Αλγόριθμοι και Πολυπλοκότητα

Λυγάτσικας Ζήνων
Βαρβάκειο Πειραματικό Λύκειο

Δεκέμβριος, 2007

1 Εισαγωγή

Η προέλευση του ονόματος του αλγορίθμου δεν είναι ελληνική! Προέρχεται απο την τελευταία λέξη του προκλητικού ονόματος ενός κάποιου *άγριου* άραβα μαθηματικού του abu ja'far mohammed ibn mûsâ al-khawārizmī, που έζησε γύρω στο 800 μΧ. Ο διάσημος αυτός *άγριος* μελέτησε την αριθμητική μεταφέροντας στην *άγρια* τότε δύση κανόνες υπολογισμού αριθμών στην δεκαδική τους αναπαράσταση, που διδάχθηκε απο κάποιους εξ ίσου *άγριους* ινδούς μαθηματικούς[1].

Ένας αλγόριθμος είναι ένα εξειδικευμένο σχέδιο υπολογισμού με τη μορφή μιας ακολουθίας στοιχειωδών πράξεων που υπακούουν σε μια καλά καθορισμένη διαδοχή.

Κατά καιρούς έχουν αναπτυχθεί διάφοροι αλγόριθμοι στην αριθμητική και στην γεωμετρία, όπως:

- Ο κανόνας υπολογισμού χορδών και επιφανειών απο τους Αιγύπτιους και Έλληνες.
- Πολλές μέθοδοι επίλυσης εξισώσεων με ακέραιες ρίζες απο τον Διόφαντο τον 4^{o} αιώνα.
- Ο περίφημος Ευκλείδειος αλγόριθμος, ($\sim$ 300 πΧ), με τον οποίο υπολογίζουμε τον μέγιστο κοινό διαιρέτη δύο αριθμών.
- Ο υπολογισμός του π απο τον Αρχιμήδη.

Στη συνέχεια έχουμε αλγορίθμους που μελετούν αριθμητικές λύσεις εξισώσεων, όπως οι αλγόριθμοι του Newton, οι μέθοδοι απαλειφής του Gauss, οι λύσεις των διαφορικών εξισώσεων κλπ. Μετά τον δεύτερο παγκόσμιο πόλεμο άλλαξε ριζικά το κεφάλαιο υπολογισμός. Η αλλαγή οφείλεται φυσικά στην ανάπτυξη ενός βασικού εργαλείου για την μελέτη και την εκτέλεση των αλγορίθμων, του καλού μας υπολογιστή. Έτσι η τόση δούλα, μέχρι τότε, έννοια του αλγορίθμου απέκτησε διάσταση πολλές φορές μεγαλύτερη απο αυτή των προβλημάτων που επίλυε. Για

[1] Παρ'ολα τα εξωτικά θέλγητρά τους, όπως θα έλεγε και ο Hardy, οι αλγόριθμοι έγιναν τελικά ελληνοπρεπείς.

παράδειγμα, η ταχύτητα εξαγωγής αποτελέσματος ή η βελτιστοποίηση των πράξεων για την επίλυση κάποιου προβλήματος, ακόμα και το αν ή όχι ένας αλγόριθμος σταματά, γίναν αντικείμενο μιας νέας επιστήμης της Θεωρίας Αλγορίθμων που απο τότε χαίρει άκρας υγείας.

Σήμερα διαθέτουμε ένα σύνολο απο αξιόλογους αλγορίθμους σε όλους τους τομείς της ανθρώπινης δραστηριότητας οι οποίοι επιλύουν με την ίδια πειθαρχία πολύ δύσκολα αλλά και πολύ εύκολα προβλήματα. Πολλούς απο αυτούς τους χρησιμοποιούμε στη καθημερινή μας ζωή όπως η εύρεση του τηλεφώνου της γιαγιάς μας στο κινητό, για παράδειγμα.

