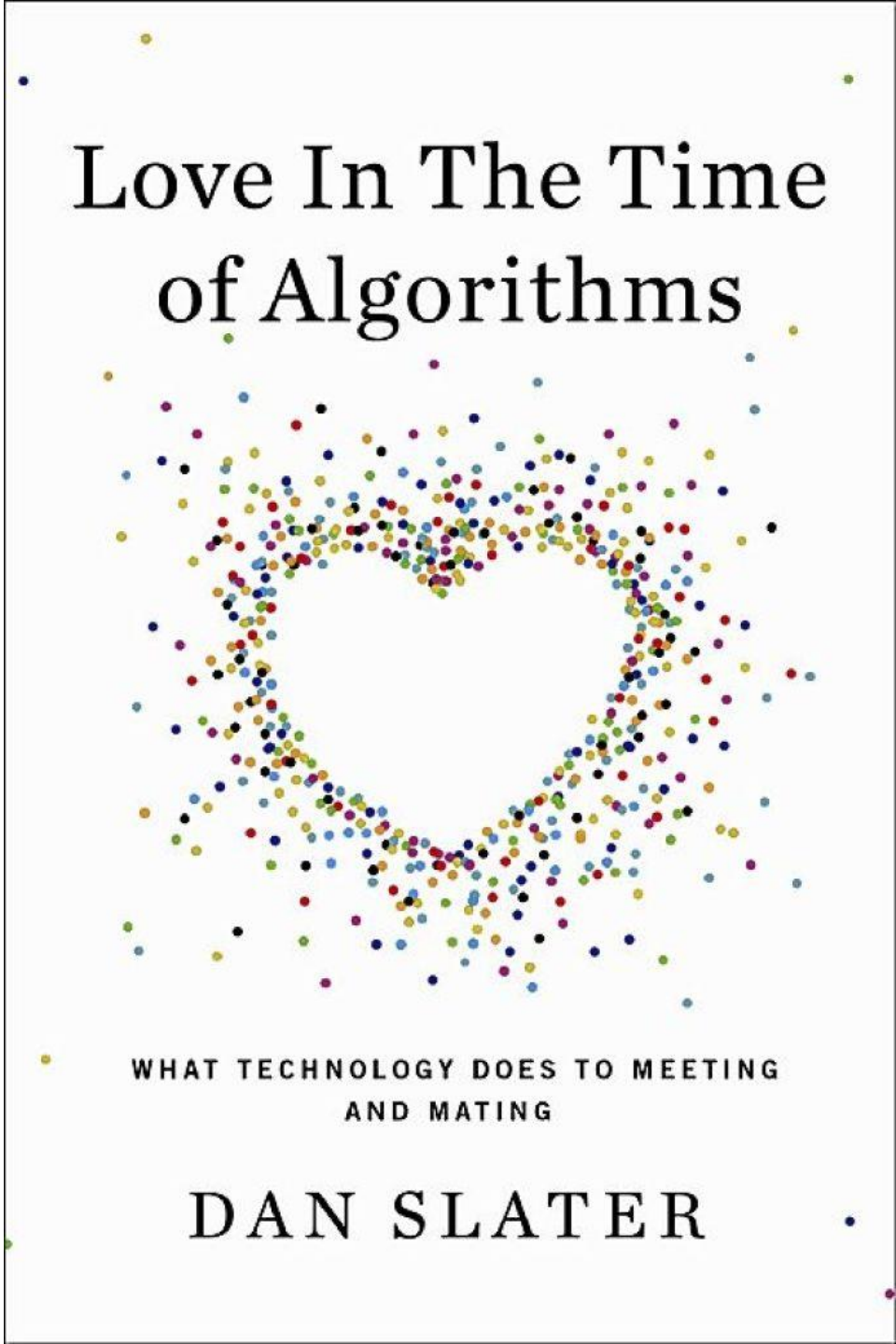
Αλγόριθμοι και Πολυπλοκότητα

"People who analyze algorithms have double happiness. First of all they experience the sheer beauty of elegant mathematical patterns that surround elegant computational procedures.

Then, they receive a practical payoff when their theories make it possible to get other jobs done more quickly and more economically."

- Donald Knuth



Αλγόριθμοι και Πολυπλοκότητα

ΕΞΑΜΗΝΟ 7ο
Υπεύθυνος μαθήματος: Τσίχλας Κωνσταντίνος
e-mail: tsichlas@delab.csd.auth.gr
Ιστότοπος:
http://delab.csd.auth.gr/~tsichlas/algorithms.html
Τηλ: 2310-991934
Γραφείο: Εθνικής Αντιστάσεως 16, 2ος όροφος, Γρ. 29

Πόσο Πολύπλοκος είναι ένας Αλγόριθμος;;;;

- Σελίδα μαθήματος με ημερολόγιο, υλικό, βιβλιογραφία, ανακοινώσεις
- http://delab.csd.auth.gr/~tsichlas/algorithms.html

e-mail: tsichlas@delab.csd.auth.gr
Τηλ: 2310-991934
Γραφείο: Εθνικής Αντιστάσεως 16, 2ος όροφος, Γρ. 29

Κάποιες Συμβουλές

- Διακόψτε με ερωτήσεις (μερικές φορές ξεφεύγω)
- Παρακολουθήστε τις διαλέξεις (ελπίζω να έχουν πλάκα)
- Λύστε ασκήσεις
- Αν δεν καταλαβαίνετε κάτι ελάτε στο γραφείο μου (ίσως το καταλάβουμε μαζί)

Κινούμενο χέρι: Θέλετε κάτι να πείτε σε σχέση με αυτά που ***λέω τώρα***

Σταθερό χέρι: Ερώτηση ή σχόλιο γενικής φύσης

Φουρνίζοντας…

Υλικά:

2 κιλά αλεύρι (περίπου)

1200 ml χλιαρό νερό σε δύο φάσεις, 600+600

70γρ νωπή μαγιά

1 κουταλιά ζάχαρη

2 κουτα

2 κουτα

(προαιρ

(προαιρ

Αλγόριθμος = Συνταγή

1. Ανέβασμα Μαγιάς
2. Ζύμωμα
3. Πλάσιμο Ψωμιού
4. Διπλασίασμα Ψωμιού
5. Φούρνισμα
6. Ξεφούρνισμα

Πόσο γρήγορα φτιάχνω ψωμί;
Πόσο αλεύρι χρειάζομαι;

εξαρτάται από το μέσο

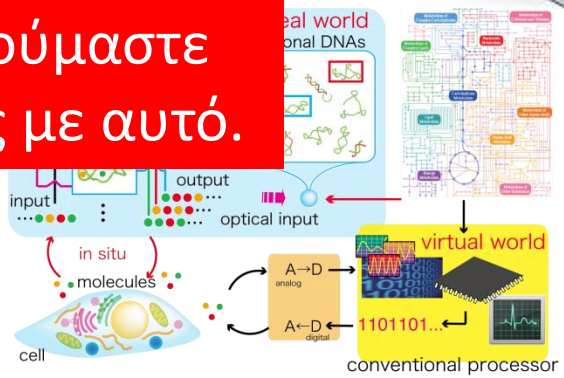
Τι μπορώ να φτιάξω με έναν ζυμωτή;

Ομοίως: Ο Αλγόριθμος εξαρτάται από το μέσο

Ποιο είναι το Μέσο;;;;

- Ένας απλός Η/Υ.
- Ένας DNA Υπολογιστής ☺
- Ένας Κβαντικός Υπολογιστής ☺ ☺
- Σαπουνόφουσκες;;; ☺ ☺ ☺
- …

Ασχολούμαστε κυρίως με αυτό.



Περί Τίνος δεν Πρόκειται

«Οι Η/Υ δεν μπορούν να λύσουν τα πάντα»

Προφανή άλυτα προβλήματα:

- Πως θα γίνω εκατομμυριούχος; (?)
- Πώς θα ενοποιήσω την κβαντική θεωρία με την βαρύτητα; (?)
- Πώς θα δω αν υπάρχει Θεός; (?)

Τα προβλήματα αυτά δεν είναι σωστά ορισμένα.

