helios.ntua.gr

# Θεμελιώδη Θέματα Επιστήμης Υπολογιστών

## 3ο εξάμηνο ΣΗΜΜΥ

## Εισαγωγή

Διδάσκοντες: Γιώργος Αλεξανδρίδης, Στάθης Ζάχος, Βηρένα Καντερέ, Άρης Παγουρτζής, Γιώργος Στάμου, Δώρα Σούλιου

Βοηθοί διδασκαλίας: Αγγέλα Χαλκή, Pouran Behrouz, Μαριάννα Σπυράκου

# Οργανωτικά του μαθήματος

- Εξ αποστάσεως διαλέξεις (MS-Teams)
- Δια ζώσης συναντήσεις για επίλυση αποριών, ασκήσεις, κ.λπ.
- 3-4 σειρές ασκήσεων: γραπτές και προγραμματιστικές
  - Παρουσίαση των λύσεων
  - «Άριστα»: 2 μονάδες
- ΤΒ = ΒΕ * (1 + ΒΑ / 10)

# Τι είναι η Επιστήμη των Υπολογιστών;

- Μήπως είναι η επιστήμη που μελετάει τους υπολογιστές;

# Edsger W. Dijkstra (1930-2002)

"*Computer Science is no more about computers than astronomy is about telescopes*."

«Η επιστήμη των υπολογιστών έχει ως αντικείμενο τους υπολογιστές όσο και η αστρονομία τα τηλεσκόπια»

# Άλλα ονόματα

- Informatics (Πληροφορική)
- Computing Science (Επιστήμη Υπολογισμών)

# Επιστήμη των Υπολογιστών

Ο επιστημονικός και τεχνολογικός κλάδος που:

- μελετάει την αναπαράσταση, αποθήκευση, επεξεργασία και μετάδοση πληροφοριών μέσω υπολογιστών και δικτύων
- αναζητά και εξετάζει τρόπους (αλγόριθμους, δομές δεδομένων, γλώσσες προγραμματισμού, αρχιτεκτονικές) για την αποδοτική υλοποίηση των παραπάνω εργασιών

# Κεντρικό ερώτημα της επιστήμης υπολογιστών

Τι μπορεί να μηχανοποιηθεί και μάλιστα αποδοτικά;

Ποια προβλήματα μπορούμε να λύσουμε με υπολογιστή, πώς και πόσο καλά;

# Παράδειγμα: τι λείπει από το $2^{29}$ ;

Ποιο ψηφίο λείπει από τον αριθμό $2^{29}$ ;

(αποτελείται από 9 διαφορετικά ψηφία, λείπει ένα)

# Τι λείπει από το $2^{29}$ ;

Πόσες πράξεις χρειαζόμαστε;

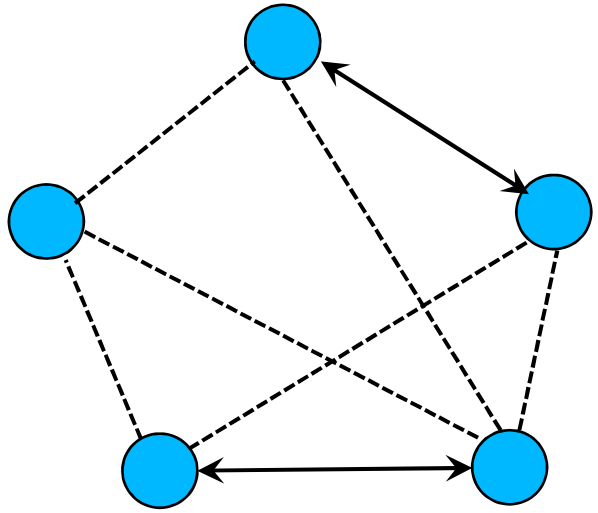
Γίνεται καλύτερα;

# Τι λείπει από το $2^{29}$ ;

Γίνεται χωρίς να υπολογίσουμε ολόκληρο τον αριθμό;

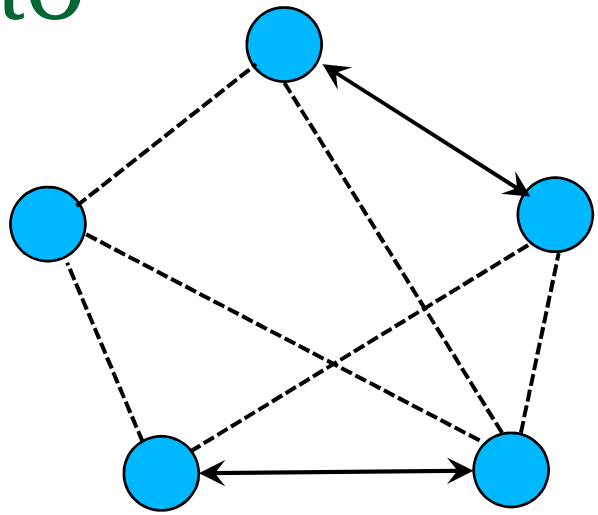
( ερώτηση από βιβλίο προετοιμασίας για συνεντεύξεις σε ‘quant jobs’ )

