



ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ

ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

Θεωρητική και Πειραματική Μελέτη Άμεσων
Μεθόδων Για Την Λύση Αραιών Γραμμικών
Συστημάτων.

Study and Evaluation of Direct Methods of Sparse
Linear System.

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

της

Κοντογιώργη Ειρήνη

Βόλος, Ιούνιος 2015



ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ

ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

Θεωρητική και Πειραματική Μελέτη Άμεσων Μεθόδων Για Την Λύση Αραιών Γραμμικών Συστημάτων.

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

της

Κοντογιώργη Ειρήνη

Επιβλέποντες :

Τσομπανοπούλου Παναγιώτα	Μποζάνης Παναγιώτης
Επίκουρος Καθηγήτρια Π.Θ.	Καθηγητής Π.Θ.

Εγκρίθηκε από την διμελή εξεταστική επιτροπή την

(Υπογραφή)

.....

ΚΥΡΙΑ ΕΠΙΒΛΕΠΟΥΣΑ

Επίκουρος Καθηγήτρια Π.Θ.

(Υπογραφή)

.....

ΔΕΥΤΕΡΕΥΩΝ ΕΠΙΒΛΕΠΩΝ

Καθηγητής Π.Θ.

(Υπογραφή)

.....

Κοντογιώργη Ειρήνη

Διπλωματούχος Μηχανικός Ηλεκτρονικών Υπολογιστών, Τηλεπικοινωνιών και
Δικτύων του Τμήματος Ηλεκτρολόγων Μηχανικών και Μηχανικών
Υπολογιστών, Πανεπιστημίου Θεσσαλίας

© 2015 – All rights reserved

Ευχαριστίες

Θα ήθελα να ευχαριστήσω όλους όσους συνέβαλαν με κάθε τρόπο στην επιτυχή εκπόνηση αυτής της διπλωματικής εργασίας. Ευχαριστώ θερμά την επίκουρο καθηγήτρια κ. Παναγιώτα Τσομπανοπούλου για την επίβλεψη της διπλωματικής μου εργασίας και για την ευκαιρία που μου έδωσε να την εκπονήσω στο Εργαστήριο Τεχνολογιών Δόμησης και Επεξεργασίας Δεδομένων, για την άριστη συνεργασία, τον πολύτιμο χρόνο που διέθεσε και την προθυμία να προσφέρει τις γνώσεις και την εμπειρία της. Ευχαριστώ ιδιαίτερα τον κ. Αθανάσιο Φεύγα για την εξαιρετική συνεργασία που είχαμε, και ελπίζω να συνεχίσουμε να έχουμε στο μέλλον, ήταν πάντα διαθέσιμος να ασχοληθεί με κάθε απορία μου σχετική με ακαδημαϊκά ζητήματα, εντός και εκτός των πλαισίων της παρούσας εργασίας και με κάθε δισταγμό μου, για τα επόμενα βήματα των σπουδών μου. Έπειτα, θα ήθελα να ευχαριστήσω τους καθηγητές της σχολής ΤΗΜΜΥ του Πανεπιστημίου Θεσσαλίας που με καθοδήγησαν τα χρόνια αυτά στο πολύ ενδιαφέρον και ευρύ αντικείμενο του μηχανικού ηλεκτρονικών υπολογιστών. Τέλος, το μεγαλύτερο ευχαριστώ το οφείλω στην οικογένειά μου που πιστεύει σε εμένα, είναι δίπλα μου και με στηρίζει σε κάθε βήμα.

Πίνακας περιεχομένων

Περίληψη	1
1.Εισαγωγή.....	3
1.1 Αραιοί Πίνακες.....	3
1.2 Δομές Δεδομένων Αραιού Πίνακα.....	4
1.3 Πολλαπλασιασμός Πίνακα με Διάνυσμα.....	6
1.4 Πολλαπλασιασμός Πίνακα με Πίνακα.....	7
1.5 Πρόσθεση Πινάκων.....	8
1.6 Πίνακας Μετάθεσης.....	8
2. Διατάξεις Fill-Reducing.....	10
2.1 Διάταξη Ελαχίστου Βαθμού (Minimum Degree).....	10
2.2 Διάταξη Μεγίστου Βαθμού (Maximum Degree).....	15
2.3 Μορφή Τριγωνικού Block (Block Triangular Form).....	20
3. Συμβολική Ανάλυση.....	25
3.1 Δένδρο Απαλοιφής	25
3.2 Μεταδιατεταγμένη διάσχιση Δένδρου	33
3.3 Πλήθος Γραμμής και Πλήθος Στήλης.....	36
3.3.1 Πλήθος Γραμμής.....	37
3.3.2 Πλήθος Στήλης.....	44
4. Παραγοντοποίηση Cholesky.....	49
4.1 Up - Looking Cholesky.....	49
4.2 Left-Looking Cholesky.....	53
4.3 Supernodal Cholesky.....	56
4.4 Right - Looking Cholesky	62
4.5 Multifrontal Cholesky	65
4.6 Χρήση της Παραγοντοποίησης Cholesky	66
5. Παραγοντοποίηση LU	67
5.1 Μερική Οδήγηση.....	67
5.2 Left - Looking LU.....	68

5.3	Right - Looking LU.....	75
5.4	Multifrontal LU	77
5.5	Χρήση της παραγοντοποίησης LU.....	85
6.	Πειραματική Μελέτη.....	87
6.1	SuiteSparse	87
6.2	CSparse.....	88
6.3	Πειραματικά Αποτελέσματα	88
7.	Συμπεράσματα και Μελλοντική Εργασία	93
7.1	Συμπεράσματα.....	93
7.2	Μελλοντική Εργασία.....	94
	Βιβλιογραφία.....	95
	ΠΑΡΑΡΤΗΜΑ Α.....	97

Περίληψη

Σκοπός της παρούσας διπλωματικής είναι η μελέτη άμεσων μεθόδων επίλυσης αραιών γραμμικών συστημάτων. Η εργασία περιλαμβάνει θεωρητικό και πειραματικό μέρος.

Στο θεωρητικό κομμάτι , περιέχεται εισαγωγή σχετικά με τον ορισμό του αραιού πίνακα , των άμεσων μεθόδων επίλυσης, των μεθόδων δόμησης και των πράξεων πολλαπλασιασμού, πρόσθεσης και μετάθεσης αραιών πινάκων. Έπειτα, περιγράφονται οι μέθοδοι μετάθεσης που αντιμετωπίζουν την ελαχιστοποίηση του πλήθους των μη μηδενικών εγγραφών του αρχικού αραιού πίνακα και η φάση της συμβολικής ανάλυσης για την εύρεση μιας ρητής αναπαράστασης του μη μηδενικού προτύπου της παραγοντοποίησης. Τέλος, καταγράφονται κάποιες από τις πιο συνηθισμένες αραιές μεθόδους παραγοντοποίησης των Cholesky και LU.

Στο πειραματικό μέρος, εκτελείται κώδικας υπολογισμού του χρόνου εκτέλεσης των μεθόδων παραγοντοποίησης Cholesky και LU , για διάφορα μεγέθη πινάκων και διεξάγεται η σύγκριση των αποτελεσμάτων των δύο μεθόδων.

Λέξεις Κλειδιά

Αραιός πίνακας, αραιό γραμμικό σύστημα, άμεσες μέθοδοι επίλυσης αραιού γραμμικού συστήματος, δένδρο απαλοιφής, πλήθος γραμμής και στήλης μερική οδήγηση, παραγοντοποίηση Cholesky, παραγοντοποίηση LU ,SuiteSparse , CSparse.

Abstract

The purpose of this diploma thesis is the study of direct methods for solving sparse linear systems. The work consists of a theoretical and experimental part.

The theoretical part contains introduction on the definition of sparse matrices and direct methods for solving linear systems, data structure manipulation and matrix multiplication, addition and transpose. Moreover, describes fill-reducing orderings which minimize the number of nonzeros in the factorization and the amount of work required to compute it. Following, the phase of symbolic analysis which includes the computation of an explicit representation of the nonzero pattern of the factorization is included and the most commonly used sparse methods of Cholesky and LU factorization are presented.

In the experimental part, code for various sizes of sparse matrices is performed. The results of execution time of both methods Cholesky and LU factorization are compared.

Key Words

Sparse matrix, sparse linear system, direct methods for solving sparse linear systems, elimination tree, row counts, column counts, partial pivoting, Cholesky factorization, LU factorization, SuiteSparse, CSparse.

1.

Εισαγωγή

Στο κεφάλαιο αυτό θα ορίσουμε τον αραιό πίνακα και τις άμεσες μεθόδους επίλυσης αραιών γραμμικών συστημάτων^[2]. Επίσης, θα παρουσιάσουμε κάποιες από τις δομές αποθήκευσης και τις πράξεις πολλαπλασιασμού και πρόσθεσης μεταξύ αραιών πινάκων και αραιού πίνακα με διάνυσμα, οι οποίες θα μας χρειαστούν στα επόμενα κεφάλαια.

1.1 Αραιοί Πίνακες

Ένα αραιό γραμμικό σύστημα είναι της μορφής $Ax = b$, όπου A είναι ένας αραιός πίνακας. Ένας πίνακας ονομάζεται αραιός όταν τα περισσότερα στοιχεία του είναι μηδέν. Το μειονέκτημα των αραιών πινάκων είναι ότι δαπανούν αρκετό χώρο μνήμης για την αποθήκευση των μηδενικών. Ωστόσο, υπάρχουν δομές δεδομένων που εκμεταλλεύονται την αραιότητα τους έτσι ώστε να αποθηκεύονται μόνο οι μη μηδενικές καταχωρήσεις.

Υπάρχουν δύο μεγάλες κατηγορίες επίλυσης αραιών γραμμικών συστημάτων: οι άμεσες και οι επαναληπτικές μέθοδοι^A. Οι άμεσες μέθοδοι επίλυσης υπολογίζουν την ακριβή λύση, παραγοντοποιώντας τον πίνακα A , δηλαδή εκφράζοντάς τον ως ένα γινόμενο δύο ή περισσότερων πινάκων, και εν συνεχεία βασιζόμενες στην ευθεία και όπισθεν αντικατάσταση υπολογίζουν την τιμή του αγνώστου διανύσματος x . Στις πιο γνωστές άμεσες μεθόδους επίλυσης περιλαμβάνονται οι μέθοδοι LU και Cholesky.

1.2 Δομές Δεδομένων Αραιού Πίνακα

Η πιο απλή μορφή δομής δεδομένων ενός αραιού πίνακα είναι η τριάδα μηδενικής βάσης (zero – based triplet form). Ονομάζεται μηδενικής βάσης γιατί οι δείκτες γραμμής και στήλης ενός πίνακα μεγέθους $m \times n$ κυμαίνονται από 0 έως $m-1$ και από 0 έως $n-1$, αντίστοιχα.

Συγκεκριμένα, η triplet form ενός αραιού πίνακα A αποτελείται από τρία διανύσματα ίδιου μεγέθους :

- T_i , με τους δείκτες γραμμής ,
- T_j , με τους δείκτες στήλης ,
- x , με τις μη μηδενικές αριθμητικές τιμές του A .

Έτσι, η $a[i][j]$ μη μηδενική εγγραφή θα μετατραπεί σε triplet form ως εξής:

- $T_i[k] = i$
- $T_j[k] = j$
- $x[k] = a[i][j]$.

όπου k είναι η αντίστοιχη θέση της εγγραφής σε κάθε πίνακα.

Η σειρά με την οποία οι μη μηδενικές εγγραφές του πίνακα A τοποθετούνται στην δομή είναι τυχαία. Επίσης, αν υπάρχουν πολλαπλές καταχωρήσεις με ταυτόσημους δείκτες γραμμής και στήλης τότε η αντίστοιχη αριθμητική τιμή είναι το άθροισμα των διπλότυπων εγγραφών.

Η triplet form δημιουργείται εύκολα, ωστόσο έχει το μειονέκτημα ότι είναι δύσκολο να χρησιμοποιηθεί στους περισσότερους αλγόριθμους αραιών πινάκων, γιατί δεν γνωρίζουμε εκ των προτέρων το μέγεθος των διανυσμάτων της, με αποτέλεσμα να πρέπει να αντιγράφουμε ολόκληρο

τον πίνακα από ένα σημείο της μνήμης σε άλλο κάθε φορά που προστίθεται μία νέα εγγραφή.

Η μορφή συμπιεσμένης στήλης (compressed-column form) είναι πιο χρήσιμη. Στην δομή αυτή ένας αραιός πίνακας μεγέθους $m \times n$ με τουλάχιστον n_{\max} μη μηδενικές εγγραφές, αναπαριστάται από τρία διανύσματα στήλες:

- p , διάνυσμα ακεραίων μεγέθους $n+1$, η κάθε θέση του οποίου περιέχει το συνολικό πλήθος μη μηδενικών εγγραφών όλων των στηλών μέχρι και πριν την τρέχουσα στήλη. Ισχύει πάντα $p[0] = 0$, μιας και η μηδενική στήλη είναι η πρώτη και δεν υπάρχει κάποια προηγούμενη.
- i , διάνυσμα ακεραίων μεγέθους n_{\max} , όπου η κάθε θέση του περιέχει τον αντίστοιχο δείκτη γραμμής της τρέχουσας στήλης στην οποία υπάρχει η μη μηδενική εγγραφή.
- x , διάνυσμα πραγματικών μεγέθους n_{\max} , όπου η κάθε θέση του περιέχει την αντίστοιχη μη μηδενική αριθμητική τιμή.

Με την μορφή αυτή , η μη μηδενική εγγραφή $a[i][j]$ αναπαρίσταται:

- $p[j] = \text{πλήθος μη μηδενικών εγγραφών μέχρι και την στήλη } j-1$.
- $i[k] = i$.
- $x[k] = a[i][j]$.

Απαιτείται, οι δείκτες γραμμής της κάθε στήλης να βρίσκονται σε αύξουσα σειρά και να μην παρουσιάζονται οι μηδενικές καταχωρήσεις.