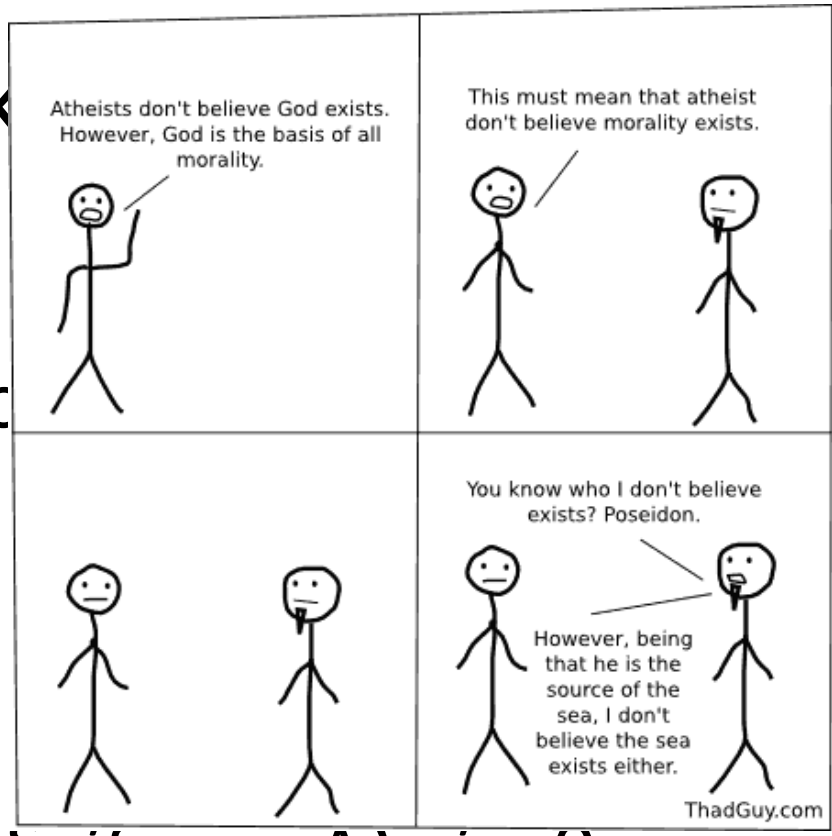
- Δεν είναι «**υπολογιστικά προβλήματα**»

- Ένα σωστά ορισμένο πρόβλημα θα πρέπει να περιγράφει την έξοδο για κάθε δυνατή είσοδο.

Περι τινοσ προκειται

- Πώς λύνουμε προβλήματα;
  Τεχνικές
- Ποιοι είναι οι θεμελιώδεις περιορισμοί των Η/Υ;
  Υπολογισιμότητα/Αποφασισιμότητα
- Τι κάνει τα προβλήματα δύσκολα/εύκολα;
  Κατηγοριοποίηση Προβλημάτων
- Τι πόρους χρειαζόμαστε για να υπολογίσουμε κάτι;
  Χρόνος / Χώρος / «Υλικό» / Πολυπλοκότητα

Η Πραγματικ



- Οι Αλγόριθμοι και Πολυπλο
  - Αποδείξεις + Αλγόριθμοι
  - Περισσότερες αποδείξεις
  - Ακόμα περισσότερες αποδείξεις + Αλγόριθμοι και....
  - Αποδείξεις (καθώς και κάποιες «ωραίες» ιδέες)

Όμως:

- Όχι παραγώγους ή ολοκληρώματα
- Όχι δυωνυμικοί συντελεστές (μην με πιστεύετε κιόλας!!!)
- Όχι πολύπλοκοι υπολογισμοί (χμ...)
- Θα έχει όμως πιθανότητες ☺
- Όχι θεωρία αριθμών
- Κάποια μαθηματική σημειογραφία και η δύναμη της λογικής σκέψης

Η πιο Δύσκολη Ερώτηση:
Γιατί να Πάρω αυτό το Μάθημα;

- Για αυτούς που συνεχίζουν (μεταπτυχιακό ή διδακτορικό) είναι απαραίτητο μιας και εισάγονται σε έννοιες που θα βρουν μπροστά τους.
- Για αυτούς που θα δουλέψουν στην βιομηχανία (άπειρα δύσκολα προβλήματα)
- Για τους υπόλοιπους – η θεμελίωση της Πληροφορικής

❖ Για τον βαθμό;;;;

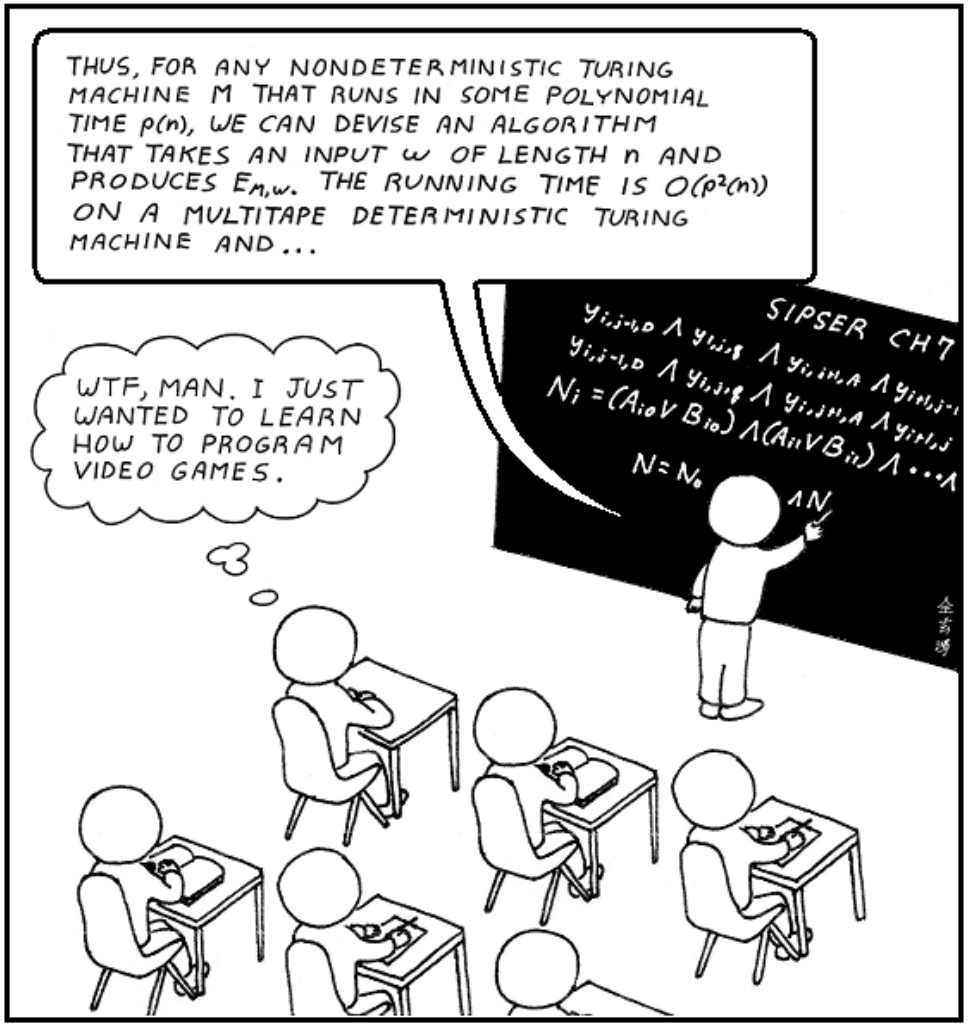
+

- Πρόκληση
- Καλό βιβλίο
- Σημαντικό για εσάς
- Έχει Πλάκα

–

- Δύσκολο
- Αν χάσετε μία διάλεξη θα έχετε πρόβλημα
- «Σαν να χτυπάτε το κεφάλι σας στο τοίχο»

Γιατί να μην Πάρω Αυτό το Μάθημα



Πολυπλοκότητα

Υπολογισιμότητα

- *Τι μπορούμε να υπολογίσουμε;*
- *Μπορεί ένας Η/Υ να λύσει οποιοδήποτε πρόβλημα δοθέντος αρκετού χρόνου και χώρου;*

Πολυπλοκότητα

- *Πόσο γρήγορα μπορούμε να λύσουμε ένα πρόβλημα;*
- *Πόσος χώρος χρειάζεται για να λύσουμε ένα πρόβλημα;*

Αυτόματα

- *Τι προβλήματα μπορούμε να λύσουμε με πολύ λίγο χώρο;*

Πολυπλοκότητα

Τι προβλήματα λύνει ένας Η/Υ;

Όχι όλα!!!

