ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΡΗΤΗΣ

Τμήμα Εφαρμοσμένων Μαθηματικών

Πτυχιακή Εργασία

Ακριβής Σύγκριση Αλγεβρικών Αριθμών Βαθμού το πολύ 3

Επιβλέπων Καθηγητής :
Μενέλαος Καραβέλας

Πιπερίδη Σοφία-Άννα
ΑΜ: 1135

Τριμελής επιτροπή :
Θ. Γαρεφαλάκης, Μ. Καραβέλας, Ν. Τζανάκης

Ηράκλειο, Φεβρουάριος 2013

Περιεχόμενα

1 Περίληψη

Παρουσιάζουμε (βέλτιστο) αλγόριθμο για : (1) την απομόνωση και (2) τον υπολογισμό των πολλαπλοτήτων, των πραγματικών ριζών πολυωνύμων και την κατασκευή, σύγκριση και υπολογισμό προσήμων πραγματικών αλγεβρικών αριθμών όταν ο βαθμός είναι ≤ 3.

Πραγματικοί αλγεβρικοί αριθμοί παρουσιάζονται στη σχεδίαση με υπολογιστή, στη γεωμετρική μοντελοποίηση, στη μη γραμμική υπολογιστική γεωμετρία. Αποδοτικοί θεωρητικά και πρακτικά αλγόριθμοι για υπολογισμούς με αλγεβρικούς αριθμούς μικρού βαθμού είναι πολύ σημαντικοί καθώς εμπλέκονται στα βασικά βήματα υπολογισμού της διάταξης επιπέδων αλγεβρικών καμπυλών, στα Voronoi διαγράμματα κυρτών αντικειμένων [1, 2, 3, 4, 5, 6, 7] και στις κινητικές δομές δεδομένων [8]. Είναι επίσης πολύ σημαντικοί σε βιβλιοθήκες λογισμικού όπως η Core [9], η Exacus [10], η Esolid [11], ο επερχόμενος καμπύλος πυρήνας της Cgal [12].

Οι πολυωνυμικές εξισώσεις βαθμού ≤ 3 μπορούν να λυθούν με ριζικά, ή με άλλα λόγια επιδέχονται αναλυτικής λύσης. Ωστόσο, θέλουμε να αποφύγουμε την αναλυτική προσέγγιση για πολλούς λόγους. Καταρχάς, ο υπολογισμός των ριζικών είναι πολύ ακριβός υπολογιστικά και δεν μπορεί να γίνει μόνο με τη χρήση των ρητών αριθμών. Ακόμα κι αν χρησιμοποιήσουμε αριθμούς κινητής υποδιαστολής οι τύποι των ριζικών είναι πολύ ασταθείς αριθμητικά, ιδιαίτερα στην περίπτωση πολλαπλών ριζών. Δεύτερον στην περίπτωση του τρίτου βαθμού, όταν το πολυώνυμο έχει πραγματικές ρίζες τότε απαιτούνται (πάντοτε) πράξεις με μιγαδικούς αριθμούς προκειμένου να τις υπολογίσουμε. Τέλος, ακόμα και αν χρειαζόμαστε μόνο κάποια (κάποιες) από τις ρίζες του πολυωνύμου δεν μπορούμε να συνάγουμε τη διάταξη από τους τύπους με τα ριζικά και πρέπει να υπολογίσουμε όλες τις ρίζες.

Προκειμένου να αντιμετωπίσουμε όλα τα προηγούμενα προβλήματα, χρειαζόμαστε αλγορίθμους που να είναι συμβολικοί, που να εμπλέκουν, αν είναι δυνατόν, μόνο τις βασικές πράξεις και να υπολογίζουν και τις πολλαπλότητες των ριζών. Για να αποφύγουμε προβλήματα αριθμητικής αστάθειας και υπολογισμούς με αριθμούς κινητής υποδιαστολής, απορρίπτουμε την αναλυτική επίλυση και αναπαριστούμε τις (πραγματικές) ρίζες χρησιμοποιώντας την αναπαράσταση με διαστήματα απομόνωσης. Η αναπαράσταση με διαστήματα απομόνωσης ενός πραγματικού αλγεβρικού αριθμού, $\alpha \in R_{alg}$, συμβολίζεται με $\alpha \cong (\mathrm{A}(x), J)$, όπου $\mathrm{A} \in Z[x]$ είναι χωρίς τετράγωνα, $\mathrm{A}(\alpha) = 0$, $\alpha \in J$, $J = (\alpha, \beta)$, $\alpha, \beta \in Q$ και το A δεν έχει άλλη πραγματική ρίζα στο $[\alpha, \beta]$. Στόχος μας είναι να παρουσιάσουμε αλγορίθμους με πολυπλοκότητα $O(1)$.

Προκειμένου να υπολογίσουμε συμβολικά το πλήθος των πραγματικών ριζών και τις πολλαπλότητές τους, μιας πολυωνυμικής εξίσωσης βαθμού ≤ 3 θεωρούμε τους συντελεστές του πολυωνύμου ως παραμέτρους και επεκτείνουμε το σύστημα διακρινουσών (discrimination system), DS,το οποίο εισήγαγε ο Yang [13], δείτε επίσης [14, 15, 16, 17, 18, 19, 20, 21], στο σύστημα διακρινουσών και απομόνωσης (isolation and discrimination system),το οποίο συμβολίζεται ως IDS. Το IDS είναι ένα σύνολο από πολυωνυμικές εκφράσεις και ανισότητες στους συντελεστές του πολυωνύμου, οι οποίες αρκούν για να περιγράψουν το πλήθος, τις πολλαπλότητες και την αναπαράσταση με διάστημα απομόνωσης των πραγματικών ριζών. Για παράδειγμα, στο γνωστό σε όλους τριώνυμο η διακρίνουσα του αρκεί για να χαρακτηρίσει (και να απομονώσει) το πλήθος των πραγματικών ριζών. Γνωρίζουμε ότι ο μηδενισμός της διακρίνουσας ενός πολυωνύμου είναι ικανή και αναγκαία συνθήκη έτσι ώστε το πολυώνυμο να έχει (μιγαδικές) ρίζες με πολλαπλότητες. Για πολυώνυμα μεγαλύτερου βαθμού χρειάζονται επιπλέον ποσότητες, όχι για να βεβαιώσουν την ύπαρξη ριζών με πολλαπλότητα αλλά για να υπολογίσουν το πλήθος και την πολλαπλότητα των ριζών, αυτές είναι το σύστημα διακρινουσών.

Η προσέγγιση που θα ακολουθήσουμε υπολογίζει το σύστημα διακρινουσών χρησιμοποιώντας τις προσημασμένες ακολουθίες υπο-επιλυουσών. Τις διάφορες ποσότητες που εμφανίζονται, τις εκφράζουμε ως συνάρτηση των αναλλοίωτων (invariant) του πολυωνύμου και ως στοιχεία του πίνακα Bezout του πολυωνύμου και τις παραγώγου του, προκειμένου να ελαχιστοποιήσουμε το υπολογιστικό κόστος. Για την αναπαράσταση με διαστήματα απομόνωσης υπολογίζουμε ρητούς αριθμούς (σημεία διαχωρισμού) που διαχωρίζουν τις πραγματικές ρίζες, τους οποίους εκφράζουμε ως ρητές πολυωνυμικές συναρτήσεις των συντελεστών του πολυωνύμου. Τα σημεία διαχωρισμού είναι πολύ σημαντικά καθώς μπορούν να αποτελέσουν αρχικές προσεγγίσεις σε επαναληπτικές μεθόδους προσέγγισης των ριζών.

Η κατασκευή αλγεβρικών αριθμών βαθμού ≤ 3 σε σταθερό χρόνο μας επιτρέπει να παρουσιάσουμε αλγορίθμους, επίσης σταθερού χρόνου, για τη σύγκριση, για τον υπολογισμό του προσήμου αποτίμησης και για την επίλυση πολυωνυμικών συστημάτων δύο μεταβλητών συνολικού βαθμού ≤ 2. Οι παρουσιαζόμενοι αλγόριθμοι είναι βέλτιστοι ή σχεδόν βέλτιστοι ως προς τον αλγεβρικό βαθμό (algebraic degree) των εξεταζόμενων (πολυωνυμικών) ποσοτήτων ως προς τους συντελεστές του πολυωνύμου.

Το γεγονός ότι μπορούμε να απομονώσουμε τις πραγματικές ρίζες πολυωνύμων βαθμού ≤ 3 έχοντας υπολογίσει συμβολικά όλες τις απαιτούμενες ποσότητες και το IDS μας επιτρέπει να προτείνουμε ειδικούς αλγορίθμους για την απαλοιφή ενός ποσοδείκτη $\exists x$ από κάποια φόρμουλα $\exists x(P)$, όπου η P αποτελείται από διαζεύξεις και συζεύξεις πολυωνυμικών εξισώσεων βαθμού ≤ 3 και ανισώσεων οποιουδήποτε βαθμού, βασιζόμενοι στην προσέγγιση της εικονικής αντικατάστασης (virtual substitution) [18, 19, 20, 21, 22, 23, 24].

Σε ότι θα ακολουθήσει θα θεωρήσουμε ότι η είσοδος είναι οι συντελεστές ενός πολυωνύμου f, όπου $\deg(f) \leq 3$, και όλοι οι αλγόριθμοι που θα παρουσιάσουμε εξετάζουν το πρόσημο κάποιων πολυωνυμικών ποσοτήτων στους συντελεστές. Από την άποψη της πολυπλοκότητας σκοπός μας είναι να ελαχιστοποιήσουμε όσο το δυνατόν το συνολικό βαθμό των εξεταζόμενων ποσοτήτων και ή το δυνατόν τις απαιτούμενες πράξεις.

2 Ιστορική αναδρομή

Η αλγεβρική λύση μίας εξίσωσης βαθμού ≤ 3 συνίσταται στον υπολογισμό κάποιων τύπων για τις ρίζες, οι οποίοι περιέχουν πεπερασμένο αριθμό βασικών πράξεων και ριζικών.

Οι πρώτοι που ασχολήθηκαν με την εύρεση των ριζών της πολυωνυμικής εξίσωσης δευτέρου βαθμού ήταν οι Αιγύπτιοι, οι Βαβυλώνιοι και οι Κινέζοι μηχανικοί. Ίσως η πρώτη γνωστή λύση της εξίσωσης να είναι ένας Αιγυπτιακός πάπυρος από την εποχή του Μέσου Βασιλείου, περίπου 2160-1700πΧ. Επίσης σε πινακίδια από πηλό τα οποία χρονολογούνται ανάμεσα στο 1800πΧ και 1600πΧ οι αρχαίοι Βαβυλώνιοι παρουσιάζουν πολυωνυμικές εξισώσεις δευτέρου βαθμού και μερικούς απλούς τρόπους επίλυσης τους. Ο Ινδός μαθηματικός Baudhayana ο οποίος έγραψε το Sulba Sutr στην αρχαία Ινδία περίπου τον 8ο αιώνα πΧ χρησιμοποίησε πρώτος εξισώσεις της μορφής $ax^2 = c$ και $ax^2 + bx = c$ και παρουσίασε τρόπους επίλυσης τους.

Βαβυλώνιοι μαθηματικοί γύρω στο 400πΧ και Κινέζοι μαθηματικοί γύρω στο 200πΧ χρησιμοποίησαν μεθόδους συμπληρώματος του τετραγώνου προκειμένου να επιλύσουν πολυωνυμικές εξισώσεις δευτέρου βαθμού με θετικές ρίζες αλλά δεν παρουσίασαν ένα γενικό τύπο επίλυσης. Ο Ευκλείδης παρουσίασε μία γενική γεωμετρική μέθοδο γύρω στο 300πΧ και ο Διόφαντος στα Αριθμητικά υπολόγιζε τη μία λύση μιας δευτεροβάθμιας εξίσωσης. Στο βιβλίο Bakshali Manuscript το οποίο γράφτηκε στην Ινδία μεταξύ 200πΧ και του 400μΧ παρουσιάστηκε ένας γενικός αλγεβρικός τύπος για την επίλυση δευτέρου βαθμού εξισώσεων. Ο πρώτος μαθηματικός που υπολόγιζε αρνητικές λύσεις της εξίσωσης ήταν ο Brahmagupta (Ινδία, 7ος αιώνας). Ο Muhammad ibn Musa al-Kwarizmi (Περσία, 9ος αιώνας) ανέπτυξε ένα σύνολο από τύπους οι οποίοι δίνουν τις θετικές ρίζες. Ο Abraham bar Hiyya Ha-Nasi, γνωστός επίσης και με το λατινικό του όνομα Savasorda, εισήγαγε στην Ευρώπη την πλήρη λύση της εξίσωσης με το βιβλίο του Liber embadorum τον 12ο αιώνα.

Ο Shridhara (Ινδίας, 9ος αιώνας) ήταν από τους πρώτους μαθηματικούς που έδωσαν ένα γενικό κανόνα επίλυσης της πολυωνυμικής εξίσωσης δευτέρου βαθμού, αλλά τα γραπτά του δεν διεσώθησαν. Επίσης ο Vieta ήταν ανάμεσα στους πρώτους που αντικατέστησαν τις γεωμετρικές μεθόδους με αλγεβρικές αν και είναι αμφίβολο αν είχε συλλάβει τον γενικό τύπο επίλυσης της εξίσωσης.

Οι προσπάθειες επίλυσης τις κυβικής εξίσωσης ξεκινούν από τα τέλη του 15ου αιώνα. Το 1494 ο Ιταλός μαθηματικός Luuca Pacioli μελετούσε την εξίσωση και εξέφρασε την πεποίθηση ότι είναι αδύνατον να επιλυθεί. Αυτή η παρατήρηση λειτούργησε ως πρόκληση και αποτέλεσε το Άγιο Δισκοπότηρο για την Ιταλική μαθηματική κοινότητα του 16ου αιώνα.

Ο πρώτος που αντιμετώπισε την πρόκληση ήταν ο Scipione dal Ferro (1465-1526), καθηγητής μαθηματικών στο Πανεπιστήμιο της Bologna. Περίπου το 1500 ο dal Ferro ανακάλυψε την λύση της συμπιεσμένης (depressed) κυβικής εξίσωσης, δηλαδή της κυβικής εξίσωσης χωρίς τον όρο δευτέρου βαθμού. Ένας απλός γραμμικός μετασχηματισμός φέρνει πάντα τη γενική εξίσωση τρίτου βαθμού σε αυτή τη μορφή. Ο dal Ferro δεν δημοσίευσε τα αποτελέσματα του πιθανώς γιατί είχε αμφιβολίες για την ορθότητα τους. Αν η λύση του ήταν λανθασμένη τότε θα έθετε σε σοβαρό κίνδυνο την πανεπιστημιακή του θέση. Ωστόσο την κοινοποίησε στον ανιψιό και μαθητή του Antonio Maria Fior.