# Παράδειγμα: ψηφιακό κουτσομπολιό



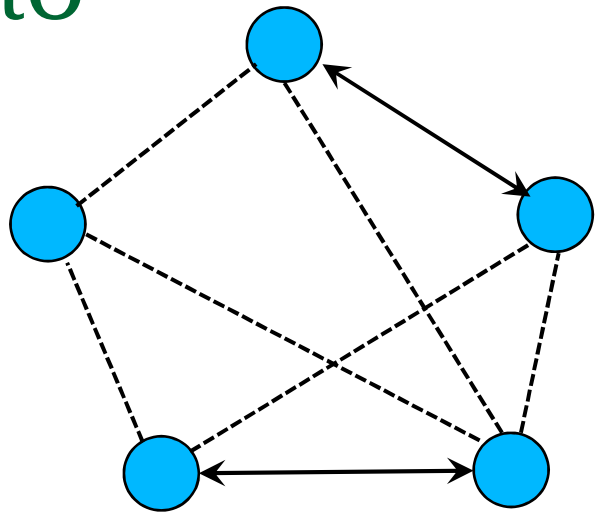
- Δίκτυο με *n* συμμετέχοντες («παίκτες»)
- Όλοι διαθέτουν από ένα μυστικό
- Μπορούν να καλέσουν ο ένας τον άλλο και να του πουν ό,τι γνωρίζουν
- Πώς μπορούν να μοιραστούν τα μυστικά τους γρήγορα;

# Ψηφιακό κουτσομπολιό



- Ποιος είναι το μικρότερο πλήθος κλήσεων;
- Ας ξεκινήσουμε από ένα μυστικό
- Πώς γενικεύουμε;
- Μπορούμε καλύτερα;

# Ψηφιακό κουτσομπολιό



- Γίνεται καλύτερα;
- Καλύτερα από $2n-2$ ;
- Πόσο καλύτερα;
- Πώς το αποδεικνύουμε;
- Πώς θα ξέρουμε ότι βρήκαμε το βέλτιστο;
- Παραλληλοποίηση;

# Κλάδοι επιστήμης υπολογιστών (i)

- Υπολογισιμότητα και πολυπλοκότητα
- Μοντελοποίηση: αυτόματα, γράφοι, λογική
- Αλγόριθμοι και δομές δεδομένων
- Γλώσσες προγραμματισμού και μεταγλωττιστές
- Τεχνολογία λογισμικού
- Βάσεις δεδομένων και διαχείριση πληροφοριών
- Αρχιτεκτονική και οργάνωση υπολογιστών
- Λειτουργικά - παράλληλα - κατανεμημένα συστήματα

# Κλάδοι επιστήμης υπολογιστών (ii)

- Δίκτυα υπολογιστών και τεχνολογίες διαδικτύου
- Ενσωματωμένα συστήματα
- Τεχνητή νοημοσύνη, μηχανική μάθηση
- Επικοινωνία ανθρώπου – μηχανής, πολυμέσα
- Κρυπτογραφία, e-voting, bitcoin
- Υπολογιστική βιολογία, βιολογικοί υπολογισμοί
- Κβαντικοί υπολογισμοί
- Ανάλυση κοινωνικο-οικονομικών δικτύων

# Είπαν…

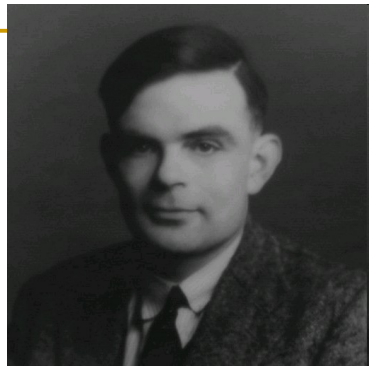

Dijkstra:

«Το ερώτημα εάν ο *υπολογιστής σκέφτεται* δεν είναι πιο ενδιαφέρον από το ερώτημα εάν το *υποβρύχιο κολυμπάει*»

Tom Watson, IBM, 1945:

«Ο κόσμος δεν χρειάζεται περισσότερους από *πέντε υπολογιστές*»

# Θεωρητικές θεμελιώσεις



- Υπολογισιμότητα: *ποιά* προβλήματα μπορούμε να λύσουμε;
- Αλγόριθμοι: *πώς* μπορούμε να τα λύσουμε;
- Υπολογιστική πολυπλοκότητα: *πόσο καλά* μπορούμε να τα λύσουμε;
  - ως προς το χρόνο
  - ως προς το χώρο/μνήμη
  - ως προς # επεξεργαστών
  - ως προς # μηνυμάτων (bandwidth)
  - ως προς την κατανάλωση ενέργειας
  - ...

# Παραδείγματα υπολογισμού

- Αριθμοί Fibonacci
- Κύκλος Euler – κύκλος Hamilton
- Κρυπτογραφικοί αλγόριθμοι

# Αριθμοί Fibonacci

- 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, ...