π.χ. Δοθέντος ενός προγράμματος σε Java δεν μπορούμε να ελέγξουμε αν θα τερματίσει σωστά!

Ο έλεγχος ορθότητας προγραμμάτων είναι **αδύνατος**!

(Ο καημός της Microsoft και δικός μας!)

Πολυπλοκότητα

Υπολογισιμότητα

Πολυπλοκότητα

Αυτόματα

Τι προβλήματα λύνει ένας Η/Υ;

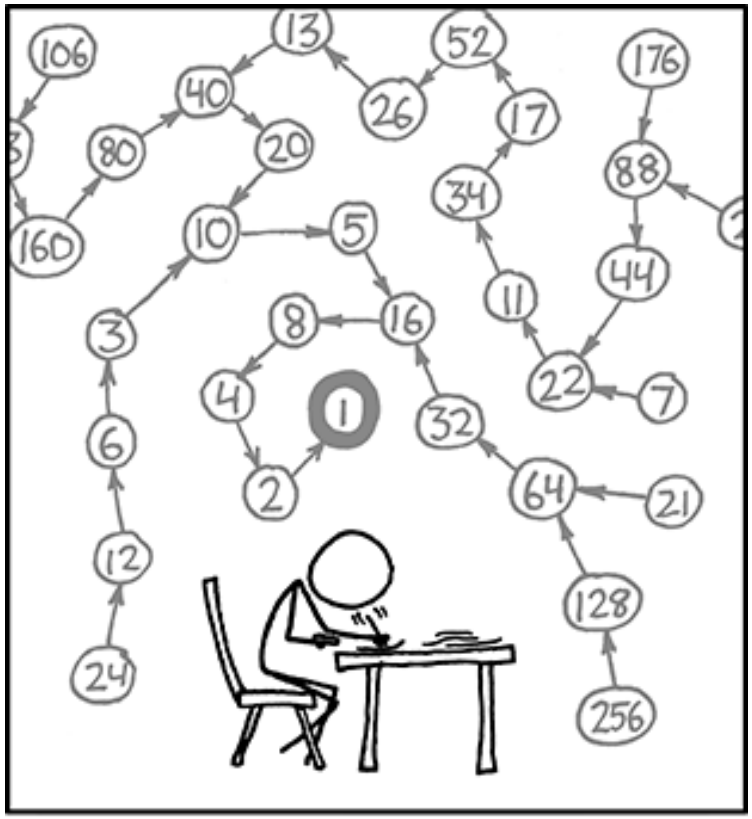
Ο έλεγχος τερματισμού είναι αδύνατος! Για παράδειγμα (Collatz Conjecture):

```
input n;
assume n>1;
while (n !=1) {
   if (n is even)
     n := n/2;
  else
     n := 3*n+1;
}
```

Κανείς δεν ξέρει αν αυτό το πρόγραμμα τερματίζει σε όλες τις εισόδους!

17, 52, 26, 13, 40, 20, 10, 5, 16, 8, 4, 2, 1.

Collatz Conjecture



THE COLLATZ CONJECTURE STATES THAT IF YOU PICK A NUMBER, AND IF IT'S EVEN DIVIDE IT BY TWO AND IF IT'S ODD MULTIPLY IT BY THREE AND ADD ONE, AND YOU REPEAT THIS PROCEDURE LONG ENOUGH, EVENTUALLY YOUR FRIENDS WILL STOP CALLING TO SEE IF YOU WANT TO HANG OUT.

x=10?

Μια Γεύση Υπολογισιμότητας

- Έστω *P*(*I*) ένα πρόγραμμα, όπου *P* τρέχει σε είσοδο *I*.
- Έστω ότι το πρόγραμμα ILT (το java πρόγραμμα για έλεγχο τερματισμού)
- Το ILT(*P*,*I*) απαντάει «άπειρη επανάληψη» αν το *P*(*I*) μπαίνει σε άπειρη επανάληψη, διαφορετικά λέει «τερματίζει».
- Θα κατασκευάζουμε ένα πρόγραμμα *ΚΛΑΨΕ* (αποΚΛείεται νΑ δουλέΨΕι) χρησιμοποιώντας το πρόγραμμα ILT.

Απόδειξη με εις Άτοπο Απαγωγή

Το *ΚΛΑΨΕ*(*P*) κάνει τα εξής:

1. Καλεί ILT(*P*,*P*).
2. Αν ILT απαντά «άπειρη επανάληψη», τότε τερμάτισε.
3. Αν ILT απαντά «τερματίζει», τότε μπες σε άπειρη επανάληψη.

Τι γίνεται αν τρέξουμε *ΚΛΑΨΕ*(*ΚΛΑΨΕ*);

- Αν η εκτέλεση τερματίσει, τότε στη γραμμή 1 ILT θα απαντούσε «τερματίζει», και άρα θα πηγαίναμε στην γραμμή 3 και…OOPS!
- Αν η εκτέλεση δεν τερματίσει, τότε μετά την γραμμή 1 θα πηγαίναμε στην γραμμή 2 και…OOPS!

Έχουμε **άτοπο** και άρα το ILT **δεν μπορεί να υπάρξει**

Γουάου...

- Η απόδειξη αυτή είναι ένα από τα πιο σημαντικά και θεμελιώδη αποτελέσματα στην Επιστήμη της Πληροφορικής.

- Για να την καταλάβετε προσπαθήστε να την περιγράψετε σε κάποιον άλλον.

Πολυπλοκότητα

Υπολογισιμότητα

Πολυπλοκότητα

Αυτόματα

- **Πόσο γρήγορα μπορούμε να υπολογίσουμε μία συνάρτηση;**
- **Πόσο χώρο χρειαζόμαστε;**

- Υπολογισμός πολυωνυμικού χρόνου
- Ανταιτιοκρατικός πολυωνυμικού χρόνου (NP)
- Προσέγγιση, Τυχαιότητα

Συναρτήσεις που δεν μπορούν να υπολογισθούν γρήγορα:

- Εφαρμογή σε ασφάλεια
  - Γρήγορη κρυπτογράφηση,
  - Η αποκρυπτογράφηση δεν είναι γρήγορη

- Κρυπτογραφία RSA

Ένα Απλό Παράδειγμα

7 x 11 = ?

Το πρόβλημα του

Πολλαπλασιασμού

(απάντηση: 77)

Ένα Ακόμα Απλό Παράδειγμα

? x ? = 77

Το πρόβλημα της

Παραγοντοποίησης

(απάντηση: 7,11)

Ένα Πιο Μεγάλο Παράδειγμα

163473364580925384844313388386509085984178367003309231218111085238933310010450815121211816751157919008712816648221131268515739354139754718967899685154936666385390880271038021044989571912614655715 X = ?

Απάντηση:

310741824049004372135075003588856793003734602284272754516194882320644051808150455634682967172328678243791627283803341547107310850191954852900733772482278352574238645401469173660247765234660926

Το Αντίστροφο όμως;

? X ? = 3107418240490043721350750035888567930037346022842727545720
16194882320644051808150455634682967172328678243791627283803341547107310850191954852900733772482278352574238645401469
173660247765234660 9

Αριθμός RSA 200. Οι παράγοντες βρέθηκαν μόλις το 2005 έπειτα από 5 μήνες ημερολογιακού χρόνου (80 AMD Opteron CPUs) – 20000$ το βραβείο

RSA Challenge 2048 – 617 ψηφία

25195908475657893494027183240048398571429282126204032027777**137836**
04366202070759555626401852588078440691829064124951508218929855914
91761845028084891200728449926873928072877767359714183472702618963
75014971824691165077613379859095700097330459748808428401797429100
64245869181719511874612151517265463228221686998754918242243363725
90851418654620435767984233871847744479207399342365848238242811981
63815010674810451660377306056201619676256133844143603833904414952
63443219011465754445417842402092461651572335077870774981712577246
79629263863563732899121548314381678998850404453640235273819513786
36564391212010397122822120720357

Βραβείο 200.000$ - Ο μεγαλύτερος που έχει παραγοντοποιηθεί μέχρι σήμερα είναι μήκους 232 ψηφίων. Θα περάσουν πολλά χρόνια μέχρι να επιτευχθεί....

