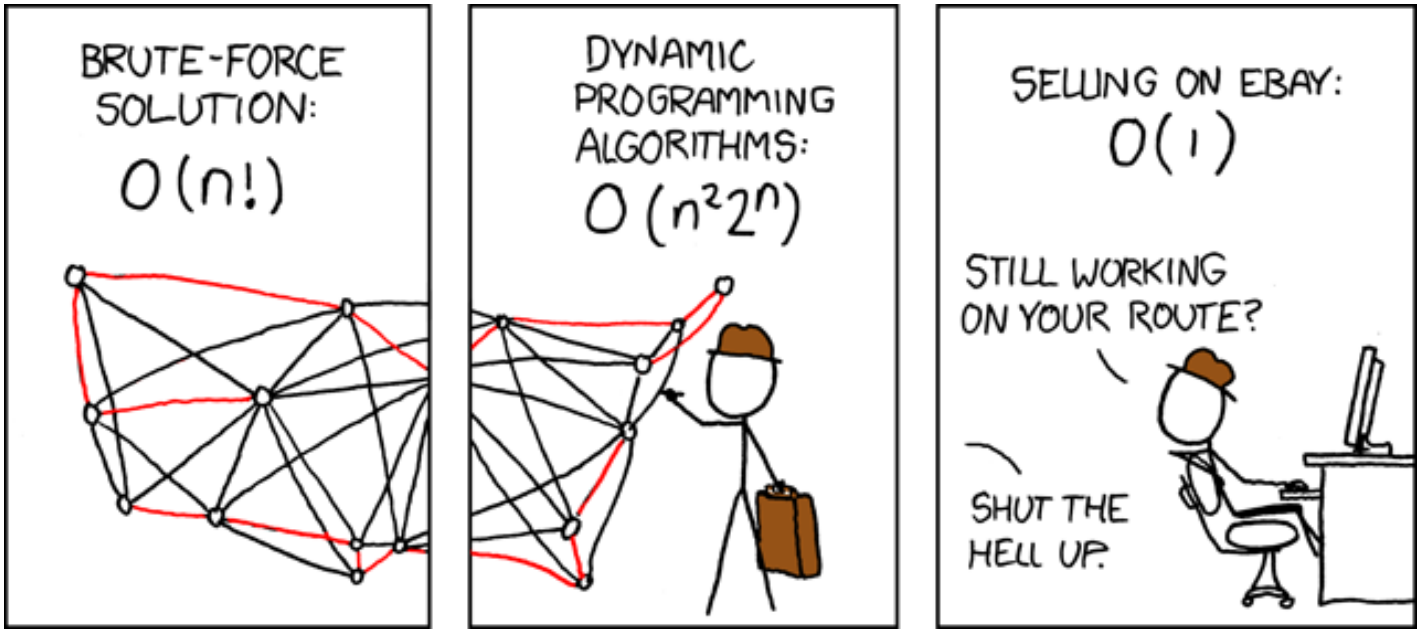
βήματα	$n = 10$	$n = 20$	$n = 50$
n	10 ns	20 ns	50 ns
n^2	0.1 μs	0.4 μs	2.5 μs
2^n	1 μs	1 ms	13 μέρες
$n!$	3.6 ms	77 χρόνια	10^{48} χρόνια

Για σύγκριση: η ηλικία του σύμπαντος είναι περίπου $1.4 \cdot 10^{10}$ χρόνια.

Γράφουμε $O(f(n))$ για «το πολύ $c \cdot f(n)$ βήματα, για κάποια σταθερά c».

Το πρόβλημα του Πλανόδιου Πωλητή

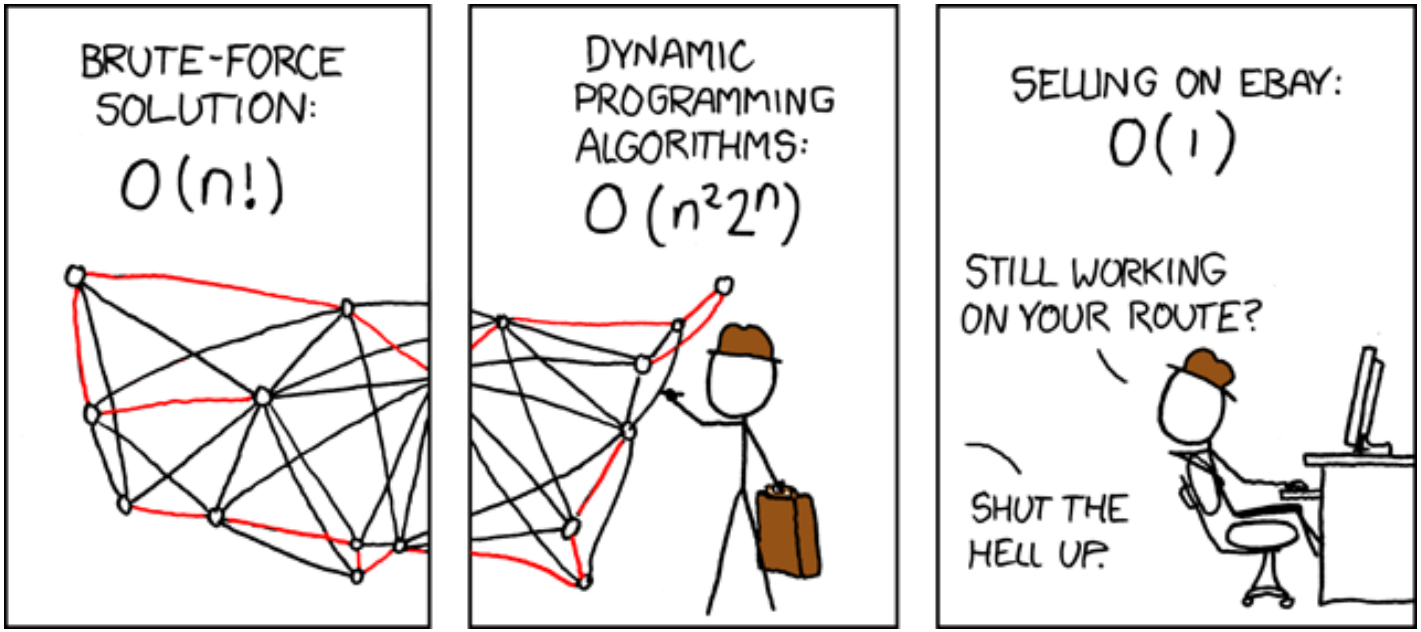
Δίνονται n πόλεις και οι αποστάσεις μεταξύ τους. Ποια είναι η συντομότερη διαδρομή που περνάει από κάθε πόλη ακριβώς μία φορά και επιστρέφει στην αφετηρία;