Οι παλαιότεροι απο εμάς θα θυμούνται τους υπολογιστές τις δεκαετίας του 80 και σύγουρα όλοι θα έχετε ακούσει τη φράση: *οι υπολογιστές σήμερα είναι καλύτεροι απο αυτούς που είχαμε κάποτε.* Φυσικά δεν μιλάμε για την αισθητική του περιβλήματος ούτε για την φιλοσοφία της μηχανής η οποία δεν άλλαξε καθόλου, αλλά μιλώντας απο την μεριά του χρήστη, είμαστε άλλωστε οι περισσότεροι, εννοούμε ότι οι σημερινοί υπολογιστές κάνουν πιο εύκολα και πιο γρήγορα περισσότερα πράγματα απο αυτά που κάναν παλαιότερα. Θα έχετε ακούσει φυσικά και χαρακτηρισμούς όπως, ο τάδε αλγόριθμος είναι αποτελεσματικότερος του δείνα ή ότι η τάδε έκδοση του τάδε λογισμικού έχει βελτιώσεις σε σχέση με τις προηγούμενες κοκ. Τι είναι άραγε η αποτελεσματικότητα ενός αλγορίθμο; Είναι ένα μετρήσιμο μαθηματικό μέγεθος; Πότε ένας αλγόριθμος είναι αποτελεσματικότερος απο έναν άλλον; Αυτό θα είναι το αντικείμενο που θα προσπαθήσουμε να σκιαγραφήσουμε παρακάτω. Δεν είναι εύκολα διαπραγματεύσιμο για τον λόγο αυτό αφήνουμε την βελτιστοποίηση και τις υπόλοιπες ιδιότητες των αλγορίθμων στο δικός σας ενδιαφέρον.

2 Πολυπλοκότητα αλγορίθμου

Η αξιολόγιση αλγορίθμων μπορεί να γίνει με διάφορα κριτήρια. Ένα απο τα πλέον διαδεδομένα είναι ο ρυθμός αύξησης, συναρτήσει του μήκους των δεδομένων της εισόδου, του χρόνου και του χώρου μνήμης που απαιτεί η εκτέλεση ενός αλγορίθμου. Έτσι για παράδειγμα, ο χρόνος εγγραφής ενός αρχείου του Word στον σκληρό δίσκο εξαρτάται απο το μέγεθος του. Αν και στους περισσότερους υπολογιστές ο χρόνος αυτός φαντάζει μηδενικός, ωστόσο η διαφορά εγγραφής ενός αρχείου 20Mbs απο αυτόν ενός αρχείου 20Kbs είναι αναγνώσιμη. Αυτό σημαίνει ότι ο αλγόριθμος φύλαξης αρχείων εκτελεί την εντολή εγγραφής σε χρόνο ανάλογο του μήκους του αρχείου.

Θα αντιστοιχίσουμε σε κάθε πρόβλημα έναν ακέραιο που θα τον ονομάσουμε μέγεθος - size - του προβλήματος και θα μετρά μια ποσοτική παράμετρο των δεδομένων στην είσοδο ενός αλγορίθμου. Ο απαιτούμενος χρόνος για τη λύση του προβλήματος απο ένα αλγόριθμο, συναρτήσει του size n, καλείται πολυπλοκότητα -complexity- του αλγορίθμου. Όταν το n αυξάνει η πολυπλοκότητα του αλγορίθμου είναι λογικό να αυξάνει. Αν υποθέσουμε ότι το n γίνει παρα πολύ μεγάλο η πολυπλοκότητα θα τείνει σε μια ποσότητα την οποία θα την λέμε ασυπτωτική πολυπλοκότητα. Ανάλογους ορισμούς μπορούμε να δώσουμε και για το χώρο κατάλυψης των δεδομένων κατα την διάρκεια επεξεργασίας τους απο τον αλγόριθμο.