Μπορούμε να μετατρέψουμε την τριαδική μορφή στην μορφή συμπιεσμένης στήλης, με την εξής διαδικασία:

Αρχικά, έχοντας τους τρεις πίνακες με ίδιο μέγεθος που φέρουν τους δείκτες γραμμής, στήλης και τις αριθμητικές τιμές θέλουμε να προσκομίσουμε από αυτούς τα στοιχεία των τριών πινάκων με το άθροισμα των μη μηδενικών των προηγούμενων στηλών, τους δείκτες γραμμής σε αύξουσα σειρά για κάθε στήλη, και τις αντίστοιχες αριθμητικές τιμές. Επομένως,

- βρίσκουμε το πλήθος μη μηδενικών στοιχείων κάθε στήλης ξεχωριστά, με την βοήθεια ενός διανύσματος-στήλης w , όπου η κάθε θέση του αντιστοιχεί σε κάθε στήλη του αρχικού αραιού πίνακα.
- έπειτα, παίρνουμε το διάνυσμα w και βρίσκουμε για κάθε θέση του το πλήθος όλων των προηγούμενων μη μηδενικών στοιχείων. Με τον τρόπο αυτό δημιουργούμε το διάνυσμα p .
- Ταξινομούμε τα διανύσματα i και x , ώστε οι δείκτες γραμμής να είναι σε αύξουσα σειρά και οι αριθμητικές τιμές να βρίσκονται στην αντίστοιχη θέση.

Έτσι, καταφέρνουμε να πάρουμε τον αραιό πίνακα A από triplet form σε compressed column form.

1.3 Πολλαπλασιασμός Πίνακα με Διάνυσμα

Ο πολλαπλασιασμός ενός πίνακα με διάνυσμα είναι ένας από τους πιο απλούς αλγορίθμους. Έστω, $z = Ax + y$ με x και y πυκνά διανύσματα και A αραιός πίνακας $m \times n$, το z αποτέλεσμα μπορεί να υπολογιστεί αν χωρίσουμε τον πίνακα A σε διανύσματα στήλες, όσα είναι και ο αριθμός

των στοιχείων του διανύσματος x και υπολογίζουμε από το τελικό αποτέλεσμα μία στήλη την φορά, δηλαδή

$$z = [A_{*1} \dots A_{*n}] * [x_1 \dots x_n]^T + y$$

Ο πολλαπλασιασμός θα πραγματοποιείται μόνο για τις τιμές όπου το στοιχείο του A θα είναι μη μηδενικό $a[i][j] \neq 0$.

1.4 Πολλαπλασιασμός Πίνακα με Πίνακα

Ο πολλαπλασιασμός αραιών πινάκων της μορφής $C = A * B$, όπου A είναι $m \times n$, B είναι $k \times n$ και C είναι $m \times n$ πίνακες, αποθηκευμένοι σε μορφή συμπιεσμένης στήλης (compressed – column) πραγματοποιείται με την διαπέραση των A και B κατά μία στήλη την φορά και την δημιουργία της αντίστοιχης στήλης του C .

Συγκεκριμένα, έστω C_{*j} και B_{*j} , η στήλη j των πινάκων C και B αντίστοιχα, τότε:

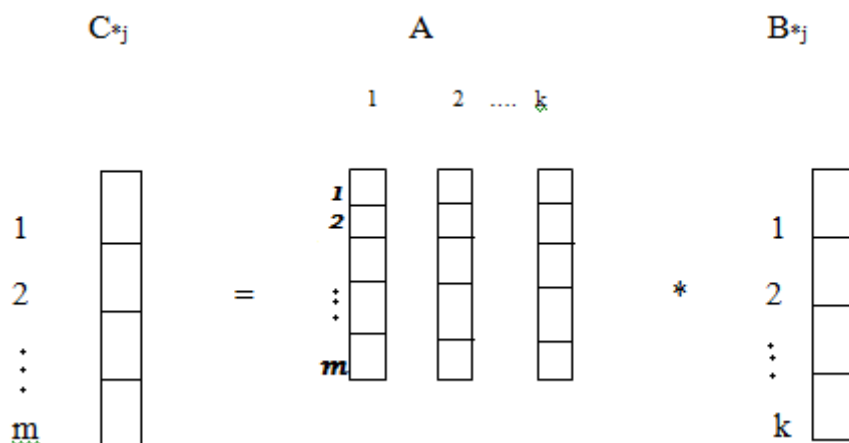
$$C_{*j} = A * B_{*j}$$

με ολόκληρο τον πίνακα A διαχωρισμένο στις k στήλες του και την j -στη στήλη του B_{*j} διαχωρισμένη στα k στοιχεία της:

$$C_{*j} = [A_{*1} \ A_{*2} \dots A_{*k}] * [b_{1j} \ b_{2j} \dots b_{kj}]^T$$

Παρατηρούμε ότι το μη μηδενικό πρότυπο της j -στης στήλης του C (δηλαδή τα μη μηδενικά στοιχεία της στήλης) προκύπτει από τα μη μηδενικά στοιχεία της j -στης στήλης του A που πολλαπλασιάζονται με τα αντίστοιχα στοιχεία του διανύσματος στήλης του B , για τα οποία ισχύει ότι $b_{ij} \neq 0$, για κάθε $i \in [0, k-1]$. Ολόκληρος ο πίνακας C θα αποτελείται από πλήθος γραμμών ίσο με το πλήθος γραμμών του πίνακα A (m) και από πλήθος στηλών ίσο με αυτό του πίνακα B (n).

Σχηματικά:



1.5 Πρόσθεση Πινάκων

Έστω, $C = \alpha A + \beta B$ η πρόσθεση μεταξύ δύο αραιών πινάκων A και B διαστάσεων $m \times n$. Η πράξη αυτή μοιάζει με τον πολλαπλασιασμό, γιατί υπολογίζουμε μία στήλη την φορά από το τελικό αποτέλεσμα. Πιο συγκεκριμένα, γράφουμε την αρχική σχέση της πρόσθεσης ως:

$$C = [A \ B] * [\alpha I \ \beta I]^T$$

όπου I ο ταυτοτικός πίνακας κατάλληλου μεγέθους, με μηδέν σε όλα τα στοιχεία $I_{ij} = 0$, $i \neq j$ και ένα στην διαγώνιο $I_{ij} = 1$, $i = j$.

Σε κάθε βήμα των υπολογισμών παίρνουμε μία στήλη του πίνακα A και μία στήλη του πίνακα B , της προσθέτουμε και φτιάχνουμε μία στήλη του πίνακα C .

1.6 Πίνακας Μετάθεσης

Ένας πίνακας P διαστάσεων $n \times n$ μπορεί να αναπαρασταθεί ως αραιός πίνακας P , με έναν άσσο σε κάθε γραμμή και στήλη και με όλες τις υπόλοιπες τιμές μηδέν. Ο αραιός πίνακας P είναι ορθογώνιος ^[1] γιατί οι γραμμές και οι στήλες του είναι μοναδιαία διανύσματα, οπότε ισχύει ότι:

$$P^T = P^{-1}.$$

Η μετάθεση είναι χρήσιμη στην παραγοντοποίηση , όπως θα δούμε παρακάτω. Αναλόγως, από ποια πλευρά πολλαπλασιάζουμε έναν πίνακα A με τον πίνακα μετάθεσης μεταθέτουμε τις γραμμές ή τις στήλες του:

- PA : μεταθέτει τις γραμμές του πίνακα A , ο πίνακας P έχει διατεταγμένες τις γραμμές του ,δηλαδή ο άσος της θέσης $P[i][j]$ θα μεταθέσει την γραμμή j του πίνακα A στην θέση γραμμής i .
- AQ : μεταθέτει τις στήλες του πίνακα A , ο πίνακας Q έχει διατεταγμένες τις στήλες του, δηλαδή ο άσος της θέσης $Q[i][j]$ θα μεταθέσει την στήλη i του πίνακα A στην θέση στήλης j .

Η συμμετρική μετάθεση ενός πίνακα, ουσιαστικά πραγματοποιείται με τον πολλαπλασιασμό του πίνακα A με τον ίδιο πίνακα μετάθεσης δεξιά και αριστερά:

$$C = PAP^T$$

Στο κεφάλαιο που ακολουθεί θα δούμε βασικές μεθόδους μετάθεσης ενός αραιού πίνακα που θα μας διευκολύνουν στην μετέπειτα παραγοντοποίηση του.

2.

Διατάξεις Fill – Reducing

Στο κεφάλαιο αυτό, παρουσιάζουμε τρεις μεθόδους μετάθεσης που αντιμετωπίζουν την ελαχιστοποίηση του πλήθους των μη μηδενικών εγγραφών που εμφανίζονται κατά την διάρκεια της παραγοντοποίησης ενός μετατεθειμένου αραιού πίνακα $A^{[3]}$. Όπως, θα δούμε στα κεφάλαια που ακολουθούν μία διάταξη fill – reducing αποτελεί βασικό βήμα πριν την εκτέλεση των Cholesky και LU παραγοντοποιήσεων.

2.1 Διάταξη Ελαχίστου Βαθμού (Minimum Degree)

Η πιο ευρείας χρήσης ευρετική και άπληστη μέθοδος για την εύρεση ενός πίνακα μετάθεσης P , τέτοιου ώστε η PAP^T να έχει όσο το δυνατόν λιγότερα μη μηδενικά στοιχεία στην παραγοντοποίηση του αραιού πίνακα A .

Έστω ότι έχουμε τον αραιό συμμετρικό πίνακα A διαστάσεων $n \times n$, με μη μηδενικά στοιχεία στην διαγώνιο και το αντίστοιχο μη κατευθυνόμενο γράφημα $G(V,E)$, με κόμβους $V=\{1,...,n\}$ και ακμές $(i,j) \in E$, όταν $A[i][j] \neq 0$.

Στο k -στο βήμα της απαλοιφής του Gauss όταν ένας κόμβος απαλείφεται από το γράφημα $G^{[k]}$ (όπου είναι το γράφημα G με τις αλλαγές που έχει υποστεί μέχρι το βήμα k) μεταξύ των γειτόνων του προστίθενται ακμές. Ο αντίστοιχος πίνακας $A^{[k]}$ είναι ο υποπίνακας που περιέχει τις νέες ακμές που δημιουργήθηκαν μαζί με τις παλιές. Ωστόσο, τον υποπίνακα $A^{[k]}$ μπορούμε να τον αναπαραστήσουμε ως $G^{[k]}$ γράφημα πηλίκου (quotient graph) όπου στο k -στο βήμα της απαλοιφής, με

διάταξη ελαχίστου βαθμού ενός κόμβου, δημιουργείται μια έμμεση κλίκα από τους γείτονες του.

Πριν προχωρήσουμε στην λειτουργία της μεθόδου, αναφέρουμε ότι σε ένα γράφημα πηλίκου δύο κόμβοι i, j είναι είτε άμεσοι γείτονες, δηλαδή υπάρχει ακμή που τους ενώνει (αντίστοιχη τιμή στον πίνακα $a[i][j] \neq 0$) είτε έμμεσοι γείτονες, γιατί βρίσκονται στην ίδια κλίκα που δημιουργεί κάποιο κοινό τους αντικείμενο e_k (με το οποίο συνδέονται άμεσα).

Κατά την εκτέλεση της διάταξης ελαχίστου βαθμού:

- επιλέγουμε προς απαλοιφή τον κόμβο k με τον μικρότερο βαθμό και τον μετατρέπουμε σε αντικείμενο (element) e_k , που αντιπροσωπεύει μια συλλογή από πυκνούς πίνακες ή κλίκες σε ένα γράφημα.
- Όπου χρειαστεί, προσθέτουμε ακμές στο γράφημα, για να απεικονίσουμε την κλίκα, μεταξύ του αντικειμένου e_k και κάποιου έμμεσου γείτονα του. Απαλείφουμε τις ήδη υπάρχουσες ακμές μεταξύ κόμβων που βρίσκονται στην ίδια νέα κλίκα, γιατί είναι περιττές λόγω τις πλέον έμμεσης γειτνίασης των κόμβων.
- Αν ένα αντικείμενο e_k είναι υποσύνολο κάποιου άλλου αντικειμένου e_j ($e_k \subseteq e_j$), τότε απαλείφουμε το πρώτο e_k . Η λειτουργία αυτή ονομάζεται απορρόφηση στοιχείου (element absorption).
- Αν ένας κόμβος i μείνει με μία ακμή μόνο προς ένα αντικείμενο e_k , δηλαδή: $\mathcal{E}_i = \{k\}$, $A_i = \{\}$, όπου \mathcal{E}_i το σύνολο των αντικειμένων που ο i είναι γείτονας και A_i είναι το σύνολο των άμεσων γειτόνων, τότε απαλείφεται άμεσα και η διαδικασία ονομάζεται μαζική απαλοιφή (mass elimination).