xkcd.com/399 (CC BY-NC 2.5)

Το πρόβλημα του Πλανόδιου Πωλητή

Δίνονται n πόλεις και οι αποστάσεις μεταξύ τους. Ποια είναι η συντομότερη διαδρομή που περνάει από κάθε πόλη ακριβώς μία φορά και επιστρέφει στην αφετηρία;



xkcd.com/399 (CC BY-NC 2.5)

Δοκιμάζοντας όλες τις διαδρομές: $(n-1)!/2$ διαδρομές. Για 25 πόλεις $\approx 3 \cdot 10^{23}$ διαδρομές: με 10^9 διαδρομές το δευτερόλεπτο, ≈ 10 εκατομμύρια χρόνια.

Πώς οργανώνουμε ένα επιτυχημένο πάρτι;

Μία εταιρεία έχει n υπαλλήλους και θέλει να οργανώσει ένα πάρτι. Κάθε υπάλληλος v έχει έναν «βαθμό χαβαλέ» $w(v) \geq 0$.

Οι διοργανωτές έχουν παρατηρήσει ότι εάν στο πάρτι βρίσκεται ένας υπάλληλος και ο άμεσος προϊστάμενός του τότε κανένας από τους δύο δεν περνάει καλά.

Πώς οργανώνουμε ένα επιτυχημένο πάρτι;

Μία εταιρεία έχει n υπαλλήλους και θέλει να οργανώσει ένα πάρτι. Κάθε υπάλληλος v έχει έναν «βαθμό χαβαλέ» $w(v) \geq 0$.

Οι διοργανωτές έχουν παρατηρήσει ότι εάν στο πάρτι βρίσκεται ένας υπάλληλος και ο άμεσος προϊστάμενός του τότε κανένας από τους δύο δεν περνάει καλά.

Ποιους πρέπει να καλέσει η εταιρεία ώστε ο συνολικός χαβαλές να είναι ο μέγιστος δυνατός;

Πώς οργανώνουμε ένα επιτυχημένο πάρτι;

Μία εταιρεία έχει n υπαλλήλους και θέλει να οργανώσει ένα πάρτι. Κάθε υπάλληλος v έχει έναν «βαθμό χαβαλέ» $w(v) \geq 0$.

Οι διοργανωτές έχουν παρατηρήσει ότι εάν στο πάρτι βρίσκεται ένας υπάλληλος και ο άμεσος προϊστάμενός του τότε κανένας από τους δύο δεν περνάει καλά.

Ποιους πρέπει να καλέσει η εταιρεία ώστε ο συνολικός χαβαλές να είναι ο μέγιστος δυνατός;

Δοκιμάζοντας όλους τους πιθανούς συνδυασμούς καλεσμένων: 2^n επιλογές. Για $n = 50$ ήδη 13 μέρες.

Μπορούμε καλύτερα; Θα επανέλθουμε...

Ένα πάρτι με 6 άτομα

Σε *οποιαδήποτε* παρέα 6 ατόμων υπάρχουν 3 που γνωρίζονται όλοι μεταξύ τους ή 3 που είναι όλοι άγνωστοι μεταξύ τους.

Ένα πάρτι με 6 άτομα

Σε *οποιαδήποτε* παρέα 6 ατόμων υπάρχουν 3 που γνωρίζονται όλοι μεταξύ τους ή 3 που είναι όλοι άγνωστοι μεταξύ τους.

Γιατί; Διαλέγουμε ένα άτομο A. Από τα άλλα 5, τουλάχιστον 3 είναι γνωστοί του ή τουλάχιστον 3 είναι άγνωστοί του (αφού $5 > 2 + 2$). Έστω ότι B, C, D είναι γνωστοί του A (η άλλη περίπτωση είναι συμμετρική).