Ο Fior δεν ήταν καλός στο να κρατάει μυστικά, φήμες κυκλοφορούσαν στην Bologna ότι η κυβική εξίσωση είχε επιλυθεί. Ορμώμενος από τις φήμες ο Nicolo of Brescia, γνωστός ως Tartaglia (αυτός που τραυλίζει) έλυσε μια ειδική περίπτωση της κυβικής την $x^3+mx^2=n$ και δεν το κράτησε κρυφό!

Ο Fior προσκάλεσε τον Tartaglia σε ένα δημόσιο διαγωνισμό. Ο καθένας θα έδινε στον άλλο 30 προβλήματα και θα είχαν στη διάθεση τους 40 ή 50 μέρες να τα λύσουν. Νικητής θα ήταν αυτός που έλυνε τα περισσότερα προβλήματα και υπήρχαν και μικρά βραβεία για κάθε πρόβλημα ξεχωριστά. Φυσικά ο Fior έθεσε ως ένα από τα προβλήματα την συμπιεσμένη κυβική εξίσωση. Ο Tartaglia έλυσε σε λιγότερο από μία μέρα όλα τα προβλήματα εκτός από την κυβική εξίσωση. Λίγο πριν εκπνεύσει η προθεσμία έλυσε και το τελευταίο πρόβλημα.

Τα νέα της νίκης του Tartaglia έφτασαν στον Gerolamo Cardano από το Milan, ο οποίος εντυπωσιάστηκε από τη λύση της εξίσωσης. Έγραψε δε πολλές φορές στον Tartaglia και τελικά τον έπεισε να τον επισκεφτεί το 1539. Στη συνάντησή τους κατάφερε να πείσει τον Tartaglia να του αποκαλύψει τη λύση της κυβικής εξίσωσης, ορκίστηκε όμως να μην την αποκαλύψει πριν ο Tartaglia την δημοσιεύσει. Το 1543 ο Cardano με το μαθητή του Ludovico Ferrari κοιτάζοντας τα χαρτιά του dal Ferro ανακάλυψε ότι η λύση του Tartaglia ήταν η ίδια με αυτή του dal Ferro και θεώρησε ότι μπορούσε να σπάσει τον όρκο του. Τελικά δημοσίευσε τη λύση του πολυωνύμου τρίτου (και τετάρτου) βαθμού στην εργασία του Ars Magna (Μεγάλη Τέχνη) το 1545, που είναι η πρώτη εργασία άλγεβρας στα λατινικά, δίνοντας τα εύσημα μόνο στον dal Ferro και στον εαυτό του.

Περισσότερα ιστορικά στοιχεία παρουσιάζονται στην ιστοσελίδα του BBC [1], στην ιστοσελίδα mathworld [2] του Eric W. Weisstein και στις εργασίες των Dunham [25] και Boyer [26]. Οι προαναφερθείσες αναφορές αποτέλεσαν και τις πηγές της σύντομης ιστορικής ανασκόπησης που παρουσιάσαμε.

[1] http://www.bbc.co.uk/dna/h2g2/A2982567

[2] http://mathworld.wolfram.com/

3 Πολυώνυμα απομόνωσης και πραγματικές ριζές

Από το θεώρημα του Rolle[3] μπορούμε εύκολα να συνάγουμε ότι ανάμεσα σε δύο πραγματικές ρίζες ενός πολυωνύμου παρεμβάλλεται τουλάχιστον μία ρίζα της παραγώγου (δείτε για παράδειγμα [27, 28, 29]). Ωστόσο, μπορούμε να υπολογίσουμε και άλλα πολυώνυμα με αυτήν την ιδιότητα. Η παρακάτω πρόταση μας παρέχει μία τέτοια δυνατότητα. Η παρακάτω πρόταση, η οποία οφείλεται στους Sederberg and Chang [30], και η οποία γενικεύει ένα θεώρημα του de Gua [27, 28] μας παρέχει μια τέτοια δυνατότητα.

Πρόταση 3.1 Έστω $f(X) \in \mathbb{R}[X]$, δύο συνεχόμενες πραγματικές ρίζες του γ_1, γ_2 και δύο άλλα (οποιαδήποτε) πολυώνυμα $B(X), C(X) \in \mathbb{R}[X]$. Θεωρούμε το πολυώνυμο

$$A(X) := B(X)f'(X) + C(X)f(X) \tag{1}$$

όπου $A \in \mathbb{R}[X]$ και f' η παράγωγος του f. Τουλάχιστον ένα από τα $A(X)$ και $B(X)$ έχει τουλάχιστον μία πραγματική ρίζα στο κλειστό $[\gamma_1, \gamma_2]$. Επιπρόσθετα είναι πάντοτε δυνατόν να έχουμε $\deg(A) + \deg(B) \leq \deg(f) - 1$.
Τα $A(X)$ και $B(X)$ ονομάζονται πολυώνυμα απομόνωσης (isolating polynomials).

Απόδειξη: Αν $A(\gamma_1) = 0$ ή $A(\gamma_2) = 0$ ή $B(\gamma_1) = 0$ ή $B(\gamma_2) = 0$ η πρόταση ισχύει. Ας υποθέσουμε ότι γ_1 και γ_2 δεν είναι ρίζες ούτε του $A(X)$ ούτε του $B(X)$.

Εφόσον $f(\gamma_1) = 0$, η (1) γίνεται

$$A(\gamma_1) = B(\gamma_1)f'(\gamma_1) \Rightarrow \mathrm{sign}(f'(\gamma_1)) = \mathrm{sign}(A(\gamma_1)B(\gamma_1)) \tag{2}$$

Εφόσον $f(\gamma_2) = 0$, η (1) γίνεται

$$A(\gamma_2) = B(\gamma_2)f'(\gamma_2) \Rightarrow \mathrm{sign}(f'(\gamma_2)) = \mathrm{sign}(A(\gamma_2)B(\gamma_2)) \tag{3}$$

Από το θεώρημα του Rolle ανάμεσα σε δύο πραγματικές ρίζες του f υπάρχει τουλάχιστον μία ρίζα του f' και από το θεώρημα του Bolzano έχουμε ότι $\mathrm{sign}(f'(\gamma_1))\mathrm{sign}(f'(\gamma_1)) < 0$. Συνεπώς, συνδυάζοντας τις (2) και (3) έχουμε ότι

$$\mathrm{sign}(A(\gamma_1)B(\gamma_1))\mathrm{sign}(A(\gamma_2)B(\gamma_2)) < 0$$

Από το θεώρημα Bolzano η συνάρτηση $A(X)B(X)$ έχει τουλάχιστον μία πραγματική ρίζα στο (γ_1, γ_2) και η πρόταση αποδείχτηκε. □

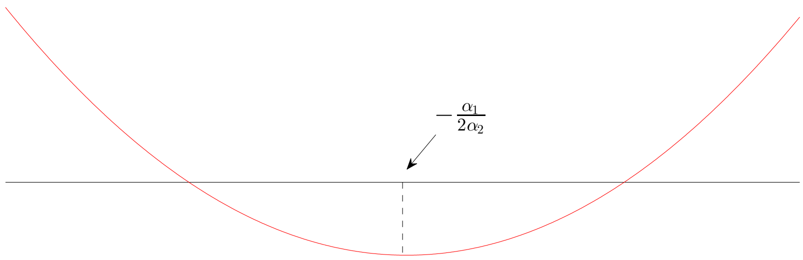
[3] Αν μία συνάρτηση f είναι :

- συνεχής στο κλειστό διάστημα$[\alpha, \beta]$
- παραγωγήσιμη στο ανοικτό διάστημα (α, β) και
- $f(\alpha) = f(\beta)$

τότε $\exists$ τουλάχιστον ένα $\xi \in (\alpha, \beta) : f'(\xi) = 0$

4 Πραγματική επίλυση

Μπορούμε τώρα να εξετάσουμε τα πολυώνυμα βαθμού ≤ 3. Στόχος μας είναι να χαρακτηρίσουμε, με συμβολικό τρόπο, τις πραγματικές τους ρίζες (πλήθος και πολλαπλότητες), να τις απομονώσουμε και τελικά να υπολογίσουμε το σύστημα διακρινουσών και απομόνωσης.



Σχήμα 1: Η τετμημένη του ακροτάτου απομονώνει τις πραγματικές ρίζες ενός πολυωνύμου $f = \alpha_2 X^2 + \alpha_1 X + \alpha_0$ για το οποίο ισχύει $\Delta > 0$.

4.1 Το δευτεροβάθμιο πολυώνυμο

Θεωρούμε το δευτεροβάθμιο πολυώνυμο (quadratic), ή τριώνυμο, $f \in \mathbb{Z}[X]$, με $\alpha_2 > 0$

$$f(X) = \alpha_2 X^2 + \alpha_1 X + \alpha_0. \tag{4}$$

Οι (μιγαδικές) λύσεις του (4) είναι

$$\frac{-\alpha_1 \pm \sqrt{\Delta}}{2\alpha_2},$$

όπου $\Delta = disc(f) = \alpha_1^2 - 4\alpha_2\alpha_0$ είναι η διακρίνουσα του f. Το πρόσημο της διακρίνουσας παρέχει όλες τις πληροφορίες που χρειαζόμαστε προκειμένου να χαρακτηρίσουμε και να απομονώσουμε τις πραγματικές ρίζες του f, όπως δείχνει και το ακόλουθο λήμμα:

Λήμμα 4.1.1 Έστω $f(X) = \alpha_2 X^2 + \alpha_1 X + \alpha_0 \in \mathbb{R}[X]$. Αν $\Delta = 0$ τότε το f έχει μία διπλή ρίζα, την $-\frac{\alpha_1}{2\alpha_2}$. Αν $\Delta > 0$ τότε ο αριθμός $-\frac{\alpha_1}{2\alpha_2}$ διαχωρίζει τις δύο πραγματικές ρίζες του f.

Απόδειξη: Αν $\Delta = 0$ η απόδειξη είναι προφανής και βασίζεται στην τεχνική της συμπλήρωσης του τετραγώνου.

Αν $\Delta > 0$ τότε το f έχει δύο διαφορετικές πραγματικές ρίζες. Παρατηρούμε ότι η παράγωγος $f' = 2\alpha_2 X + \alpha_1$ έχει ρίζα το $-\frac{\alpha_1}{2\alpha_2}$ και σύμφωνα με το θεώρημα του Rolle είναι ανάμεσα στις ρίζες του f. Η περίπτωση αυτή παρουσιάζεται στο Σχ.1. $\square$

Συνεπώς έχουμε την ακόλουθη πρόταση:

Πρόταση 2.1.2 Το ΙΔΣ του δευτεροβάθμιου πολυωνύμου (4) είναι

(1)	$\Delta < 0$	$\{\ \}$	
(2)	$\Delta = 0$	$\{2\}$	$\gamma_1 = -\frac{\alpha_1}{2\alpha_2}$
(3)	$\Delta > 0$	$\{1{,}1\}$	$\gamma_1 \cong (f, (-\infty, -\frac{\alpha_1}{2\alpha_2}))$ $\gamma_2 \cong (f, (-\frac{\alpha_1}{2\alpha_2}, +\infty))$

Σημείωση 2.1.3 (Πως διαβάζουμε το IDS). Η πρώτη στήλη περιέχει την απαρίθμηση των περιπτώσεων. Η δεύτερη στήλη περιέχει τις πολυωνυμικές ανισότητες που ισχύουν. Η τρίτη στήλη περιέχει τις πολλαπλότητες και το πλήθος των πραγματικών ριζών και η τελευταία στήλη την αναπαράσταση με διαστήματα απομόνωσης. Ο συμβολισμός $\{2\}$ σημαίνει ότι έχουμε μία διπλή ρίζα και ο $\{1{,}1\}$ ότι έχουμε δύο διαφορετικές πραγματικές ρίζες πολλαπλότητας 1.

Για παράδειγμα αν ένα (δευτεροβάθμιο) πολυώνυμο, όπως στην (4), είναι τύπου (2) τότε ισχύει $\Delta = 0$, έχει μία διπλή ρίζα, η οποία είναι ο αριθμός $-\frac{\alpha_1}{2\alpha_2}$.

4.2 Το τριτοβάθμιο πολυώνυμο

Θεωρούμε το κυβικό πολυώνυμο (cubic) $f \in \mathbb{Z}[X]$, όπου $\alpha_3 > 0$,

$$f(X) = \alpha_3 X^3 + \alpha_2 X^2 + \alpha_1 X + \alpha_0 \tag{5}$$

του οποίου οι μιγαδικές λύσεις είναι[4]

$$\begin{aligned}
\gamma_1 &= -\frac{\alpha_2}{3\alpha_3} + (S+T) \\
\gamma_2 &= -\frac{\alpha_2}{3\alpha_3} - \frac{1}{2}(S+T) + \frac{1}{2}i\sqrt{3}(S-T) \\
\gamma_3 &= -\frac{\alpha_2}{3\alpha_3} - \frac{1}{2}(S+T) - \frac{1}{2}i\sqrt{3}(S-T)
\end{aligned} \tag{6}$$

όπου,

$$S = \sqrt[3]{R + \sqrt{R^2 + Q^3}} \qquad T = \sqrt[3]{R - \sqrt{R^2 + Q^3}} \qquad Q = \frac{\Delta_2}{\alpha_3^2} \qquad R = -\frac{\Delta_1}{2\alpha_3^3}$$

Χρειαζόμαστε να ορίσουμε τις ακόλουθες ποσότητες

$$\begin{aligned}
\Delta_2 &= \alpha_2^2 - 3\alpha_3\alpha_1 & \Delta_3 &= \alpha_1^2 - 3\alpha_2\alpha_0 \\
W &= \alpha_2\alpha_1 - 9\alpha_3\alpha_0 & P &= 2\alpha_2\Delta_2 - 3\alpha_3 W
\end{aligned} \tag{7}$$

Οι οποίες είναι είτε αναλλοίωτες του κυβικού πολυωνύμου, είτε στοιχεία του πίνακα Bezout του f και του f'. Η διακρίνουσα του κυβικού πολυωνύμου είναι:

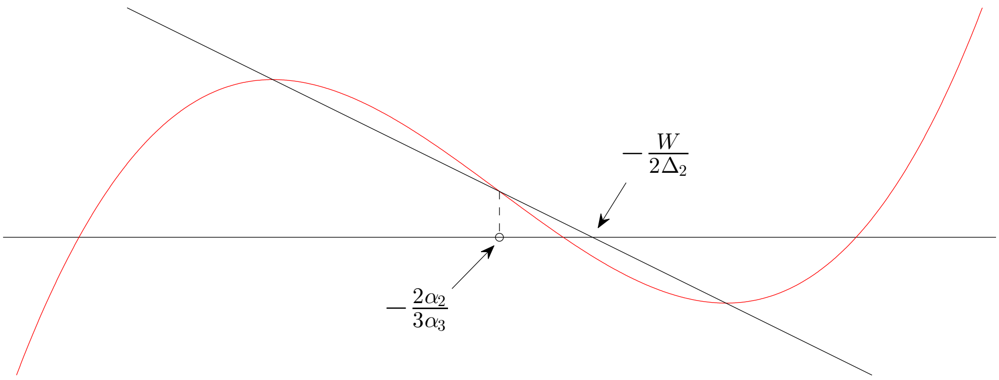
$$\Delta_1 = disc(f) = W^2 - 4\Delta_2\Delta_3 \tag{8}$$

[4] Οι τύποι αυτοί είναι γνωστοί και ως «τύποι του Cardano».