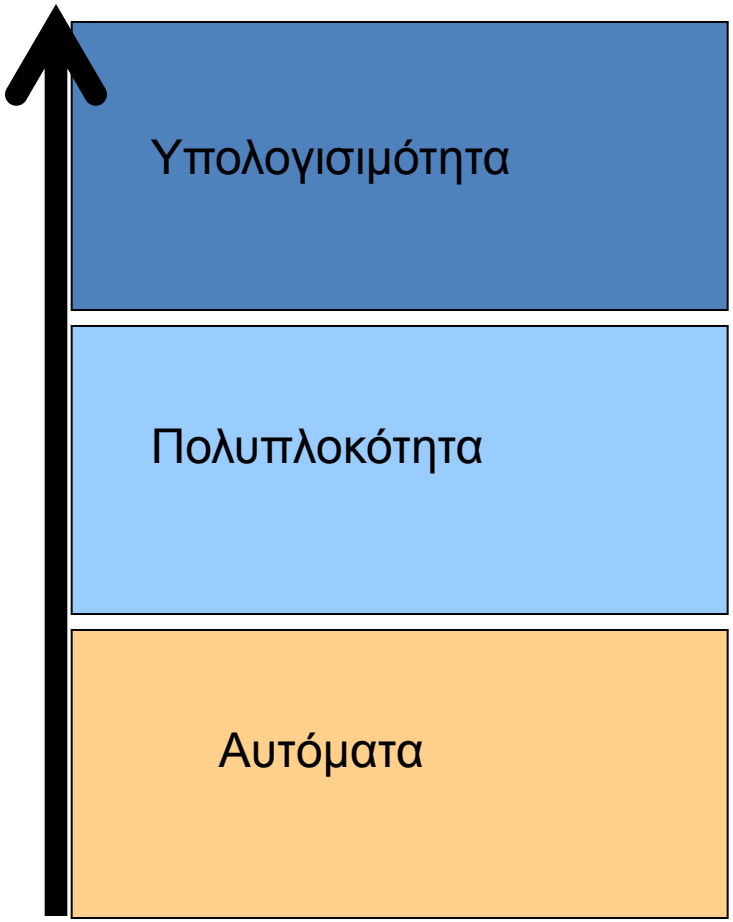
Θεωρια Υπολογισμου

Αυτόματα:
- Θεμελιώσεις υπολογισμού
- Μαθηματικές μέθοδοι
- Απλότητα

Έχουν ήδη γίνει στο μάθημα
Θεωρία Υπολογισμού

Πολυπλοκότητα

ΑΥΞΗΜΕΝΗ ΠΟΛΥΠΛΟΚΟΤΗΤΑ



Τι υπολογίζουμε;
- Γενικές έννοιες υπολογισιμότητας
- Μη υπολογίσιμες συναρτήσεις

Τι μπορούμε να υπολογίσουμε γρήγορα;
- Γρήγοροι αλγόριθμοι, πολυωνυμικός χρόνος
- Προβλήματα που δεν λύνονται γρήγορα
- Κρυπτογραφία

Τι υπολογίζουμε με μικρό χώρο;
- Σταθερός χώρος (+σωρός)
- Εύρεση αλφαριθμητικών, επαλήθευση υλικού κτλ.

ΚΆΤΙ ΠΟΥ ΈΧΕΙ ΠΛΆΚΑ

Quines

Πώς ένα πρόγραμμα εκτυπώνει τον εαυτό του;

```
main()
{
printf(“Hello World”);
}
```

```
main()
{
printf(“main() { printf(\”Hello World\”);}
}
```

??????????

Quines

http://www.nyx.net/~gthompso/quine.htm

Ένα παράδειγμα.

Ένα πιο απλό παράδειγμα.

Διαλογικά Αποδεικτικά Συστήματα (Πιθανοκρατικά)

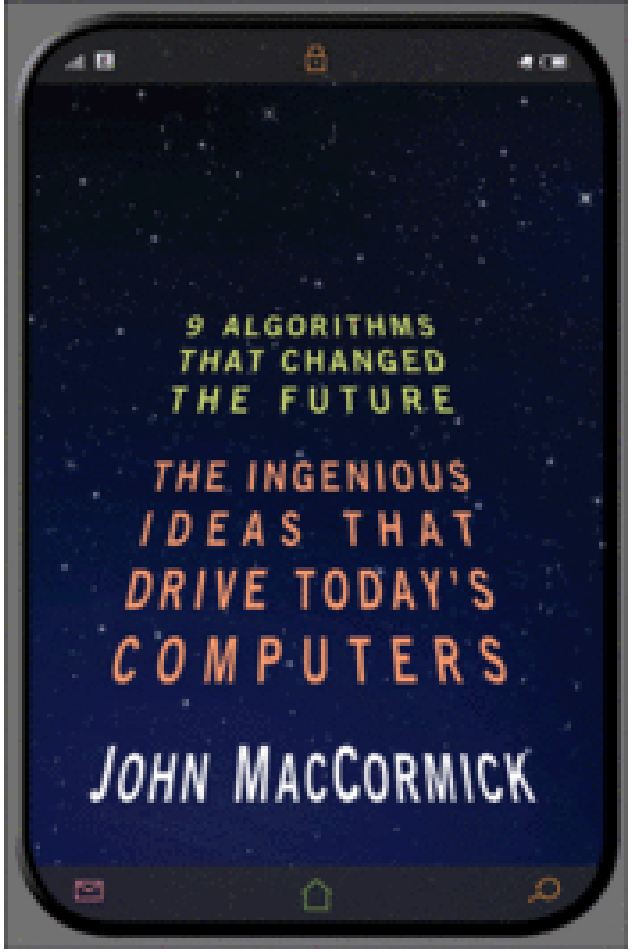
Η Marla έχει μία κόκκινη κάλτσα και μία κίτρινη κάλτσα. Ο φίλος της ο Arthur έχει αχρωματοψία και δεν την πιστεύει ότι οι κάλτσες έχουν διαφορετικό χρώμα. Πώς θα τον πείσει ότι πράγματι έχουν διαφορετικό χρώμα;

Αποδείξεις με Μηδενική Γνώση (Zero-Knowledge Proofs) - Αυθεντικοποίηση

1. Δύο περιπολίες συναντιούνται σε ένα στρατόπεδο χωρίς να ξέρει ο ένας για τον άλλο ότι κάνουν περιπολία. Και οι δύο ξέρουν το σύνθημα που είναι ένας ακέραιος αριθμός στο διάστημα [1,52]. Τι θα κάνουν αν:
    1. Έχουν μία τράπουλα και μία τσάντα.
    2. Αν ένας σκαστός φαντάρος εμφανίστηκε μπροστά τους και βάλουν αυτόν να αποφασίσει αν είναι γνήσια περίπολα και όχι κατάσκοποι.