- Αν δύο από τους B, C, D γνωρίζονται, μαζί με τον A έχουμε 3 γνωστούς.
- Αλλιώς οι B, C, D είναι 3 άγνωστοι μεταξύ τους. □

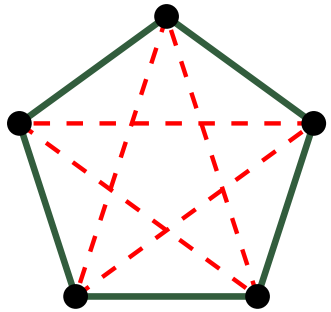
Ένα πάρτι με 6 άτομα

Σε *οποιαδήποτε* παρέα 6 ατόμων υπάρχουν 3 που γνωρίζονται όλοι μεταξύ τους ή 3 που είναι όλοι άγνωστοι μεταξύ τους.

Γιατί; Διαλέγουμε ένα άτομο A. Από τα άλλα 5, τουλάχιστον 3 είναι γνωστοί του ή τουλάχιστον 3 είναι άγνωστοί του (αφού $5 > 2 + 2$). Έστω ότι B, C, D είναι γνωστοί του A (η άλλη περίπτωση είναι συμμετρική).

- Αν δύο από τους B, C, D γνωρίζονται, μαζί με τον A έχουμε 3 γνωστούς.
- Αλλιώς οι B, C, D είναι 3 άγνωστοι μεταξύ τους. □

Με 5 άτομα δεν ισχύει πάντα:



— γνωστοί - - άγνωστοι

Ας οργανώσουμε ένα πάρτι...

Αν καλέσουμε μόνο k αγνώστους ή άτομα που γνωρίζονται όλοι μεταξύ τους;

Ας οργανώσουμε ένα πάρτι...

Αν καλέσουμε μόνο k αγνώστους ή άτομα που γνωρίζονται όλοι μεταξύ τους;

Υπάρχουν $\binom{n}{k} \leq n^k$ πιθανά σύνολα k ατόμων, και για καθένα ελέγχουμε $\binom{k}{2}$ ζεύγη: $O(k^2 n^k)$ βήματα.

Ας οργανώσουμε ένα πάρτι...

Αν καλέσουμε μόνο k αγνώστους ή άτομα που γνωρίζονται όλοι μεταξύ τους;

Υπάρχουν $\binom{n}{k} \leq n^k$ πιθανά σύνολα k ατόμων, και για καθένα ελέγχουμε $\binom{k}{2}$ ζεύγη: $O(k^2 n^k)$ βήματα.

Μπορούμε να κάνουμε την αναζήτηση αυτών των συνόλων πιο γρήγορα;

Ας οργανώσουμε ένα πάρτι...

Αν καλέσουμε μόνο k αγνώστους ή άτομα που γνωρίζονται όλοι μεταξύ τους;

Υπάρχουν $\binom{n}{k} \leq n^k$ πιθανά σύνολα k ατόμων, και για καθένα ελέγχουμε $\binom{k}{2}$ ζεύγη: $O(k^2\, n^k)$ βήματα.

Μπορούμε να κάνουμε την αναζήτηση αυτών των συνόλων πιο γρήγορα;

Λίγο: περίπου $n^{0.8k}$ βήματα αντί για n^k (Nešetřil–Poljak, 1985, με γρήγορο πολλαπλασιασμό πινάκων).

Ας οργανώσουμε ένα πάρτι...

Αν καλέσουμε μόνο k αγνώστους ή άτομα που γνωρίζονται όλοι μεταξύ τους;

Υπάρχουν $\binom{n}{k} \leq n^k$ πιθανά σύνολα k ατόμων, και για καθένα ελέγχουμε $\binom{k}{2}$ ζεύγη: $O(k^2\, n^k)$ βήματα.

Μπορούμε να κάνουμε την αναζήτηση αυτών των συνόλων πιο γρήγορα;

Λίγο: περίπου $n^{0.8k}$ βήματα αντί για n^k (Nešetřil–Poljak, 1985, με γρήγορο πολλαπλασιασμό πινάκων).

Ουσιαστικά όχι (πιστεύουμε): αν ισχύει η Υπόθεση Εκθετικού Χρόνου (ETH), δεν υπάρχει αλγόριθμος με $f(k) \cdot n^{o(k)}$ βήματα. Δηλαδή ο εκθέτης του n πρέπει να μεγαλώνει ανάλογα με το k.

Δύο σημαντικές κλάσεις πολυπλοκότητας

Προβλήματα απόφασης: η απάντηση είναι ΝΑΙ ή ΟΧΙ.

P: βρίσκουμε την απάντηση *εύκολα*.

NP: όταν η απάντηση είναι ΝΑΙ, υπάρχει μία ενδεχόμενη λύση (ένα «πιστοποιητικό») και ελέγχουμε *εύκολα* ότι είναι σωστή.

Εύκολα για εμάς σημαίνει σε πολυωνυμικό χρόνο, π.χ. $O(n^2)$ ή $O(n^3)$, όχι 2^n ή $n!$.

Δύο σημαντικές κλάσεις πολυπλοκότητας

Προβλήματα απόφασης: η απάντηση είναι ΝΑΙ ή ΟΧΙ.