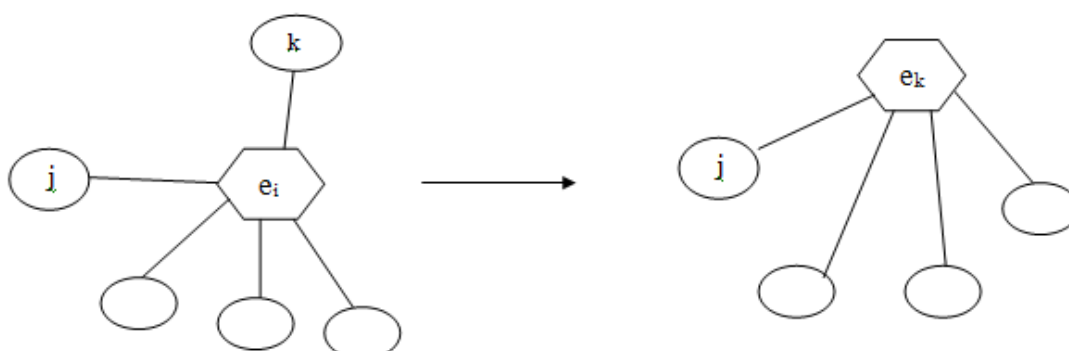
Στην τελική ανάλυση μία ακμή μπορεί να υπάρξει μεταξύ ενός κόμβου j και ενός αντικειμένου e_k , επειδή:

- είτε οι j και k ήταν εξ αρχής άμεσα συνδεδεμένοι, πριν ο δεύτερος γίνει το αντικείμενο e_k , όπως φαίνεται στο Σχήμα 2.1:



Σχήμα 2.1 Άμεσα συνδεδεμένοι κόμβοι

- είτε ήταν έμμεσα συνδεδεμένοι στην κλίκα ενός άλλου αντικειμένου, Σχήμα 2.2:



Σχήμα 2.2 Άμεσα έμμεσα συνδεδεμένοι κόμβοι

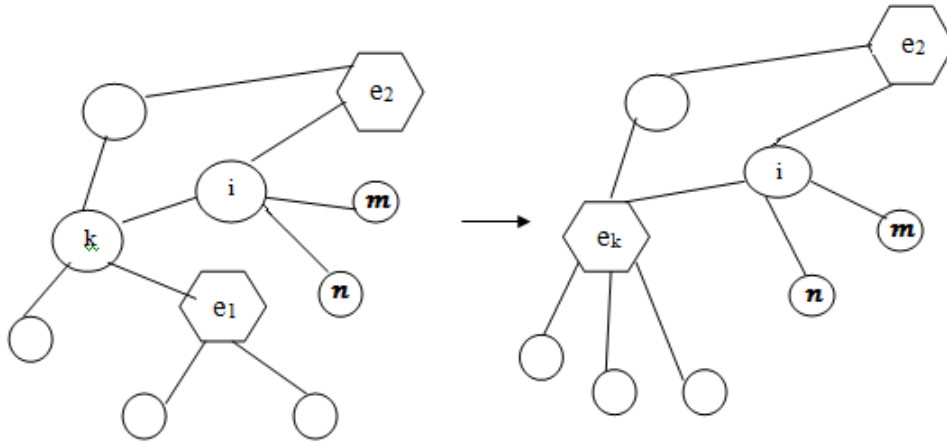
Χρειάζεται να προσέξουμε αρκετά τους βαθμούς των κόμβων που γίνονται αντικείμενα, όπως ο κόμβος k . Γενικότερα, για έναν κόμβο k , η λίστα γειτνίασης L_k αποτελείται από τον συνδυασμό: του συνόλου όλων των αντικειμένων που συμμετέχει στις κλίκες τους L_e και των άμεσων γειτόνων του A_k :

$$L_k = (\cup L_e) \cup A_k, \quad e \in \mathcal{E}_k \quad (2.1)$$

καθώς, ο βαθμός του κόμβου είναι:

$$d_k = |L_k| \leq |(\sum L_e)| + |A_k| \quad (2.2)$$

Επίσης, όταν ένα αντικείμενο e_k δημιουργείται πρέπει να επαναυπολογιστούν οι βαθμοί όλων των κόμβων που συμμετέχουν στην κλίκα του ($i \in L_k$, i είναι κόμβος) και όχι όλων των κόμβων του γραφήματος. Στο σχήμα που ακολουθεί, δίνεται ένα παράδειγμα:



Ο κόμβος k απαλείφεται και γίνεται το αντικείμενο e_k . Για τον κόμβο i :

$$\mathcal{E}_i = \{e_2\} \text{ μετατρέπεται σε } \mathcal{E}_i = \{e_k, e_2\}$$

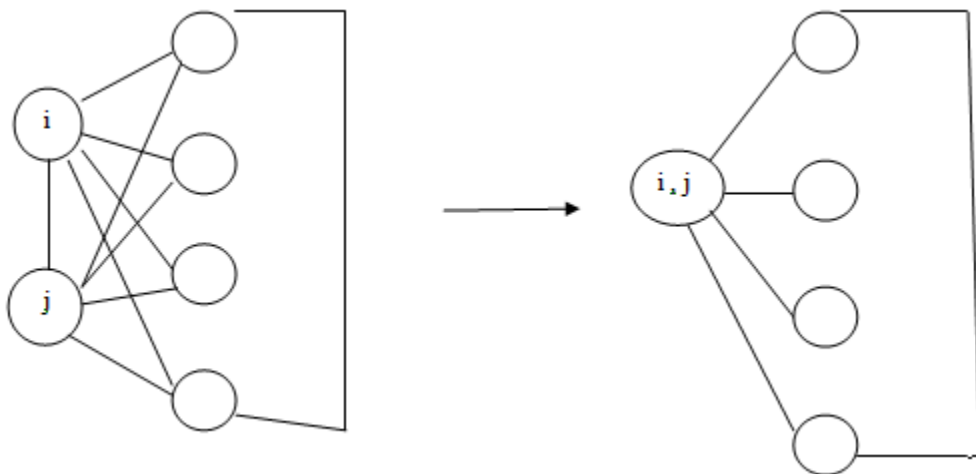
$$A_i = \{k, n, m\} \text{ μετατρέπεται σε } A_i = \{n, m\}.$$

Οι κόμβοι χωρίς όνομα, στο σχήμα, δεν μας απασχολούν. Παρατηρούμε ότι οι ακμές του γραφήματος δεν αυξάνονται παραπάνω από τις ακμές του αρχικού πίνακα A , δηλαδή το G δεν υπερβαίνει το $|A|$, γιατί ο κόμβος i πριν ο κόμβος k απαλειφτεί είχε συνολικά $d_i = 4$ ακμές και μετά την απαλοιφή παρέμεινε το ίδιο. Ο k είχε πριν την απαλοιφή $d_k = 4$ και μετά $d_k = 5$, αυξήθηκε το πλήθος ακμών του κατά ένα, γιατί η ακμή του προς τον e_1 κλαδεύτηκε και στην θέση της μπήκαν οι υπάρχουσες ακμές της κλίκας του e_1 , δηλαδή οι υπάρχουσες ακμές παρέμειναν κρατώντας το $|A|$ σταθερό και ίσως μικρότερο (λόγω απαλοιφής κάποιων ακμών).

Έστω οι δύο κόμβοι i και j οι οποίοι μπορεί να είναι άμεσοι ή έμμεσοι γείτονες ή να μην είναι, τους ονομάζουμε πανομοιότυπους κόμβους (indistinguishable nodes) όταν ισχύει ότι:

$$\mathcal{E}_i = \mathcal{E}_j \text{ και } A_i = A_j, \quad (2.3)$$

και έχουν την εξής ιδιότητα: όταν ένας από τους δύο γίνει ελαχίστου βαθμού, είναι σίγουρο πως θα είναι και ο άλλος, αφού $L_i = L_j$ λόγω των (2.1) και (2.3), επομένως μπορούμε να επιλέξουμε προς απαλοιφή τον ένα εκ των δύο, να δημιουργήσουμε την αντίστοιχη κλίκα, να απαλείψουμε τυχόν πλεονάζουσες ακμές και αντικείμενα και έπειτα να απαλείψουμε και τον πανομοιότυπο του χωρίς να εκτελέσουμε κάποια παραπάνω ενέργεια. Οι πανομοιότυποι κόμβοι μπορεί να είναι τουλάχιστον δύο και μπορούμε να τους συγχωνεύσουμε σε έναν υπερκόμβο (supernode) που τους αναπαριστά και δημιουργεί μία εσωτερική κλίκα, μιας και μεταξύ τους συνδέονται κατά ζεύγη, ώστε να πάρουμε μικρότερο γράφημα.



Σχήμα 2.3 Πανομοιότυποι κόμβοι

Οι κόμβοι στην αρχή της διαδικασίας ξεκινούν με την απλή αναπαράσταση του εαυτού τους, με την διάταξη ελαχίστου βαθμού επιλέγουμε σε κάθε βήμα έναν κόμβο ή υπερκόμβο k με τον ελάχιστο βαθμό, που αντιπροσωπεύει την αραιότερη γραμμή και στήλη του A .

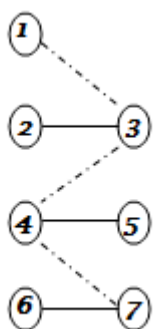
2.2 Διάταξη Μεγίστου Βαθμού (Maximum Degree)

Πριν προχωρήσουμε στην επεξήγηση της λειτουργίας της μεθόδου θα αναφερθούμε σε κάποιους βοηθητικούς όρους της γραμμικής άλγεβρας. Ο βαθμός ενός πίνακα A είναι το μέγιστο μέγεθος των υποσυνόλων των στηλών του που είναι γραμμικά ανεξάρτητες. Ένας πίνακας A διαστάσεων $n \times n$ ονομάζεται ιδιάζων (δηλαδή δεν έχει αντίστροφο) αν η τάξη του k είναι μικρότερη από το n , ενώ ένας πίνακας διαστάσεων $m \times n$ είναι ελλιπούς τάξης, αν η τάξη του είναι μικρότερη από τον μικρότερο εκ των m, n ($k < \min(m, n)$) αλλιώς, ονομάζεται πλήρους τάξης.

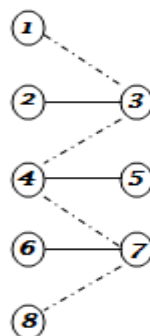
Μέγιστο ταίριασμα (ή διάσχιση) είναι η μετάθεση ενός πίνακα A , έτσι ώστε η k -στη διαγώνια εγγραφή του να είναι μη μηδενική και $|k|$ να ελαχιστοποιείται μοναδικά, εκτός της περίπτωσης που ο πίνακας είναι ολόκληρος μηδέν. Αυτή η μετάθεση καθορίζει την δομική τάξη του πίνακα A . Συγκεκριμένα, ο πίνακας έχει δομική τάξη αν και μόνο αν $k = 0$, αλλιώς είναι ανεπαρκούς δομικής τάξης. Ο αριθμός των εγγραφών στην διαγώνιο μας δίνει την δομική τάξη του πίνακα A και είναι το άνω όριο της αριθμητικής τάξης κάθε πίνακα με το ίδιο μη μηδενικό πρότυπο.

Για το αντίστοιχο γράφημα $G(V, E)$ του πίνακα A με V το σύνολο κορυφών και E το σύνολο ακμών, έχουμε μέγιστο ταίριασμα αν δεν υπάρχει ταίριασμα μεγαλύτερο του πλήθους των ακμών. Έστω, ένα ταίριασμα M στο γράφημα G , εναλλακτικό (alternating) μονοπάτι του M ονομάζουμε εκείνο το μονοπάτι που οι ακμές του εναλλάσσονται μεταξύ ταιριασμένων και αταίριαστων ακμών. Ένα εναλλακτικό μονοπάτι με άκρα ελεύθερες κορυφές ονομάζεται επαυξητικό (augmenting) μονοπάτι για το M . Ένα εναλλακτικό μονοπάτι απεικονίζεται στο Σχήμα 2.4, όπου με διακεκομμένες ακμές συμβολίζονται οι αταίριαστες ακμές και με συνεχείς ακμές οι

ταιριασμένες. Στο Σχήμα 2.5 απεικονίζεται ένα επαυξητικό μονοπάτι μεταξύ των ελεύθερων κορυφών 1 και 8.



Σχήμα 2.4 Εναλλασσόμενο μονοπάτι



Σχήμα 2.5 Επαυξητικό Μονοπάτι

Έστω, ο πίνακας A μεγέθους $m \times n$ και το αντίστοιχο διμερές γράφημα $G(V,E)$, με m κόμβους γραμμής και n κόμβους στήλης και μη κατευθυνόμενες ακμές $E = \{ (i,j) \mid A[i][j] \neq 0 \}$, κάθε μία από αυτές συνδέει μόνο ζεύγη γραμμής και στήλης. Ωστόσο, να σημειώσουμε ότι η ακμή (i,j) δεν είναι όμοια με την (j,i) . Επίσης, έστω ότι ο A_j είναι το μη μηδενικό πρότυπο της j -στης στήλης, δηλαδή αποτελείται από τις γραμμές που είναι γείτονες στο γράφημα G .

Επιπρόσθετα, έστω ένα σύνολο γραμμών R και ένα σύνολο στηλών C . Ταίριασμα, ονομάζουμε κάθε ζεύγος που αποτελείται από μία γραμμή $i \in R$, μία στήλη $j \in C$ και $(i,j) \in E$, οι οποίες ονομάζονται ταιριασμένες. Οι ταιριασμένες ακμές που ανήκουν στο E είναι τα αντίστοιχα διαγώνια στοιχεία του μετατεθειμένου πίνακα. Όσες ακμές, στήλες και γραμμές δεν ανήκουν σε ζεύγος(ταίριασμα) ονομάζονται αταίριαστες. Το ταίριασμα είναι τέλειας γραμμής ή τέλειας στήλης αν όλες οι γραμμές ή στήλες είναι ταιριασμένες, αντίστοιχα. Το μέγιστο ταίριασμα του G είναι μεγαλύτερο ή ίσο με οποιοδήποτε ταίριασμα στο G , το οποίο μπορεί να έχει πολλά μέγιστα ταιριάσματα ($n!$ για τετραγωνικό πίνακα).

Η διάταξη μεγίστου βαθμού λειτουργεί ως εξής:

- επεκτείνουμε ,κάθε φορά, το σύνολο ταιριασμένων στηλών C κατά μια στήλη $j \notin C$, με την βοήθεια ενός επαυξητικού μονοπατιού.

Κάθε στήλη φέρει μία λίστα γειτνίασης, η οποία προέρχεται από τον πίνακα A .

- Στο επαυξητικό μονοπάτι, έχοντας ξεκινήσει από την αταίριαστη στήλη j διαπερνάμε μία ακμή προς μία γειτονική γραμμή $i1$. Εφόσον, η j είναι αταίριαστη η $(j,i1)$ είναι και αυτή.
- Αν η $i1$ είναι ταιριασμένη έστω με την στήλη $j1$, τότε διαπερνάμε την ταιριασμένη ακμή $(i1, j1)$ και προχωράμε στην επόμενη γραμμή $i2$ μέσω της αταίριαστης ακμής $(j2,i1)$ (αταίριαστη γιατί δεν είναι ζευγάρι) και συνεχίζουμε.
- Αλλιώς, σταματάμε (η $i1$ είναι αταίριαστη) και προσθέτουμε την στήλη j στο C και την γραμμή $i1$ στο R .
- Επαναλαμβάνουμε για την επόμενη αταίριαστη στήλη.
- Αν δεν βρεθεί αταίριαστη γραμμή, τότε δεν υπάρχει αντίστοιχο μονοπάτι και δεν επεκτείνεται το ταίριασμα. (Αυτό συμβαίνει συνήθως όταν ο A είναι δομικά ανεπαρκούς τάξης). Έτσι, παίρνουμε έναν από τους ήδη ταιριασμένους και τον κάνουμε ταίρι του j , τροποποιώντας τους ταιριασμένους κόμβους και ψάχνοντας τώρα το ταίρι του κόμβου που εγκαταλείφθηκε για χάρη του j , επαναλαμβάνοντας την διαδικασία.