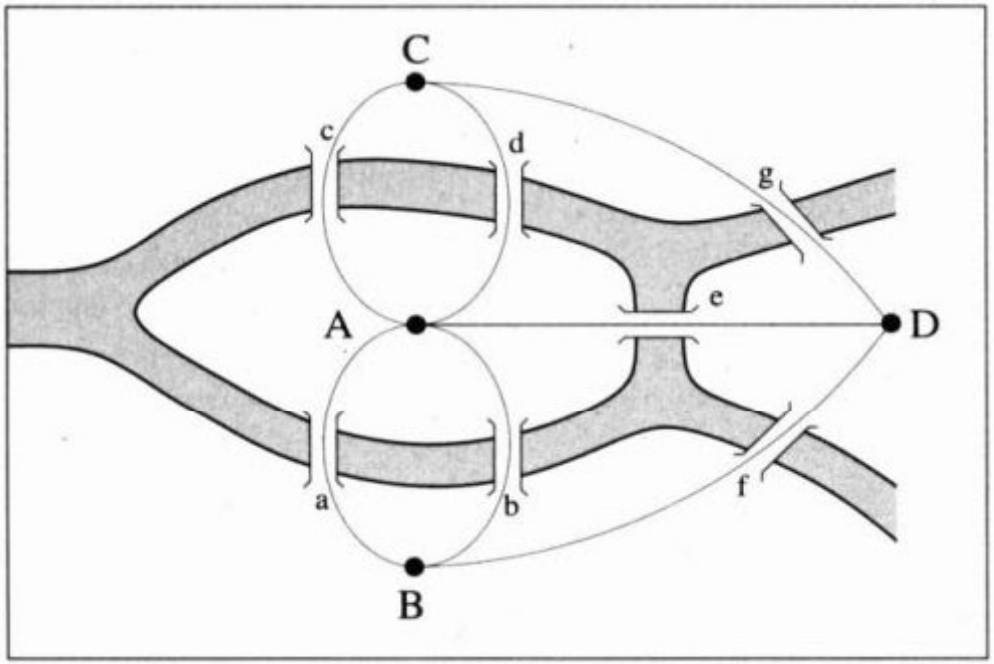
$$F_0 = 0,\ F_1 = 1$$

$$F_n = F_{n-1} + F_{n-2},\ n >= 2$$

- Πρόβλημα: Δίνεται $n$, να υπολογιστεί το $F_n$
- Πόσο αργό μπορεί να γίνει το πρόγραμμά μας;
- Πόσο γρήγορα μπορούμε να απαντήσουμε;

# Το πρόβλημα του Euler

Δίνεται γράφος. Υπάρχει τρόπος να περάσουμε από κάθε ακμή μια ακριβώς φορά;

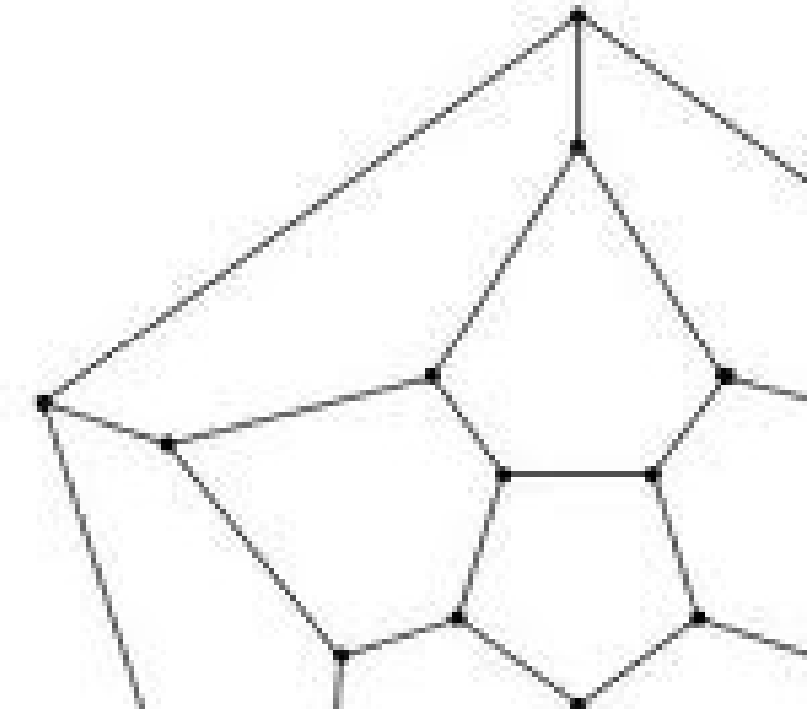



**Seven Bridges of Königsberg**

Source:
http://physics.weber.edu/carroll/honors_images/BarbasiBridges.jpg

# Το πρόβλημα του Hamilton

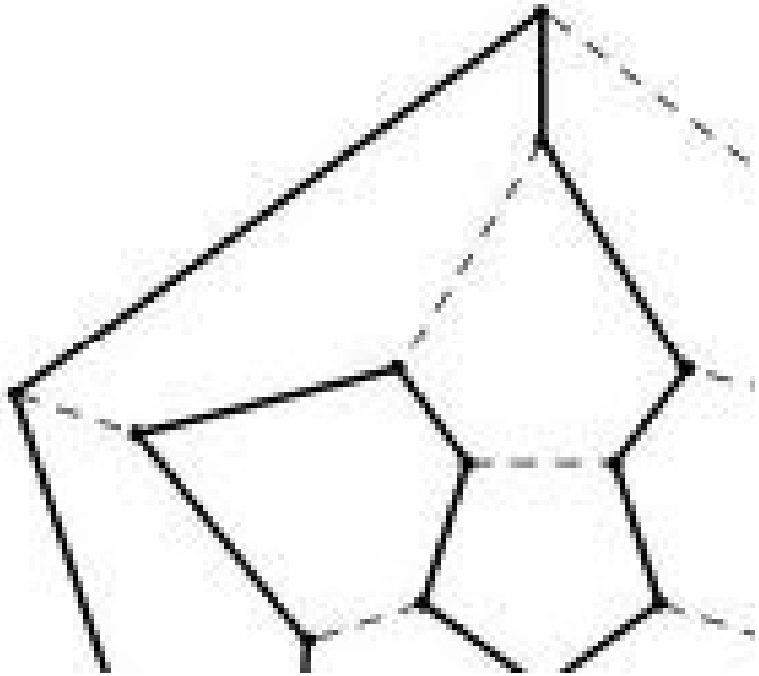
Δίνεται γράφος.
Υπάρχει τρόπος
να περάσουμε
από κάθε κορυφή
μια ακριβώς
φορά;