P: βρίσκουμε την απάντηση *εύκολα*.

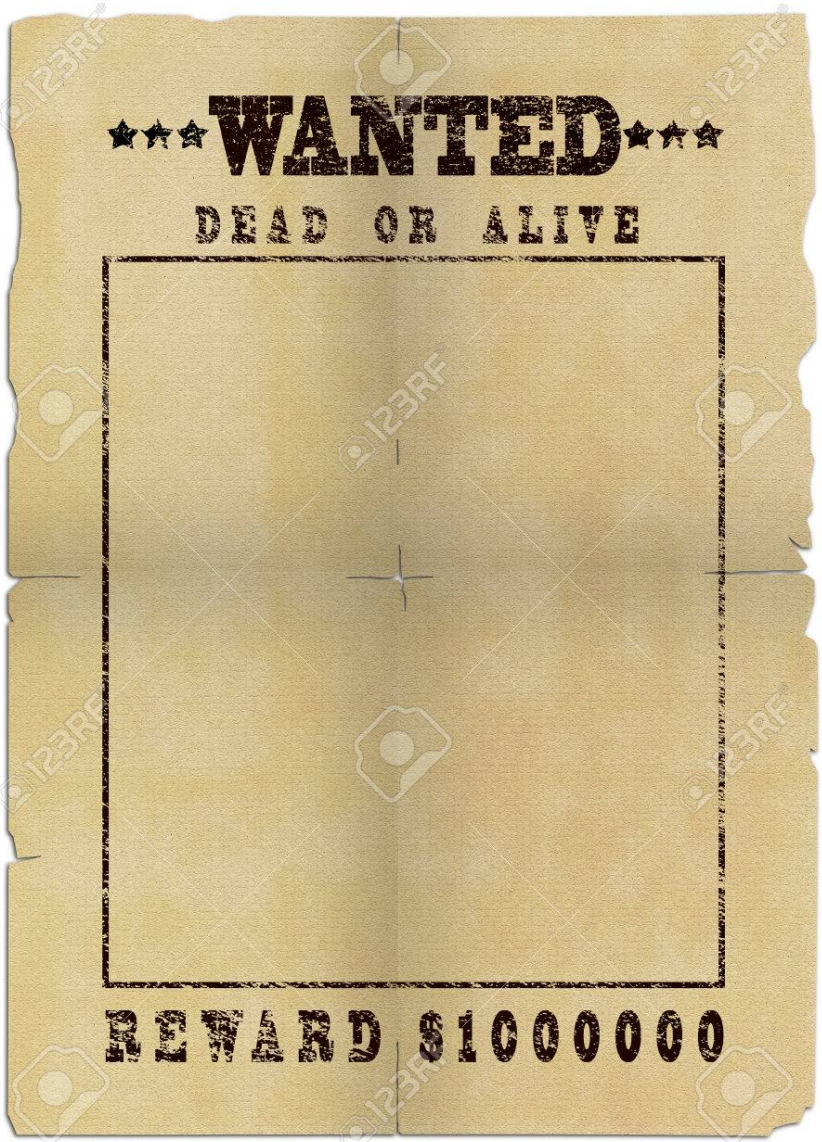
NP: όταν η απάντηση είναι ΝΑΙ, υπάρχει μία ενδεχόμενη λύση (ένα «πιστοποιητικό») και ελέγχουμε *εύκολα* ότι είναι σωστή.

Εύκολα για εμάς σημαίνει σε πολυωνυμικό χρόνο, π.χ. $O(n^2)$ ή $O(n^3)$, όχι 2^n ή $n!$.

Παράδειγμα: «Υπάρχουν k άτομα που γνωρίζονται όλοι μεταξύ τους;»

- Αν κάποιος μας δώσει k άτομα, ελέγχουμε τα $\binom{k}{2}$ ζεύγη: εύκολο. Άρα το πρόβλημα είναι στο NP.
- Να τα *βρούμε* όμως (με το k μέρος της εισόδου); Δεν ξέρουμε αλγόριθμο πολυωνυμικού χρόνου.

Επικηρυγμένα προβλήματα



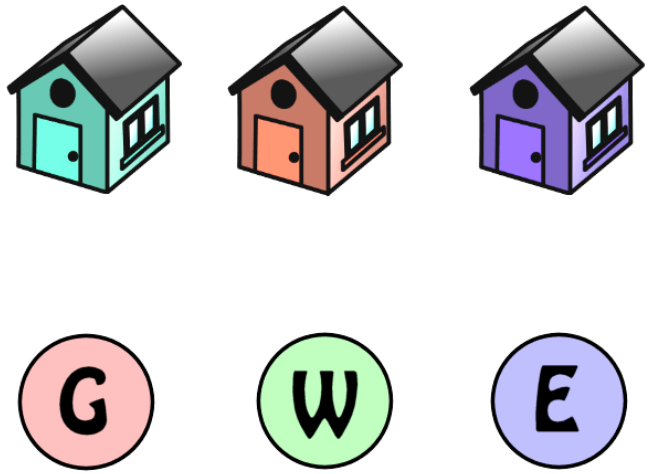
Είναι οι κλάσεις P και NP ίσες;

Ένα από τα 7 Προβλήματα της Χιλιετίας του Clay Mathematics Institute (2000).