Στο σημείο αυτό, αξίζει να σημειώσουμε κάποιες σημαντικές παρατηρήσεις :

- το μονοπάτι μπορεί να είναι οσοδήποτε μεγέθους και κανένας κόμβος ή ακμή δεν μπορεί να εμφανιστεί πάνω από μία φορά σε αυτό.

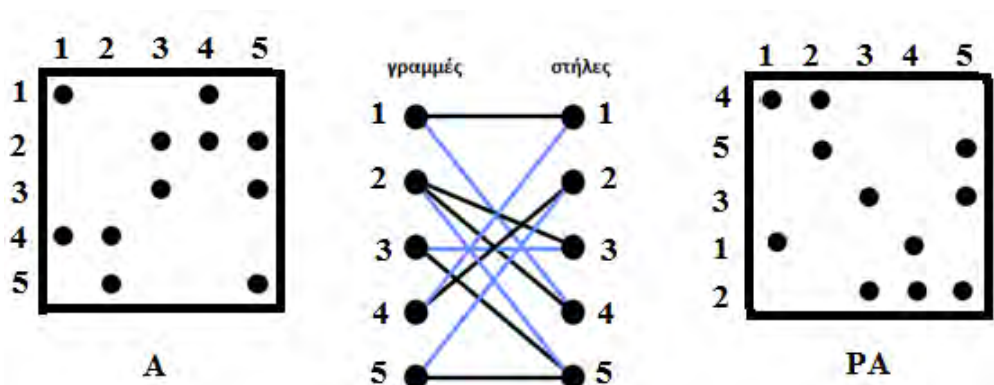
- Κάθε ταιριασμένη στήλη ή γραμμή παραμένει ταιριασμένη και δεν μπορεί να μετατραπεί σε αταίριαστη. Ωστόσο, το αντίστοιχο στοιχείο που είναι ταιριασμένη μπορεί να αλλάξει κατά το πέρας της μεθόδου.
- Η λίστα γειτνίασης κάθε κόμβου στήλης διαπερνάται προς μία κατεύθυνση μέχρι να βρεθεί αταίριαστη γραμμή. Ουσιαστικά, το πρώτο μέρος της είναι γραμμές ήδη ταιριασμένες που έχουμε ήδη προσπελάσει και γραμμές που δεν γνωρίζουμε ακόμα τι είναι.
- Βρίσκουμε το εναλλακτικό επαυξητικό μονοπάτι από την αταίριαστη στήλη j πραγματοποιώντας την κατά βάθος αναζήτηση στην λίστα γειτνίασης (depth first search, dfs¹). Αν χρειαστεί να διαπεράσουμε ολόκληρο το γράφημα σε κάθε βήμα ο χρόνος που απαιτείται είναι $O(|A|n)$.
- Για να μειώσουμε το χρόνο, μπορούμε να τροποποιήσουμε τον αλγόριθμο πραγματοποιώντας κατά πλάτος αναζήτηση (breadth first search) σε κάθε στήλη j , πριν προχωρήσουμε σε κατά βάθος αναζήτηση. Συγκεκριμένα, χωρίζουμε το A_j σε δύο μέρη, όπου το πρώτο περιέχει τις ταιριασμένες και ελέγχουμε για αταίριαστη γραμμή στο δεύτερο επεκτείνοντας το πρώτο κομμάτι του A_j όσο δεν βρίσκουμε κάποια. Με αυτό τον τρόπο απαιτείται χρόνος $O(|A|)$ για την εύρεση ολόκληρου του μεγίστου ταιριάσματος.

Ακολουθεί χαρακτηριστικό παράδειγμα:

Έστω ο πίνακας A , με μηδενικά στην διαγώνιο και το αντίστοιχο διμερές γράφημα στο οποίο με μπλε γραμμές φαίνονται τα τελικά ζεύγη

¹ Ο αλγόριθμος (DFS – Depth first search) ^[15] επιτυγχάνει διάσχιση ή αναζήτηση σε δέντρο ή γράφημα. Η διάσχιση ξεκινά από τη ρίζα και εξερευνά όσο το δυνατόν περισσότερο κατά μήκος κάθε κλαδί του δέντρου μέχρι να φτάσει σε αδιέξοδο.

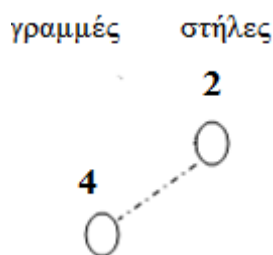
και με μαύρες οι ακμές που υπάρχουν λόγω μη μηδενικών εγγραφών στον πίνακα A ^[4].



Σχήμα 2.6 Πίνακας, Διμερές Γράφημα, Τελικός πίνακας

Όπως, μπορούμε να δούμε και από τον πίνακα αρχικά έχουμε τα ζεύγη (1,1) , (3,3) ,(5,5).Εφόσον είναι ταιριασμένα θα παραμείνουν και ας αλλάξουν ταίρι το κάθε ένα από τα επιμέρους στοιχεία.

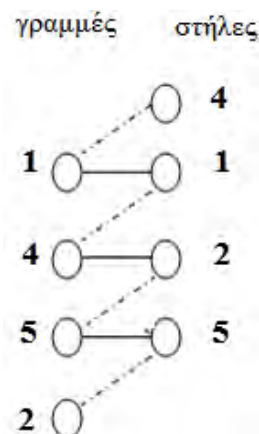
- Για την στήλη 2 που είναι αταίριαστη: προσπελάνουμε την λίστα γειτνίασης μέσω του διμερούς γραφήματος : $2 \rightarrow 4$, με ακμή (2,4). Άρα τα ταιριασμένα ζεύγη είναι τα (1,1) ,(3,3), (4,2), (5,5),καθώς προσθέτουμε τα 2 και 4 στα σύνολα C και R , αντίστοιχα. Οπότε μένει να ταιριάζουμε την στήλη 4.



Σχήμα 2.7 Επαυξητικό Μονοπάτι από την στήλη 2

- Για την στήλη 4: προσπελάνουμε την λίστα γειτνίασης, ομοίως: με εναλλασσόμενο και επαυξητικό μονοπάτι (4,1) , (1,1) ,(1,4), (4,2), (2,5), (5,5),(5,2) , όπου με σκούρο μαύρο δηλώνονται οι

ταιριασμένες ακμές. Έτσι, για να επεκτείνουμε το ταίριασμα, η στήλη 4 προστίθεται στο σύνολο C και η γραμμή 2 στο R., δημιουργώντας τα ζεύγη: (1,4), (4,1), (5,2),(2,5) και το μονοπάτι γίνεται : **(4,1)** , (1,1) ,**(1,4)**, (4,2) , **(2,5)**, (5,5) ,**(5,2)** .



Σχήμα 2.8 Επαυξητικό Μονοπάτι από την στήλη 4

- Μεταθέτοντας τις γραμμές του A ώστε το διαγώνιο στοιχείο να αντιπροσωπεύει ταιριασμένο ζεύγος γραμμής και στήλης παίρνουμε το πίνακα PA του Σχήματος 2.6.

2.3 Διάταξη Μορφής Τριγωνικού Block (Block Triangular Form)

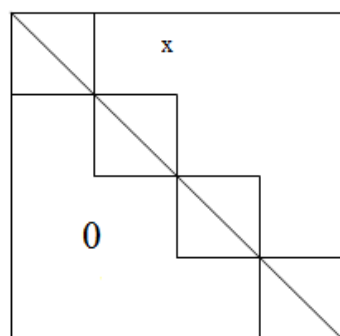
Αρχικά, θα εξηγήσουμε κάποιες σημαντικές έννοιες που θα μας βοηθήσουν στην κατανόηση της μεθόδου. Έστω ένας πίνακας A μεγέθους $m \times n$, θεωρούμε ότι αποτελείται από την ισχυρή ιδιότητα Hall αν κάθε σύνολο k στηλών του έχει μη μηδενικές εγγραφές σε τουλάχιστον k γραμμές για κάθε k μεταξύ 1 έως n . Ισοδύναμα, αν και μόνο αν ο πίνακας A έχει στήλη πλήρους δομικής τάξης.

Η μορφή τριγωνικού block (block triangular form) είναι μία μετάθεση ενός πίνακα A, ώστε η διαγώνιος του να μην αποτελείται από

μηδενικά καθώς ο ίδιος μετατρέπεται σε άνω ή κάτω τριγωνικό πίνακα. Συγκεκριμένα, έστω ο τετραγωνικός πλήρους δομικής τάξης πίνακας A , που δεν αποτελείται από την ισχυρή ιδιότητα Hall. Μπορούμε να τον μεταθέσουμε σε μορφή τριγωνικού block , με το κάθε διαγώνιο block να είναι τετραγωνικό, χωρίς μηδέν στην διαγώνιο και με την ισχυρή ιδιότητα Hall. Η διάταξη αυτή είναι μοναδική, παρόλο που τα διαγώνια block μπορούν να ανταλλάσσονται με την προϋπόθεση ότι η διαγώνιος θα είναι πάντα μη μηδενική.

Η μετάθεση ενός τετραγωνικού πίνακα A με μηδενικά στοιχεία στην διαγώνιο, σε block τριγωνικής μορφής ισοδυναμεί με την εύρεση των ισχυρά συνεκτικών συνιστωσών ενός κατευθυνόμενου γραφήματος. Το αντίστοιχο κατευθυνόμενο γράφημα $G(V,E)$, με κορυφές $V = \{1, \dots, n\}$ και ακμές $E = \{(i,j) \mid A[i][j] \neq 0\}$ είναι ο αντίστοιχος πίνακας γειτνίασης του μη μηδενικό προτύπου του πίνακα A . Η ισχυρά συνεκτική συνιστώσα είναι το μέγιστο σύνολο κόμβων στο οποίο ισχύει ότι για κάθε ζεύγος κόμβων i, j που ανήκουν στην συνιστώσα υπάρχει στο γράφημα μονοπάτι από την i προς την j και αντίστροφα. Μπορούμε να βρούμε με πολλούς τρόπους τις ισχυρά συνεκτικές συνιστώσες ενός γραφήματος.

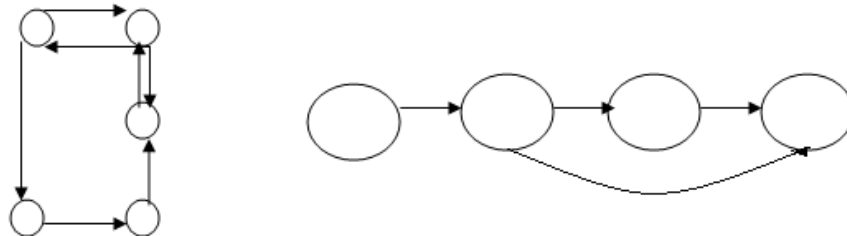
Μορφή Τριγωνικού block



ισχυρά συνεκτική συνιστώσα

Σχήμα 2.9 Μορφή Πίνακα και Τριγωνικού block

Στο Σχήμα 2.9 απεικονίζεται η δομή του πίνακα με τις ισχυρά συνδεδεμένες συνιστώσες στην διαγώνιο. Στο Σχήμα 2.10 απεικονίζεται μία ισχυρά συνεκτική συνιστώσα (δεξιά) και το γράφημα μεταξύ των ισχυρά συνεκτικών συνιστωσών(αριστερά).



Σχήμα 2.10 Δομή Ισχυρά Συνεκτικής Συνιστώσας και Γράφημα Ισχυρά συνεκτικών συνιστωσών

Παρατηρούμε ότι :

- Μία ισχυρά συνεκτική συνιστώσα, λόγω της ιδιότητας της να έχει μονοπάτι για κάθε ζεύγος κόμβων που επιστρέφει από εκεί που ξεκίνησε δεν μπορεί να μετακινηθεί σε κάποια άλλη .
- Η μετάβαση μεταξύ συνεκτικών συνιστωσών πραγματοποιείται μόνο μέσω των μη μηδενικών εγγραφών του πίνακα A , όπως στο Σχήμα 2.9 όπου συμβολίζεται με x μία ση μηδενική εγγραφή .
- Οι ακμές στο γράφημα των συνεκτικών συνιστωσών δείχνουν προς μία κατεύθυνση , από αριστερά προς τα δεξιά και αυτό γιατί κάθε μη μηδενική εγγραφή x στον πίνακα A δεν είναι συμμετρική. Έτσι, το γράφημα είναι άκυκλο.
- Για αυτό το λόγο και ο πίνακας A μετατρέπεται σε άνω ή κάτω τριγωνικός.

Για την διάταξη μορφής τριγωνικού block, εκτελούμε τα εξής βήματα:

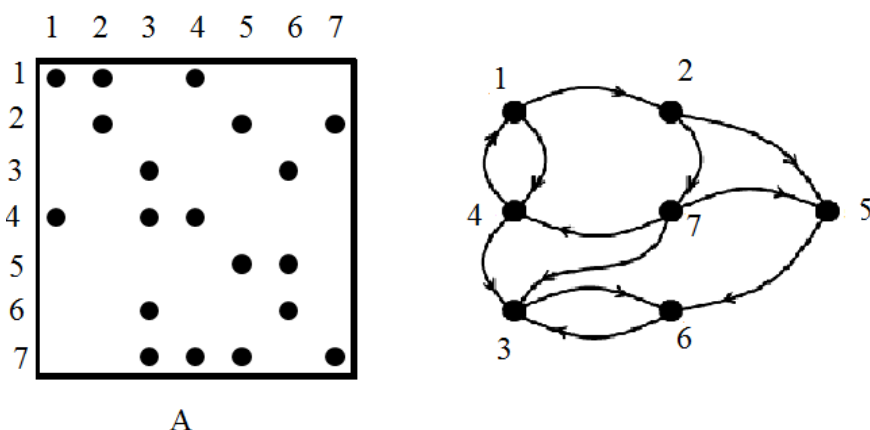
- Αρχικά, μεταθέτουμε τον πίνακα έτσι ώστε η διαγώνιός του να μην περιέχει μηδενικά. Μπορούμε να τον μοντελοποιήσουμε ως διμερές γράφημα και να εφαρμόσουμε διάταξη μεγίστου βαθμού.