Source:
http://jwilson.coe.uga.edu/emat6680/yamaguchi/emat6690/essay1/gt.html

# Το πρόβλημα του Hamilton

Δίνεται γράφος.
Υπάρχει τρόπος
να περάσουμε
από κάθε κορυφή
μια ακριβώς
φορά;



Source:
http://jwilson.coe.uga.edu/emat6680/yamaguchi/emat6690/essay1/gt.html

# P =? NP

- Μπορεί να λυθεί το πρόβλημα του Hamilton τόσο γρήγορα όσο και το πρόβλημα του Euler;
- Αυτό είναι ουσιαστικά το P =? NP πρόβλημα, που αποτελεί το πιο σημαντικό ανοικτό πρόβλημα της Θεωρητικής Πληροφορικής σήμερα.
- Στο http://www.claymath.org προσφέρονται 1εκ. δολάρια για τη λύση του !
- Προβλήματα *ενδιάμεσης* πολυπλοκότητας: πολύ σημαντικές εφαρμογές!

# Από τη θεωρία στις εφαρμογές

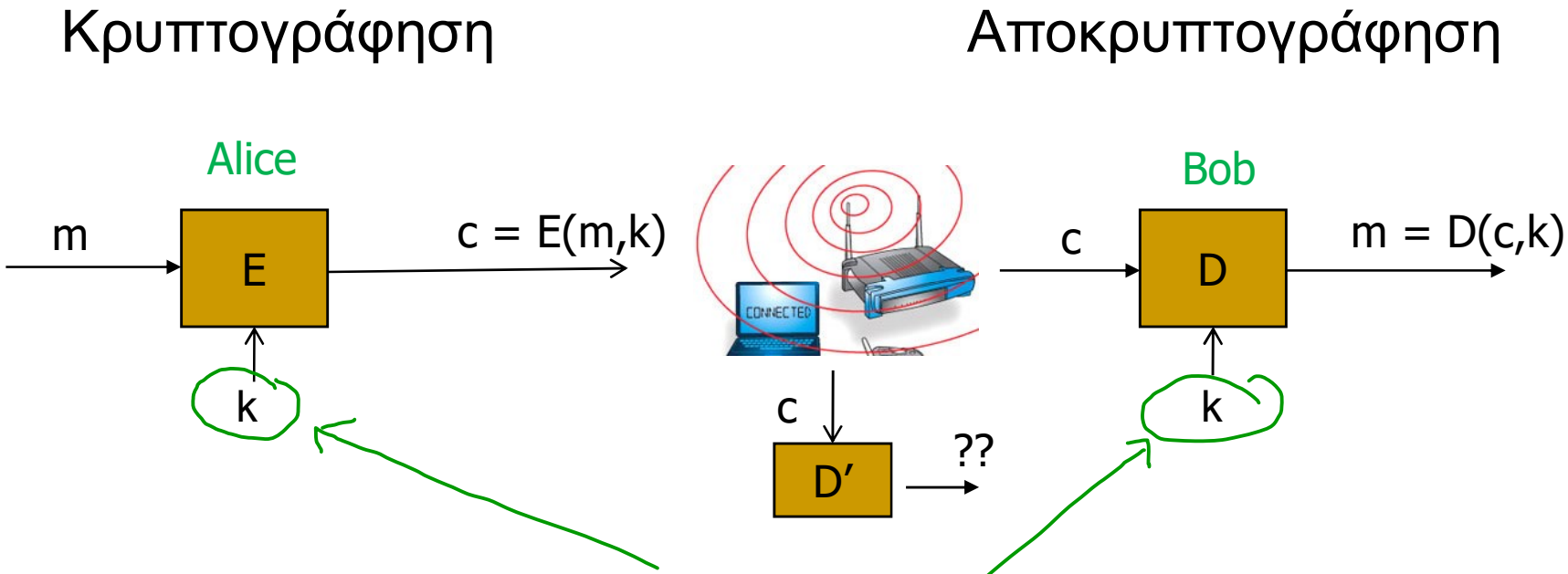
- Κρυπτογραφία
- Τεχνητή Νοημοσύνη
- Βάσεις Δεδομένων
- ... και πολλές άλλες