Αμοιβή: $1.000.000$ δολάρια.

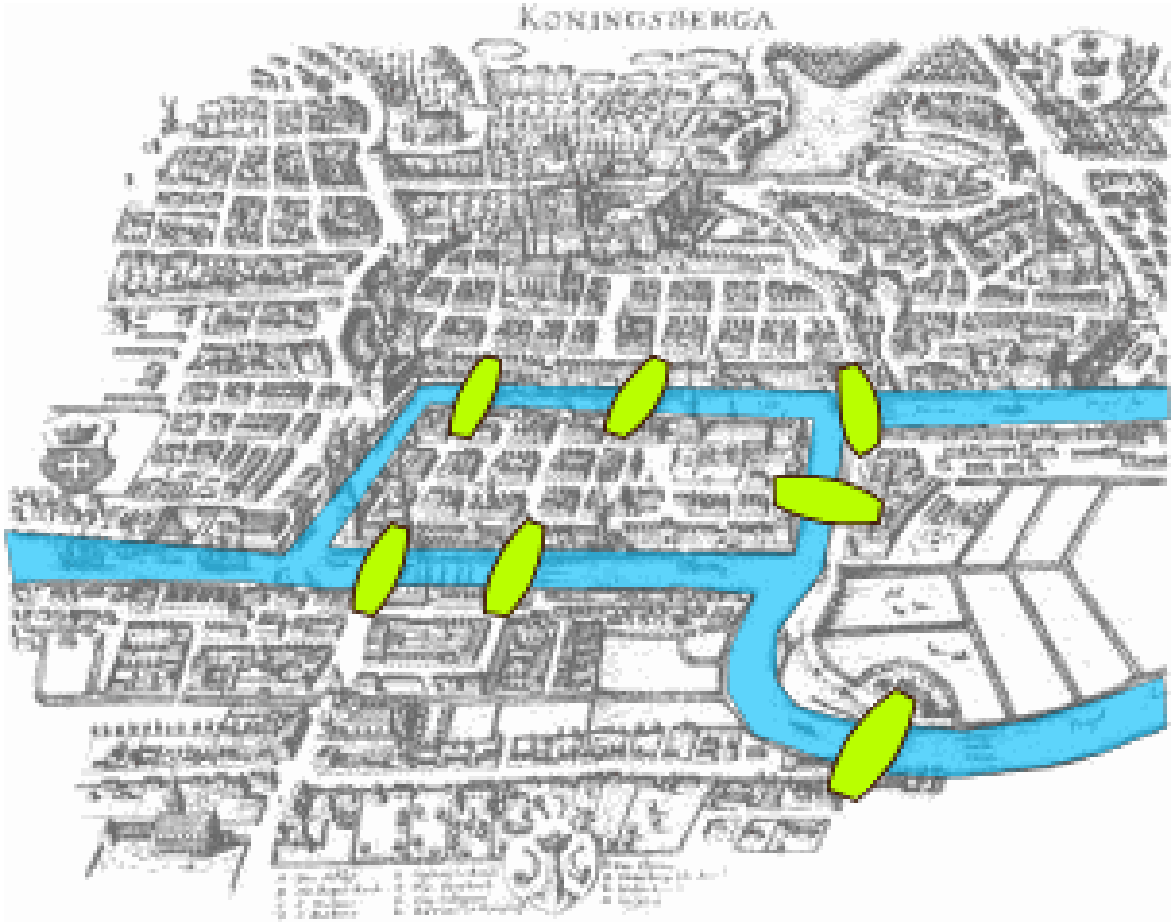
Ένας γρίφος

Τρία σπίτια πρέπει να συνδεθούν με το γκάζι (G), το νερό (W) και το ρεύμα (E).



Μπορούμε να συνδέσουμε κάθε σπίτι με κάθε παροχή χωρίς να διασταυρώνονται οι γραμμές;

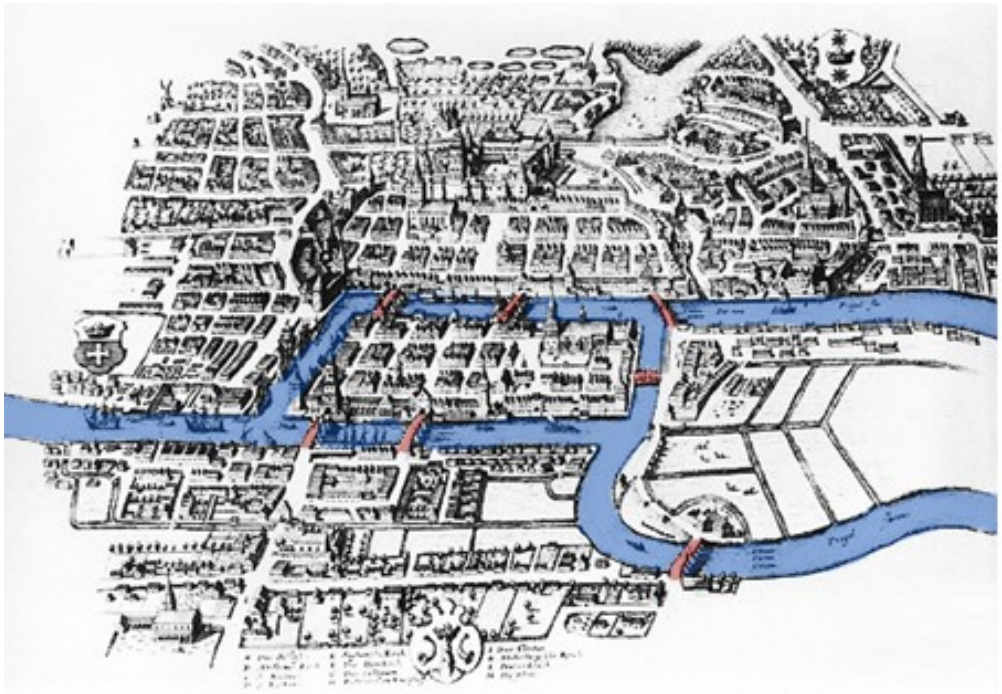
Πώς λύνουμε προβλήματα;