Και Κάτι πιο Εξωτικό

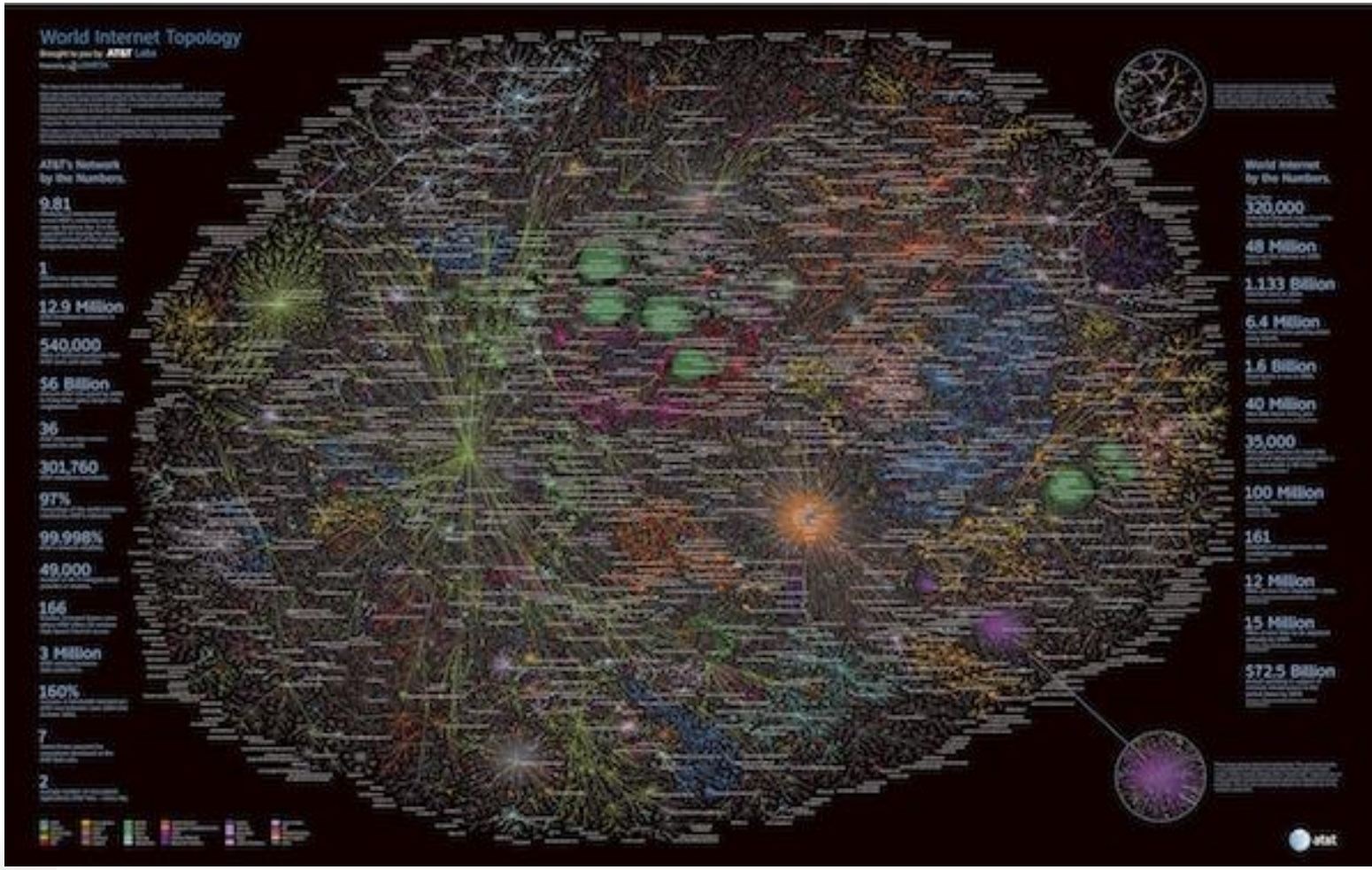
- Εναλλακτικά μοντέλα Υπολογισμού:
  - Φούσκες σαπουνιού
  - Κβαντικός Υπολογισμός
  - Αναλογικός Υπολογισμός και Σχετικότητα
  - Κβαντική Βαρύτητα
  - Ταξίδι στον Χρόνο
  - Ανθρωποκεντρικός Υπολογισμός