# Κρυπτογραφία

Κρυπτογράφηση

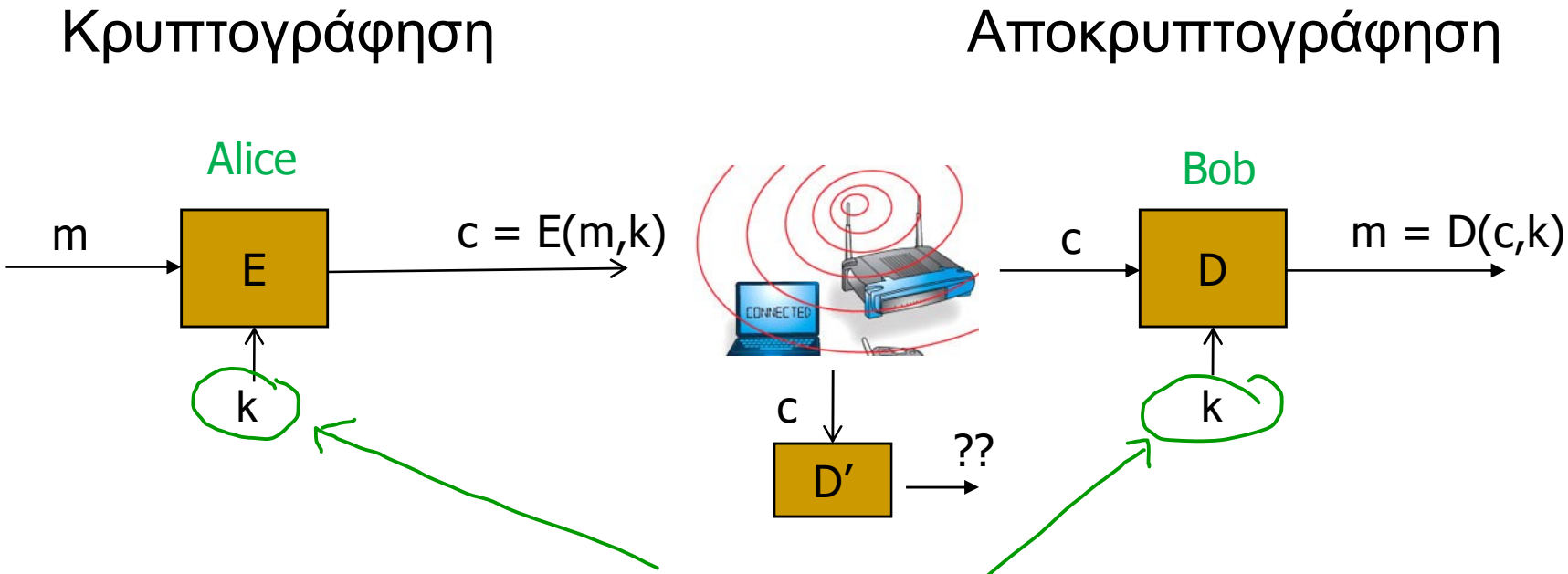
Αποκρυπτογράφηση

# Κρυπτογραφία ιδιωτ. κλειδιού



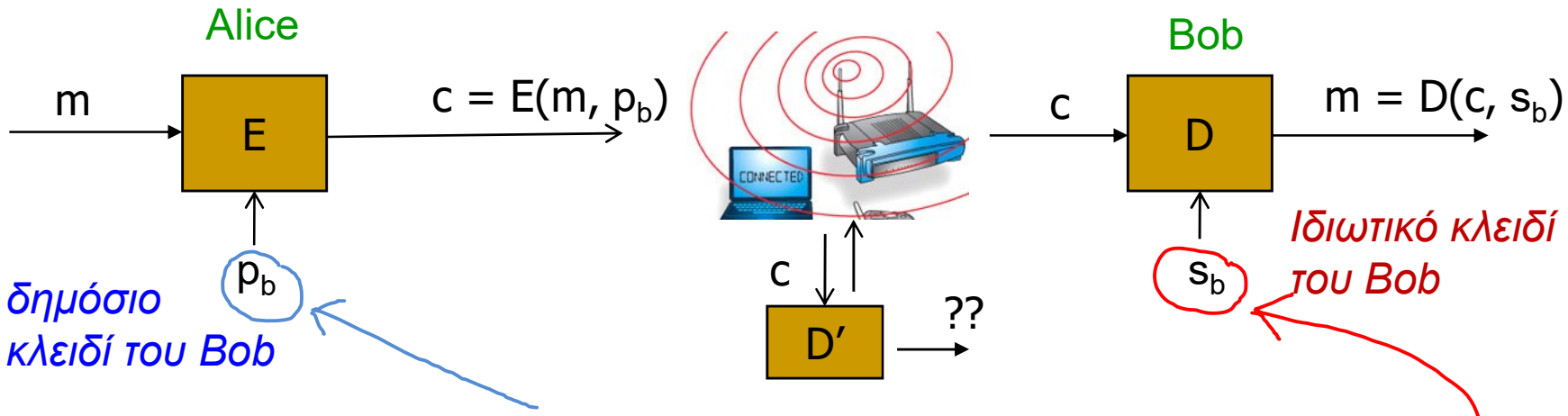

- … με χρήση *κοινού ιδιωτικού κλειδιού* (συμμετρική κρυπτογραφία)

# Πρόβλημα: ανταλλαγή κλειδιού




- … με χρήση *κοινού ιδιωτικού κλειδιού* (συμμετρική κρυπτογραφία)
- Ασφαλής ανταλλαγή *εξ αποστάσεως*;

# Λύση: κρυπτογραφία δημόσιου κλειδιού




- … με χρήση δημοσίου κλειδιού (κρυπτογραφία μονής κατεύθυνσης), μαζί με *απόλυτα* ιδιωτικό, γνωστό στον παραλήπτη μόνο