Μπορούμε να περάσουμε από όλες τις γέφυρες ακριβώς μία φορά;

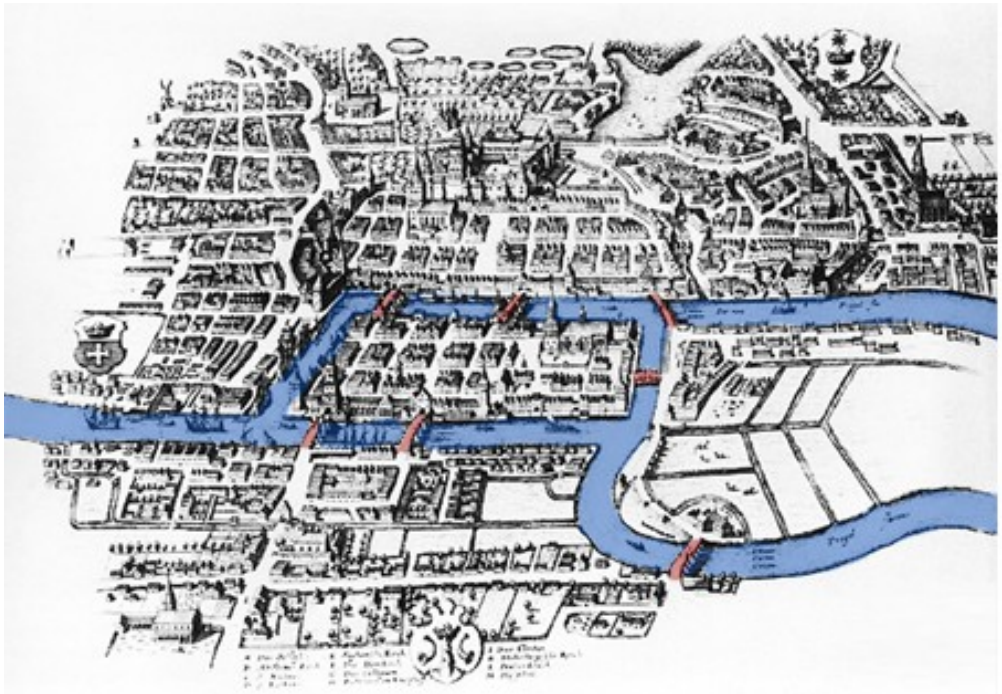
Δοκιμάστε!

Πιο γενικές δομές





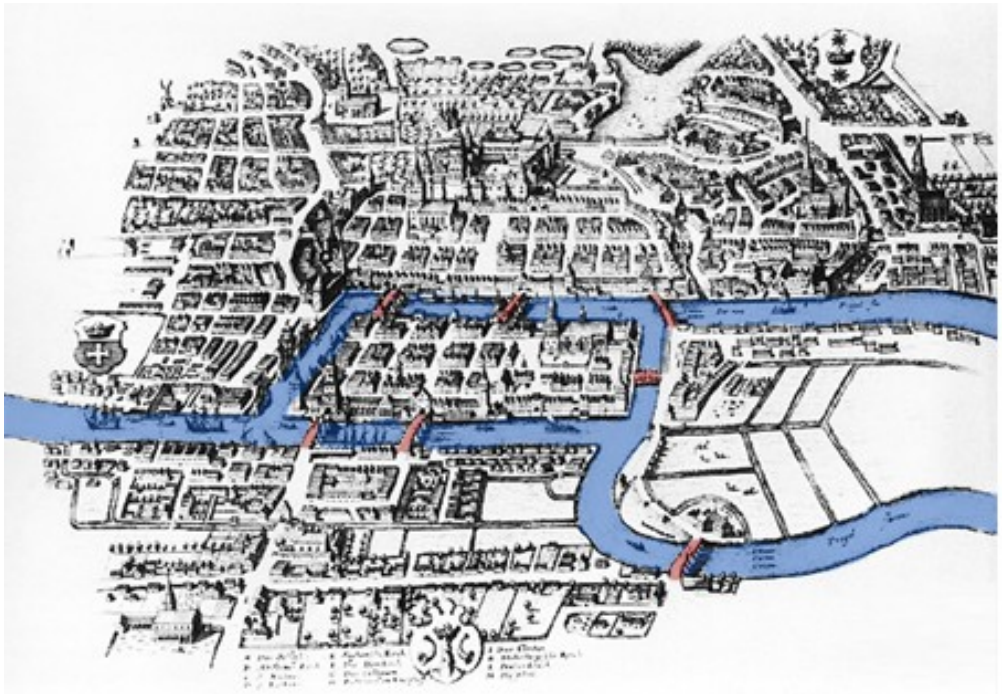
Πιο γενικές δομές





Κάθε φορά που *περνάμε* από μία περιοχή, χρησιμοποιούμε 2 γέφυρες (μία για να μπούμε, μία για να βγούμε). Άρα κάθε κορυφή, εκτός ίσως από την αρχή και το τέλος, έχει άρτιο πλήθος γεφυρών.