9 Αλγόριθμοι (μάλλον αλγοριθμικές ιδέες) που Άλλαξαν το Μέλλον

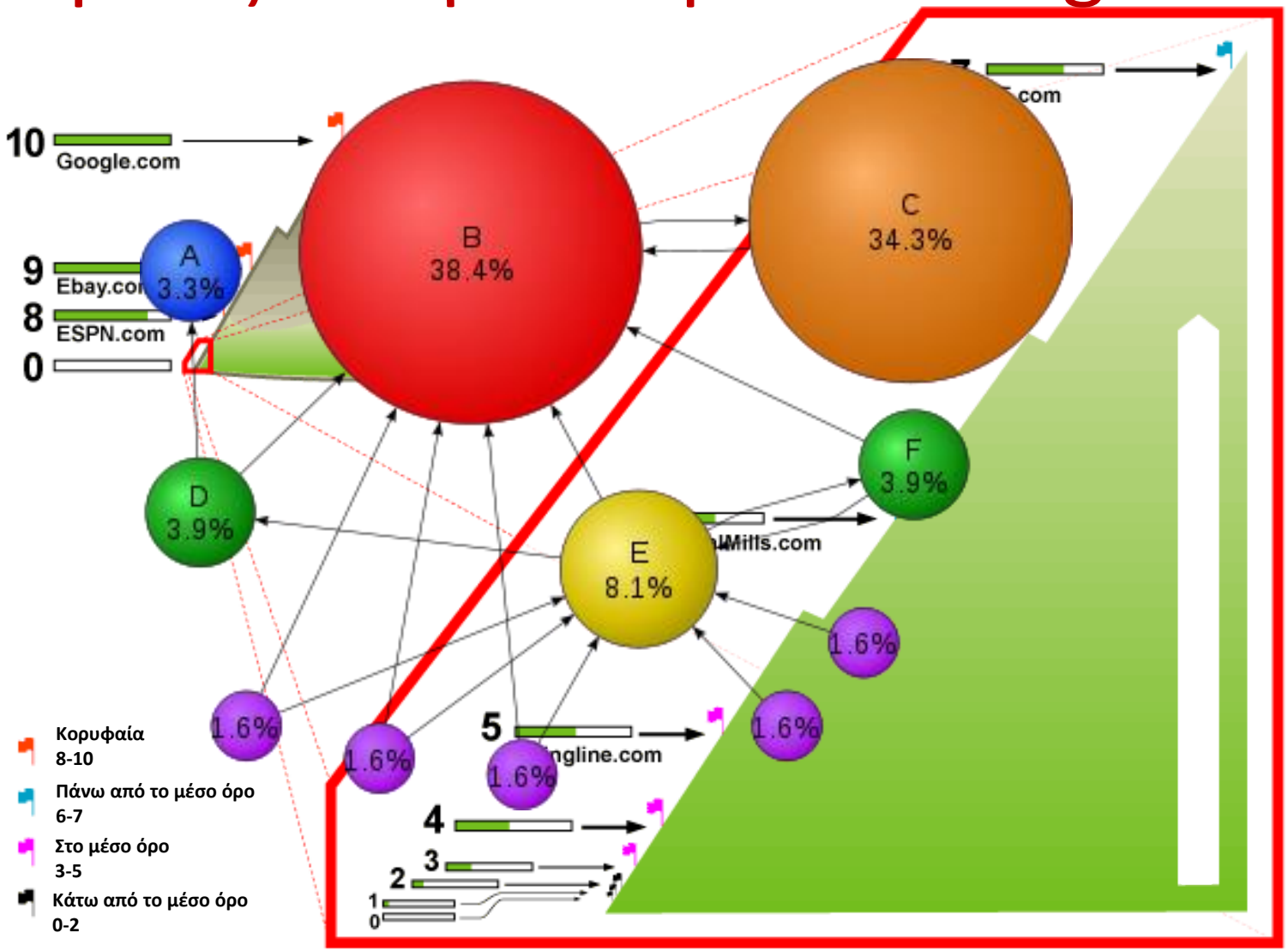
Δεικτοδότηση Μηχανών Αναζήτησης

ή πώς να ψάχνεις μέσα στα άχυρα



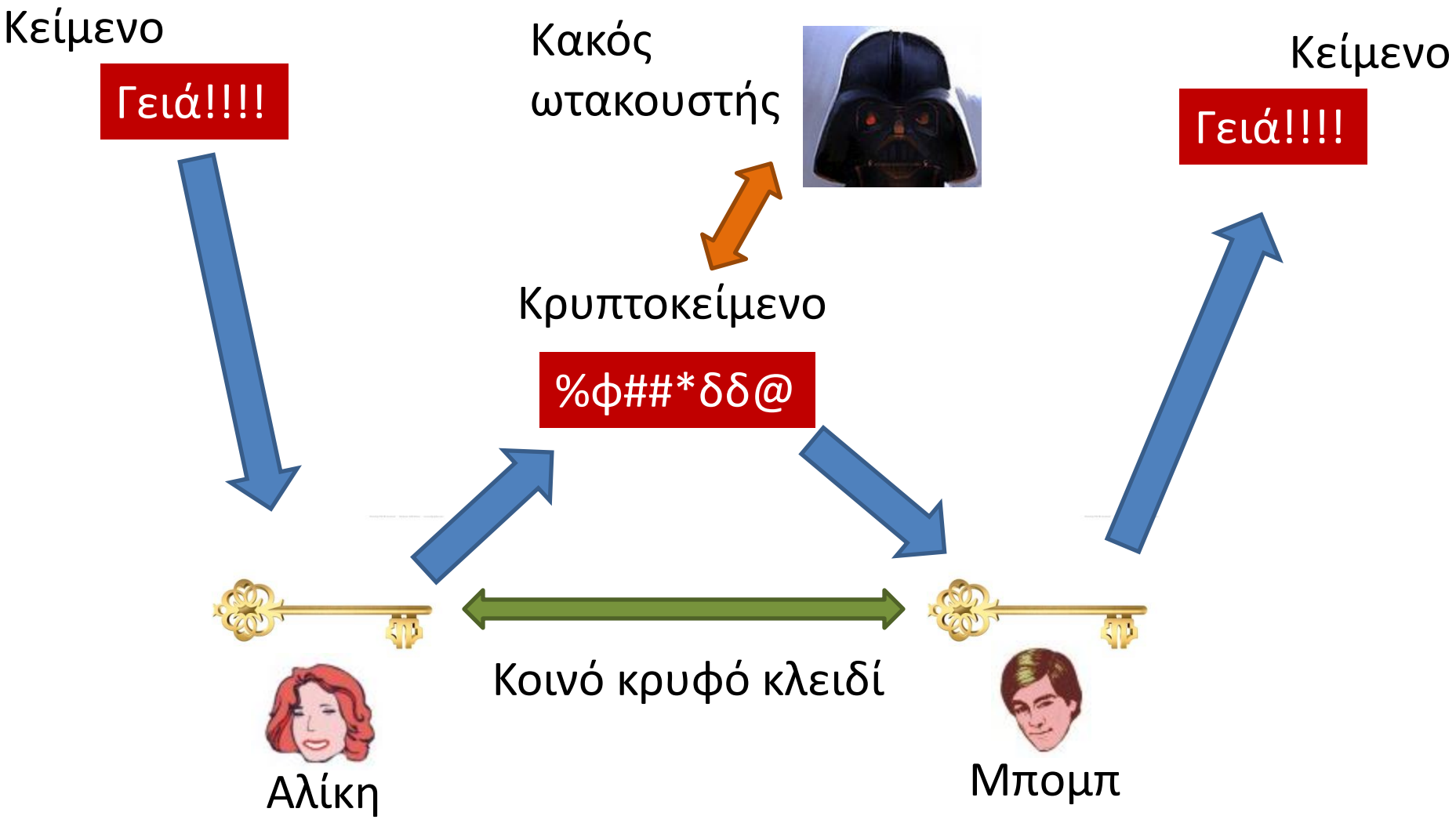
Pagerank

ή πώς απογειώθηκε το Google



Κρυπτογραφία Δημόσιου Κλειδιού:

ή πώς να στέλνεις μυστικά με μία κάρτα



Κώδικες Διόρθωσης Λαθών

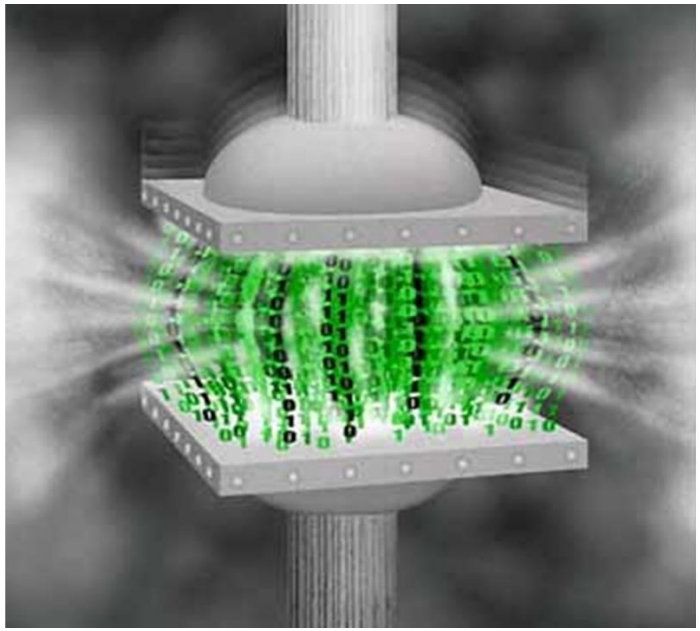
ή πώς να διορθώνεις τα λάθη σου

Αναγνώριση Μοτίβων
ή ποιος είναι στη φωτογραφία

Ποιος είναι αυτός;

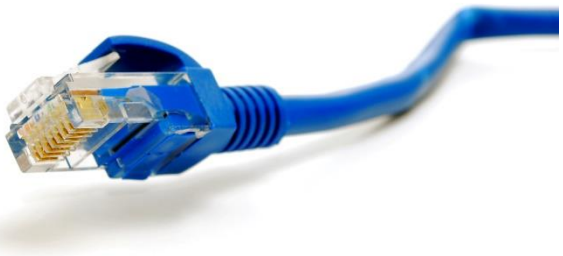
Συμπίεση Δεδομένων
ή πώς χωρά μία ταινία σε ένα δίσκο

- Για ένα video 2 ωρών 1080p χωρίς συμπίεση (μόνο εικόνα) το συνολικό μέγεθος είναι περίπου (με 30 fps): 440Gb
- Αυτό όμως χωρά άνετα σε ένα δισκάκι 8 GB;



Βάσεις Δεδομένων

ή πως θα γίνεις συνεπής



Ψηφιακές Υπογραφές

ή πώς θα δείξεις ότι είσαι αυτός που είσαι

1. Ο Γιάννης μαζί με το email στέλνει και την ψηφιακή του υπογραφή χρησιμοποιώντας το προσωπικό του κλειδί.

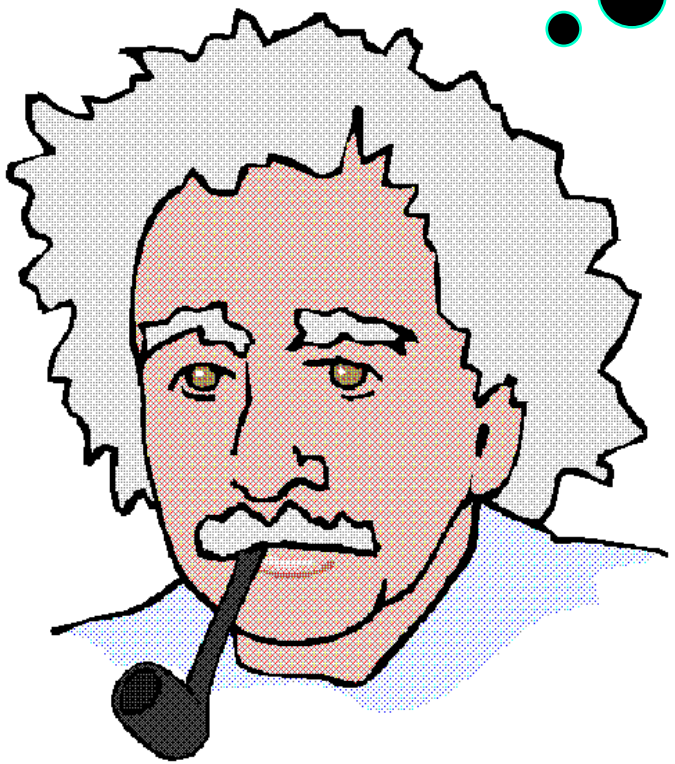
2. Με το που λαμβάνει το email η Μαρία ελέγχει με το δημόσιο κλειδί του Γιάννη την ψηφιακή του υπογραφή

Υπολογισιμότητα
ή τι μπορούμε να υπολογίσουμε

Το Δόγμα των Church-Turing

Κάθε πρόβλημα που μπορεί να λυθεί από μία μηχανιστική διαδικασία, μπορεί να λυθεί και από έναν υπολογιστή.