Από τις σχέσεις (6) είναι εμφανές ότι απαιτούνται πράξεις με μιγαδικούς αριθμούς προκειμένου να υπολογίζουμε με ριζικά τις (πραγματικές) ρίζες της (5).

Προκειμένου να υπολογίσουμε το σύστημα διακρινουσών του πολυωνύμου τρίτου βαθμού θεωρούμε την (προσημασμένη) ακολουθία υπο-επιλυουσών του f και του f', $SR(f, f')$. Υπολογίζουμε την ακολουθία συμβολικά και είμαστε βέβαιοι ότι για οποιαδήποτε ανάθεση τιμών στις παραμέτρους, με τον περιορισμό $\alpha_3 \neq 0$, η ακολουθία θα είναι νόμιμη. Η ακολουθία $SR(f, f')$ είναι

$$SR(f, f') = \begin{cases} SR_3(X) = f(X) \\ SR_2(X) = f'(X) \\ SR_1(X) = 2\Delta_2 X + W \\ SR_0(X) = -3\Delta_1 \end{cases} \tag{9}$$



Σχήμα 2: Η ευθεία η οποία ενώνει τα δύο ακρότατα ενός κυβικού πολυωνύμου $f = \alpha_3 X^3 + \alpha_2 X^2 + \alpha_1 X + \alpha_0$ για το οποίο ισχύει $\Delta_1 > 0 \wedge P > 0$.

όπου οι συντελεστές των πολυωνύμων της ακολουθίας παρουσιάζονται με τη βοήθεια των ποσοτήτων (7) και (8).

Αν θεωρήσουμε τη γραφική παράσταση ενός κυβικού πολυωνύμου f, με τρεις πραγματικές ρίζες, τότε ισχύει το παρακάτω (γεωμετρικό) λήμμα :

Λήμμα 4.2.1 Έστω κυβικό πολυώνυμο f όπως στην (5), με τρεις πραγματικές ρίζες. Το μέγιστο, το ελάχιστο και το σημείο καμπής της γραφικής παράστασης του f είναι συνευθειακά. Η ευθεία που ενώνει τα τρία αυτά σημεία ονομάζεται ευθεία απομόνωσης του κυβικού πολυωνύμου (isolating line).

Απόδειξη: Οι τετμημένες των ακροτάτων του f υπολογίζονται αν επιλύσουμε την εξίσωση $f'(X) = 0$. Συνεπώς τα ακρότατα του f είναι τα σημεία $p_1 = (w_1, f(w_1))$ και $p_2 = (w_2, f(w_2))$ όπου

$$w_{1,2} = \frac{-\alpha_2 \pm \sqrt{\Delta_2}}{3\alpha_3}$$

Θεωρούμε την ευθεία που διέρχεται από τα p_1 και p_2, η εξίσωση της οποίας είναι

$$Y = kX + l, \text{ όπου } k = -\frac{2\Delta_2}{\alpha_3} \text{ και } l = -\frac{W}{\alpha_3} \tag{10}$$

Η τετμημένη του σημείου καμπής της f προκύπτει από τη λύση της $f''(X) = 3\alpha_3 X + \alpha_2 = 0$ και άρα το σημείο καμπής είναι το $(-\frac{2\alpha_2}{3\alpha_3}, f(-\frac{2\alpha_2}{3\alpha_3}))$. Με αντικατάσταση αποδεικνύουμε ότι η ευθεία απομόνωσης διέρχεται από το σημείο καμπής και η απόδειξη του λήμματος ολοκληρώνεται. □

Στο Σχ.2 παρουσιάζεται η ευθεία απομόνωσης ενός κυβικού πολυωνύμου με τρεις πραγματικές ρίζες. Μπορούμε τώρα να παρουσιάσουμε το σύστημα διακρινουσών και απομόνωσης.

Πρόταση 4.2.2 Το IDS του κυβικού πολυωνύμου (5) είναι

(1)	$\Delta_1 < 0 \wedge P = 0$	$\{1,1,1\}$	$\gamma_1 \cong (h, (-\infty, -\frac{\alpha_2}{3\alpha_3}))$ $\gamma_2 = -\frac{\alpha_2}{3\alpha_3}$ $\gamma_3 \cong (h, (-\frac{\alpha_2}{3\alpha_3}, +\infty))$
(2)	$\Delta_1 < 0 \wedge P < 0$	$\{1,1,1\}$	$\gamma_1 \cong (f, (-\infty, -\frac{W}{2\Delta_2}))$ $\gamma_2 \cong (f, (-\frac{W}{2\Delta_2}, -\frac{\alpha_2}{3\alpha_3}))$ $\gamma_3 \cong (f, (-\frac{\alpha_2}{3\alpha_3}, +\infty))$
(3)	$\Delta_1 < 0 \wedge P > 0$	$\{1,1,1\}$	$\gamma_1 \cong (f, (-\infty, -\frac{\alpha_2}{3\alpha_3}))$ $\gamma_2 \cong (f, (-\frac{\alpha_2}{3\alpha_3}, -\frac{W}{2\Delta_2}))$ $\gamma_3 \cong (f, (-\frac{W}{2\Delta_2}, +\infty))$
(4)	$\Delta_1 > 0 \wedge \alpha_0 = 0$	$\{1\}$	$\gamma_1 = 0$
(5)	$\Delta_1 > 0 \wedge \alpha_0 < 0$	$\{1\}$	$\gamma_1 \cong (f, (0, +\infty))$
(6)	$\Delta_1 > 0 \wedge \alpha_0 > 0$	$\{1\}$	$\gamma_1 \cong (f, (-\infty, 0))$
(7)	$\Delta_1 = 0 \wedge \Delta_2 \neq 0$	$\{1,2\}$	$\gamma_1 = \min\{ \frac{-W}{2\Delta_2}, \frac{-\alpha_2\Delta_2+\alpha_3 W}{\alpha_3\Delta_2} \}$ $\gamma_2 = \max\{ \frac{-W}{2\Delta_2}, \frac{-\alpha_2\Delta_2+\alpha_3 W}{\alpha_3\Delta_2} \}$
(8)	$\Delta_1 = 0 \wedge \Delta_2 = 0$	$\{3\}$	$\gamma_1 = -\frac{\alpha_2}{3\alpha_3}$

όπου $h = 9\alpha_3^2 X_2 + 6\alpha_2\alpha_3 X + 9\alpha_3\alpha_1 - 2\alpha_2^2$.

Θα αναφέρουμε ένα Πόρισμα και ένα Θεώρημα που θα μας βοηθήσουν στην απόδειξη μας.
Πόρισμα 4.2.2.1 Έστω A $\in \mathbb{R}[X]$. Αν $\alpha < b, \alpha, b \in \mathbb{R} \cup \{\pm\infty\}$ δεν είναι ρίζες του A τότε

$$\mathrm{VAR}(SR(A, A'; \alpha)) - \mathrm{VAR}(SR(A, A'; b)) = \#\{\gamma | A(\gamma) = 0 \wedge \alpha < \gamma < b\}$$

Ορίζουμε ως VAR (variation) τον αριθμό των εναλλαγών των προσήμων από την ακολουθία των υπο-επιλυουσών. Με τον όρο υπο-επιλύουσες (ή υπο-απαλοίφουσες) ορίζονται ως ορίζουσες κάποιας παραλλαγής του πίνακα Sylvester. Επίσης, με την έκφραση $\mathrm{VAR}(SR(A, A'; \alpha))$ εννοούμε ότι υπολογίζουμε το variation των υπο-επιλυουσών του πολυωνύμου A και του A' στο σημείο α.

Θεώρημα 4.2.2.2 Το πολυώνυμο $SR_0(A,B)$ είναι (εκτός ίσως εκτός από το πρόσημο του) η επιλύουσα των A και B, δηλαδή $SR_0(A,B) = \pm res(A,B)$. Από την ακολουθία $SR(A,B)$ μπορούμε να υπολογίσουμε τον ΜΚΔ των A και B. Πιο συγκεκριμένα ισχύει ότι:

$$SR_k(A,B) = gcd(A,B) \Leftrightarrow psc_0(A,B) = ... = psc_{k-1}(A,B) = 0 \text{ ή } psc_k(A,B) \neq 0$$

και deg(gcd(A,B)) $= \kappa$.[5]

Απόδειξη: Θα δείξουμε τι συμβαίνει σε κάθε περίπτωση του ΙΔΣ.

Η ποσότητα VAR$(SR(f;-\infty))$ − VAR$(SR(f;+\infty))$ σύμφωνα με το (Πορ.4.2.2.1) μας δίνει το πλήθος των πραγματικών ριζών της f. Επίσης το πρόσημο της αποτίμησης του SR_j στο $\pm\infty$ υπολογίζεται ως $\text{sign}(\lim_{X\to\pm\infty} SR_j(X))$.

Διακρίνουμε τις παρακάτω περιπτώσεις :

- Το f είναι τύπου (1) ή (2) ή (3). Ισχύει ότι

$$\text{V}^- = \text{VAR}(SR(f;-\infty)) = \text{VAR}(-,+,-,+) = 3$$
$$\text{V}^+ = \text{VAR}(SR(f;+\infty)) = \text{VAR}(+,+,+,+) = 0$$

και συνεπώς $\text{V}^- - \text{V}^+ = 3 - 0 = 3$, οπότε το f έχει 3 πραγματικές ρίζες (Πορ. 2.27) που είναι ο μέγιστος αριθμός ριζών που μπορεί να έχει και άρα είναι απλές, δηλαδή $\{1,1,1\}$.

Παρατηρούμε ότι για την τετμημένη του σημείου καμπής ισχύει ότι

$$\text{sign}(f(-\frac{2\alpha_2}{3\alpha_3})) = \text{sign}(2\alpha_2\Delta_2 - 3\alpha_3 W) = \text{sign}(P)$$

Αν το f είναι τύπου (1) τότε $\text{sign}(f(-\frac{2\alpha_2}{3\alpha_3})) = \text{sign}(P) = 0$ και γνωρίζουμε τη μία ρίζα ακριβώς, συνεπώς $f = (X + \frac{2\alpha_2}{3\alpha_3})h$. Αναπαριστούμε τις άλλες δύο ρίζες χρησιμοποιώντας το δευτεροβάθμιο πολυώνυμο h και το $-\frac{2\alpha_2}{3\alpha_3}$ ως σημείο διαχωρισμού τους.

Αν το f είναι τύπου (2) ή (3) τότε θεωρούμε την ευθεία απομόνωσης (Λήμμα 4.2.1). Αντικαθιστώντας στην (10) $U = 0$, παρατηρούμε ότι η ευθεία απομόνωσης τέμνει τον άξονα X στο σημείο $(-\frac{W}{2\Delta_2}, 0)$. Επειδή το f έχει μόνο πραγματικές ρίζες ισχύει $\Delta_2 > 0$. Τετμημένη του σημείου καμπής, $-\frac{2\alpha_2}{3\alpha_3}$, βρίσκεται ανάμεσα στις ρίζες καθώς αποτελεί τον αριθμητικό τους μέσο. Υποθέτουμε ότι $\text{sign}(f(-\frac{2\alpha_2}{3\alpha_3})) = P > 0$. Τότε υπάρχει μία ρίζα του f αριστερά του $-\frac{2\alpha_2}{3\alpha_3}$ και δύο δεξιά. Παρατηρούμε ότι

[5] psc_k είναι : ο κ-οστός πρωτεύων συντελεστής της υπο-επιλύουσας των A και B, δηλαδή

$$psc_k(A,B) = det_k^{(k)}$$

$$-\frac{\alpha_2}{3\alpha_3} < -\frac{W}{2\Delta_2} \Longleftrightarrow P > 0$$

Λόγω του γεγονότος ότι η f είναι κυρτή δεξιά από το σημείο καμπής, η ευθεία απομόνωσης κόβει τον άξονα X μετά τη δεύτερη ρίζα και πριν την τρίτη. Κατά συνέπεια οι $-\frac{2\alpha_2}{3\alpha_3}$ και $-\frac{W}{2\Delta_2}$ διαχωρίζουν τις ρίζες του f . Η περίπτωση $P < 0$ είναι όμοια και ισχύει $-\frac{\alpha_2}{3\alpha_3} > -\frac{W}{2\Delta_2}$.

Παρατηρούμε ότι το $-\frac{W}{2\Delta_2}$ δεν μπορεί να είναι ρίζα της f καθώς ισχύει

$$\mathrm{sign}(f(-\frac{W}{2\Delta_2})) = \mathrm{sign}(P\Delta_1) \neq 0$$

- Η f είναι τύπου (4) ή (5) ή (6). Ισχύει ότι

$$\mathrm{V}^- = \mathrm{VAR}(SR(f;-\infty)) = \mathrm{VAR}(-,+,\pm \vee 0,-) = 2$$
$$\mathrm{V}^+ = \mathrm{VAR}(SR(f;+\infty)) = \mathrm{VAR}(+,+,\mp \vee 0,-) = 1$$

και συνεπώς $\mathrm{V}^- - \mathrm{V}^+ = 2-1 = 1$, οπότε το f έχει 1 πραγματική ρίζα (Πορ.4.2.2.1). Εφόσον δεν μηδενίζεται κάποιο $psc_i, i \in \{0,1\}$ η ρίζα έχει πολλαπλότητα 1 (Θεωρ.4.2.2.2). Η ρίζα είναι αρνητική, θετική ή μηδέν ανάλογα με το πρόσημο του α_0.

- Η f είναι τύπου (7). Ισχύει ότι

$$\mathrm{V}^- = \mathrm{VAR}(SR(f;-\infty)) = \mathrm{VAR}(-,+,-,0) = 2$$
$$\mathrm{V}^+ = \mathrm{VAR}(SR(f;+\infty)) = \mathrm{VAR}(+,+,+,0) = 0$$

και συνεπώς $\mathrm{V}^- - \mathrm{V}^+ = 2 - 0 = 2$, οπότε το f έχει 2 πραγματικές ρίζες (Πορ.4.2.2.1). Παρατηρούμε ότι από το Θεώρημα 5.2 συνάγουμε ότι $\Delta_2 \geq 0$.

Από τις ιδιότητες των υπο-επιλυουσών (Θεωρ.4.2.2.2) εφόσον $SR_0 = psc_0 = 0$ τότε $\gcd(f, f') = SR_1$. Οπότε η μία ρίζα, αυτή με πολλαπλότητα 2, είναι η λύση της εξίσωσης $SR_1 = 0$, δηλαδή η $-\frac{W}{2\Delta_2}$. Η απλή ρίζα είναι η $\frac{-\alpha_2\Delta_2+\alpha_3 W}{\alpha_3\Delta_2}$.