Πιο γενικές δομές





Κάθε φορά που *περνάμε* από μία περιοχή, χρησιμοποιούμε 2 γέφυρες (μία για να μπούμε, μία για να βγούμε). Άρα κάθε κορυφή, εκτός ίσως από την αρχή και το τέλος, έχει άρτιο πλήθος γεφυρών.

Εδώ οι βαθμοί είναι $3, 3, 3, 5$: τέσσερις περιττοί. Άρα **δεν** γίνεται.

Το θεώρημα του Euler

Θεώρημα

Ένα συνεκτικό γράφημα (επιτρέπονται πολλαπλές ακμές) έχει διαδρομή που περνάει από κάθε ακμή ακριβώς μία φορά αν και μόνο αν έχει 0 ή 2 κορυφές περιττού βαθμού.

- ($\Rightarrow$) Euler, 1736: το επιχείρημα της προηγούμενης διαφάνειας.
- ($\Leftarrow$) Hierholzer, 1873.

Με αυτή τη δουλειά του Euler ξεκίνησε η Θεωρία Γραφημάτων.

Γιατί μας απασχολούν τα γραφήματα;

Μοντελοποιούν:

- Κοινωνικά δίκτυα,
- Δίκτυα,
- Οικονομικές συναλλαγές,
- Χημικές δομές,
- Βιολογία,

⋮

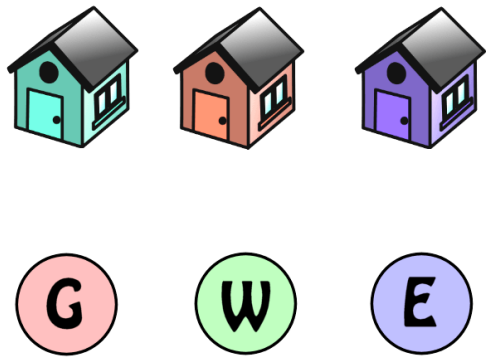
Αλγοριθμική Θεωρία Γραφημάτων

Προσπαθούμε να βρούμε:

Δομικές ιδιότητες των γραφημάτων που θα μας βοηθήσουν να σχεδιάσουμε αποδοτικούς αλγόριθμους.

Παραδείγματα

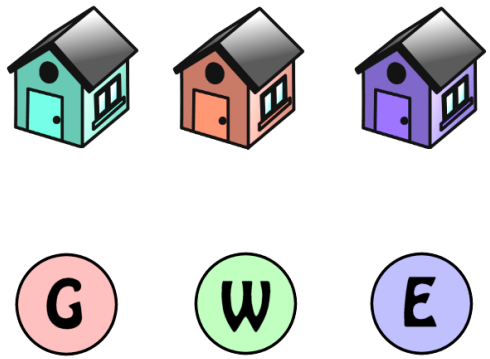
- Δέντρα (συνεκτικά γραφήματα χωρίς κύκλους)
- Επίπεδα Γραφήματα (ζωγραφίζονται στο επίπεδο χωρίς τομές)
- Διμερή (οι κορυφές χωρίζονται σε δύο ομάδες και κάθε ακμή ενώνει κορυφές από διαφορετικές ομάδες)



Ο γρίφος: είναι το διμερές γράφημα $K_{3,3}$ επίπεδο;

Παραδείγματα

- Δέντρα (συνεκτικά γραφήματα χωρίς κύκλους)
- Επίπεδα Γραφήματα (ζωγραφίζονται στο επίπεδο χωρίς τομές)
- Διμερή (οι κορυφές χωρίζονται σε δύο ομάδες και κάθε ακμή ενώνει κορυφές από διαφορετικές ομάδες)



Ο γρίφος: είναι το διμερές γράφημα $K_{3,3}$ επίπεδο;

Όχι: το $K_{3,3}$ δεν είναι επίπεδο, άρα ο γρίφος δεν έχει λύση.

Πίσω στο πάρτι της εταιρείας

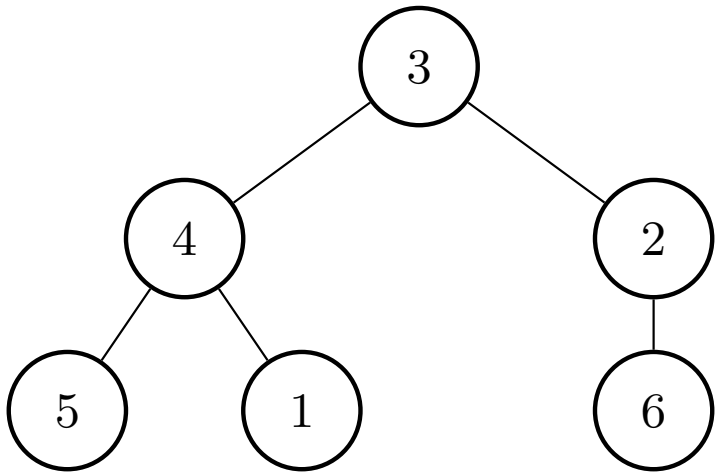
Κορυφές: οι υπάλληλοι. Ακμές: υπάλληλος – άμεσος προϊστάμενος.