- Βρίσκουμε τις ισχυρά συνεκτικές συνιστώσες πραγματοποιώντας δύο αναζητήσεις κατά βάθος:
 - με την πρώτη στο γράφημα $G(A)$, η οποία επιστρέφει το σύνολο X όλων των κόμβων του γραφήματος , με την σειρά στην οποία εμφανίζονται οι κόμβοι. Συγκεκριμένα, ένας κόμβος j τοποθετείται στο X μόλις τελειώσει η κατά βάθος αναζήτηση που ξεκίνησε από τον ίδιο.
 - Η δεύτερη αναζήτηση κατά βάθος διασχίζει το $G(A^T)$, με τους κόμβους να βρίσκονται στην αντίστροφη σειρά με την οποία είναι στο σύνολο X , δημιουργώντας τις ισχυρά συνεκτικές συνιστώσες. Κάθε νέος κόμβος i και όλοι οι προσπελάσιμοι από αυτόν στο $G(A^T)$ ορίζουν μία ισχυρά συνεκτική συνιστώσα στο $G(A)$.

Έχοντας αποθηκεύσει τον αραιό πίνακα A σε μορφή συμπιεσμένης στήλης, το μη μηδενικό πρότυπο A_j περιέχει την λίστα γειτνίασης του κόμβου j στο $G(A^T)$.

Ακολουθεί χαρακτηριστικό παράδειγμα:

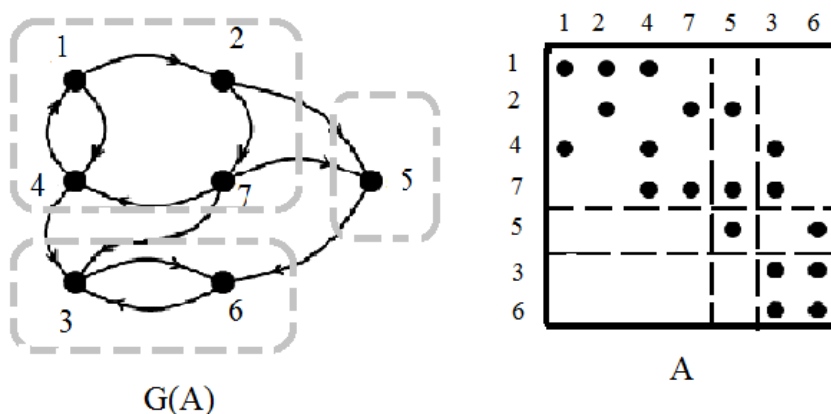
Έστω ο πίνακας $A^{[4]}$, του Σχήματος 2.11 με μη μηδενικά στοιχεία στην διαγώνιο , ώστε να αποφύγουμε τον υπολογισμό μεγίστου βαθμού και το αντίστοιχο κατευθυνόμενο γράφημα $G(A)$.



Σχήμα 2.11 Αραιός Πίνακας και γράφημα $G(A)$

- Πραγματοποιούμε αναζήτηση κατά βάθος στο γράφημα $G(A)$, η οποία μας επιστρέφει το σύνολο $X = \{ 6,3,5,4,7,2,1 \}$.
- Έπειτα εκτελούμε την δεύτερη αναζήτηση σε βάθος με την αντίστροφη σειρά που έχουν τα στοιχεία στο σύνολο X και μας επιστρέφονται οι ισχυρές συνεκτικές συνιστώσες $\{1,2,7,4\}$, $\{5\}$, $\{3,6\}$.

Ως αποτέλεσμα έχουμε τον άνω τριγωνικό πίνακα A που φαίνεται στο σχήμα 2.15, μαζί με τις συνεκτικές συνιστώσες στο γράφημα $G(A)$.



Σχήμα 2.12 Ισχυρά Συνεκτικές Συνιστώσες και Τελικός πίνακας A

3.

Συμβολική Ανάλυση

Η επίλυση του αραιού γραμμικού συστήματος $Ax = b$, με τον αραιό πίνακα A περνά από φάσεις επεξεργασίας, ώστε να είναι αποδοτικότερη. Έχοντας μεταθέσει τον αραιό αρχικό πίνακα με κάποια από τις μεθόδους fill-reducing, εφαρμόζουμε την συμβολική ανάλυση στον παράγοντα L που θα παραγοντοποιήσει αριθμητικά τον A , μέσω κάποιας από τις μεθόδους Cholesky ή LU. Στο κεφάλαιο αυτό θα περιγράψουμε την φάση της συμβολικής ανάλυσης ^[5], η οποία περιλαμβάνει υπολογισμούς που εξαρτώνται τυπικά από το μη μηδενικό πρότυπο του πίνακα και όχι από τις αριθμητικές του τιμές, για την διεξαγωγή μιας ρητής αναπαράστασης του μη μηδενικού προτύπου της παραγοντοποίησης. Θα περιγράψουμε την εύρεση του δένδρου απαλοιφής (elimination tree), τον υπολογισμό της μεταδιατεταγμένης (post-ordering) διάσχισης του και του πλήθους των μη μηδενικών σε κάθε στήλη (column count) και γραμμή (row count) του πίνακα L .

3.1 Δένδρο Απαλοιφής

Έστω το σύστημα παραγοντοποίησης $LL^T = A$, με τον 2x2 block διαχωρισμό των γραμμών και στηλών των πινάκων:

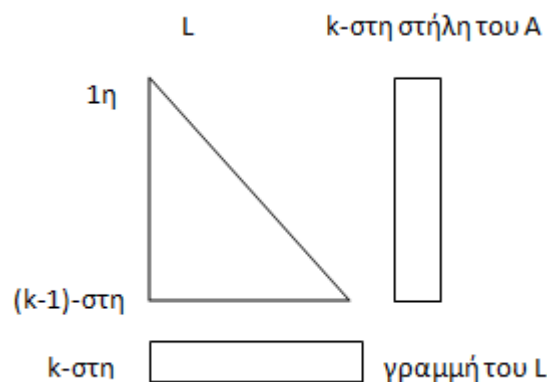
$$\begin{bmatrix} L_{11} & 0 \\ l_{12}^T & l_{22} \end{bmatrix} \begin{bmatrix} L_{11}^T & l_{12} \\ 0 & l_{22} \end{bmatrix} = \begin{bmatrix} A_{11} & a_{12} \\ a_{12}^T & a_{22} \end{bmatrix}$$

με L_{11} , A_{11} είναι πίνακες διαστάσεων $(n-1) \times (n-1)$, τα a_{12} και a_{12}^T είναι

διανύσματα στήλης και γραμμής αντίστοιχα και ο a_{22} είναι βαθμωτό μέγεθος. Στο σημείο αυτό αναφέρουμε πως με πεζά θα δηλώνουμε τις εγγραφές - στοιχεία ενός πίνακα με τους αντίστοιχους δείκτες γραμμής και στήλης και με κεφαλαία υποπίνακες των αρχικών πινάκων. Οι τρεις εξισώσεις που προκύπτουν είναι οι εξής:

- $L_{11} L_{11}^T = A_{11}$ (3.1) , η οποία επιλύεται αναδρομικά ως προς τον L_{11} με παραγοντοποίηση Cholesky(όπως θα δούμε στο κεφ.4).
- $L_{11} l_{12}^T = a_{12}$ (3.2) που είναι η τριγωνική λύση ως προς το l_{12} , γνωρίζοντας τον L_{11} .
- $l_{12}^T l_{12} + l_{22} = a_{22}$, όπου αν ο A θετικά ορισμένος² ισχύει ότι $a_{22} > l_{12}^T l_{12}$ και η σχέση γίνεται $l_{22} = \sqrt{a_{22} - l_{12}^T l_{12}}$ (3.3) .

Οι εγγραφές που περιέχονται στον πίνακα L αλλά δεν βρίσκονται στον πίνακα A , ονομάζονται γεμίσματα (fill – in). Κατασκευάζουμε τον L μία γραμμή την φορά μέσω της αντίστοιχης στήλης του A , δηλαδή, η k -στη γραμμή του L υπολογίζεται από την k -στη στήλη του A , μέσω της επίλυσης της τριγωνικής λύσης.



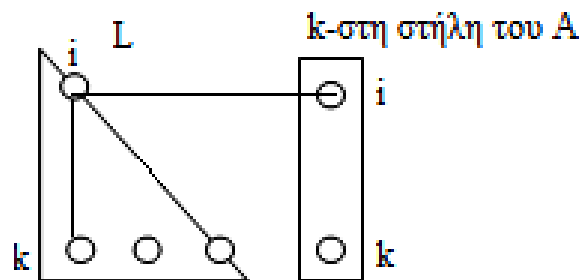
Σχήμα 3.1 Κατασκευή k-στης γραμμής του πίνακα L

² **Θετικά Ορισμένος Πίνακας [6]:**

Ένας πίνακας A είναι θετικά ορισμένος αν και μόνο αν $x^T A x > 0$, για κάθε μη μηδενικό διάνυσμα x . Ένας συμμετρικός πίνακας A ονομάζεται θετικά ορισμένος πίνακας εάν $x^T A x$ είναι θετικά ορισμένη τετραγωνική μορφή.

Παρατηρώντας λεπτομερέστερα την κατασκευή αυτή, έστω πως η πρώτη μη μηδενική εγγραφή του πίνακα A στην στήλη k είναι η a_{ik} , η οποία κατά την επίλυση της τριγωνικής λύσης $L_{kk}l_{ik}^T = a_{ik}$, που είναι η (3.2) σε γενικότερη μορφή, γίνεται η l_{ik} . Για παράδειγμα στην εξίσωση (3.2) ισχύει ότι αν $a_{12} \neq 0$, τότε $l_{12}^T \neq 0$. Άρα, γενικότερα ισχύει ότι αν υπάρχει το a_{ik} και είναι μη μηδενικό τότε υπάρχει και το l_{ik} και είναι μη μηδενικό.

$$a_{ik} \neq 0 \quad \Rightarrow \quad l_{ik} \neq 0 \quad (3.4)$$



Σχήμα 3.2 Ύπαρξη της εγγραφής l_{ik} λόγω της εγγραφής a_{ik}

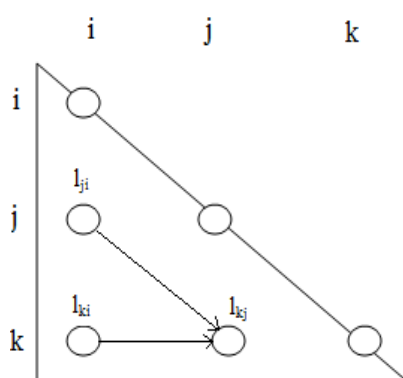
Έστω το κατευθυνόμενο γράφημα $G_{k-1} = (V, E)$, το οποίο αντιστοιχεί στο άνω τριγωνικό κομμάτι του L που έχουμε υπολογίσει μέχρι τώρα με κόμβους $V = \{1, \dots, n\}$ και ακμές $E = \{(j, i) \mid l_{ij} \neq 0\}$, L_k το μη μηδενικό πρότυπο της λύσης του τριγωνικού συστήματος, το οποίο γίνεται η k -στη γραμμή του πίνακα L και X_k το σύνολο των κόμβων που είναι προσπελάσιμοι από τον κόμβο k μέσω των μονοπατιών του γραφήματος G_{k-1} . Κατά την διαπέραση του κατευθυνόμενου γραφήματος:

- ξεκινάμε από τον κόμβο i , όπου $i \in L_k$ και άρα το στοιχείο $l_{ki} \neq 0$.
- έπειτα, επισκεπτόμαστε τον κόμβο j , με $j > i$, μέσω της ακμής (i, j) , η οποία αντιστοιχεί στην τιμή l_{ji} , άρα

παρατηρούμε ότι και το $j \in L_k$. Επομένως, ισχύει ότι και το $l_{kj} \neq 0$, όπου $i < j < k$.

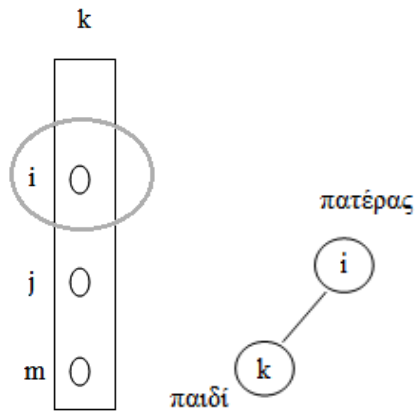
Συμπεραίνουμε ότι αν υπάρχουν οι ακμές (i,j) και (i,k) στο κατευθυνόμενο γράφημα G_L , δηλαδή τα στοιχεία l_{ji} και l_{ki} είναι μη μηδενικά, τότε υπάρχει και η ακμή (j,k) και αντίστοιχα το στοιχείο l_{kj} είναι μη μηδενικό. Σχηματικά, η σχέση αυτή απεικονίζεται στο Σχήμα

3.3: $l_{ji} \neq 0 \text{ και } l_{ki} \neq 0 \Rightarrow l_{kj} \neq 0$, για $i < j < k$ (3.5)



Σχήμα 3.3 Σχέση μεταξύ των l_{ji} , l_{ki} , l_{kj}

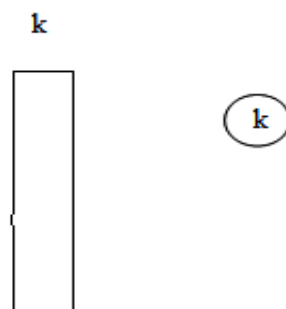
Από την στιγμή που βρήκαμε μονοπάτι από τον κόμβο i προς τον κόμβο k μέσω του j , χωρίς η ακμή (i,k) να χρησιμοποιείται, μπορούμε να την απαλείψουμε χωρίς να επηρεάσουμε το σύνολο X του κόμβου i . Γενικότερα, δεν επηρεάζουμε το σύνολο των κόμβων που είναι προσπελάσιμοι από έναν κόμβο με την απομάκρυνση όλων των «περιττών» ακμών και την διατήρηση το πολύ μίας ακμής, εκείνης που αντιστοιχεί στο μεγαλύτερο μήκος μονοπατιού. Έτσι, αν έχουμε έστω $i > k$ να είναι ο ελάχιστος αριθμημένος κόμβος (δηλαδή έχει τον μικρότερο δείκτη γραμμής στην στήλη k) για τον οποίο $l_{ik} \neq 0$, όλα τα υπόλοιπα μη μηδενικά l_{jk} , με $j > i$ είναι περιττά. Ουσιαστικά, από την κάθε στήλη κρατάμε την πρώτη μη μηδενική εγγραφή με τον μικρότερο δείκτη γραμμής, όπως απεικονίζεται στο Σχήμα 3.4.