- Η f τύπου (8). Ισχύει ότι

$$\mathrm{V}^- = \mathrm{VAR}(SR(f;-\infty)) = \mathrm{VAR}(-,+,0,0) = 1$$

$$\mathrm{V}^+ = \mathrm{VAR}(SR(f;+\infty)) = \mathrm{VAR}(+,+,0,0) = 0$$

και συνεπώς $\mathrm{V}^- - \mathrm{V}^+ = 1 - 0 = 1$, οπότε το f έχει 1 πραγματική ρίζα (Πορ.4.2.2.1).

Από τις ιδιότητες των υπο-επιλυουσών (Θεωρ.4.2.2.2) εφόσον $SR_0 = psc_0 = psc_1 = 0$ τότε $gcd(f, f') = SR_2$. Η ρίζα έχει πολλαπλότητα 3 και υπολογίζεται ως διπλή ρίζα του $SR_2 = f'$.

Εφόσον έχουμε θεωρήσει όλους τους δυνατούς συνδυασμούς προσήμων των psc, η πρόταση αποδείχτηκε. $\square$

Στη συνέχεια θα δείξουμε πως προκύπτει το h.
Έχουμε το κυβικό πολυώνυμο $f \in \mathbb{Z}[X]$, όπου $\alpha_3 > 0$,

$$f(X) = \alpha_3 X^3 + \alpha_2 X^2 + \alpha_1 X + \alpha_0$$

και το

$$g(X) = 3\alpha_3 X + \alpha_2$$

το οποίο προκύπτει από τη ρίζα γ_2 της περίπτωσης (1) του προηγούμενου πίνακα.
Άρα πρέπει $h(X)g(X) = f(X)$, όπου $h(X) = \beta_2 X^2 + \beta_1 X + \beta_0$. Αυτό δίνει το σύστημα:

$$\begin{aligned} 3\beta_2\alpha_3 &= \alpha_3 \\ \beta_2\alpha_2 + 3\beta_1\alpha_3 &= \alpha_2 \\ \beta_1\alpha_2 + 3\alpha_3\beta_0 &= \alpha_1 \\ \beta_0\alpha_2 &= \alpha_1 \end{aligned}$$

Χρησιμοποιώντας τις παραπάνω σχέσεις βρίσκω:

$$\begin{aligned} \beta_2 &= \frac{1}{3} \\ \beta_1 &= \frac{2\alpha_2}{9\alpha_3} \\ \beta_0 &= \frac{9\alpha_3\alpha_1 - 2\alpha_2^2}{27\alpha_3^2} \end{aligned}$$

και για να μην έχω παρανομαστές πολλαπλασιάζω με το $27\alpha_3^2$ οπότε έχω το πολυώνυμο:

$$\frac{1}{3}27\alpha_3^2 X^2 + \frac{2\alpha_2}{9\alpha_3}27\alpha_3^2 X + \frac{9\alpha_3\alpha_1 - 2\alpha_2^2}{27\alpha_3^2} = 9\alpha_3^2 X^2 + 6\alpha_2\alpha_3 X + 9\alpha_3\alpha_1 - 2\alpha_2^2 = h$$

5 Σύγκριση αλγεβρικών αριθμών

Ορίζουμε αναπαράσταση με διαστήματα απομόνωσης ενός πραγματικού αλγεβρικού αριθμού, $\alpha \in R_{alg}$, συμβολίζεται με $\alpha \cong (\mathrm{A}(x), J)$, όπου $\mathrm{A} \in Z[x]$ είναι χωρίς τετράγωνα, $\mathrm{A}(\alpha) = 0$, $\alpha \in J$, $J = (\alpha, \beta)$, $\alpha, \beta \in Q$ και το A δεν έχει άλλη πραγματική ρίζα στο $[\alpha, \beta]$.
Προκειμένου να συγκρίνουμε δύο αλγεβρικούς αριθμούς βαθμού ≤ 3 θα χρησιμοποιήσουμε τον αλγόριθμο $COMPARE$ (παρουσιάζεται παρακάτω). Ωστόσο, θα υπολογίσουμε τις ακολουθίες υποεπιλυουσών, θεωρώντας τους συντελεστές των πολυωνύμων ως παραμέτρους, προκειμένου αφενός για να αποδείξουμε ότι η σύγκριση απαιτεί σταθερό πλήθος αριθμητικών πράξεων και αφετέρου για να επιταχύνουμε τον υπολογισμό στην πράξη.

Αλγόριθμος $COMPARE$

$Input : \gamma_1 \cong (A_1, J_1 = (\alpha_1, \beta_1)), \gamma_2 \cong (A_2, J_2 = (\alpha_2, \beta_2))$
$Output$: Σύγκριση των γ_1 και γ_2

1. $J \leftarrow J_1 \cap J_2 = (c, d)$
2. $if\ J = \emptyset\ then$
3. $\quad if\ J_1 < J_2\ then\ return - 1$
4. $\quad else\ return\ 1$
5. $\quad endif$
6. $endif$
7. $if\ \gamma_1 \notin J\ then$
8. $\quad if\ J_1 \setminus J < J\ then\ return\ - 1$
9. $\quad else\ return\ 1$
10. $\quad endif$
11. $endif$
12. $if\ \gamma_2 \notin J\ then$
13. $\quad if\ J_2 \setminus J < J\ then\ return\ 1$
14. $\quad else\ return\ - 1$
15. $\quad endif$
16. $endif$
17. $return\ sign\_at(A_2, \gamma_1) sign(A_2'(\gamma_2))$

Ο συμβολισμός $I_1 < I_2$ έχει την εξής έννοια: δεδομένων δύο ανοικτών διαστημάτων $I_1 = (\ell_1, h_1)$ και $I_2 = (\ell_2, h_2)$, θα λέμε ότι $I_1 < I_2$ αν και μόνο $h_1 \leq \ell_2$.

5.1 Ο βαθμός 2

Έστω $\alpha \cong (f_1 = \alpha_2 X^2 - 2\alpha_1 X + \alpha_0, J_1)$ και $\beta \cong (f_2 = \beta_2 X^2 - 2\beta_1 X + \beta_0, J_2)$. Υπολογίζουμε την ακολουθία $SR(f_1, f_2)$ η οποία είναι

$$
\begin{aligned}
SR_3 &= f_1 \\
SR_2 &= f_2 \\
SR_1 &= 2JX - G \\
SR_0 &= -\alpha_2(G^2 - 4JJ_1)
\end{aligned}
$$

όπου $J = \alpha_2\beta_1 - \beta_2\alpha_1$, $J_1 = \alpha_1\beta_0 - \beta_1\alpha_0$ και $G = \alpha_2\beta_0 - \beta_2\alpha_0$. Ανάλογα με τον τύπο των α και β η ακολουθία πρέπει να αποτιμηθεί σε δύο από τους αριθμούς $\{\pm\infty, -\frac{\alpha_1}{\alpha_2}, -\frac{\beta_1}{\beta_2}\}$ και στη συνέχεια να μετρήσουμε τις εναλλαγές προσήμων. Η προσέγγιση που παρουσιάζουμε ομοιάζει με αυτή των Emiris and Karavelas [31, 6], αλλά χρησιμοποιούμε μόνο μία ακολουθία υπολοίπων ενώ η προσέγγιση των Emiris and Karavelas [31, 6] χρησιμοποιεί πολλές αλλά δεν υποθέτει ότι οι αλγεβρικοί αριθμοί είναι σε αναπαράσταση με διαστήματα απομόνωσης. Επίσης οι Devillers et al. [32, 33] αναδεικνύουν τη γεωμετρική πληροφορία πίσω από τη σύγκριση αλγεβρικών αριθμών βαθμού 2 και παρουσιάζουν έναν (βέλτιστο) αλγόριθμο για τη σύγκριση των δύο μικρότερων ριζών.

Ο μέγιστος αλγεβρικός βαθμός που απαιτείται για τη σύγκριση δύο πραγματικών αλγεβρικών αριθμών βαθμού 2 παρουσιάζεται και με τον υπολογισμό της επιλύουσας, SR_0 και είναι 4. Καθώς η επιλύουσα είναι η βέλτιστη συνθήκη για την ύπαρξη κοινών ριζών και ο αλγεβρικός βαθμός της είναι βέλτιστος, συνάγουμε ότι, όσο αφορά τον αλγεβρικό βαθμό, ο αλγόριθμος σύγκρισης είναι επίσης βέλτιστος.

5.2 Ο βαθμός 3

Έστω $\alpha \cong (f_1, J_1)$ και $\beta \cong (f_2, J_2)$ όπου $\deg(f_1) \leq 3$ και $\deg(f_2) \leq 3$. Προκειμένου να συγκρίνουμε τους α και β, σύμφωνα με τον αλγόριθμο $COMPARE$, πρέπει να υπολογίσουμε ένα κοινό διάστημα, να υπολογίσουμε την $SR(f_1, f_2)$ και να την αποτιμήσουμε στα άκρα του διαστήματος.
Παρουσιάζουμε αναλυτικά την ακολουθία υπο-επιλυουσών όταν και τα δύο πολυώνυμα είναι τρίτου βαθμού.
Έστω δύο τριτοβάθμια πολυώνυμα:

$$f_1 = \alpha_3 X^3 - 3\alpha_2 X^2 + 3\alpha_1 X - \alpha_0$$

$$f_2 = \beta_3 X^3 - 3\beta_2 X^2 + 3\beta_1 X - \beta_0$$

όπου $\alpha_3, \beta_3 > 0$. Η ακολουθία υπο-επιλυουσών $SR(f_1, f_2)$ είναι:

$$
\begin{aligned}
SR_4 &= f_1 \\
SR_3 &= f_2 \\
SR_2 &= -3JX^2 + 3GX - M \\
SR_1 &= [J(P + 12J_1) - 3G^2]X - (GM - 3JM_1) = Z_1X + Z_2 \\
SR_0 &= J^2(P^3 - 27Q) = J^2[M(M^2 - 9JM_2) - 9M_1Z_2 - 9M_2Z_1]
\end{aligned}
$$

όπου,

$\Delta_{11} = W_1^2 - 2\Delta_{12}\Delta_{13}$
$\Delta_{21} = W_2^2 - 2\Delta_{22}\Delta_{23}$
$J = \alpha_3\beta_2 - \beta_3\alpha_2$
$J_1 = \alpha_2\beta_1 - \beta_2\alpha_1$
$M = \alpha_3\beta_0 - \beta_3\alpha_0$
$P = M - 3J$
$Z_1 = J(M + 9J_1) - 3G^2$
$Z_2 = GM - 3JM_1$
$G = \alpha_3\beta_1 - \beta_3\alpha_1$
$M_1 = \alpha_2\beta_0 - \beta_2\alpha_0$
$M_2 = \alpha_1\beta_0 - \beta_1\alpha_0$
$W_1 = \alpha_3\alpha_0 - \alpha_1\alpha_2$
$W_2 = \beta_3\beta_0 - \beta_1\beta_2$

6 Ο υπολογισμός προσήμου

Με τον όρο υπολογισμό προσήμου αναφερόμαστε στον υπολογισμό του προσήμου της αποτίμησης ενός πολυωνύμου πάνω σε έναν αλγεβρικό αριθμό. Στο παρόν κείμενο ο αλγεβρικός αριθμός είναι βαθμού ≤ 3. Για τον υπολογισμό χρησιμοποιούμε τον αλγόριθμο $SIGN\_AT$, ο οποίος, όπως και ο αλγόριθμος $COMPARE$, χρειάζεται την αποτίμηση μιας ακολουθίας υπο-απαλλοιφουσών πάνω σε δύο ρητά σημεία.

Ωστόσο το IDS μας επιτρέπει, εκτός από τον υπολογισμό του προσήμου, να αντιμετωπίσουμε ένα ακόμα πιο γενικό πρόβλημα, το πρόβλημα της απαλοιφής ποσοδεικτών (quantifier elimination). Στα προβλήματα απαλοιφής ποσοδεικτών η είσοδος είναι φόρμουλες του τύπου $\exists x(P)$, όπου το P είναι ένας συνδυασμός από πολυωνυμικές εξισώσεις και ανισώσεις στο x και άλλες ελεύθερες (free variables) μεταβλητές $y_1, ..., y_n$. Για κάποιες καθορισμένες πραγματικές τιμές των $y_1, ..., y_n$ το σύνολο S των πραγματικών τιμών του x που ικανοποιούν την P είναι η ένωση ξένων μεταξύ τους διαστημάτων. Τα άκρα αυτών των διαστημάτων είναι οι πραγματικές ρίζες κάποιων πολυωνύμων του P και των $\pm\infty$. Συνεπώς προκειμένου να ελέγξουμε εάν το S είναι κενό ή όχι αρκεί να ελέγξουμε αν μία από αυτές τις ρίζες, ή το $\pm\infty$ ανήκει στο S. Αυτοί οι υπολογισμοί μπορούν να εκφραστούν ως πεπερασμένες συζεύξεις και διαζεύξεις από φόρμουλες $P[\alpha/x]$ όπου οι εκφράσεις α , που εξαρτώνται από τα $y_1, ..., y_n$ έχουν αντικαταστήσει το x στο P. Η μέθοδος αυτή είναι γνωστή ως εικονική αντικατάσταση (virtual substitution) [19, 20, 21, 23, 24].

Χρησιμοποιώντας τις φόρμουλες χωρίς ποσοδείκτες μπορούμε να παράγουμε αλγορίθμους για την αποτίμηση προσήμου οι οποίοι δεν περιέχουν βρόγχους (straight-line programs). Καθώς το πρόβλημα της απαλοιφής ποσοδεικτών είναι πιο γενικό.

6.1 Ο βαθμός 2 και 3

Ο βασικός υπολογισμός είναι ο υπολογισμός του προσήμου της αποτίμησης ενός πολυωνύμου πάνω σε ένα αλγεβρικό αριθμό. Θα παρουσιάσουμε παρακάτω πως μπορούμε να υπολογίσουμε την ακολουθία υπο-επιλυουσών για δύο πολυώνυμα, ένα 3ου βαθμού και ένα 2ου.

Έστω $\gamma \cong (f, (\alpha, \beta))$, όπου $f = \alpha_3 X^3 + \alpha_2 X^2 + \alpha_1 X + \alpha_0$, $g = \beta_2 X^2 + \beta_1 X + \beta_0$ και $\alpha_3 \beta_2 \neq 0$. Θέλουμε να υπολογίσουμε το sign[g(γ)].