Η ασυμπτωτική πολυπλοκότητα (απλώς πολυπλοκότητα στη συνέχεια), ενός αλγορίθμου είναι αυτό που καθορίζει αν μπορεί ή όχι να λυθεί ένα πρόβλημα απο έναν αλγόριθμο. Επίσης είναι και το μέτρο που θα χαρακτηρίσει την αποτελεσματικότητα του συγκεκριμμένου αλγορίθμου. Αν έχουμε ένα πρόβλημα size n με χρόνο επίλυσης απο κάποιον αλγόριθμο $10n^2$ μονάδες μέτρησης χρόνου, συμβολίζουμε την πολυπλοκότητα του με $O(n^2)$, σύμβολο που πρωτοχρησιμοποιήθηκε απο τον Landau (1877-1938), και διαβάζουμε: ή πολυπλοκότητα του αλγορίθμου είναι της τάξης n^2.

Ας δούμε για παράδειγμα πέντε αλγορίθμους A_1, A_2, A_3, A_4, A_5 που λύνουν το ίδιο πρόβλημα Π, (μπορείτε να υποθέσετε για να γίνει αυτό κατανοητό, ότι το πρόβλημα Π αφορά τη γνωστή ανάλυση ενός ακεραίου αριθμού σε γινόμενο πρώτων παραγόντων), με τις αντίστοιχες πολυπλοκότητες τους. Ας δούμε τον παρακάτω πίνακα:

Αλγόριθμος	*Πολυπλοκότητα τάξης*	*Μέγιστο μέγεθος προβλήματος*		
		1 sec	*1 min*	*1 h*
A_1	n	1000	$6 \cdot 10^4$	$3,6 \cdot 10^6$
A_2	$n \cdot log(n)$	140	4893	$2,0 \cdot 10^5$
A_3	n^2	31	245	1897
A_4	n^3	10	39	153
A_5	2^n	9	15	21

Υποθέτω ότι η μονάδα χρόνου είναι ίση με 1 millisecond. Ο αλγόριθμος A_1, με πολυπλοκότητα τάξεως n, μπορεί να λύση σε 1 sec το πρόβλημα Π με size 1000, (δηλαδή να δώσει την ανάλυση ενός αριθμού με 1000 ψηφία), ενώ ο A_5, με πολυπλοκότητα τάξεως 2^n, σε 1 sec λύνει το πρόβλημα Π με size μόνο 9, (δηλαδή δίνει την ανάλυση ενός αριθμού με 9 ψηφία). Άρα ο αλγόριθμος A_1 είναι ικανότερος του A_5 και περίπου 100 φορές πιο γρήγορος για size $n = 1000$.

Αλλά όπως ήδη γνωρίζετε δεν είναι μόνο η πολυπλοκότητα του αλγορίθμου που παίζει ρόλο στην ταχύτητα επίλυσης ενός προβλήματος. Ένας άλλος παράγοντας είναι φυσικά ο επεξεργαστής. Για να δούμε πως θα αλλάξει το size του προβλήματος όταν η νέα γενιά υπολογιστών H2 διαθέτει έναν επεξεργαστή 10 φορές πιο γρήγορο απο αυτόν της γενιάς των υπολογιστών H1 που κάναμε το test στον προηγούμενο πίνακα.

Αλγόριθμος	*Πολυπλοκότητα*	*Μέγιστο size του προβ. Π σε χρόνο t με τον H1*	*Μέγιστο size του προβ. Π σε χρόνο t με τον H2*
A_1	n	s_1	$10s_1$
A_2	$n \cdot log(n)$	s_2	$\sim 10s_2$
A_3	n^2	s_3	$3,16s_3$
A_4	n^3	s_4	$2,15s_4$
A_5	2^n	s_5	$s_5 + 3,3$

Ο αλγόριθμος A_3 που λύνει σήμερα το πρόβλημα Π σε χρόνο t με size s_3, αύριο ο ίδιος ο αλγόριθμος θα μπορεί να δεχθεί στον ίδιο χρόνο t το πρόβλημα με size 3 φορές περίπου μεγαλύτερο απο το σημερινό.