Φιλοσοφικές
Σκέψεις



Το Αξίωμα των Church-Turing

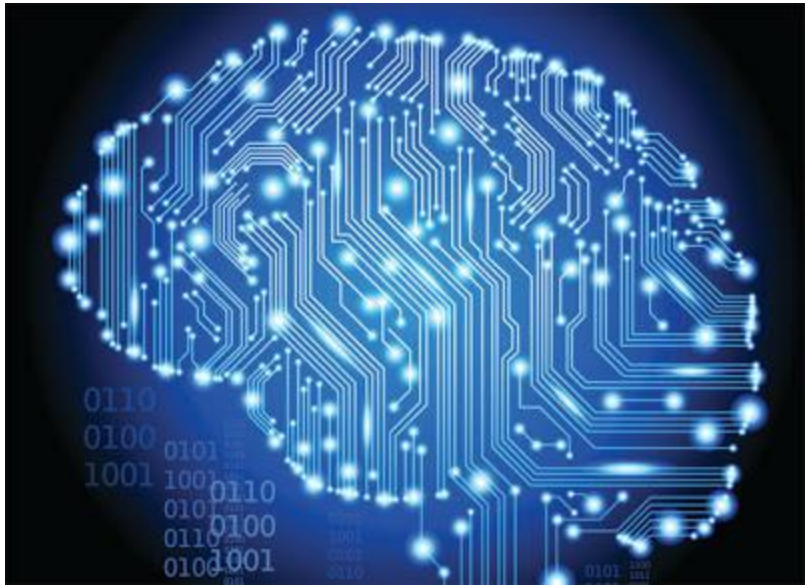
- Κάθε υπολογιστική μέθοδος που μπορεί να σχεδιαστεί και να εκτελεστεί από το ανθρώπινο μυαλό, μπορεί να εκτελεστεί και σε έναν Υπολογιστή

Το Αξίωμα των Church-Turing

- Η θέση αυτή δεν είναι Θεώρημα (δεν έχει αποδειχθεί) αλλά είναι μία εικασία σχετικά με το σύμπαν στο οποίο ζούμε.

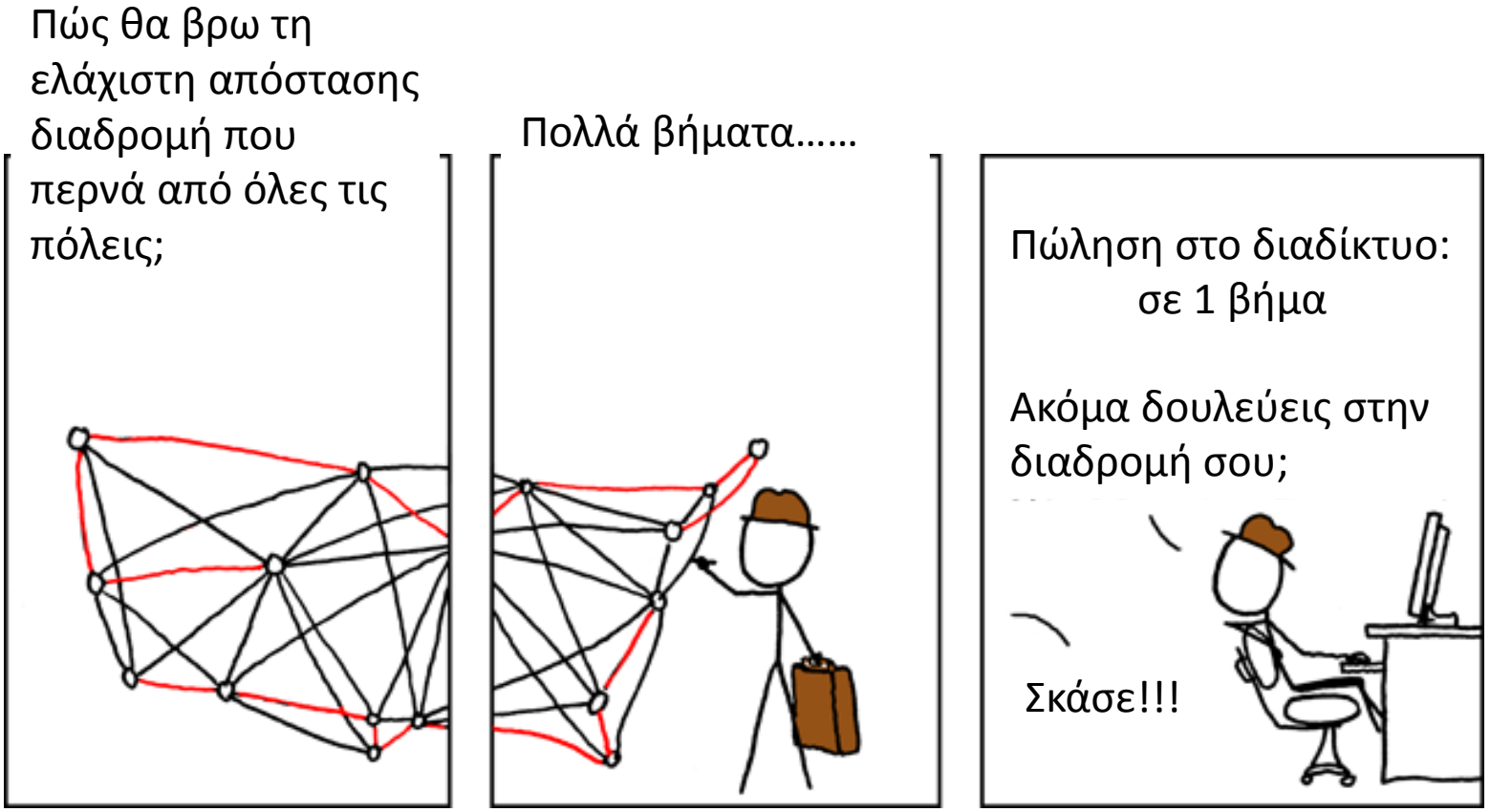
- Η άποψή σας σχετικά με αυτό το αξίωμα μπορεί να επηρεαστεί από τις θρησκευτικές, επιστημονικές και φιλοσοφικές σας αντιλήψεις.

Πολυπλοκότητα



Πόσο γρήγορα μπορεί ο υπολογιστής να λύσει ένα πρόβλημα;

Το Πρόβλημα του Περιοδεύοντος Πωλητή



P vs. **NP**
Ή Τρόπος vs Κόπος

Τι είναι πιο εύκολο από τα παρακάτω:

- Να βρεις τη λύση σε ένα πρόβλημα.
- Να επαληθεύσεις ότι μία δοθείσα λύση σε ένα πρόβλημα είναι σωστή.

Clay Mathematics Institute

MILLENNIUM PRIZE PROBLEMS

P versus NP

The Hodge Conjecture

The Poincaré Conjecture

Λύθηκε (Perelman)!

The Riemann Hypothesis

Yang-Mills Existence and Mass Gap

Navier-Stokes Existence and Smoothness

The Birch and Swinnerton-Dyer Conjecture

Announced 16:00, on Wednesday, May 24, 2000
Collège de France

P vs. **NP** και Μαθηματικά

- Αν *P=NP*, τότε θα αντικαθιστούσαμε τους μαθηματικούς από (πολύ πιο αξιόπιστους) υπολογιστές:

P=NP: Υπάρχει αλγοριθμική διαδικασία που παίρνει ως είσοδο οποιαδήποτε τυπική μαθηματική πρόταση και πάντα παράγει την μικρότερη δυνατή απόδειξη σε χρόνο ανάλογο με το μήκος της απόδειξης.