Υπολογίζουμε την προσημασμένη ακολουθία υποαπαλοιφουσών των f και g, η οποία είναι

$$
\begin{aligned}
SR_3 &= f \\
SR_2 &= sr_{22} X^2 + sr_{21} X + sr_{20} \\
SR_1 &= sr_{11} X + sr_{10} \\
SR_0 &= sr_0
\end{aligned}
$$

όπου,

$$
\begin{aligned}
&sr_{22} = K_1 \\
&sr_{21} = K_2 \\
&sr_{20} = K_3 \\
&sr_{11} = -K_1 K_7 + K_2 K_6 \\
&sr_{10} = -K_1 K_4 + K_3 K_6 \\
&sr_0 = -K_1 K_7 K_5 + K_1 K_4 K_8 - K_6 K_3 K_8 + K_6 K_2 K_5 - K_7 K_2 K_4 + K_3 K_7^2
\end{aligned}
$$

όπου, $K_i, 1 \leq i \leq 8$ είναι τα στοιχεία του πίνακα Bezout των f και g, δηλαδή

$$
\begin{aligned}
&K_1 = \beta_2 \\
&K_2 = \beta_1 \\
&K_3 = \beta_0 \\
&K_4 = -\beta_2 \alpha_0 \\
&K_5 = -\alpha_0 \beta_1 \\
&K_6 = \alpha_3 \beta_1 - \beta_2 \alpha_2 \\
&K_7 = \beta_0 \alpha_3 - \beta_2 \alpha_1 \\
&K_8 = \beta_0 \alpha_2 - \beta_2 \alpha_0 - \alpha_1 \beta_1
\end{aligned}
$$

Ας θεωρήσουμε τον υπολογισμό

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & sr_{22} & sr_{21} & sr_{20} \\ 0 & 0 & sr_{11} & sr_{10} \\ 0 & 0 & 0 & sr_0 \end{pmatrix} \begin{pmatrix} \alpha^3 & \beta^3 \\ \alpha^2 & \beta^2 \\ \alpha & \beta \\ 1 & 1 \end{pmatrix} = \begin{pmatrix} SR_3(\alpha) & SR_3(\beta) \\ SR_2(\alpha) & SR_2(\beta) \\ SR_1(\alpha) & SR_1(\beta) \\ SR_0 & SR_0 \end{pmatrix} = (M_1 \ M_2) = M$$

ο οποίος περιγράφει την αποτίμηση της ακολουθίας $SR(f,g)$ πάνω στα α και β. Η πρώτη στήλη του M περιέχει την $SR(f,g;\alpha)$ και η δεύτερη την $SR(f,g;\beta)$. Αν μετρήσουμε τις εναλλαγές προσήμων στην πρώτη στήλη ($\mathrm{VAR}(M_1)$) και στη δεύτερη ($\mathrm{VAR}(M_2)$) τότε $\mathrm{sign}(\mathrm{g}(\gamma)) = (\mathrm{VAR}(M_1)) - (\mathrm{VAR}(M_2))$ σύμφωνα με τον αλγόριθμο $COMPARE$. Όλοι οι δυνατοί συνδυασμοί προσήμων είναι $\leq 2 \times 3^5 = 486$, αν και δεν είναι όλοι εφικτοί.

7 Κώδικας

7.1 Η βιβλιοθήκη Pol.h

```c
#ifndef POL_H
#define POL_H 1

// h vivliothiki Pol.h periexei ta structs pou xreiazomaste
// kathws episis kai ton orismo twn sunartisewn
// pou mas voithane gia tous upologismous analoga tin periptwsi
// pou vriskomaste(1ou vathmmou, 2ou h 3ou)

#include <stdio.h>
#include <math.h>
#include <stdlib.h>
#include <assert.h>
#include <stdbool.h>

// Vector functions prototypes

// edw orizoume apo ti apoteleitai to polywnimo mas.
// o pinakas a, periexei tous suntelestes tou,
// to deg dilwnei to vathmo tou polywnimou
typedef struct Polywnimo
{
        double a[4];
        int deg;
}Polywnimo;

// edw exoume to polywnimo mas p, low einai to katw akro
// tou diastimatos pou periexetai i riza tou polywnimou
// kai to high einai to anw akro, an low = high tote to
// is_exact pairnei tin timi 1, alliws tin timi 0
typedef struct Root
{
        struct Polywnimo *p;
        double low, high;
        bool is_exact;
}Root;

// edw exoume sto nroots to plithos twn rizwn pou exei to polywnimo
// mas, ton pinaka roots pou einai tupou Root
```

```
// kai tin pollaplotita kathe rizas mas ston pinaka multiplicity
typedef struct Result
{
        int nroots;
        struct Root *roots[3];
        int multiplicity[3];
}Result;

// diavazei to polywnimo apo to arxeio Ptuxiaki.txt
struct Polywnimo* read_Polywnimo( FILE *, Polywnimo * ) ;

// epilegei tis periptwseis gia ti lusi tou polywnimou
struct Result* Solve(Polywnimo *);

// auti i sunartisi ektupwnei to polywnimo mas
void printPol(Polywnimo *);

// auti i sunartisi tha ektupwnei ta apotelesmata gia tis
// rizes tou polywnimou mas, dld olo to Result
void print_Result2(Result *res, Polywnimo *f);

int Sugkrisi_rizwn( Root *, Root * ) ;
// vriskw to f(k) epistrefw to apotelesma, edw to xrisimopoiw gia to f'(k)
double timh_polywnimou( Polywnimo *f, double x );

#endif /* POL_H */
```

7.1.1 Επεξήγηση του Pol.h

Στη δομή “Polywnimo” ορίζουμε από τι αποτελείται το πολυώνυμο μας. Αποθηκεύουμε σε ένα πίνακα από doubles τους συντελεστές του και σε μία ακέραια μεταβλητή το βαθμό του πολυωνύμου μας.

Η δομή “Root” αποτελείται απο τη δομή “Polywnimo”, αποθηκεύει σε δύο μεταβλητές κινητής υποδιαστολής τα δύο άκρα του διαστήματος στο οποίο βρίσκεται η ρίζα μας και αν είναι ίσα σημαίνει ότι γνωρίζουμε τη ρίζα ακριβώς και έτσι η μεταβλητή is_exact γίνεται true, αλλιώς γίνεται false.

Η δομή “Result” αποθηκεύει σε μία ακέραια μεταβλητή τον αριθμό των ριζών του πολυωνύμου μας. Αποτελείται από τη δομή “Root” και από έναν πίνακα ακεραίων multiplicity, ο οποίος αποθηκεύει την πολλαπλότητα της κάθε ρίζα της συνάρτησης μας.

7.2 Το αρχείο Pol.c

```
// ulopoiisi twn sunartisewn pou vriskontai mesa sti vivliothiki Pol.h

// Include the header file
#include "Pol.h"
#include "Pol_internal.h"

// ektupwnoume to polywnimo pou tou stelnoume
void printPol( Polywnimo *f )
{
 int i;

 if( f->deg == 0 )        //an ine 0ikou va8mou
 {
  printf("%g\n", f->a[0]);
  return;
 }
 printf("%gx^%d", f->a[f->deg], f->deg);
 for( i=f-> deg-1;i>0;i-- )
 {
  if( f->a[i] != 0 )
  {
   printf(" ");
   if( f->a[i] >= 0 )
   {
        printf("+");
   }
   printf("%gx^%d", f->a[i], i);
  }
 }
 if( f->a[0] != 0 )
 {
 printf(" ");
 if(f->a[0] >= 0)
 {
  printf("+");
 }
 printf("%g", f->a[0]);
 }

 return;
}
```

```
// vriskw to f(k) epistrefw to apotelesma, edw to xrisimopoiw gia to f'(k)
double timh_polywnimou( Polywnimo *f, double x )
{
 int i;
 double value = f->a[f->deg];
 for ( i = f->deg - 1; i >= 0; --i )
 {
  value *= x;
  value += f->a[i] ;
 }
 return value;
}

struct Result* Solve(Polywnimo *f)
{
 // analoga ti vathmou einai to polywnimo exoume tis antistoixes periptwseis
 if( f->deg == 3 )
 {
  return(periptwsis_3ou_vathmou(f));
 }
 else if( f->deg == 2 )
 {
  return(periptwsis_2ou_vathmou(f));
 }
 else if( f->deg == 1 )
 {
  return(periptwsis_1ou_vathmou(f));
 }
 else if( f->deg == 0 )
 {
  return(periptwsis_0ou_vathmou(f));
 }
}


int Sugkrisi_rizwn( Root *r_f, Root *r_g )
{

 //an kseroume akrivws kai tis 2 rizes twn polywnimwn
 if ( r_f->is_exact == true && r_g->is_exact == true )
```

```
{
//h gia 2 polywnima 1ou vathmou
 if ( r_f->low < r_g->low )
 {
  return -1;
 }
 else if ( r_f->low == r_g->low )
 {
  return 0;
 }
 else if ( r_f->low > r_g->low )
 {
  return 1;
 }
}

// an gnwrizoume ti riza tis f akrivws
if ( r_f->is_exact == true )
{
 if ( r_f->low <= r_g->low )// ..f..[gl...gh]
 {
  return -1;
 }
 else if ( r_f->low >= r_g->high )//[gl...gh]..f..
 {
  return 1;
 }
 else  //  [gl...f...gh]
 {
  int sign_g_rho = Sign(timh_polywnimou(r_g->p, r_f->low));
  if ( sign_g_rho == 0 ) { return 0; }

  int sign_g_low = Sign(timh_polywnimou(r_g->p, r_g->low));
  assert( sign_g_low != 0 );
  return ( sign_g_rho == sign_g_low ) ? -1 : 1;
 }
}

// an gnwrizoume ti riza tis g akrivws
if ( r_g->is_exact == true )
{
```

```
  if ( r_g->low <= r_f->low )//..g..[fl...fh]
  {
   return 1;
  }
  else if ( r_g->low >= r_f->high )//[fl...fh]..g..
  {
   return -1;
  }
  else                                          //  [fl...g...fh]
  {
   int sign_f_rho = Sign(timh_polywnimou(r_f->p, r_g->low));
   if ( sign_f_rho == 0 ) { return 0; }

   int sign_f_low = Sign(timh_polywnimou(r_f->p, r_f->low));
   assert( sign_f_low != 0 );
   return ( sign_f_rho == sign_f_low ) ? 1 : -1;
  }
}

// apo edw kai sto e3hs kai apo ta polywnyma einai 2ou h
// 3ou va8mou kai malista DEN 3erw tis rizes tous akrivws
// an iksera 8a eixa hdh apanthsei (kwdikas parapanw)
if( r_f->low < r_g->low )
{
  if( r_f->high < r_g->low )// { f }....[ g ]
  {
   return -1;
  }
  else if( r_f->high < r_g->high )//{f..[g..f}..g]
  {

   int f_lowg = Sign( timh_polywnimou( r_f->p, r_g->low ) ) ;
   if ( f_lowg == 0 ) { return -1; }

   int g_highf = Sign( timh_polywnimou( r_g->p, r_f->high ) ) ;
   if ( g_highf == 0 ) { return -1; }

   int f_lowf = Sign( timh_polywnimou( r_f->p, r_f->low ) ) ;
   int s = Sign( f_lowf * f_lowg );
   assert( s != 0 );
```

```
  if( s < 0 ) { return -1; }

  int g_highg = Sign( timh_polywnimou( r_g->p, r_g->high ) ) ;
  s = Sign( g_highg * g_highf );
  assert( s != 0 );
  if( s < 0 ) { return -1; }

  int D, dD ;

  D = dialegw_periptwsi_gia_ta_SR ( r_f, r_g ) ;

  dD = f_lowg ;

  if( D == 0 )  { return 0; }
  if( D == dD ) { return 1; }
  if( D != dD ) { return -1; }
}
else  // {f..[ g ]..f}
{
  int f_lowg = Sign( timh_polywnimou( r_f->p, r_g->low ) ) ;
  if ( f_lowg == 0 ) { return -1; }

  int f_highg = Sign( timh_polywnimou( r_f->p, r_g->high ) ) ;
  if ( f_highg == 0 ) { return 1; }

  int f_lowf = Sign( timh_polywnimou( r_f->p, r_f->low ) ) ;
  int s = Sign( f_lowf * f_lowg );
  assert( s != 0 );
  if( s < 0 ) { return -1; }

  int f_highf = Sign( timh_polywnimou( r_f->p, r_f->high ) ) ;
  s = Sign( f_highg * f_highf );
  assert( s != 0 );
  if( s < 0 ) { return 1; }

  int D, dD ;

  D = dialegw_periptwsi_gia_ta_SR ( r_f, r_g ) ;

  dD = f_lowg ;
```

```
   if( D == 0 )  { return 0; }
   if( D == dD ) { return 1; }
   if( D != dD ) { return -1; }

  }
 }
 else if( r_g->low < r_f->low )
 {
  if( r_g->high < r_f->low )//[ g ]...{ f }
  {
   return 1;
  }
  else if( r_g->high < r_f->high )//[g..{f..g]..f}
  {
   int g_lowf = Sign( timh_polywnimou( r_g->p, r_f->low ) ) ;
   if ( g_lowf == 0 ) { return 1; }

   int f_highg = Sign( timh_polywnimou( r_f->p, r_g->high ) ) ;
   if ( f_highg == 0 ) { return 1; }

   int g_lowg = Sign( timh_polywnimou( r_g->p, r_g->low ) ) ;
   int s = Sign( g_lowf * g_lowg );
   assert( s != 0 );
   if( s < 0 ) { return 1; }

   int f_highf = Sign( timh_polywnimou( r_f->p, r_f->high ) ) ;
   s = Sign( f_highf * f_highg );
   assert( s != 0 );
   if( s < 0 ) { return 1; }

   int D, dD ;

   D = dialegw_periptwsi_gia_ta_SR ( r_f, r_g ) ;

   dD = g_lowf ;

   if( D == 0 )  { return 0; }
   if( D == dD ) { return -1; }
   if( D != dD ) { return 1; }

  }
```

```
  else  // [g..{ f }..g]
  {
   int g_lowg = Sign( timh_polywnimou( r_g->p, r_f->low ) ) ;
   if ( g_lowg == 0 ) { return 1; }

   int g_highg = Sign( timh_polywnimou( r_g->p, r_f->high ) ) ;
   if ( g_highg == 0 ) { return -1; }

   int g_lowf = Sign( timh_polywnimou( r_g->p, r_g->low ) ) ;
   int s = Sign( g_lowf * g_lowg );
   assert( s != 0 );
   if( s < 0 ) { return 1; }

   int g_highf = Sign( timh_polywnimou( r_g->p, r_g->high ) ) ;
   s = Sign( g_highg * g_highf );
   assert( s != 0 );
   if( s < 0 ) { return -1; }

   int D, dD ;

   D = dialegw_periptwsi_gia_ta_SR ( r_f, r_g ) ;

   dD = g_lowf ;

   if( D == 0 )  { return 0; }
   if( D == dD ) { return -1; }
   if( D != dD ) { return 1; }
  }
 }
}

}

struct Polywnimo* read_Polywnimo( FILE *indata, Polywnimo *f )
{
 int i;

 fscanf( indata, "%d", &f->deg );

 for ( i=0; i<(f->deg)+1; i++ )
 {
```

```
    fscanf( indata, "%lf", &f->a[i] );
  }

  for( i=(f->deg)+1; i<4; i++ )
  {
    f->a[i] = 0;//tous upoloipous tous kanoume isous me 0
  }

  return f;
}

void print_Result2(Result *res, Polywnimo *f)
{
    int i;
    for (i = 0; i < res->nroots; ++i) {
        printf("{");
        if ( res->roots[i]->is_exact ) {
            printf("%lf", res->roots[i]->low);
        } else {
            printf("(%lf, %lf)", res->roots[i]->low, res->roots[i]->high);
        }
        printf(", ");
        printf("%d, ", res->multiplicity[i]);
        printPol(res->roots[i]->p);
        printf("} ");
    }
}
```

7.2.1 Επεξήγηση του Pol.c

Η συνάρτηση $void\ printPol(\ Polywnimo\ *f)$ δέχεται σαν όρισμα τη δομή ενός πολυωνύμου και το εκτυπώνει. Δηλαδή αν η δομή του πολυωνύμου f αποτελείται από : $\alpha[0] = 1,\ \alpha[1] = -3,\ \alpha[2] = 7,\ \alpha[3] = 4$ και $\deg = 3$ τότε θα εκτυπώσει στην οθόνη μας : $4X^3 + 7X^2 - 3X + 1$

Η συνάρτηση $struct\ Result*\ Solve(Polywnimo\ *f)$ δέχεται σαν όρισμα τη δομή ενός πολυωνύμου και ελέγχοντας τι βαθμό έχει και επιλέγει ποια συνάρτηση θα πρέπει να καλέσει για την επίλυση του. Για παράδειγμα, αν έχουμε το πολυώνυμο $X^3 + 3X^2 - 5$, το πολυώνυμο μας είναι 3ου βαθμού και άρα θα καλούσαμε τη συνάρτηση $periptwsis\_3ou\_vathmou(f)$. Στην περίπτωση αυτή θα επιλύσουμε το πολυώνυμο μας με τη βοήθεια του IDS του κυβικού πολυωνύμου (Πρόταση 4.2.2).