Ταξινομούμε τις πολυπλοκότητες στις παρακάτω κατηγορίες:

Σύμβολο	*Κατηγορία πολυπλοκότητας*
$O(1)$	Σταθερή, ανεξάρτητη του μεγέθους των δεδομένων
$O(n)$	Γραμμική
$O(log(n))$	Λογαριθμική
$O(nlog(n))$	Σχεδόν γραμμική
$O(n^2)$	Τετραγωνική
$O(n^3)$	Κυβική
$O(n^p)$	Πολυωνυμική
$O(n^{nlog(n)})$	Σχεδόν πολυωνιμική
$O(2^n)$	Εκθετική
$O(n!)$	Παραγωντική

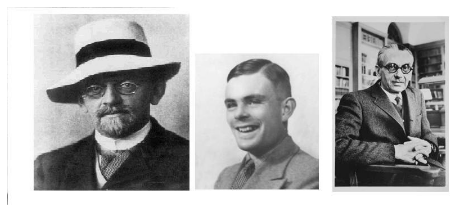
Πέρα του ίδιου αλγορίθμου και του επεξεργαστή του υπολογιστή, υπάρχει ένας άλλος βασικότερος παράγοντας που αν και δεν φαίνεται να συμμετέχει ενεργά στην αποτελεσματικότητα των αλγορίθμων, είναι θα λέγαμε η *ψυχή* της μορφής του σύγχρονου υπολογισμού. Αγγίζουμε έτσι το μαθηματικό μοντέλο του υπολογιστή: τη μηχανή Turing.

3 Η μηχανή Turing

Η μηχανή Turing πήρε το όνομά της απο τον άγγλο μαθηματικό Alan Mathison Turing (1912-1954), και είναι μια *φανταστική* μηχανή που περιγράφει με σύμβολα τις πράξεις της μαθηματικής λογικής. Είναι η βάση της θεωρίας των Αυτομάτων και η βάση των σημερινών υπολογιστών, αφού μελετάμε με αυτήν τη μηχανή, τη λογική δομή του υπολογιστή τυποποιώντας την έννοια του αλγορίθμου[2]. Η θεωρία της μηχανής Turing είναι ισοδύναμη της θεωρίας των επαναληπτικών συναρτήσεων, (recursive functions). Έτσι μπορούμε να πούμε ότι, ο σημερινός υπολογιστής είναι μια επαναληπτική συνάρτηση, μια περίπτωση στο απέραντο δάσος των μαθηματικών συναρτήσεων.

Η ιστορία της μηχανής Turing έχει ως εξής: Λοιπόν, στις αρχές του περασμένου αιώνα ο μαθηματικός D. Hilbert (1862-1943), διατύπωσε μια διάσημη ερώτηση (Entscheidungsproblem): *Υπάρχει μέθοδος (που πηγάζει απο τη μαθηματική λογική και τα αξιώματα μιας θεωρίας) έτσι ώστε να μπορούμε να αποφασίσουμε αν όλες οι προτάσεις της θεωρίας είναι αληθείς ή όχι μετά απο έναν πεπερασμένο αριθμό βημάτων*; Μια θεωρία για την οποία υπάρχει μια μέθοδο που αποδεικνύει την αλήθεια ή όχι των προτάσεών της λέγεται πλήρης θεωρία. Στο πρόβλημα αυτό απάντησε ο K. Gödel (1906-1978) και είπε ότι κάθε μαθηματική θεωρία που περιέχει την αριθμητική δεν είναι πλήρης, ότι δηλαδή υπάρχει ένα σημείο πέρα του οποίου η αλήθεια ή όχι των προτάσεων της θεωρίας δεν μπορεί να δειχθεί μόνο

[2]Το άλλο μοντέλο που χρησιμοποιούμε είναι η μηχανή RAM - Random Access Machine.