# Κρυπτογραφία δημοσίου κλειδιού

- Κρυπτογραφία δημοσίου κλειδιού: κατάργησε την ανάγκη ανταλλαγής κλειδιών! Στηρίζεται στην ύπαρξη συναρτήσεων μονής κατεύθυνσης.
- Συναρτήσεις μονής κατεύθυνσης (one-way functions): *εύκολο* να υπολογιστούν, *δύσκολο* να αντιστραφούν
- Κρυπτοσύστημα RSA [Rivest-Shamir-Adleman, 1977]
  - κρυπτογράφηση: $c = m^e \mod n$
  - αποκρυπτογράφηση: $m = c^d \mod n$
  - δημόσιο κλειδί: ***e, n***
  - ιδιωτικό κλειδί: ***d***

# Παράδειγμα RSA [http://nmichaels.org/rsa.py]

- κρυπγ/ση: $c = m^e \mod n$ απκργ/ση: $m = c^d \mod n$
- δημόσιο κλειδί (1ο μέρος) $n$ =
  d543be11021217e30589b41f796fac8f54a8905a4ddcd2007e2d004
  7d7b751a1aa60db5a080545a4ee2b33a2a119cc7aa3ff5b022d895
  4eeb5b72d1eec7cf40dfdc7947da9f49009c62be9d89fda3c71137bb
  d009d3631bfa83bcde81a7bbc261890d2edd2fb20a4f0cb904b40bd
  5662c3c006634a7fcd7eae87a6d494e5fb5 (hex)
- δημόσιο κλειδί (2ο μέρος) $e$ = 10001 (hex)
- ιδιωτικό κλειδί: $d$ =
  47b5fb04312ecb57d78a082c8151ff65547b49d108743678b663f37
  46feeee18d81523463327c84b786ba78515601c69081437c3e23ef
  4b6b2b0ad99d47e7c0228333da1594f774c8a73d4093f476635557
  209945423cbd1e9b6a358f8254ed831c30d61f85cf57a49b8c7b1a2
  1282d2fad548c12aa10f2ed0e5ccd5c7e32841 (hex)

# ΠΑΡΑΓΟΝΤΟΠΟΙΗΣΗ

- Δίνεται σύνθετος αριθμός *n*, βρείτε τους πρώτους παράγοντές του:

  12301866845301177551304949583849627207728535695953347921973224521517264005072636575187452021997864693899564749427740638459251925573263034537315482685079170261221429134616704292143116022212404792747377940806653514195974598569021434 13

  =

  33478071698956898786044169848212690817704794983713768568912431388982883793878002287614711652531743087737814467999489

  x

  36746043666799590428244633799627952632279158164343087642676032283815739666511279233373417143396810270092798736308917

# Παραγοντοποιηση

- *Κοινή πεποίθηση:* υπολογιστικά δύσκολο (όχι στην κλάση **P**)
- *Ευεπίλυτο με* κβαντικό υπολογιστή
- Αν αποδειχθεί πρακτικά ευεπίλυτο «εξανεμίζεται» η ασφάλεια του κρυπτοσυστήματος RSA

# Βασικά συστατικά του RSA

- Υπολογιστική ευκολία της ύψωσης σε δύναμη (modulo *p*) αριθμού χιλιάδων ψηφίων, με εκθέτη χιλιάδων ψηφίων
- Υπολογιστική ευκολία ελέγχου και εύρεσης πρώτων αριθμών με χιλιάδες ψηφία
- Υπολογιστική ευκολία υπολογισμού αντιστρόφου a modulo *n* (a, *n* με χιλιάδες ψηφία) – μέσω αλγορίθμου Ευκλείδη!
- Υπολογιστική δυσκολία παραγοντοποίησης αριθμών με χιλιάδες ψηφία

# Σημασία πολυωνυμικού χρόνου

- Έχει ταυτιστεί με την υπολογιστική ευκολία
- Επιτρέπει (συνήθως) την επίλυση πολύ μεγάλων «στιγμιοτύπων» (εισόδων)
- Πρακτικά και «χοντρικά»:

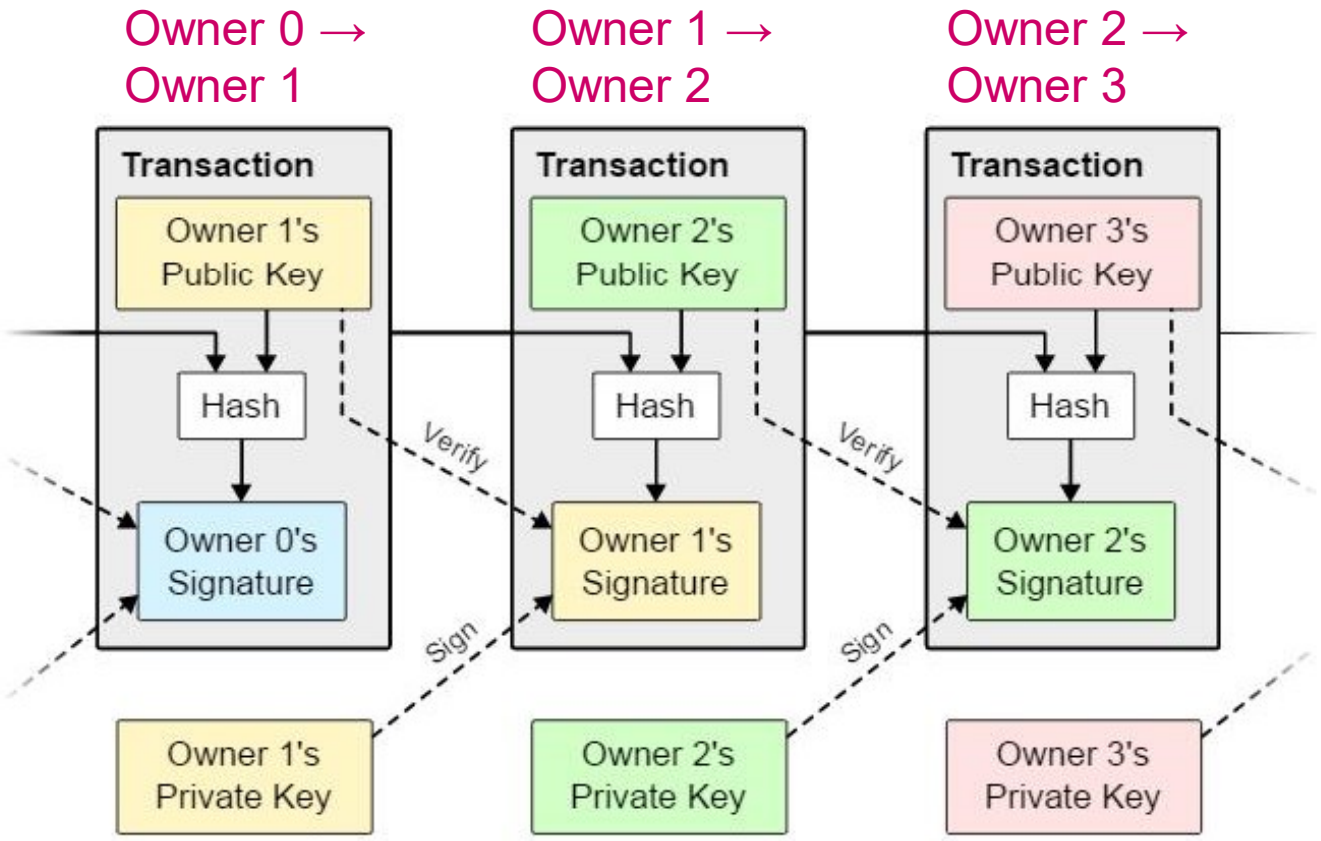
  *αν μπορείς να το γράψεις μπορείς και να το υπολογίσεις!*

# Εφαρμογές κρυπτογραφίας

- Ψηφιακές υπογραφές
- Ασφάλεια επικοινωνιών
- Ασφάλεια συναλλαγών
- Ηλεκτρονικές ψηφοφορίες – και άλλοι υπολογισμοί χωρίς αποκάλυψη δεδομένων!
- Bitcoin: νομίσματα και συναλλαγές χωρίς μεσάζοντες!

# Bitcoin [Satoshi Nakamoto 2008]

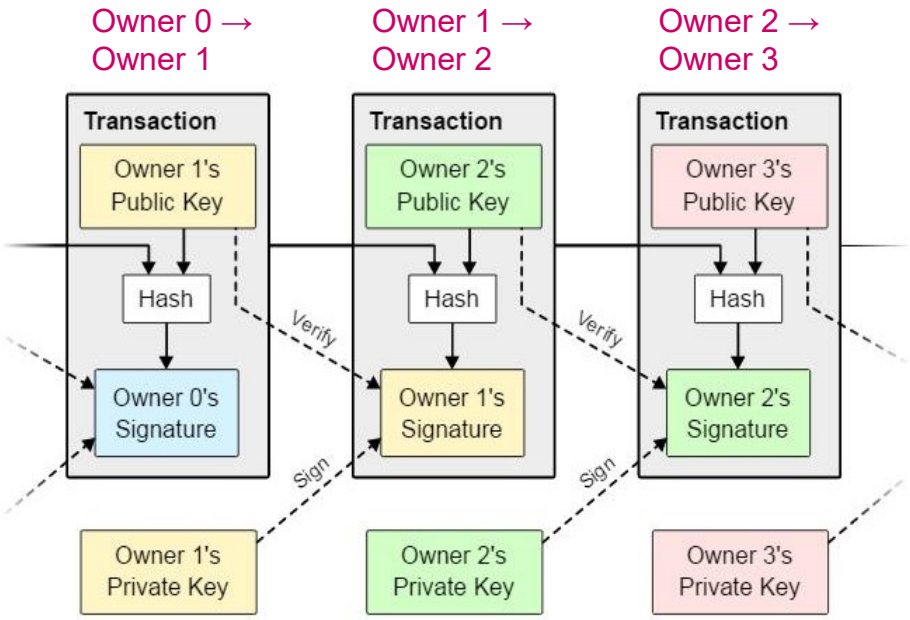
- Μια **επανάσταση** σε εξέλιξη!



# Bitcoin [Satoshi Nakamoto 2008]

- Μια **επανάσταση** σε εξέλιξη!