Με τη συνάρτηση $void\ print\_Result2(Result*res,\ Polywnimo*f)$ εμφανίζουμε στην οθόνη το πολυώνυμο μας, πόσες ρίζες έχει, ποιές είναι ή σε ποιό διάστημα ανήκουν και τι πολλαπλότητας είναι. Στην περίπτωση που το πολυώνυμο μας είναι το $4X^2 - 4X + 1$ θα εμφανιστεί στην οθόνη μας $p : 4X^2 - 4X + 10.5, 2, 8X - 4$, όπου το p είναι το πολυώνυμο $4X^2 - 4X + 1$ και 0.5 είναι η μία ρίζα που έχει, με το 2 καταλαβαίνουμε ότι είναι διπλή και το πολυώνυμο $8X - 4$ είναι το πολυώνυμο h (βλέπε Πρόταση 4.2.2).

Η συνάρτηση $int\ Sugkrisi\_rizwn(\ Root\ *r_f,\ Root\ *r_g)$ δέχεται μία ρίζα από το καθένα από δύο πολυώνυμα το πολύ μέχρι τρίτου βαθμού, και έπειτα από τους απαιτούμενους ελέγχους μας επιστρέφει αν η ρίζα του πρώτου πολυωνύμου είναι μεγαλύτερη από του δεύτερου, αν είναι ίσες ή αν είναι μικρότερη.

Με τη βοήθεια της συνάρτησης $struct\ Polywnimo*\ read\_Polywnimo(\ FILE\ *indata,\ Polywnimo*f)$ διαβάζουμε από αρχείο και αποθηκεύουμε στις κατάλληλες μεταβλητές το βαθμό του εκάστοτε πολυωνύμου καθώς και τους συντελεστές του. Για παράδειγμα όταν στο αρχείο μας γράφουμε : 3 -6 11 -6 1, αναφερόμαστε σε ένα πολυώνυμο 3ου βαθμού με $\alpha[0] = -6,\ \alpha[1] = 11,\ \alpha[2] = -6,\ \alpha[3] = 1$, δηλαδή αναφερόμαστε στο πολυώνυμο $X^3 - 6X^2 + 11X - 6$.

8 Το αρχείο Ptuxiaki.txt

Είναι το αρχείο εκείνο που διαβάζει το πρόγραμμα μας και εκτελείται. Παρουσιάζεται παρακάτω με όλους τους δυνατούς συνδυασμούς ως προς το βαθμό πολυωνύμων.

3 −6 11 −6 1 3 0 2 3 1 1 1
3 −6 11 −6 1 3 0 2 3 1 1 2
3 −6 11 −6 1 3 0 2 3 1 1 3
3 −6 11 −6 1 3 0 2 3 1 2 1
3 −6 11 −6 1 3 0 2 3 1 2 2
3 −6 11 −6 1 3 0 2 3 1 2 3
3 −6 11 −6 1 3 0 2 3 1 3 1
3 −6 11 −6 1 3 0 2 3 1 3 2
3 −6 11 −6 1 3 0 2 3 1 3 3
3 −1 3 −3 1 3 0 0 0 1 1 1
3 −1 0 0 1 3 −8 0 0 1 1 1
3 −2 5 −4 1 3 −4 8 −5 1 1 1
3 −2 5 −4 1 3 −4 8 −5 1 1 2
3 −2 5 −4 1 3 −4 8 −5 1 2 1
3 −2 5 −4 1 3 −4 8 −5 1 2 2
3 −6 11 −6 1 3 0 0 0 1 1 1
3 −6 11 −6 1 3 0 0 0 1 2 1
3 −6 11 −6 1 3 0 0 0 1 3 1
3 −6 11 −6 1 3 −2 5 −4 1 1 1
3 −6 11 −6 1 3 −2 5 −4 1 1 2
3 −6 11 −6 1 3 −2 5 −4 1 2 1
3 −6 11 −6 1 3 −2 5 −4 1 2 2
3 −6 11 −6 1 3 −2 5 −4 1 3 1
3 −6 11 −6 1 3 −2 5 −4 1 3 2
3 0 2 3 1 3 0 0 0 1 1 1
3 0 2 3 1 3 0 0 0 1 2 1
3 0 2 3 1 3 0 0 0 1 3 1
3 0 0 0 1 3 −4 8 −5 1 1 1
3 0 0 0 1 3 −4 8 −5 1 1 2
3 −1 3 −3 1 3 −1 0 0 1 1 1
3 −2 5 −4 1 3 −8 0 0 1 1 1
3 −2 5 −4 1 3 −8 0 0 1 2 1
3 −6 11 −6 1 2 0 1 1 1 1
3 −6 11 −6 1 2 0 1 1 1 2
3 −6 11 −6 1 2 0 1 1 2 1
3 −6 11 −6 1 2 0 1 1 2 2

3 −6 11 −6 1 2 0 1 1 3 1

3 −6 11 −6 1 2 0 1 1 3 2

3 −6 11 −6 1 2 0 0 1 1 1

3 −6 11 −6 1 2 0 0 1 2 1

3 −6 11 −6 1 2 0 0 1 3 1

3 −6 11 −6 1 2 1 0 1 1 1

3 0 0 0 1 2 0 1 1 1 1

3 0 0 0 1 2 0 1 1 1 2

3 0 0 0 1 2 1 2 1 1 1

3 0 0 1 1 2 2 −3 1 1 1

3 0 0 1 1 2 2 −3 1 1 2

3 0 0 1 1 2 2 −3 1 2 1

3 0 0 1 1 2 2 −3 1 2 2

3 0 0 1 1 2 1 −2 1 1 1

3 0 0 1 1 2 1 −2 1 2 1

3 −1 0 0 1 2 0 1 1 1 1

3 −1 0 0 1 2 0 1 1 1 2

3 −1 0 0 1 2 1 2 1 1 1

2 0 1 1 2 2 −3 1 1 1

2 0 1 1 2 2 −3 1 1 2

2 0 1 1 2 2 −3 1 2 1

2 0 1 1 2 2 −3 1 2 2

2 1 −2 1 2 1 2 1 1 1

2 1 1 1 2 1 0 1 1 1

2 2 −3 1 2 1 −2 1 1 1

2 2 −3 1 2 1 −2 1 2 1

1 −1 1 3 −6 11 −6 1 1 1

1 −1 1 3 −6 11 −6 1 1 2

1 −1 1 3 −6 11 −6 1 1 3

1 −3 1 3 −4 8 −5 1 1 1

1 −3 1 3 −4 8 −5 1 1 2

1 −4 1 3 −2 5 −4 1 1 1

1 −4 1 3 −2 5 −4 1 1 2

1 0 1 3 0 0 0 1 1 1

1 −7 1 3 −1 0 0 1 1 1

1 2 1 2 0 1 1 1 1

1 2 1 2 0 1 1 1 2

1 −3 1 2 1 −2 1 1 1

1 1 1 2 1 0 1 1 1

1 −5 1 1 6 1 1 1

0 6 3 −6 11 −6 1 1 1

$0 \quad 8 \quad 0 \quad -8 \quad 1 \quad 1$

$0 \quad 0 \quad 2 \quad 0 \quad 2 \quad 1 \quad 1 \quad 1$

$0 \quad 0 \quad 0 \quad 0 \quad 1 \quad 1$

$3 \quad -6 \quad 11 \quad -6 \quad 1 \quad 2 \quad 0 \quad 0 \quad 1 \quad 2 \quad 5$