Σχήμα 3.4 Πρώτη μη μηδενική εγγραφή της k -στης στήλης και σχέση των κόμβων στο δένδρο

Το αποτέλεσμα που προκύπτει είναι το δένδρο απαλοιφής (elimination tree) T , όπου ο γονέας του κόμβου k είναι ο κόμβος i , όταν το πρώτο μη μηδενικό στοιχείο (εκτός της διαγωνίου) της στήλης k έχει δείκτη γραμμής i , με $i > k$, όπως φαίνεται στο Σχήμα 3.4.

Η ρίζα του δένδρου είναι ο κόμβος k , αν η στήλη k δεν έχει κάτω από την διαγώνιο μη μηδενικές εγγραφές, γιατί δεν έχει ως πατέρα κάποιον κόμβο (Σχήμα 3.5). Επίσης, το δένδρο μπορεί να είναι δάσος, μέσω της ύπαρξης πολλών ριζών, αλλά λόγω σύμβασης το ονομάζουμε δέντρο.



Σχήμα 3.5 Ο κόμβος k ρίζα του δένδρου

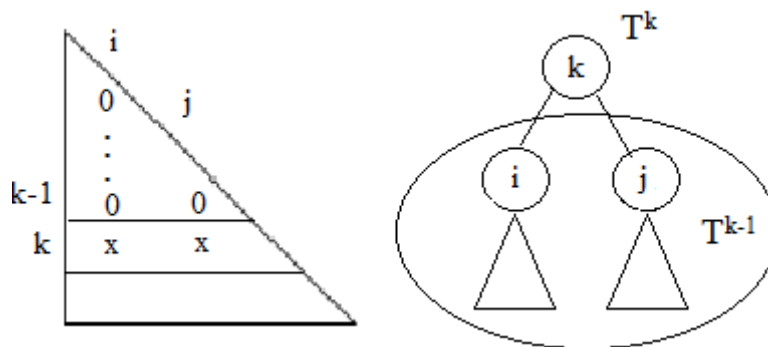
Ο υπολογισμός του δένδρου απαλοιφής T , το οποίο αποτελείται από τα υποδέντρα γραμμής του L , στηρίζεται στις παρακάτω παρατηρήσεις:

- οι μη μηδενικές τιμές του πίνακα L χωρίζονται σε δύο κατηγορίες: σε αυτές που δεν υπάρχουν στον πίνακα A , λόγω της σχέσης (3.4) και σε εκείνες που προέρχονται από τον A λόγω της (3.5).
- η μη μηδενική εγγραφή l_{ki} με $k > i$ υποδηλώνει την ύπαρξη μονοπατιού στο T από τον κόμβο i (απόγονος) στον k (πρόγονος).
- το υποδέντρο γραμμής T^k , είναι το υποδέντρο που παράγεται από τους κόμβους που αντιστοιχούν στο μη μηδενικό πρότυπο της k -στης γραμμής του L , L_k . Για το λόγο αυτό, το T^k είναι μία συλλογή μονοπατιών προς τον κόμβο k , ο οποίος αποτελεί την ρίζα του.
- τα υποδέντρα γραμμής λαμβάνουν όλη την δομή των μη μηδενικών εγγραφών του πίνακα L .
- Κάθε φύλλο ενός υποδέντρου αντιστοιχεί σε μία μη μηδενική εγγραφή του πίνακα A , η οποία έχει την μοναδική ικανότητα να μην έχει απογόνους. Οι μη μηδενικές εγγραφές της k -στης γραμμής του πίνακα A χωρίζονται σε δύο κατηγορίες σε εκείνες που είναι φύλλα του υποδένδρου T^k και σε εκείνες που δεν είναι φύλλα και βρίσκονται απλά στο μονοπάτι. Για να αποτελεί φύλλο μία εγγραφή θα πρέπει να ισχύει το παρακάτω θεώρημα:³
ο κόμβος j είναι φύλλο στο T^k αν και μόνο αν ισχύει ότι $a_{jk} \neq 0$ και $a_{ik} = 0$ για κάθε κόμβο i απόγονο του κόμβου j .
- κάθε υποδέντρο χαρακτηρίζεται από τα φύλλα του που αντιστοιχούν στις εγγραφές του πίνακα A .

Υποθέτοντας ότι το T^{k-1} είναι ήδη γνωστό και περιλαμβάνει όλα τα υποδέντρα γραμμής των αντίστοιχων $k-1$ γραμμών, προχωράμε στην

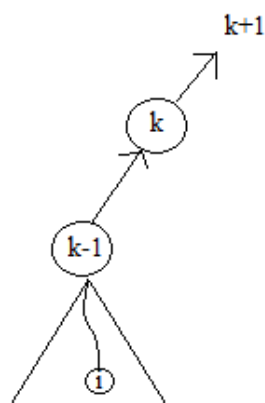
³ Direct Methods for Sparse Linear Systems, Tim Davis, ch.4, Theorem 4.6 (Liu [148]).

κατασκευή του T^k , το οποίο θα έχει ως ρίζα του τον κόμβο k και θα αποτελείται από τους κόμβους όπου ανήκουν στο μη μηδενικό πρότυπο της k -στης γραμμής. Σε όποια θέση ,έστω j , βρούμε μη μηδενικό στοιχείο, ο κόμβος j θα έχει πρόγονό του τον κόμβο k . Τα άμεσα παιδιά του k είναι οι κόμβοι ρίζες του T^{k-1} , για τους οποίους ισχύει ότι στις προηγούμενες $k-1$ γραμμές ήταν μηδέν. Η κατασκευή του υποδέντρου γραμμής απεικονίζεται στο Σχήμα 3.6.



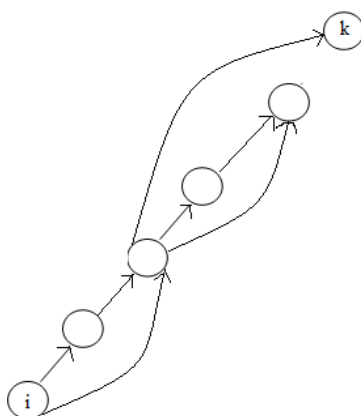
Σχήμα 3.6 Κατασκευή του υποδέντρου γραμμής T^k

Με αυτό τον αλγόριθμο κατασκευής του δένδρου απαλοιφής, ο χρόνος υπολογισμού είναι $O(|A|)$. Ωστόσο, θα μπορούσαμε να επιταχύνουμε την διαπέραση του υποδέντρου γραμμής T^{k-1} , κρατώντας ένα σύνολο προγόνων, όπου ο κάθε κόμβος στο υποδέντρο γραμμής θα έδειχνε στον πρόγονό του. Ιδανικά, η απλή ρίζα $k-1$ στο υποδέντρο γραμμής T^{k-1} θα είναι πρόγονος του κόμβου i και η προσπέλαση του μονοπατιού από τον i στον $k-1$ παίρνει χρόνο $O(1)$, απλά βρίσκοντας μόνο τον πρόγονο και χωρίς να διαπεράσουμε ολόκληρο το μονοπάτι που οδηγεί σε αυτόν. Στο Σχήμα 3.7 φαίνεται πως ο πρόγονος του κόμβου i είναι ο $k-1$:



Σχήμα 3.7 Μονοπάτι Προγόνου

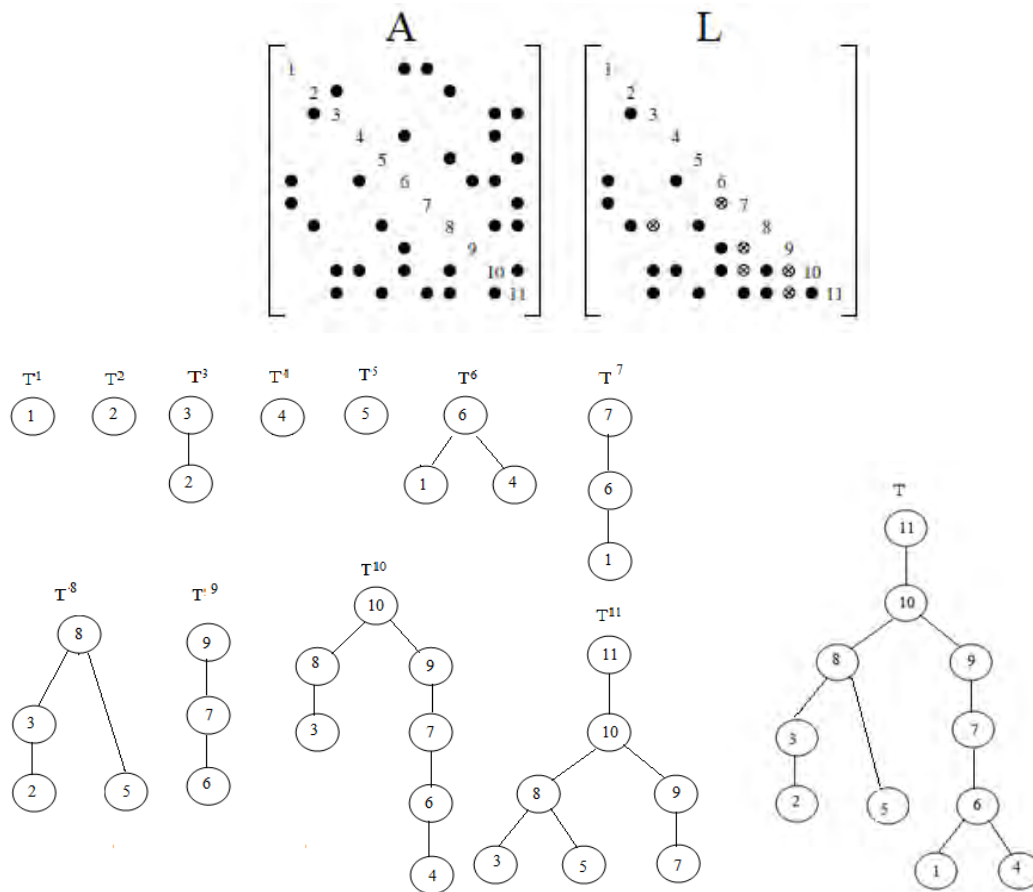
Η ιδέα είναι πως καθώς προχωράμε το μονοπάτι ο στόχος μας είναι να φτάσουμε στην ρίζα του υποδέντρου που έχουμε βρει μέχρι τώρα, χωρίς να διαπεράσουμε ολόκληρο το μονοπάτι αλλά να φτάσουμε στον κόμβο k κατευθείαν. Αυτό πραγματοποιείται αν κρατάμε για κάθε κόμβο i τον υψηλότερο πρόγονο χωρίς να είναι απαραίτητα η ρίζα του υποδέντρου στο οποίο ανήκει. Επειδή στο υποδέντρο γραμμής T^k η διαδρομή οδηγεί εγγυημένα στον κόμβο k , πρόγονος όλων των κόμβων κατά μήκος του μονοπατιού είναι ο k , όπως απεικονίζεται και στο Σχήμα 3.8.



Σχήμα 3.8 Σύνολο μονοπατιών που οδηγούν στον k -στο κόμβο.

Ωστόσο, η μέθοδος αυτή δεν είναι καθόλου βολική όταν θέλουμε να απαριθμήσουμε τους κόμβους του δένδρου.

Σαν παράδειγμα απεικονίζεται στο Σχήμα 3.9 ένας αραιός πίνακας A ^[5] και ο αντίστοιχος παράγοντας του L , με τα αντίστοιχα υποδέντρα γραμμής και το δένδρο απαλοιφής.



Σχήμα 3.9 Πίνακας A , Παράγοντας L , τα υποδέντρα γραμμής και δένδρο απαλοιφής.

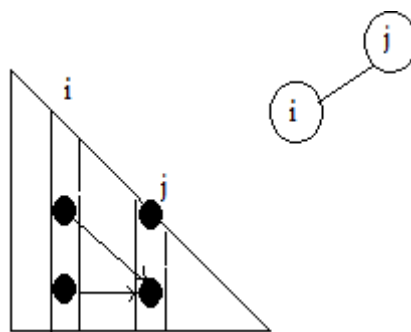
3.2 Μεταδιατεταγμένη διάσχιση δένδρου

Μόλις βρούμε το δένδρο απαλοιφής απαιτείται να υπολογίσουμε την μεταδιατεταγμένη διάταξή του (post ordering) ώστε να προχωρήσουμε στον υπολογισμό του πλήθους των μη μηδενικών στοιχείων κάθε στήλης του L .

Με την μεταδιάταξη του πίνακα L συγχωνεύουμε τις στήλες του και παίρνουμε τον ίδιο πίνακα ισομορφικά. Το μη μηδενικό πρότυπο του L , που προκύπτει από την μεταδιάταξη $PAP^T = LL^T$, θα είναι περισσότερο

δομημένο και η αριθμητική παραγοντοποίηση που θα ακολουθήσει πιο γρήγορη.

Αναλυτικότερα, στηριζόμενοι στη σχέση (3.5) και γνωρίζοντας ότι ο κόμβος i είναι παιδί του κόμβου j , κάθε μη μηδενική εγγραφή στην στήλη i εκτός της διαγωνίου θα βρίσκεται στην στήλη j , άρα είναι ταυτόσημες και τις τοποθετούμε μαζί. Όπως φαίνεται στο Σχήμα 3.10 οι στήλες i, j ταυτίζονται εκτός της διαγώνιας εγγραφής της πρώτης, καθώς συνδέονται με σχέση γονέα και παιδιού.