Σχήμα 1: D. Hilbert - A. Turing - K. Gödel

με την θεωρία την ίδια. Αυτό φαίνεται ότι κατέστρεψε το πρόγραμμα του Hilbert που αφορούσε την ενοποίηση της μαθηματικής θεωρίας, αλλά το πεδίο που μια θεωρία ικανοποιεί το αίτημα του Hilbert φάνηκε ότι ήταν ένας μαθηματικός παράδεισος παρά το αρνητικό αποτέλεσμα του Gödel, ισάξιος με αυτόν τον παράδεισο του Cantor, για να θημηθούμε τη φιλάρεσκη φράση του Hilbert.

Ας δούμε λίγο όμως τα εργαλεία που χρησιμοποίησε ο Gödel. Στην απόδειξη αυτή λοιπόν[3] εισάγει κάποιες παράξενες έως τότε συναρτήσεις, που δεν είναι τίποτα άλλο παρά οι επαναληπτικές συναρτήσεις. Αυτές τις συναρτήσεις χρησιμοποίησε ο Turing για να αποδείξει ότι οι έννοιες μιας αξιωματικής θεωρίας μπορεί να καθορισθούν και όλα τα αποτελέσματα της μη-πληρότητας του Gödel μπορεί να εκφρασθούν.

Το μαθηματικό αυτό μοντέλο του Turing έχει τη λογική δομή του σημερινού υπολογιστή. Με άλλα λόγια οι επαναληπτικές συναρτήσεις μπορούν να αναπαράγουν την ανθρώπινη λογική διαδιακασία σε συστήματα αποκλειστικά αξιωματικά. Με το μοντέλο αυτό ο Turing θέτει τα θεμέλια της θεωρητικής πληροφορικής. Το έργο του επεκτείνεται και σε άλλους τομείς όπως για παράδειγμα στον μικροπρογραμματισμό, και ειδικότερα στη κατασκευή της γλώσσσας assemply, γλώσσα που αποκοδικοποιεί αριθμητικούς κώδικες σε κώδικες μηχανής.

[3]From Frege to Gödel, *Some mathematical results on completeness and consistency. On formally undecidable propositions of Principia mathematics and related systems I, and On completeness and consistency, 1930-1931a-193b*,p. 592-617, ed. J.v Heijenoort, Harvard Univ. Press, 1967.

4 Δύο παραδείγματα

Σαν πρώτο παράδειγμα της πολυπλοκότητας αλγορίθμου θα αναφέρουμε τον αλγόρθμο πρόσθεσης και πολλαπλασιασμού αριθμών στο δυαδικό σύστημα και θα ολοκληρώσουμε την παρουσίαση με μια σύγκριση τριών αλγορίθμων για τον υπολογισμό του αριθμού π.

4.1 Πρόσθεση και αφαίρεση

Λέμε δεκαδική αναπαράσταση του αριθμού α μήκους n, το πεπερασμένο άθροισμα:

$$\alpha = \alpha_n \cdot 10^n + \alpha_{n-1} \cdot 10^{n-1} + \ldots + \alpha_1 \cdot 10^1 + \alpha_0 \cdot 10^0 = \sum_{i=0}^{n} \alpha_i \cdot 10^i \tag{1}$$

όπου $\alpha_i \in \{0, 1, \ldots, 9\}$. Θεωρήστε τώρα δύο αριθμούς α και β στη δεκαδική τους αναπαράσταση 1, μήκους n, και υποθέστε ότι θέλετε να γράψετε τους αλγορίθμους $s = \alpha + \beta$ και $p = \alpha \cdot \beta$.

Για το άθροισμα s στη δεκαδική μορφή $\sum_{i=0}^{n} s_i \cdot 10^i$ κάθε s_i είναι ίσο με:

$$\begin{cases} s_i = (a_i + b_i + r_i) \bmod 10 \\ r_{i+1} = 0 \text{ αν } a_i + b_i + r_i < 10 \\ r_{i+1} = 1 \text{ αν όχι} \\ r_0 = 0 \end{cases}$$

Ο αλγόριθμος εκτελεί σε κάθε βήμα $i \geq n$ δύο προσθέσεις και μια αφαίρεση, συνολικά $3n$ προσθέσεις. Άρα η πολυπλοκότητα της πρόσθεσης είναι γραμμική και ίση με $O(n)$.