9 Τα αποτελέσματα του Προγράμματος μας

Εδώ παρουσιάζονται τα αποτελέσματα από το πρόγραμμα μας όταν εκτελείται με τα δεδομένα της προηγούμενης παραγράφου.

```
p: 1x^3 -6x^2 +11x^1 -6
{(-12.000000, 2.000000), 1, 9x^2 -36x^1 +27} {2.000000, 1, 3x^1 -6} {(2.000000, 12.000000), 1, 9x^2 -36x^1 +27}

q: 1x^3 +3x^2 +2x^1
{(-4.000000, -1.000000), 1, 9x^2 +18x^1} {-1.000000, 1, 3x^1 +3} {(-1.000000, 4.000000), 1, 9x^2 +18x^1}

cmp(root 1 of p, root 1 of q) = larger
```

```
p: 1x^3 -6x^2 +11x^1 -6
{(-12.000000, 2.000000), 1, 9x^2 -36x^1 +27} {2.000000, 1, 3x^1 -6} {(2.000000, 12.000000), 1, 9x^2 -36x^1 +27}

q: 1x^3 +3x^2 +2x^1
{(-4.000000, -1.000000), 1, 9x^2 +18x^1} {-1.000000, 1, 3x^1 +3} {(-1.000000, 4.000000), 1, 9x^2 +18x^1}

cmp(root 1 of p, root 2 of q) = larger
```

```
p: 1x^3 -6x^2 +11x^1 -6
{(-12.000000, 2.000000), 1, 9x^2 -36x^1 +27} {2.000000, 1, 3x^1 -6} {(2.000000, 12.000000), 1, 9x^2 -36x^1 +27}

q: 1x^3 +3x^2 +2x^1
{(-4.000000, -1.000000), 1, 9x^2 +18x^1} {-1.000000, 1, 3x^1 +3} {(-1.000000, 4.000000), 1, 9x^2 +18x^1}

cmp(root 1 of p, root 3 of q) = equal
```

```
p: 1x^3 -6x^2 +11x^1 -6
{(-12.000000, 2.000000), 1, 9x^2 -36x^1 +27} {2.000000, 1, 3x^1 -6} {(2.000000, 12.000000), 1, 9x^2 -36x^1 +27}

q: 1x^3 +3x^2 +2x^1
{(-4.000000, -1.000000), 1, 9x^2 +18x^1} {-1.000000, 1, 3x^1 +3} {(-1.000000, 4.000000), 1, 9x^2 +18x^1}

cmp(root 2 of p, root 1 of q) = larger
```

```
p: 1x^3 -6x^2 +11x^1 -6
{(-12.000000, 2.000000), 1, 9x^2 -36x^1 +27} {2.000000, 1, 3x^1 -6} {(2.000000, 12.000000), 1, 9x^2 -36x^1 +27}

q: 1x^3 +3x^2 +2x^1
{(-4.000000, -1.000000), 1, 9x^2 +18x^1} {-1.000000, 1, 3x^1 +3} {(-1.000000, 4.000000), 1, 9x^2 +18x^1}

cmp(root 2 of p, root 2 of q) = larger
```

```
p: 1x^3 -6x^2 +11x^1 -6
{(-12.000000, 2.000000), 1, 9x^2 -36x^1 +27} {2.000000, 1, 3x^1 -6} {(2.000000, 12.000000), 1, 9x^2 -36x^1 +27}

q: 1x^3 +3x^2 +2x^1
{(-4.000000, -1.000000), 1, 9x^2 +18x^1} {-1.000000, 1, 3x^1 +3} {(-1.000000, 4.000000), 1, 9x^2 +18x^1}

cmp(root 2 of p, root 3 of q) = larger
```

```
p: 1x^3 -6x^2 +11x^1 -6
{(-12.000000, 2.000000), 1, 9x^2 -36x^1 +27} {2.000000, 1, 3x^1 -6} {(2.000000, 12.000000), 1, 9x^2 -36x^1 +27}

q: 1x^3 +3x^2 +2x^1
{(-4.000000, -1.000000), 1, 9x^2 +18x^1} {-1.000000, 1, 3x^1 +3} {(-1.000000, 4.000000), 1, 9x^2 +18x^1}
```

```
cmp(root 3 of p, root 1 of q) = larger
```

```
p: 1x^3 -6x^2 +11x^1 -6
{(-12.000000, 2.000000), 1, 9x^2 -36x^1 +27} {2.000000, 1, 3x^1 -6} {(2.000000, 12.000000), 1, 9x^2 -36x^1 +27}

q: 1x^3 +3x^2 +2x^1
{(-4.000000, -1.000000), 1, 9x^2 +18x^1} {-1.000000, 1, 3x^1 +3} {(-1.000000, 4.000000), 1, 9x^2 +18x^1}

cmp(root 3 of p, root 2 of q) = larger
```

```
p: 1x^3 -6x^2 +11x^1 -6
{(-12.000000, 2.000000), 1, 9x^2 -36x^1 +27} {2.000000, 1, 3x^1 -6} {(2.000000, 12.000000), 1, 9x^2 -36x^1 +27}

q: 1x^3 +3x^2 +2x^1
{(-4.000000, -1.000000), 1, 9x^2 +18x^1} {-1.000000, 1, 3x^1 +3} {(-1.000000, 4.000000), 1, 9x^2 +18x^1}

cmp(root 3 of p, root 3 of q) = larger
```

```
p: 1x^3 -3x^2 +3x^1 -1
{1.000000, 3, 3x^1 -3}

q: 1x^3
{0.000000, 3, 3x^1}

cmp(root 1 of p, root 1 of q) = larger
```

```
p: 1x^3 -1
{(0.000000, 2.000000), 1, 1x^3 -1}

q: 1x^3 -8
{(0.000000, 9.000000), 1, 1x^3 -8}

cmp(root 1 of p, root 1 of q) = smaller
```

```
p: 1x^3 -4x^2 +5x^1 -2
{1.000000, 2, 2x^1 -2} {2.000000, 1, 1x^1 -2}

q: 1x^3 -5x^2 +8x^1 -4
{1.000000, 1, 1x^1 -1} {2.000000, 2, 2x^1 -4}

cmp(root 1 of p, root 1 of q) = equal
```

```
p: 1x^3 -4x^2 +5x^1 -2
{1.000000, 2, 2x^1 -2} {2.000000, 1, 1x^1 -2}

q: 1x^3 -5x^2 +8x^1 -4
{1.000000, 1, 1x^1 -1} {2.000000, 2, 2x^1 -4}

cmp(root 1 of p, root 2 of q) = smaller
```

```
p: 1x^3 -4x^2 +5x^1 -2
{1.000000, 2, 2x^1 -2} {2.000000, 1, 1x^1 -2}

q: 1x^3 -5x^2 +8x^1 -4
{1.000000, 1, 1x^1 -1} {2.000000, 2, 2x^1 -4}

cmp(root 2 of p, root 1 of q) = larger
```

```
====================

p: 1x^3 -4x^2 +5x^1 -2
{1.000000, 2, 2x^1 -2} {2.000000, 1, 1x^1 -2}

q: 1x^3 -5x^2 +8x^1 -4
{1.000000, 1, 1x^1 -1} {2.000000, 2, 2x^1 -4}

cmp(root 2 of p, root 2 of q) = equal

====================

p: 1x^3 -6x^2 +11x^1 -6
{(-12.000000, 2.000000), 1, 9x^2 -36x^1 +27} {2.000000, 1, 3x^1 -6} {(2.000000, 12.000000), 1, 9x^2 -36x^1 +27}

q: 1x^3
{0.000000, 3, 3x^1}

cmp(root 1 of p, root 1 of q) = larger

====================

p: 1x^3 -6x^2 +11x^1 -6
{(-12.000000, 2.000000), 1, 9x^2 -36x^1 +27} {2.000000, 1, 3x^1 -6} {(2.000000, 12.000000), 1, 9x^2 -36x^1 +27}

q: 1x^3
{0.000000, 3, 3x^1}

cmp(root 2 of p, root 1 of q) = larger

====================

p: 1x^3 -6x^2 +11x^1 -6
{(-12.000000, 2.000000), 1, 9x^2 -36x^1 +27} {2.000000, 1, 3x^1 -6} {(2.000000, 12.000000), 1, 9x^2 -36x^1 +27}

q: 1x^3
{0.000000, 3, 3x^1}

cmp(root 3 of p, root 1 of q) = larger

====================

p: 1x^3 -6x^2 +11x^1 -6
{(-12.000000, 2.000000), 1, 9x^2 -36x^1 +27} {2.000000, 1, 3x^1 -6} {(2.000000, 12.000000), 1, 9x^2 -36x^1 +27}

q: 1x^3 -4x^2 +5x^1 -2
{1.000000, 2, 2x^1 -2} {2.000000, 1, 1x^1 -2}

cmp(root 1 of p, root 1 of q) = equal

====================

p: 1x^3 -6x^2 +11x^1 -6
{(-12.000000, 2.000000), 1, 9x^2 -36x^1 +27} {2.000000, 1, 3x^1 -6} {(2.000000, 12.000000), 1, 9x^2 -36x^1 +27}

q: 1x^3 -4x^2 +5x^1 -2
{1.000000, 2, 2x^1 -2} {2.000000, 1, 1x^1 -2}

cmp(root 1 of p, root 2 of q) = smaller

====================

p: 1x^3 -6x^2 +11x^1 -6
{(-12.000000, 2.000000), 1, 9x^2 -36x^1 +27} {2.000000, 1, 3x^1 -6} {(2.000000, 12.000000), 1, 9x^2 -36x^1 +27}

q: 1x^3 -4x^2 +5x^1 -2
{1.000000, 2, 2x^1 -2} {2.000000, 1, 1x^1 -2}

cmp(root 2 of p, root 1 of q) = larger
```

```
p: 1x^3 -6x^2 +11x^1 -6
{(-12.000000, 2.000000), 1, 9x^2 -36x^1 +27} {2.000000, 1, 3x^1 -6} {(2.000000, 12.000000), 1, 9x^2 -36x^1 +27}

q: 1x^3 -4x^2 +5x^1 -2
{1.000000, 2, 2x^1 -2} {2.000000, 1, 1x^1 -2}

cmp(root 2 of p, root 2 of q) = equal
```

```
p: 1x^3 -6x^2 +11x^1 -6
{(-12.000000, 2.000000), 1, 9x^2 -36x^1 +27} {2.000000, 1, 3x^1 -6} {(2.000000, 12.000000), 1, 9x^2 -36x^1 +27}

q: 1x^3 -4x^2 +5x^1 -2
{1.000000, 2, 2x^1 -2} {2.000000, 1, 1x^1 -2}

cmp(root 3 of p, root 1 of q) = larger
```

```
p: 1x^3 -6x^2 +11x^1 -6
{(-12.000000, 2.000000), 1, 9x^2 -36x^1 +27} {2.000000, 1, 3x^1 -6} {(2.000000, 12.000000), 1, 9x^2 -36x^1 +27}

q: 1x^3 -4x^2 +5x^1 -2
{1.000000, 2, 2x^1 -2} {2.000000, 1, 1x^1 -2}

cmp(root 3 of p, root 2 of q) = larger
```

```
p: 1x^3 +3x^2 +2x^1
{(-4.000000, -1.000000), 1, 9x^2 +18x^1} {-1.000000, 1, 3x^1 +3} {(-1.000000, 4.000000), 1, 9x^2 +18x^1}

q: 1x^3
{0.000000, 3, 3x^1}

cmp(root 1 of p, root 1 of q) = smaller
```

```
p: 1x^3 +3x^2 +2x^1
{(-4.000000, -1.000000), 1, 9x^2 +18x^1} {-1.000000, 1, 3x^1 +3} {(-1.000000, 4.000000), 1, 9x^2 +18x^1}

q: 1x^3
{0.000000, 3, 3x^1}

cmp(root 2 of p, root 1 of q) = smaller
```

```
p: 1x^3 +3x^2 +2x^1
{(-4.000000, -1.000000), 1, 9x^2 +18x^1} {-1.000000, 1, 3x^1 +3} {(-1.000000, 4.000000), 1, 9x^2 +18x^1}

q: 1x^3
{0.000000, 3, 3x^1}

cmp(root 3 of p, root 1 of q) = equal
```

```
p: 1x^3
{0.000000, 3, 3x^1}

q: 1x^3 -5x^2 +8x^1 -4
{1.000000, 1, 1x^1 -1} {2.000000, 2, 2x^1 -4}

cmp(root 1 of p, root 1 of q) = smaller
```

```
p: 1x^3
{0.000000, 3, 3x^1}

q: 1x^3 -5x^2 +8x^1 -4
{1.000000, 1, 1x^1 -1} {2.000000, 2, 2x^1 -4}

cmp(root 1 of p, root 2 of q) = smaller
```

```
p: 1x^3 -3x^2 +3x^1 -1
{1.000000, 3, 3x^1 -3}

q: 1x^3 -1
{(0.000000, 2.000000), 1, 1x^3 -1}

cmp(root 1 of p, root 1 of q) = equal
```

```
p: 1x^3 -4x^2 +5x^1 -2
{1.000000, 2, 2x^1 -2} {2.000000, 1, 1x^1 -2}

q: 1x^3 -8
{(0.000000, 9.000000), 1, 1x^3 -8}

cmp(root 1 of p, root 1 of q) = smaller
```

```
p: 1x^3 -4x^2 +5x^1 -2
{1.000000, 2, 2x^1 -2} {2.000000, 1, 1x^1 -2}

q: 1x^3 -8
{(0.000000, 9.000000), 1, 1x^3 -8}

cmp(root 2 of p, root 1 of q) = equal
```

```
p: 1x^3 -6x^2 +11x^1 -6
{(-12.000000, 2.000000), 1, 9x^2 -36x^1 +27} {2.000000, 1, 3x^1 -6} {(2.000000, 12.000000), 1, 9x^2 -36x^1 +27}

q: 1x^2 +1x^1
{(-2.000000, -0.500000), 1, 1x^2 +1x^1} {(-0.500000, 2.000000), 1, 1x^2 +1x^1}

cmp(root 1 of p, root 1 of q) = larger
```

```
p: 1x^3 -6x^2 +11x^1 -6
{(-12.000000, 2.000000), 1, 9x^2 -36x^1 +27} {2.000000, 1, 3x^1 -6} {(2.000000, 12.000000), 1, 9x^2 -36x^1 +27}

q: 1x^2 +1x^1
{(-2.000000, -0.500000), 1, 1x^2 +1x^1} {(-0.500000, 2.000000), 1, 1x^2 +1x^1}

cmp(root 1 of p, root 2 of q) = larger
```

```
p: 1x^3 -6x^2 +11x^1 -6
{(-12.000000, 2.000000), 1, 9x^2 -36x^1 +27} {2.000000, 1, 3x^1 -6} {(2.000000, 12.000000), 1, 9x^2 -36x^1 +27}

q: 1x^2 +1x^1
{(-2.000000, -0.500000), 1, 1x^2 +1x^1} {(-0.500000, 2.000000), 1, 1x^2 +1x^1}

cmp(root 2 of p, root 1 of q) = larger
```

```
p: 1x^3 -6x^2 +11x^1 -6
{(-12.000000, 2.000000), 1, 9x^2 -36x^1 +27} {2.000000, 1, 3x^1 -6} {(2.000000, 12.000000), 1, 9x^2 -36x^1 +27}
```

```
q: 1x^2 +1x^1
{(-2.000000, -0.500000), 1, 1x^2 +1x^1} {(-0.500000, 2.000000), 1, 1x^2 +1x^1}

cmp(root 2 of p, root 2 of q) = larger


==================

p: 1x^3 -6x^2 +11x^1 -6
{(-12.000000, 2.000000), 1, 9x^2 -36x^1 +27} {2.000000, 1, 3x^1 -6} {(2.000000, 12.000000), 1, 9x^2 -36x^1 +27}

q: 1x^2 +1x^1
{(-2.000000, -0.500000), 1, 1x^2 +1x^1} {(-0.500000, 2.000000), 1, 1x^2 +1x^1}

cmp(root 3 of p, root 1 of q) = larger


==================

p: 1x^3 -6x^2 +11x^1 -6
{(-12.000000, 2.000000), 1, 9x^2 -36x^1 +27} {2.000000, 1, 3x^1 -6} {(2.000000, 12.000000), 1, 9x^2 -36x^1 +27}

q: 1x^2 +1x^1
{(-2.000000, -0.500000), 1, 1x^2 +1x^1} {(-0.500000, 2.000000), 1, 1x^2 +1x^1}

cmp(root 3 of p, root 2 of q) = larger


==================

p: 1x^3 -6x^2 +11x^1 -6
{(-12.000000, 2.000000), 1, 9x^2 -36x^1 +27} {2.000000, 1, 3x^1 -6} {(2.000000, 12.000000), 1, 9x^2 -36x^1 +27}

q: 1x^2
{-0.000000, 2, 2x^1}

cmp(root 1 of p, root 1 of q) = larger


==================

p: 1x^3 -6x^2 +11x^1 -6
{(-12.000000, 2.000000), 1, 9x^2 -36x^1 +27} {2.000000, 1, 3x^1 -6} {(2.000000, 12.000000), 1, 9x^2 -36x^1 +27}

q: 1x^2
{-0.000000, 2, 2x^1}

cmp(root 2 of p, root 1 of q) = larger


==================

p: 1x^3 -6x^2 +11x^1 -6
{(-12.000000, 2.000000), 1, 9x^2 -36x^1 +27} {2.000000, 1, 3x^1 -6} {(2.000000, 12.000000), 1, 9x^2 -36x^1 +27}

q: 1x^2
{-0.000000, 2, 2x^1}

cmp(root 3 of p, root 1 of q) = larger


==================

To 2o polywnimo mou den exei rizes, opote den mporoume na kanoume sugkrisi rizwn!
p: 1x^3 -6x^2 +11x^1 -6
{(-12.000000, 2.000000), 1, 9x^2 -36x^1 +27} {2.000000, 1, 3x^1 -6} {(2.000000, 12.000000), 1, 9x^2 -36x^1 +27}

q: 1x^2 +1


H riza pou zitisate den uparxei!


==================

p: 1x^3
{0.000000, 3, 3x^1}
```

```
q: 1x^2 +1x^1
{(-2.000000, -0.500000), 1, 1x^2 +1x^1} {(-0.500000, 2.000000), 1, 1x^2 +1x^1}

cmp(root 1 of p, root 1 of q) = larger
```

```
p: 1x^3
{0.000000, 3, 3x^1}

q: 1x^2 +1x^1
{(-2.000000, -0.500000), 1, 1x^2 +1x^1} {(-0.500000, 2.000000), 1, 1x^2 +1x^1}

cmp(root 1 of p, root 2 of q) = equal
```

```
p: 1x^3
{0.000000, 3, 3x^1}

q: 1x^2 +2x^1 +1
{-1.000000, 2, 2x^1 +2}

cmp(root 1 of p, root 1 of q) = larger
```

```
p: 1x^3 +1x^2
{-1.000000, 1, 1x^1 +1} {0.000000, 2, 2x^1}

q: 1x^2 -3x^1 +2
{(-4.000000, 1.500000), 1, 1x^2 -3x^1 +2} {(1.500000, 4.000000), 1, 1x^2 -3x^1 +2}

cmp(root 1 of p, root 1 of q) = smaller
```

```
p: 1x^3 +1x^2
{-1.000000, 1, 1x^1 +1} {0.000000, 2, 2x^1}

q: 1x^2 -3x^1 +2
{(-4.000000, 1.500000), 1, 1x^2 -3x^1 +2} {(1.500000, 4.000000), 1, 1x^2 -3x^1 +2}

cmp(root 1 of p, root 2 of q) = smaller
```

```
p: 1x^3 +1x^2
{-1.000000, 1, 1x^1 +1} {0.000000, 2, 2x^1}

q: 1x^2 -3x^1 +2
{(-4.000000, 1.500000), 1, 1x^2 -3x^1 +2} {(1.500000, 4.000000), 1, 1x^2 -3x^1 +2}

cmp(root 2 of p, root 1 of q) = smaller
```

```
p: 1x^3 +1x^2
{-1.000000, 1, 1x^1 +1} {0.000000, 2, 2x^1}

q: 1x^2 -3x^1 +2
{(-4.000000, 1.500000), 1, 1x^2 -3x^1 +2} {(1.500000, 4.000000), 1, 1x^2 -3x^1 +2}

cmp(root 2 of p, root 2 of q) = smaller
```

```
p: 1x^3 +1x^2
{-1.000000, 1, 1x^1 +1} {0.000000, 2, 2x^1}

q: 1x^2 -2x^1 +1
{1.000000, 2, 2x^1 -2}
```

```
cmp(root 1 of p, root 1 of q) = smaller
```

```
p: 1x^3 +1x^2
{-1.000000, 1, 1x^1 +1} {0.000000, 2, 2x^1}

q: 1x^2 -2x^1 +1
{1.000000, 2, 2x^1 -2}

cmp(root 2 of p, root 1 of q) = smaller
```

```
p: 1x^3 -1
{(0.000000, 2.000000), 1, 1x^3 -1}

q: 1x^2 +1x^1
{(-2.000000, -0.500000), 1, 1x^2 +1x^1} {(-0.500000, 2.000000), 1, 1x^2 +1x^1}

cmp(root 1 of p, root 1 of q) = larger
```

```
p: 1x^3 -1
{(0.000000, 2.000000), 1, 1x^3 -1}

q: 1x^2 +1x^1
{(-2.000000, -0.500000), 1, 1x^2 +1x^1} {(-0.500000, 2.000000), 1, 1x^2 +1x^1}

cmp(root 1 of p, root 2 of q) = larger
```

```
p: 1x^3 -1
{(0.000000, 2.000000), 1, 1x^3 -1}

q: 1x^2 +2x^1 +1
{-1.000000, 2, 2x^1 +2}

cmp(root 1 of p, root 1 of q) = larger
```

```
p: 1x^2 +1x^1
{(-2.000000, -0.500000), 1, 1x^2 +1x^1} {(-0.500000, 2.000000), 1, 1x^2 +1x^1}

q: 1x^2 -3x^1 +2
{(-4.000000, 1.500000), 1, 1x^2 -3x^1 +2} {(1.500000, 4.000000), 1, 1x^2 -3x^1 +2}

cmp(root 1 of p, root 1 of q) = smaller
```

```
p: 1x^2 +1x^1
{(-2.000000, -0.500000), 1, 1x^2 +1x^1} {(-0.500000, 2.000000), 1, 1x^2 +1x^1}

q: 1x^2 -3x^1 +2
{(-4.000000, 1.500000), 1, 1x^2 -3x^1 +2} {(1.500000, 4.000000), 1, 1x^2 -3x^1 +2}

cmp(root 1 of p, root 2 of q) = smaller
```

```
p: 1x^2 +1x^1
{(-2.000000, -0.500000), 1, 1x^2 +1x^1} {(-0.500000, 2.000000), 1, 1x^2 +1x^1}

q: 1x^2 -3x^1 +2
{(-4.000000, 1.500000), 1, 1x^2 -3x^1 +2} {(1.500000, 4.000000), 1, 1x^2 -3x^1 +2}

cmp(root 2 of p, root 1 of q) = smaller
```

```
p: 1x^2 +1x^1
{(−2.000000, −0.500000), 1, 1x^2 +1x^1} {(−0.500000, 2.000000), 1, 1x^2 +1x^1}

q: 1x^2 −3x^1 +2
{(−4.000000, 1.500000), 1, 1x^2 −3x^1 +2} {(1.500000, 4.000000), 1, 1x^2 −3x^1 +2}

cmp(root 2 of p, root 2 of q) = smaller
```

```
p: 1x^2 −2x^1 +1
{1.000000, 2, 2x^1 −2}

q: 1x^2 +2x^1 +1
{−1.000000, 2, 2x^1 +2}

cmp(root 1 of p, root 1 of q) = larger
```

```
Ta polywnima mou den exoun rizes, opote den mporoume na kanoume sugkrisi rizwn!
p: 1x^2 +1x^1 +1


q: 1x^2 +1


H riza pou zitisate den uparxei!
```

```
p: 1x^2 −3x^1 +2
{(−4.000000, 1.500000), 1, 1x^2 −3x^1 +2} {(1.500000, 4.000000), 1, 1x^2 −3x^1 +2}

q: 1x^2 −2x^1 +1
{1.000000, 2, 2x^1 −2}

cmp(root 1 of p, root 1 of q) = equal
```

```
p: 1x^2 −3x^1 +2
{(−4.000000, 1.500000), 1, 1x^2 −3x^1 +2} {(1.500000, 4.000000), 1, 1x^2 −3x^1 +2}

q: 1x^2 −2x^1 +1
{1.000000, 2, 2x^1 −2}

cmp(root 2 of p, root 1 of q) = larger
```

```
p: 1x^1 −1
{1.000000, 1, 1x^1 −1}

q: 1x^3 −6x^2 +11x^1 −6
{(−12.000000, 2.000000), 1, 9x^2 −36x^1 +27} {2.000000, 1, 3x^1 −6} {(2.000000, 12.000000), 1, 9x^2 −36x^1 +27}

cmp(root 1 of p, root 1 of q) = equal
```

```
p: 1x^1 −1
{1.000000, 1, 1x^1 −1}

q: 1x^3 −6x^2 +11x^1 −6
{(−12.000000, 2.000000), 1, 9x^2 −36x^1 +27} {2.000000, 1, 3x^1 −6} {(2.000000, 12.000000), 1, 9x^2 −36x^1 +27}

cmp(root 1 of p, root 2 of q) = smaller
```

```
p: 1x^1 -1
{1.000000, 1, 1x^1 -1}

q: 1x^3 -6x^2 +11x^1 -6
{(-12.000000, 2.000000), 1, 9x^2 -36x^1 +27} {2.000000, 1, 3x^1 -6} {(2.000000, 12.000000), 1, 9x^2 -36x^1 +27}

cmp(root 1 of p, root 3 of q) = smaller
```

```
p: 1x^1 -3
{3.000000, 1, 1x^1 -3}

q: 1x^3 -5x^2 +8x^1 -4
{1.000000, 1, 1x^1 -1} {2.000000, 2, 2x^1 -4}

cmp(root 1 of p, root 1 of q) = larger
```

```
p: 1x^1 -3
{3.000000, 1, 1x^1 -3}

q: 1x^3 -5x^2 +8x^1 -4
{1.000000, 1, 1x^1 -1} {2.000000, 2, 2x^1 -4}

cmp(root 1 of p, root 2 of q) = larger
```

```
p: 1x^1 -4
{4.000000, 1, 1x^1 -4}

q: 1x^3 -4x^2 +5x^1 -2
{1.000000, 2, 2x^1 -2} {2.000000, 1, 1x^1 -2}

cmp(root 1 of p, root 1 of q) = larger
```

```
p: 1x^1 -4
{4.000000, 1, 1x^1 -4}

q: 1x^3 -4x^2 +5x^1 -2
{1.000000, 2, 2x^1 -2} {2.000000, 1, 1x^1 -2}

cmp(root 1 of p, root 2 of q) = larger
```

```
p: 1x^1
{0.000000, 1, 1x^1}

q: 1x^3
{0.000000, 3, 3x^1}

cmp(root 1 of p, root 1 of q) = equal
```

```
p: 1x^1 -7
{7.000000, 1, 1x^1 -7}

q: 1x^3 -1
{(0.000000, 2.000000), 1, 1x^3 -1}

cmp(root 1 of p, root 1 of q) = larger
```

```
p: 1x^1 +2
{-2.000000, 1, 1x^1 +2}

q: 1x^2 +1x^1
{(-2.000000, -0.500000), 1, 1x^2 +1x^1} {(-0.500000, 2.000000), 1, 1x^2 +1x^1}

cmp(root 1 of p, root 1 of q) = smaller


=================

p: 1x^1 +2
{-2.000000, 1, 1x^1 +2}

q: 1x^2 +1x^1
{(-2.000000, -0.500000), 1, 1x^2 +1x^1} {(-0.500000, 2.000000), 1, 1x^2 +1x^1}

cmp(root 1 of p, root 2 of q) = smaller


=================

p: 1x^1 -3
{3.000000, 1, 1x^1 -3}

q: 1x^2 -2x^1 +1
{1.000000, 2, 2x^1 -2}

cmp(root 1 of p, root 1 of q) = larger


=================

To 2o polywnimo mou den exei rizes, opote den mporoume na kanoume sugkrisi rizwn!
p: 1x^1 +1
{-1.000000, 1, 1x^1 +1}

q: 1x^2 +1


H riza pou zitisate den uparxei!


=================

p: 1x^1 -5
{5.000000, 1, 1x^1 -5}

q: 1x^1 +6
{-6.000000, 1, 1x^1 +6}

cmp(root 1 of p, root 1 of q) = larger


=================

Den xreiazetai na kanoume sugkrisi rizwn, dioti to 1o polywnimo den exei rizes!


=================

Den xreiazetai na kanoume sugkrisi rizwn, dioti ta polywnima den exoun rizes!


=================

Den mporw na exw sumperasma gia ti sugkrisi rizwn, logo tou oti exw to mideniko polywnimo, ara exw apeires rizes!


=================

Den mporw na exw sumperasma gia ti sugkrisi rizwn, logo tou oti exw to mideniko polywnimo, ara exw apeires rizes!


=================

p: 1x^3 -6x^2 +11x^1 -6
{(-12.000000, 2.000000), 1, 9x^2 -36x^1 +27} {2.000000, 1, 3x^1 -6} {(2.000000, 12.000000), 1, 9x^2 -36x^1 +27}
```

```
q: 1x^2
{-0.000000, 2, 2x^1}

H riza pou zitisate den uparxei!
```