- Πρωτόκολλο bitcoin: στηρίζεται σε ευκολία και δυσκολία επίλυσης υπολογιστικών προβλημάτων



# Πρώτα συμπεράσματα

- Πολλές επαναστατικές ιδέες και εφαρμογές του καιρού μας στηρίζονται στον υπολογισμό
- ιδιαίτερα στο πόσο εύκολα (γρήγορα) μπορούμε να υπολογίσουμε κάποια πράγματα
- και στο πόσο δύσκολο είναι να υπολογίσουμε κάποια άλλα

# Τεχνητή Νοημοσύνη

- Αντικείμενο:
  - Μελέτη *διαδικασίας σκέψης και συλλογισμού* ή της *ευφυούς συμπεριφοράς* ή κατασκευή μηχανών που **αντιμετωπίζουν προβλήματα που προς το παρόν τα επιλύουν καλύτερα οι άνθρωποι**
- Στόχοι της ΤΝ είναι η ανάπτυξη συστημάτων που:
  - «Σκέφτονται» όπως οι άνθρωποι
  - «Συμπεριφέρονται» όπως οι άνθρωποι
  - «Σκέφτονται» λογικά
  - Αντιδρούν λογικά

# Αξιολόγηση

- **Δοκιμασία Turing**
  - Ο υπολογιστής περνά τη δοκιμασία αν ένας άνθρωπος-εξεταστής, αφού αλληλεπιδράσει με το σύστημα, *δεν μπορεί να συμπεράνει* αν οι αποκρίσεις προέρχονται από **άνθρωπο** ή από **μηχανή**
- Ικανότητες μηχανών για να περάσουν τη δοκιμασία
  - Αντίληψη
    - Επεξεργασία σημάτων από αισθητήρες, αλληλεπίδραση με άνθρωπο, …
  - Γνώση, συλλογισμός, διορατικότητα, προσαρμοστικότητα, δημιουργικότητα, …

# Εφαρμογές

- Παίγνια
- Απόδειξη θεωρημάτων
- Δημιουργικά συστήματα
- Αυτόνομα ρομπότ
- Σχεδιασμός ενεργειών και χρονοπρογραμματισμός
- Ευφυείς υπηρεσίες διαδικτύου
- Ιατρική, νομικές επιστήμες, βιοτεχνολογία, γεωπονία, διαστημική τεχνολογία, αρχαιολογία, …
- Όραση υπολογιστών, επεξεργασία φυσικής γλώσσας, ανάλυση δεδομένων, εξόρυξη γνώσης, …

# Επίλυση Προβλημάτων

- Βασικό χαρακτηριστικό ΤΝ
  - Βρισκόμαστε σε μια **αρχική κατάσταση** (initial state)
  - Έχουμε ένα **στόχο** (goal)
  - Μπορούμε να επιλέξουμε μια ενέργεια για εκτέλεση από **ένα σύνολο διαθέσιμων ενεργειών** (set of actions)
  - Μετά από **σκέψη**, πρέπει να βρούμε την **αλληλουχία των ενεργειών** (set of actions) που οδηγούν στην επίτευξη του στόχου (**λύση**)
  - Η κάθε ενέργεια έχει ένα **κόστος** (cost) που επηρεάζει την απόφασή μας
  - Πράκτορας επίλυσης προβλημάτων

# Επίλυση Προβλημάτων

- Μπορεί να υπάρχουν πολλές λύσεις για το ίδιο πρόβλημα!
  - Κάποιες είναι καλύτερες και κάποιες χειρότερες από πλευράς κόστους
- Επιθυμούμε να αναθέσουμε την επίλυση του προβλήματος σε ευφυή πράκτορα
  - Πράκτορας επίλυσης προβλημάτων

# Διαχείριση Δεδομένων

Η διαχείριση δεδομένων αφορά στις διαδικασίες:

- Λήψης πρωτογενών δεδομένων που παράγονται από διάφορες πηγές δεδομένων
- Αποθήκευσης των δεδομένων σε διάφορα μέσα
- Οργάνωσης των δεδομένων σε συγκεκριμένες δομές
- Επεξεργασίας των δεδομένων
- Συντήρησης και επικαιροποίησης των δεδομένων

# Διαχείριση Δεδομένων

Τα δεδομένα καθώς και η ανάγκη για τη συστηματική τους διαχείριση βρίσκονται παντού

Κλασσικά παραδείγματα εφαρμογών διαχείρισης δεδομένων σχετίζονται με:

- Διάφορα κρατικά συστήματα
- Τραπεζικά συστήματα
- Φυσικά και on-line καταστήματα
- Αεροπορικές εταιρίες
- Επιστημονική έρευνα
- .... και άλλα πολλά

# Διαχείριση Δεδομένων

Από τη στιγμή που εμφανίστηκαν οι υπολογιστές, εμφανίστηκε και η ανάγκη για διαχείριση δεδομένων:

- Αρχή με διάτρητες κάρτες (punch cards) τη δεκαετία του 50
- Αναπτύχθηκε το σχεσιακό μοντέλο και οι σχεσιακές βάσεις τις δεκαετίες του 70-80
- Αναπτύχθηκαν άλλα μοντέλα και βάσεις τη δεκαετία του 90
- Τα δεδομένα άρχισαν να αυξάνονται με την ολοένα μεγαλύτερη χρήση του διαδικτύου και την αλματώδη πρόοδο της τεχνολογίας από το 2000 και μετά
- Σήμερα μιλάμε για διαχείριση Μεγάλων Δεδομένων (Big Data)

# Συνοψίζοντας

- Είναι σημαντικό να γνωρίζουμε τί μπορούμε και τί *δεν* μπορούμε να κάνουμε με τους υπολογιστές, πώς και πόσο καλά
- Αυτό μελετάει η Επιστήμη των Υπολογιστών
- Η εξοικείωση με τις μεθοδολογίες σχεδιασμού και ανάλυσης αλγορίθμων, συστημάτων και εφαρμογών είναι απαραίτητη για να κατανοήσουμε τη σύγχρονη τεχνολογία
- …και για να συμμετέχουμε στην ανάπτυξή της!
- Τα μαθηματικά είναι πάντα επίκαιρα!