Για το γινόμενο p ο αλγόριθμος θα εκτελέσει n^2 πολλαπλασιασμούς, πιθανόν μερικοί να είναι με το 0, και στη συνέχεια n προσθέσεις οι οποίες είναι απαραίτητες για το αποτέλεσμα. Σύνολο $n^2 + n$ στοιχειώδεις πράξεις και η πολυπλοκότητα θα είναι $O(n^2 + n) = O(n^2)$.

Μπορούμε να το κάνουμε καλύτερα. Ο αλγόριθμος του τσετσένου μαθηματικού Karatsuba κάνει τον πολλαπλασιασμό με πολυπλοκότητα $O(n^{ln(3)/ln(2)}) \sim O(n^{1.58})$. Κάνει δηλαδή περίπου $\sqrt{n}$ λιγότερες πράξεις. Ο αλγόριθμος αυτός βασίζεται σε μια τεχνική που ονομάζεται *διαίρει και βασίλευε*. Αφού φέρει τους δύο αριθμούς α και β στη μορφή $a \cdot 10^k + b$ και $c \cdot 10^k + d$ αντίστοιχα, εκτελεί τον πολλαπλασιασμό ως εξής:

$$(a \cdot 10^k + b)(c \cdot 10^k + d) = ac \cdot 10^{2k} + (ac + bd - (a - b)(c - d)) \cdot 10^k + bd.$$

4.2 Τρείς αλγόριθμοι για το π

Ο υπολογισμός του $\pi = 3.141592653589793\ldots$ (το πηλίκο της περιφέρειας κύκλου δια της διαμέτρου του) έχει μελετηθεί απο την αρχαιότητα και είναι ένα κλασσικό αλγοριθμικό παράδειγμα. Ο Αρχιμήδης (287-212 πχ) έδωσε πρώτος έναν αλγόριθμο για τον υπολογισμό του π.

Η αρχή του Αρχιμήδη ήταν η εξής: Αν p_n είναι η περίμετρος ενός κανονικού n-γώνου εγγεγραμμένου σε κύκλο ακτίνας $1/2$, τότε $\pi = \lim_{n \longrightarrow \infty} p_n$. Τότε μπορούμε να υπολογίσουμε το π αν υπολογίσουμε την ακολουθία των περιμέτρων $p_6, p_{12}, p_{24}, \ldots$ διπλασιάζοντας κάθε φορά τον αριθμό των πλευρών. Αν η πλευρά ενός n-γώνου είναι $u_n = (1/n)p_n$ στοιχειώδεις πράξεις μας δίνουν τον αναγωγικό τύπο:

$$u_{2n} = \sqrt{\frac{1}{2}\left(1 - \sqrt{1 - u_n^2}\right)}, \text{ με } p_{2n} = 2nu_{2n} \tag{2}$$

Ξεκινώντας απο το κανονικό εξάγωνο με $u_6 = 1/2$ και $p_6 = 3$ μπορούμε να υπολογίσουμε τις διάφορες προσεγγίσεις του π (δες τον παρακάτω πίνακα).

Ό άλλος αλγόριθμος οφείλεται στον J. Machin (1680-1752), και δείνει προσεγγίσεις του π για διαφορετικές τιμές του δείκτη n απο τον τύπο:

$$\pi_n = 16\text{Arc } tg_n \frac{1}{5} - 4\text{Arc } tg_n \frac{1}{239} \tag{3}$$

όπου, $\text{Arc } tg_n x = \sum_{k=0}^{n} (-1)^k \frac{x^{2k+1}}{(2k+1)}$. Μερικές τιμές των π_n δίνονται στον πίνακα. Με τον τύπο αυτό οι Guilloud-Filliatre υπολόγισαν το 1966 μερικά εκατομύρια δεκαδικά του π.