10 Βιβλιογραφία

[1] Laurent Dupont, Daniel Lazard, Sylvain Lazard, and Sylvain Petitjean. Nearoptimal parameterization of the intersection of quadrics. In Proc. Annual ACM Symp. on Comp. Geometry, pages 246–255. ACM, June 2003.

[2] Arno Eigenwillig, Lutz Kettner, Elmar Schomer, and Nicola Wolpert. Complete, exact, and efficient computations with cubic curves. In Symposium on Computational Geometry, pages 409–418, 2004.

[3] I.Z. Emiris, E.P. Tsigaridas, and G.M. Tzoumas. The InCircle predicates for ellipses. In Proc. European Workshop Computat. Geometry, 2006.

[4] I.Z. Emiris, E.P. Tsigaridas, and G.M. Tzoumas. The predicates for the Voronoi diagram of ellipses. In Proc. Annual ACM Symp. on Computational Geometry, 2006. (to appear).

[5] M. Hemmer, E. Schomer, and N. Wolpert. Computing a 3dimensional cell in an arrangement of quadrics: Exactly and actually! In Proc. Annual ACM Symp. Comput. Geometry, pages 264–273, 2001.

[6] M.I. Karavelas and I.Z. Emiris. Root comparison techniques applied to the planar additively weighted Voronoi diagram. In Proc. SODA, pages 320–329, January 2003.

[7] L. Lazard, S.and Pe˜naranda and S. Petitjean. Intersecting quadrics: an efficient and exact implementation. In Symposium of Computational Geometry, pages 419–428, 2004.

[8] L.J. Guibas, M.I. Karavelas, and D. Russel. A computational framework for handling motion. In Proc. 6th Workshop Algor. Engin. and Experim. (ALENEX), pages 129–141, Jan 2004.

[9] V. Karamcheti, C. Li, I. Pechtchanski, and C. Yap. A Core Library For Robust Numeric and Geometric Computation. In Proc. 15th Annual ACM Symp. on Comp. Geometry, pages 351–359, 1999.

[10] E. Berberich, A. Eigenwillig, M. Hemmer, S. Hert, L. Kettner, K. Mehlhorn, J. Reichel, S. Schmitt, E. Schomer, and N. Wolpert. EXACUS: Efficient and Exact Algorithms for Curves and Surfaces. In ESA, volume 1669 of LNCS, pages 155–166. Springer, 2005.

[11] J. Keyser, T. Culver, D. Manocha, and S. Krishnan. ESOLID: A system for exact boundary evaluation. Comp. Aided Design, 36(2):175–193, 2004.

[12] I.Z. Emiris, A.V. Kakargias, M. Teillaud, E.P. Tsigaridas, and S. Pion. Towards an open curved kernel. In Proc. Annual ACM Symp. on Computational Geometry, pages 438–446, New York, 2004. ACM Press.

[13] L. Yang. Recent advances on determining the number of real roots of parametric polynomials. J. Symbolic Computation, 28:225–242, 1999.

[14] S. Basu, R. Pollack, and MF. Roy. Algorithms in Real Algebraic Geometry, volume 10 of Algorithms and Computation in Mathematics. SpringerVerlag, 2003. ISBN 3540009736.

[15] I. Z. Emiris and E. P. Tsigaridas. Real algebraic numbers and polynomial systems of small degree. submitted for journal publication, June 2005.

[16] I. Z. Emiris and E. P. Tsigaridas. Computations with real algebraic numbers of degree up to 4. In Proc. ICPSS (in honor of D. Lazard), pages 64–66, Paris, 2004.

[17] I.Z. Emiris and E.P. Tsigaridas. Computing with real algebraic numbers of small degree. In S. Albers and T. Radzik, editors, Proc. ESA, volume 3221 of LNCS, pages 652–663. Springer Verlag, 2004.

[18] Daniel Lazard. Quantifier elimination: optimal solution for two classical examples. J. Symb. Comput., 5(12): 261–266, 1988. ISSN 07477171.

[19] Rudiger Loos and Volker Weispfenning. Applying linear quantifier elimination. The Computer Journal, 36(5):450–462, 1993.

[20] V. Weispfenning. Quantifier elimination for real algebra the cubic case. In ISSAC, pages 258–263, 1994.

[21] V. Weispfenning. Quantifier elimination for real algebra the quadratic case and beyond. Applicable Algebra in Engineering, Communication and Computing, 8(2):85–101, 1997.

[22] H. Hong. Quantifier elimination for formulas constrained by quadratic equations. In ISSAC, pages 264–274, 1993.

[23] V. Weispfenning. The complexity of linear problems in fields. J. Symb. Comput., 5(1/2):3–27, 1988.

[24] V. Weispfenning. A new approach to quantifier elimination for real algebra and geometry. In Abstracts 9th European Workshop Comput. Geom., pages 31–35. FernUniversitat Hagen, 1993.

[25] W. Dunham. The mathematical universe: an alphabetical journey through great proofs, problems and personalities. Wiley and Sons, Inc, New York, 1994.

[26] C. B. Boyer. A history of mathematics. Wiley and Sons, Inc, New York, 1991.

[27] M. Mignotte. Mathematics for computer algebra. SpringerVerlag, New York, 1991.

[28] M. Mignotte and D. Stefanescu. Polynomials: An algorithmic approach. Springer, 1999.

[29] C.K. Yap. Fundamental Problems of Algorithmic Algebra. Oxford University Press, New York, 2000.

[30] T. W. Sederberg and GengZhe Chang. Isolating the real roots of polynomials using isolator polynomials. In C. Bajaj, editor, Algebraic Geometry and Applications. Springer, 1993.

[31] I.Z. Emiris and M.I. Karavelas. The predicates of the Apollonius diagram: algorithmic analysis and implementation. Comp. Geom.: Theory and Appl., Spec. Issue on Robust Geometric Algorithms and their Implementations, 33(12): 18–57, 2006.

[32] Olivier Devillers, Alexandra Fronville, Bernard Mourrain, and Monique Teillaud. Algebraic methods and arithmetic filtering for exact predicates on circle arcs. In Proc. 16th Annu. ACM Sympos. Comput. Geom., pages 139–147, 2000.

[33] Olivier Devillers, Alexandra Fronville, Bernard Mourrain, and Monique Teillaud. Algebraic methods and arithmetic filtering for exact predicates on circle arcs. Comput. Geom. Theory Appl., 22:119–142, 2002.

[34] Ηλίας Π. Τσιγαρίδας, "Αλγεβρικοί αλγόριθμοι και εφαρμογές στην υπολογιστική γεωμετρία ", Ιούνιος 2006