Σχήμα 3.10 Όμοιες στήλες

Αν υποθέσουμε ότι ο πίνακας $post$ αντιπροσωπεύει την μεταδιατεταγμένη διάταξη τότε ισχύει ότι:

$$post[k] = i$$

με i να είναι ο κόμβος του πραγματικού δέντρου και k ο αντίστοιχος στο μεταδιατεταγμένο δένδρο. Ουσιαστικά, ο πίνακας $post$ είναι μία μετάθεση της μορφής $p[new] = old$ και λειτουργεί σαν πίνακας συγχώνευσης. Η φυσική μεταδιάταξη διατηρεί την διάταξη των παιδιών ενός κόμβου:

έστω t τα παιδιά του κόμβου k , με διάταξη : $c_1 < c_2 < \dots < c_t$ μετά την εφαρμογή της μεταδιάταξης θα ισχύει $post[c_1] < post[c_2] < \dots < post[c_t]$.

Ο υπολογισμός της μεταδιατεταγμένης διάταξης του δένδρου γίνεται αναδρομικά μέσω της αναζήτησης κατά βάθος (dfs).

Χαρακτηριστικά, κάθε κόμβος j του δένδρου T που είναι ρίζα έχει μία λίστα με τα παιδιά του και κάθε παιδί έχει μόνο έναν γονέα, δηλαδή εμφανίζεται σε μία μόνο λίστα. Στο k -στο βήμα της μεταδιάταξης ελέγχουμε:

- αν ο κόμβος j δεν έχει παιδιά τότε είναι φύλλο και μπαίνει στην μεταδιάταξη ως k -στος κόμβος.
- αν έχει παιδιά, τότε ξεκινάμε μια αναζήτηση κατά βάθος για το πρώτο παιδί και όταν επιστρέψουμε συνεχίζουμε με αναζήτηση κατά βάθος στο επόμενο κ.ο.κ. Μόλις, τελειώσουμε με τα παιδιά του τον τοποθετούμε στην μεταδιάταξη ως k -στο κόμβο.

Για την καλύτερη κατανόηση της μεθόδου παραθέτουμε τον παρακάτω ψευδοκώδικα:

Post order T:

$k = 0$

Για κάθε κόμβο j που είναι ρίζα στο T
 $dfs(j)$

$dfs(j)$:

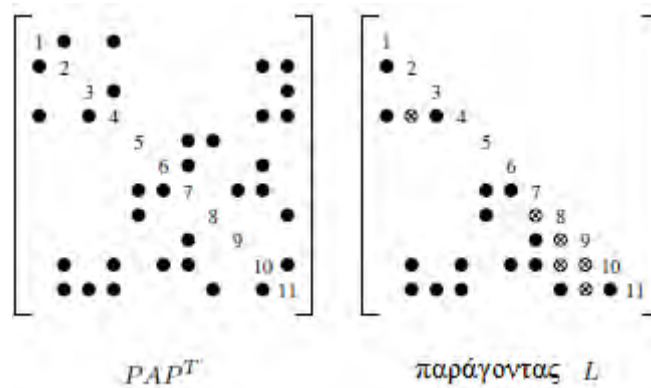
Για κάθε παιδί i του κόμβου j

$dfs(i)$

$post[k] = i$

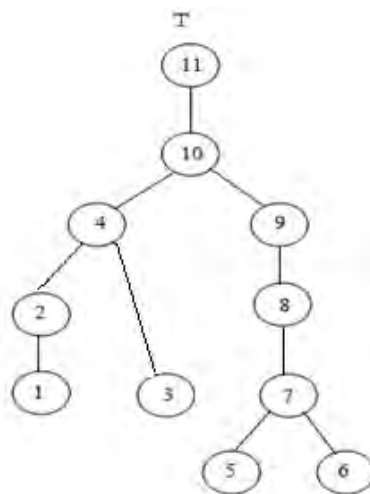
$k = k + 1$

όπου στην Post Order προσπαθούμε να βρούμε τις πιθανές ρίζες για κάθε δένδρο στο δάσος και μόλις βρούμε κάποια καλούμε την dfs για την ρίζα αυτή. Κατά την εκτέλεση της dfs για κάθε παιδί που βρίσκουμε καλούμε αναδρομικά την συνάρτηση και μόλις τελειώσει η αναδρομική dfs που αντιστοιχεί στη ρίζα, την τοποθετούμε στον πίνακα $post$.



Σχήμα 3.11 Μεταδιατεταγμένοι πίνακες A και L

Στο Σχήμα 3.11 έχουμε τον πίνακα A ^[5] μεταδιατεταγμένο, όπου P είναι ο πίνακας της μεταδιάταξης και τον αντίστοιχο πίνακα του μεταδιατεταγμένου παράγοντα L . Το αντίστοιχο μεταδιατεταγμένο δένδρο απαλοιφής T , απεικονίζεται στο Σχήμα 3.12 :



Σχήμα 3.12 Το μεταδιατεταγμένο δένδρο απαλοιφής

3.3 Πλήθος Γραμμής και Πλήθος Στήλης

Το πλήθος γραμμής και στήλης σε μία παραγοντοποίηση Cholesky είναι ο αριθμός των μη μηδενικών σε κάθε γραμμή και στήλη του πίνακα L , αντίστοιχα. Αρχικά, θα μελετήσουμε την εύρεση του πλήθους γραμμής και έπειτα του πλήθους στήλης.

3.3.1 Πλήθος Γραμμής

Υπάρχουν δύο τρόποι υπολογισμού του πλήθους γραμμής. Η πιο απλή αλλά όχι βέλτιστη στηρίζεται στην διαπέραση κάθε υποδέντρου γραμμής και την διατήρηση ενός μετρητή με το πλήθος των κόμβων που αποτελούν το συγκεκριμένο υποδέντρο:

πλήθος κόμβων στο i -στο υποδέντρο γραμμής = πλήθος γραμμής.

Με χρόνο υπολογισμού $O(|L|)$.

Η μέθοδος που ακολουθεί μειώνει τον χρόνο σε $O(|A|)$ και στηρίζεται στην εξής βασική ιδέα:

- διασπούμε το κάθε υποδέντρο γραμμής σε σύνολο ασύνδετων μονοπατιών, όπου το καθένα ξεκινά από κόμβο – φύλλο και σταματά στον ελάχιστο κοινό πρόγονο με το προηγούμενο φύλλο.
- το μήκος των μονοπατιών αυτών είναι η διαφορά των επιπέδων μεταξύ του κόμβου έναρξης και τερματισμού.
- εκμεταλλευόμαστε το γεγονός πως όλα τα υποδέντρα σχετίζονται μεταξύ τους γιατί αποτελούν υποδέντρα του δένδρου απαλοιφής.

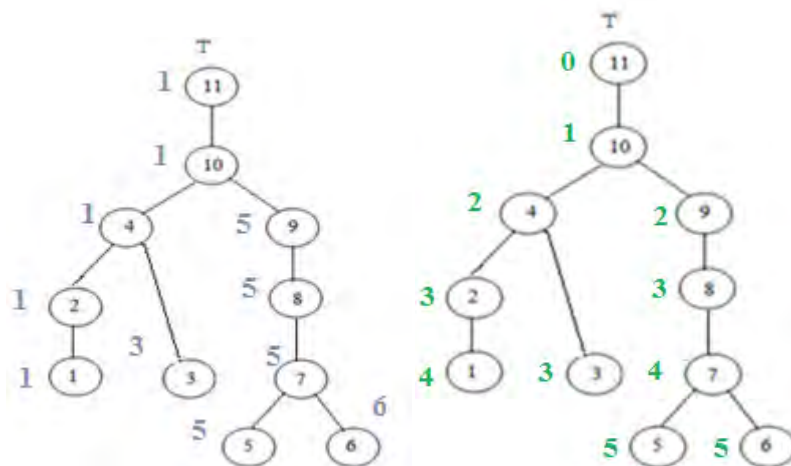
Το πρώτο βήμα της μεθόδου είναι η εύρεση του πρώτου απογόνου και του επιπέδου κάθε κόμβου, έστω $first$ και $level$, αντίστοιχα. Ο πρώτος απόγονος ενός κόμβου j είναι ο μικρότερος μεταδιατεταγμένος απόγονος του, δηλαδή αυτός με το μικρότερο όνομα, συμπεριλαμβανομένου και του εαυτού του. Ο υπολογισμός του πραγματοποιείται ως εξής:

- Αρχικά, ξεκινάμε με τον πρώτο κόμβο $k = 1$ του μεταδιατεταγμένου δένδρου και προχωράμε προς την ρίζα. Για κάθε κόμβο που προσπελάνουμε κατά μήκος του μονοπατιού από τον κόμβο 1 έως την ρίζα, θέτουμε ως πρώτο απόγονο $first$ τον k και επαναπροσπελάνουμε το

μονοπάτι προς την αντίθετη φορά (από την ρίζα προς τον k) για να καθορίσουμε τα επιμέρους επίπεδα level των κόμβων του μονοπατιού. Για την ρίζα $root$ ισχύει ότι $level(root) = 0$ και καθώς προχωράμε για κάθε κόμβο ορίζουμε το level από το μήκος του μονοπατιού που έχουμε διασχίσει μέχρι τώρα, έως ότου φτάσουμε στον κόμβο $k=1$.

- Έπειτα, για κάθε $k > 1$ κόμβο θέτουμε και διασχίζουμε, με τον ίδιο τρόπο, μέχρι να βρούμε κάποιον κόμβο που έχει ήδη καθοριστεί ο πρώτος απόγονος και το επίπεδό του. Ομοίως, καθορίζουμε τα επιμέρους επίπεδα των κόμβων κατά μήκος του μονοπατιού.

Συμπεραίνουμε ότι το κάθε μονοπάτι διασχίζεται δύο φορές, μία για τον καθορισμό του μικρότερου απογόνου και μία για την εύρεση των επιπέδων των κόμβων, καθώς και ότι όλοι οι πρόγονοι ενός κόμβου k βρίσκονται πάνω από αυτόν.



Σχήμα 3.13 Το δένδρο με τον πρώτο απόγονο και το επίπεδο κάθε κόμβου.

Ένα παράδειγμα απεικονίζεται στο Σχήμα 3.13, στο οποίο η εύρεση του πρώτου απογόνου κάθε κόμβου στο μεταδιατεταγμένο δένδρο

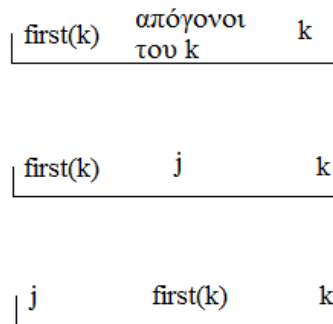
συμβολίζεται με μπλε χρώμα , στο αριστερό δένδρο του σχήματος. Ξεκινάμε για $k=1$, είναι ο μικρότερος απόγονος first για τον εαυτό του αλλά και για τους κόμβους στο μονοπάτι από αυτόν μέχρι την ρίζα. Από την ρίζα επιστρέφουμε στον $k = 1$ βρίσκοντας τώρα το επίπεδο level κάθε κόμβου. Επίσης, στο Σχήμα 3.13 απεικονίζονται τα επίπεδα των κόμβων με πράσινο χρώμα στο δεξιό δένδρο. Για τον $k = 3$ έχοντας τον εαυτό του ως ελάχιστο απόγονο, σταματάμε το μονοπάτι στον κόμβο 4 που έχει ήδη βρεθεί ο ελάχιστος κόμβος και το επίπεδο του. Το επίπεδο του κόμβου 3 βρίσκεται μέσω του επιπέδου του κόμβου 4 και του μήκους του μονοπατιού μεταξύ του 3 και 4. Η διαδικασία επαναλαμβάνεται για τους υπόλοιπους κόμβους.

Στο επόμενο βήμα, θα υπολογίσουμε τα φύλλα των υποδέντρων γραμμής και θα διασπάσουμε το καθένα σε ασύνδετα μονοπάτια. Για τον πρώτο υπολογισμό αρκεί να βρούμε τις εγγραφές που σχηματίζουν τα φύλλα, μέσω της πληροφορίας που έχουμε από το προηγούμενο βήμα, με τον πρώτο απόγονο first κάθε κόμβου.

Σε αυτό το σημείο θα εισάγουμε κάποια νέα στοιχεία που θα μας βοηθήσουν στην διεξαγωγή των υπολογισμών. Τα φύλλα των μεταδιατεταγμένων υποδέντρων γραμμής σχηματίζουν τον πίνακα σκελετού (skeleton) του μεταδιατεταγμένου πίνακα A , όπου ισχύει ότι αν ο κόμβος j είναι φύλλο του i -στου υποδέντρου γραμμής τότε το στοιχείο $a_{ij} \neq 0$. Ωστόσο, το αντίστροφο δεν ισχύει. Οι εγγραφές του πίνακα skeleton στηρίζονται στην πληροφορία του πρώτου απογόνου και του επιπέδου κάθε κόμβου, από την οποία μπορούμε να καταλάβουμε αν υπάρχει σχέση προγόνου – απογόνου μεταξύ δύο κόμβων, από το γεγονός ότι όλοι οι απόγονοι ενός κόμβου είναι μικρότεροι του στην μεταδιάταξη. Από δύο κόμβους που θέλουμε να βρούμε την σχέση τους, έστω j και k , αρκεί να πάρουμε τον μικρότερο από αυτούς, έστω ότι είναι

ο $j(j = \min(j,k))$ και να τον συγκρίνουμε με τον μικρότερο απόγονο του άλλου, $\text{first}(k)$. Αν $j < \text{first}(k)$ τότε δεν αποτελεί απόγονό του k , οπότε στο μονοπάτι από τον j προς την ρίζα δεν θα συναντήσουμε τον k . Διαφορετικά, ο j αποτελεί απόγονο του k . Στο Σχήμα 3.14 απεικονίζεται η σχέση των κόμβων με μορφή διάταξης, όπου για κάθε κόμβο k ισχύει ότι το σύνολο των απογόνων του ξεκινά με τον ελάχιστο απόγονο του, ακολουθούν οι επιμέρους απόγονοι και τέλος ο ίδιος. Για δύο κόμβους j και k :

- αν $j < \text{first}(k)$, δεν ανήκει στο σύνολο των απογόνων.
- διαφορετικά, είναι απόγονος του k .