Κάθε υπάλληλος, εκτός από τον διευθυντή, έχει ακριβώς έναν άμεσο προϊστάμενο, και ακολουθώντας τους προϊσταμένους φτάνουμε πάντα στον διευθυντή. Άρα το γράφημα είναι δέντρο.

Πίσω στο πάρτι της εταιρείας

Κορυφές: οι υπάλληλοι. Ακμές: υπάλληλος – άμεσος προϊστάμενος.

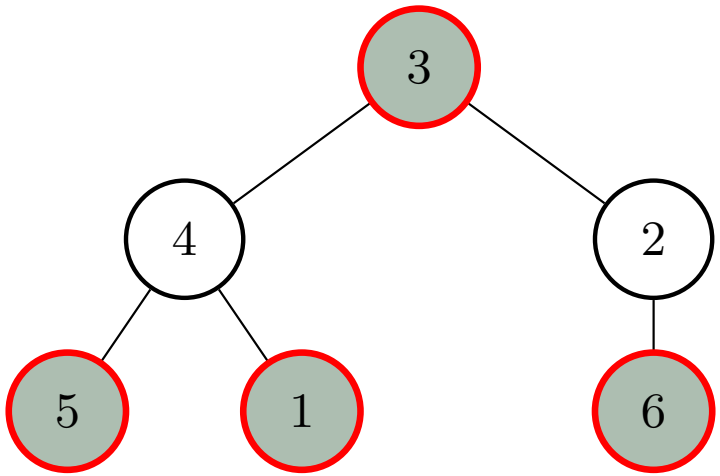
Κάθε υπάλληλος, εκτός από τον διευθυντή, έχει ακριβώς έναν άμεσο προϊστάμενο, και ακολουθώντας τους προϊσταμένους φτάνουμε πάντα στον διευθυντή. Άρα το γράφημα είναι δέντρο.



Πίσω στο πάρτι της εταιρείας

Κορυφές: οι υπάλληλοι. Ακμές: υπάλληλος – άμεσος προϊστάμενος.

Κάθε υπάλληλος, εκτός από τον διευθυντή, έχει ακριβώς έναν άμεσο προϊστάμενο, και ακολουθώντας τους προϊσταμένους φτάνουμε πάντα στον διευθυντή. Άρα το γράφημα είναι δέντρο.



Βέλτιστη επιλογή: συνολικός χαβαλές $3 + 5 + 1 + 6 = 15$.

Η δομή κάνει τη διαφορά

Πρόβλημα: βρες το μεγαλύτερο σύνολο κορυφών χωρίς καμία ακμή μεταξύ τους (*ανεξάρτητο σύνολο*).
(π.χ. k άγνωστοι μεταξύ τους στο γράφημα των γνωριμιών)

Γράφημα	Πόσο γρήγορα;
Οποιοδήποτε	NP-δύσκολο: δεν ξέρουμε πολυωνυμικό αλγόριθμο
Δέντρο	$O(n)$ (το πάρτι της εταιρείας)
Διμερές	πολυωνυμικός χρόνος (ταιριάσματα, θεώρημα König)

Το ίδιο πρόβλημα: δύσκολο γενικά, εύκολο όταν το γράφημα έχει δομή.

Πού θα τα δείτε ξανά

- Κ17. Αλγόριθμοι και Πολυπλοκότητα: πώς λύνουμε προβλήματα αποδοτικά.

- ΘΠ20. Υπολογιστική Πολυπλοκότητα (και μεταπτυχιακό): κατάταξη προβλημάτων με βάση την ταχύτητα επίλυσης.

- Μεταπτυχιακό μάθημα Αλγοριθμικής Θεωρίας Γραφημάτων (στο ΠΜΣ Πληροφορική και στο ΔΠΜΣ Αλγόριθμοι, Λογική και Διακριτά Μαθηματικά).

Α.Λ.ΜΑ

Αλγόριθμοι, Λογική και Διακριτά Μαθηματικά: Μεταπτυχιακό Πρόγραμμα Ειδίκευσης.

Συνδιοργανώνεται από

- το ΕΚΠΑ: Τμήμα Πληροφορικής και Τηλεπικοινωνιών, Τμήμα Μαθηματικών,
- το ΕΜΠ: Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών, Σχολή Εφαρμοσμένων Μαθηματικών και Φυσικών Επιστημών.

Τρεις κεντρικές περιοχές:

- Αλγόριθμοι και Υπολογιστική Πολυπλοκότητα
- Μαθηματική και Υπολογιστική Λογική
- Διακριτά Μαθηματικά

`alma.di.uoa.gr`

Ευχαριστώ πολύ για την προσοχή σας!