- Αυτός είναι ένα λόγος που συνήθως θεωρούμε (ιδιαίτερα οι μαθηματικοί!) ότι οι κλάσεις *P* και *NP* είναι διαφορετικές.

Κάποια Λίγα Δύσκολα Προβλήματα…
(η ορολογία μπορεί να είναι λάθος)

- Βιολογία: αναδίπλωση πρωτεϊνών
- Μηχανική : ισορροπία ροών κίνησης εντός πόλης
- Οικονομικά: υπολογισμός σε αγορές με τριβή
- Περιβαλλοντική Μηχανική: βέλτιστη τοποθέτηση ανιχνευτών μόλυνσης
- Οικονομική Μηχανική: εύρεση ελαχίστου κόστους πορτφόλιο για δεδομένο κέρδος
- Θεωρία Παιγνίων: εύρεση Nash ισορροπιών που μεγιστοποιεί την κοινωνική ευημερία
- Γενομική: φυλογενετική ανακατασκευή
- Μηχανική: δομή αναταραχών σε ροές υγρών
- Φαρμακευτική: ανακατασκευή 3-D σχήματος από διεπίπεδο καρδιογράφημα
- Επιχειρησιακή Έρευνα: βέλτιστη κατανομή πόρων
- Φυσική: Συνάρτηση διαμέρισης (partition) του 3-D Ising μοντέλου στη στατιστική μηχανική
- Πολιτική: Shapley-Shubik ισχύς ψήφου
- Ποπ κουλτούρα: Συνέπεια του παιχνιδιού Minesweeper
- Στατιστική: Βέλτιστη πειραματική σχεδίαση
- …

7 Παρεξηγήσεις Σχετικά με την Θεωρία Αλγορίθμων

Ή “γιατί η θεωρία είναι περιορισμένης χρήσης στον πραγματικό κόσμο”

Προσοχή: αυτά που ακολουθούν είναι μερικώς αληθή και είναι αντικείμενο διαφωνίας.

7: Η Θεωρία Αφορά την Ανάλυση Χειρότερης Περίπτωσης

ή, παραδέχομαι ότι που και που υπάρχει μία ανάλυση μέσης περίπτωσης αλλά εγώ ενδιαφέρομαι για τα στιγμιότυπά μου.

Πολλές λύσεις είναι αυτής της μορφής αλλά:

1. Η κατανόηση της χειρότερης περίπτωσης βοηθά στην κατανόηση των κακών παραδειγμάτων
2. Υπάρχει αρκετή δουλειά σε φράγματα με δεδομένο κάποιους περιορισμούς στην κατανομή των δεδομένων
3. Έχει γίνει πολύ δουλειά στη μελέτη των περιπτώσεων που είναι δύσκολες

Παρατήρηση (hot)

Χαρακτηρισμός κλάσεων εισόδου ή ιδιοτήτων και ανάλυση των αλγορίθμων σε αυτές τις εισόδους

Ιδέες για γραφήματα;

Ιδέες για ταξινόμηση;

Ιδέες για κατανομές σημείων;

6: Το Μοντέλο Υπολογισμού δεν είναι Σωστό μιας και δεν Λαμβάνει Υπόψη την Τοπικότητα

Πράγματι, η RAM δεν είναι τέλειο μοντέλο, αλλά

1. Υπάρχουν πολλά αποτελέσματα με διάφορες βελτιώσεις του μοντέλου (I/O, cache, streaming).

2. Τα περισσότερα αποτελέσματα δεν αλλάζουν ποιοτικά μεταξύ μοντέλων. π.χ. P vs. NP παραμένει μεταξύ των περισσότερων μοντέλων

3. Πολλές καλές ιδέες προέρχονται από απλά μοντέλα που μεταφέρονται σε πιο πολύπλοκα και ρεαλιστικά μοντέλα.

5: Η Θεωρία έχει Σχέση με την Ανάλυση

Όχι, η θεωρία έχει σχέση με την μοντελοποίηση:

1. Κρυπτογραφία
   - Τι σημαίνει ότι είναι δύσκολο να σπάσει;
   - Τι είναι ένα κρυπτοσύστημα δημοσίου κλειδιού;
   - Τι είναι μία απόδειξη μηδενικής γνώσης;

2. Θεωρία Πληροφορίας

3. Θεωρία Γραφημάτων

4: Τα Περισσότερα Προβλήματα στη Θεωρία Είναι Άσχετα

Το σημερινό άσχετο πρόβλημα μπορεί να είναι η αυριανή βιομηχανία.

Ποιος ενδιαφέρεται άλλωστε για παραγοντοποίηση αριθμών ή γραφήματα επέκτασης (expander graphs);

3: Τα Σχετικά Προβλήματα Είναι Υπεραπλουστευμένα

Μερικές φορές ναι, αλλά

1. Η επίλυση μίας απλής εκδοχής μπορεί να οδηγήσει στην επίλυση της πλήρους εκδοχής.

2. Η επίλυση της απλής εκδοχής μπορεί να βοηθήσει να κατανοήσουμε το λόγο για τον οποίο η πλήρης εκδοχή είναι δύσκολη

2: Η Θεωρία Αφορά Μόνο το Μεγάλο-Ο και Αδιαφορεί για Σταθερές

Πολλά αποτελέσματα χρησιμοποιούν ασυμπτωτική ανάλυση

1. Υπάρχουν πολλά αποτελέσματα που χρησιμοποιούν ακριβή ανάλυση, πολλές φορές στον υψηλότερο όρο, π.χ. 7n + O(log n)

2. Πολλές φορές αλγόριθμοι με μεγάλες σταθερές βελτιώνονται αργότερα, πολλές φορές με μικρές αλλαγές

3. Μερικές φορές η μεγάλη σταθερά είναι απλά στην απόδειξη (ίσως για απλοποίηση απόδειξης) και όχι στον πραγματικό αλγόριθμο.

1. Γεωγραφική Τοπικότητα :-)

- Αφού αρέσει στον Τσίχλα η θεωρία, καλύτερα άστο…

Ο Αλγόριθμος Του Strassen για Πολλαπλασιασμό Πινάκων

Από το βιβλίο *Numerical Recipes* (Press et. al. 1986)

> "We close the chapter with a little entertainment, a bit of algorithmic prestidigitation

και αργότερα στο ίδιο κεφάλαιο

> "This is fun, but let's look at practicabilities: If you estimate how large N has to be before the difference between exponent 3 and exponent 2.807 is substantial enough to outweigh the bookkeeping overhead, arising from the complicated nature of the recursive Strassen algorithm, you will find that LU decomposition is in no immediate danger of coming obsolete."

Από Μία Δημοσίευση με Πειράματα

(Bailey, Lee and Simon):

> "For many years it was thought that [the level at which Strassen's algorithm is more efficient] was well over 1000x1000. In fact, for some new workstations, such as the Sun-4 and SGI IRIS 4D, Strassen is faster for matrices as small as 16x16.

Οι περισσότερες μοντέρνες βιβλιοθήκες πινάκων δίνουν τη δυνατότητα χρήσης του αλγόριθμου Strassen

- IBM ESSL library
- Cray Libraries
- LAPACK library

Υλικό

- Βιβλίο:

J. Kleinberg and E. Tardos: Σχεδιασμός Αλγορίθμων, Κλειδάριθμος, 2008.

- Ενδεχομένως να χρησιμοποιήσω όμως και υλικό από άλλα βιβλία, σημειώσεις και ερευνητικά άρθρα και επομένως η καλύτερη πηγή θα είναι οι διαλέξεις.

Περιεχόμενα

Πολυπλοκότητα:

- Κλάσεις P, NP και co-NP
- Πληρότητα
- Αναγωγές
- PSPACE

Αλγόριθμοι:

- Εκθετικοί αλγόριθμοι
- Προσεγγιστικοί αλγόριθμοι
- Τυχαιοποιημένοι αλγόριθμοι
- Τοπική Αναζήτηση

ΤΕΛΟΣ