Τέλος, ο αλγόριθμος του Salamin εξαιτίας τις καλές ιδιότητες σύγκλισης του π έδωσε τον εξής τύπο:

$$\begin{cases} \pi_n = \dfrac{4a_n^2}{1 - 2\displaystyle\sum_{m=1}^{n} 2^m (a_m^2 - b_m^2)} \\[2ex] a_{m+1} = \dfrac{1}{2}(a_m + b_m);\ b_{m+1} = \sqrt{a_m b_m} \\[2ex] a_0 = 1;\ b_0 = \dfrac{1}{\sqrt{2}} \end{cases} \tag{4}$$

Ο αλγόριθμος βασίζεται στη θεωρία των ελλιπτικών ολοκληρωμάτων.

Ας δούμε τώρα τον πίνακα με τις διάφορες προσεγγίσεις των τριών τύπων.

Αρχιμήδης	*Machin*	*Salamin*
$\pi_0 = 3$	$\pi_0 = 3.183...$	$\pi_0 = 4$
$\pi_1 = 3.105...$	$\pi_1 = 3.1405...$	$\pi_0 = 3.187...$
$\pi_2 = 3.132...$	$\pi_2 = 3.14162...$	$\pi_2 = 3.14168...$
$\pi_3 = 3.139...$	$\pi_3 = 3.1415917...$	$\pi_3 = 3.14159265389...$
$\pi_4 = 3.14103...$	$\pi_4 = 3.141592682...$	$\|\pi - \pi_4\| < 10^{-20}$
$\pi_5 = 3.14145...$	$\pi_5 = 3.1415926526...$	$\|\pi - \pi_5\| < 10^{-42}$
$\pi = 3.14159265358979323846264338327950288419716939937 51...$		

Η ταχεία σύγκλιση του αλγορίθμου του Salamin είναι άρκετά ικανοποιητική όπως φαίνεται στις τελευταίες γραμμές του πίνακα. Η αξιολόγιση των τριών αυτών αλγορίθμων για το ίδιο πρόβλημα μας οδηγεί στην μέτρηση της πολυπλοκότητας που είναι για τον αλγόριθμο του Αρχιμήδη και τον αλγόριθμο του Machin είναι $O(n)$, ενώ για αυτόν του Salamin είναι $O(log(n))$, όπου n ο ακέραιος που εκφράζει την επιθυμιτή προσέγγιση μετά απο εκτέλεση n βημάτων του κάθε αλγορίθμου.

5 Η εποχή της ενηλικίωσης

Η δεκαετία του '80 κυριαρχείτε απο έναν υπολογιστικό οργασμό, σαν ανώριμοι συγγραφείς που είμαστε, χαρακτηρίζουμε την εποχή μετά το '80 σαν εποχή ενηλικίωσης του υπολογισμού. Αλγόριθμοι βελτιώνονται και γίνονται αποτελεσματικότεροι, οι μηχανές ανανεώνουν τα υλικά τους και εμφανίζονται εξαιρετικά λογισμικά που φιλοδοξούν να πάρουν ένα μερίδιο στην έρευνα είτε στα μαθηματικά είτε στη φυσική είτε στη βιολογία. Τα μαθηματικά μετά απο έναν γάμο πάνω απο 4 αιώνες με τη φυσική κοιτάζουν τώρα το *άπειρο* πλήθος των πληροφοριών που δίνει η βιολογία για την κατανόηση των μηχανισμών και την φυσιολογία των κυτάρων.

Στην τελευταία εικοσαετία έχουν περιγραφεί νέοι αλγόριθμοι για να λύσουν είτε γνωστά προβλήματα είτε νέα, όπως αυτά της οικονομίας, της θεωρίας παιγνίων και βιολογίας. Η εύρεση όμως αποτελεσματικών και βέλτιστων αλγορίθμων είναι μια πρόκληση στη Αλγοριθμική Θεωρία η οποία μόλις τώρα άρχισε.

...