Σχήμα 3.14 Σχέση κόμβων σε μορφή διάταξης

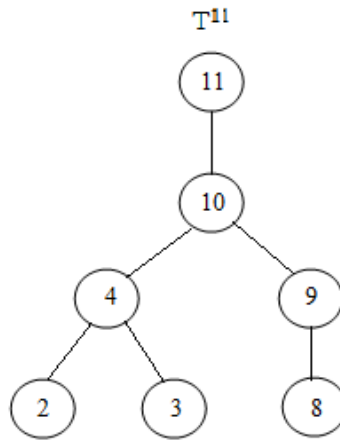
Τώρα, μπορούμε να προχωρήσουμε στην μέθοδο εύρεσης των φύλλων κάθε i -στου υποδέντρου γραμμής με την βοήθεια της μεταβλητής maxfirst , η οποία θα κρατά τον μέγιστο ελάχιστο απόγονο που έχουμε δει μέχρι τώρα για κάθε μη μηδενική εγγραφή a_{ij} του i -στου υποδέντρου γραμμής. Έτσι, για κάθε κόμβο j που βρίσκεται σε κάθε υποδέντρο i :

- Αν $\text{first}(j) > \text{maxfirst}(i)$, ο j δεν έχει απόγονο στο i -στο υποδέντρο γραμμής και είναι φύλλο.
- Αλλιώς, έχει κάποιον απόγονο έστω d , για τον οποίο ισχύει ότι :

$$\text{first}(d) = \text{maxfirst}(i) \geq \text{first}(j)$$

και άρα δεν είναι φύλλο.

Σαν παράδειγμα, ερευνούμε τα φύλλα του μεταδιατεταγμένου υποδέντρου γραμμής T^{11} , που φαίνεται στο Σχήμα 3.15:



Σχήμα 3.15 Μεταδιατεταγμένο Υποδέντρο γραμμής T^{11}

Αρχικά, $\text{maxfirst}(11) = \text{none}$.

Για $j = 2$: $\text{first}(2) = 1$, ο οποίος δεν βρίσκεται στο υποδέντρο αλλά μας βοηθά στην διαδικασία, : $1 > \text{none}$, ο 2 αποτελεί φύλλο και $\text{maxfirst}(11) = 1$.

Για $j = 3$: $\text{first}(3) = 3 > \text{maxfirst}(11)$, άρα αποτελεί φύλλο και $\text{maxfirst}(11)=3$.

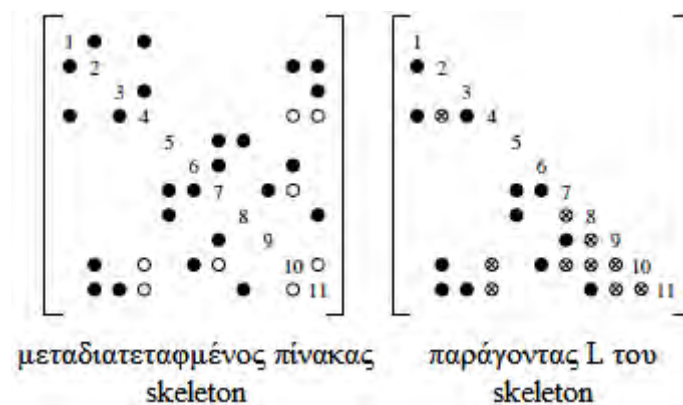
Για $j = 4$: $\text{first}(4) = 1 < \text{maxfirst}(11)$, άρα δεν είναι φύλλο γιατί έχουμε επισκεφτεί κάποιον απόγονο του με μεγαλύτερο ελάχιστο απόγονο από τον δικό του, τον 3.

Για $j = 8$: $\text{first}(8) = 5 > \text{maxfirst}(11)$, άρα είναι φύλλο και $\text{maxfirst}(11) = 5$.

Για $j = 10$: $\text{first}(10) = 1 < \text{maxfirst}(11)$, άρα δεν είναι φύλλο.

Όντως τα φύλλα του T^{11} είναι τα: 2, 3, 8 που βρήκαμε.

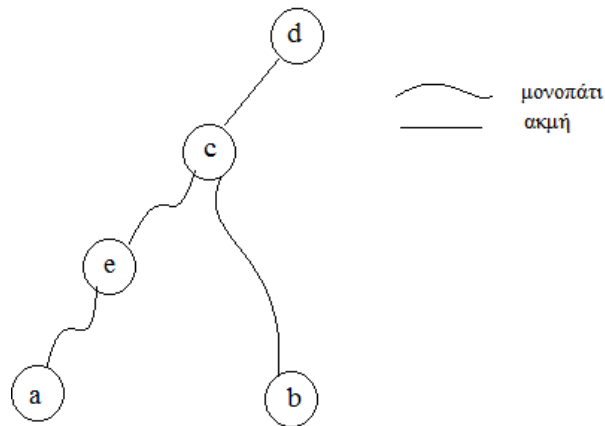
Με τον τρόπο αυτό διαπερνάμε μία φορά τον πίνακα A κατά γραμμές και βρίσκουμε τα στοιχεία του πίνακα skeleton, τα οποία θα χρησιμοποιήσουμε για την αποσύνθεση των υποδέντρων γραμμής. Στο Σχήμα 3.16 φαίνεται ο αντίστοιχος μεταδιατεταγμένος πίνακας skeleton και ο παράγοντας του L , για τον πίνακα μεταδιάταξης του Σχήματος 3.11. Οι εγγραφές με λευκό κύκλο είναι εκείνες που δεν εμφανίζονται στον skeleton ενώ υπάρχουν στον πίνακα A ^[5]. Επίσης, παρατηρούμε ότι η εγγραφή $a_{ij} \neq 0$ του skeleton για $i > j$ αν και μόνο αν ο κόμβος j είναι φύλλο στο i -στο υποδέντρο μεταδιάταξης. Το αντίστοιχο δέντρο απαλοιφής είναι όμοιο με το μεταδιατεταγμένο δέντρο του Σχήματος 3.12.



Σχήμα 3.16 Οι πίνακες skeleton των πινάκων A και L

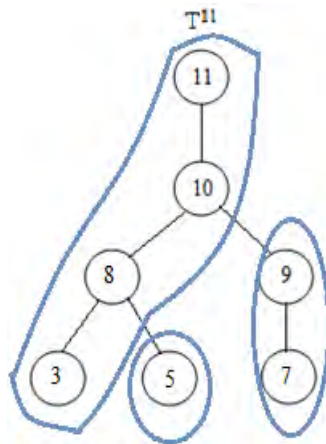
Η διαδικασία διαχωρισμού του υποδέντρου γραμμής σε ξένα μονοπάτια ονομάζεται αποσύνθεση μονοπατιού. Αρχικά, παίρνουμε το πρώτο φύλλο του υποδέντρου που βρήκαμε και ακολουθούμε το μονοπάτι που οδηγεί στην ρίζα. Αυτό είναι το πρώτο μονοπάτι διαχωρισμού. Το επόμενο μονοπάτι ξεκινά, από το αμέσως επόμενο φύλλο του υποδέντρου και σταματά στον ελάχιστο κοινό πρόγονο αυτού με το αμέσως προηγούμενο φύλλο. Ο ελάχιστος κοινός πρόγονος δύο κόμβων έστω a και b είναι ο ελάχιστος αριθμημένος κόμβος που είναι από κοινού πρόγονος των a και b . Στο Σχήμα 3.17 ο κόμβος c είναι ο

ελάχιστος κοινός πρόγονος των a, b καθώς οι κόμβοι διατεταγμένοι είναι:
 $a < e < b < c < d$.



Σχήμα 3.17 Ελάχιστος Κοινός Πρόγονος

Για το μεταδιατεταγμένο υποδέντρο γραμμής T^{11} τα μονοπάτια διαχωρισμού είναι 3, όπως φαίνονται στο Σχήμα 3.18.



Σχήμα 3.18 Μονοπάτια Διαχωρισμού στο T^{11}

Το πλήθος γραμμής, που αντιστοιχεί στο σύνολο μη μηδενικών κάθε γραμμής του πίνακα L υπολογίζεται μέσω του συνολικού αθροίσματος των μηκών των διαχωρισμένων μονοπατιών. Αναλυτικότερα, έχοντας βρει τον ελάχιστο κοινό πρόγονο δύο συνεχόμενων φύλλων, έστω c και a, b , αντίστοιχα, υπολογίζουμε το μήκος μονοπατιού από τον b στον c , το οποίο είναι ίσο με την διαφορά

των επιπέδων μεταξύ του c και b ($\text{level}(c) - \text{level}(b)$), και το προσθέτουμε στο πλήθος γραμμής. Γενικά, ισχύει για κάθε i -στο μονοπάτι διαχωρισμού:

$$\text{πλήθος γραμμής} = \sum_i |\text{μήκος μονοπατιού}|.$$

Για την γραμμή 11 του πίνακα L : Έστω $\text{row_count} = 0$, ο μετρητής για το πλήθος γραμμής. Ξεκινάμε, από τον κόμβο 2 και φτάνουμε ως την ρίζα : $\text{row_count} = |\text{level}(11)| - \text{level}(2)| = 3$. Ο ελάχιστος κοινός πρόγονος των 2, 3 είναι ο 4, οπότε $\text{row_count} = |3| + |1| = 4$. Ομοίως, για τους κόμβους 3, 8 ο ελάχιστος κοινός πρόγονος είναι ο κόμβος 10 : $\text{row_count} = |4| + |2| = 6$. Για να επαληθεύσουμε το αποτέλεσμα του προέκυψε: ο πίνακας L του Σχήματος 3.16 στην 11-η γραμμή, όντως αποτελείται από 6 μη μηδενικές τιμές, εκτός της διαγώνιας εγγραφής.

3.3.2 Πλήθος Στήλης

Η εύρεση του πλήθους στήλης μπορεί να βρεθεί ομοίως με δύο τρόπους, την πιο απλή αλλά όχι βέλτιστη μέθοδο που αναφέραμε για το πλήθος γραμμής με την διαφορά ότι κρατάμε το πλήθος των υποδέντρων γραμμής, στα οποία συμμετέχει ο κόμβος j :

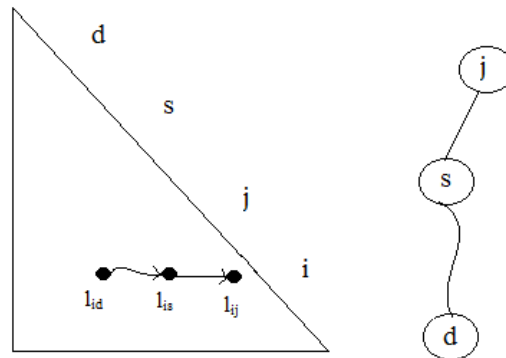
πλήθος υποδέντρων γραμμής που υπάρχει ο $j = \text{πλήθος στήλης}$.

Αναλυτικότερα, έστω L_j το μη μηδενικό πρότυπο της στήλης j του μεταδιατεταγμένου πίνακα L , στον οποίο έχουμε τοποθετήσει όμοιες στήλες μαζί και φέρουν σχέσεις απογόνου – προγόνου. Προσπαθούμε να βρούμε το πλήθος μη μηδενικών στοιχείων κάθε στήλης, έστω c_j , για το οποίο ισχύει : $c_j = |L_j|$.

Το μη μηδενικό πρότυπο της j -στης στήλης του L , γνωρίζοντας τις γραμμές του μπορεί να βρεθεί από την εξής παρατήρηση: έστω οι τρεις στήλες του κάτω τριγωνικού πίνακα L , d , s , j , όπου d και s απόγονος και παιδί του j , αντίστοιχα, για τις οποίες ισχύει ότι οι εγγραφές ενός κόμβου

παιδιού εμφανίζονται και στον πατέρα σε κάθε γραμμή i του πίνακα L , όπως αναφέραμε στην Παράγραφο 3.2 μέσω του Σχήματος 3.10. Για τον λόγο αυτό, για τις στήλες d, s, j θα ισχύει:

αν η εγγραφή $l_{id} \neq 0$ τότε και αν η εγγραφή $l_{is} \neq 0$ τότε ισχύει σίγουρα ότι $l_{ij} \neq 0$, γιατί ο κόμβος s είναι παιδί του j . Η σχέση αυτή απεικονίζεται στο Σχήμα 3.19 ^[5]:



Σχήμα 3.19 Σχέση Μεταξύ Κόμβων

Επομένως, συμπεραίνουμε με σιγουριά πως ότι εμφανίζεται σε κάποιο απόγονο d ενός κόμβου j , τότε εμφανίζεται και στον ίδιο, μιας και πάει διαδοχικά η πληροφορία από παιδί σε πατέρα στο δέντρο. Άρα, αρκεί να ξέρουμε την πληροφορία των μη μηδενικών εγγραφών σε κάθε παιδί s του κόμβου j χωρίς να ελέγχουμε όλους τους απογόνους του. Γενικότερα, καταλήγουμε στο συμπέρασμα πως το μη μηδενικό πρότυπο της στήλης j , L_j είναι η ένωση των μη μηδενικών προτύπων των παιδιών του. Ως αποτέλεσμα όσων προηγήθηκαν η γραμμή i θα εμφανιστεί στο L_j και ο κόμβος j θα βρίσκεται στο υποδέντρο γραμμής T^i .

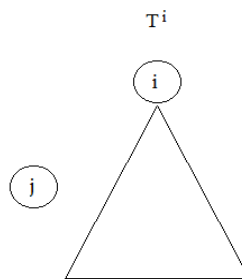
Η μέθοδος διάσχισης κάθε υποδέντρου γραμμής για την καταγραφή της εμφάνισης του κόμβου j σε κάθε ένα από τα υποδέντρα χρησιμοποιεί την παραπάνω παρατήρηση απαιτώντας χρόνο $O(|L|)$.

Η μείωση του χρόνου υπολογισμού σε $O(|A|)$ επιτυγχάνεται μέσω της εύρεσης των ελαχίστων προγόνων των διαδοχικών φύλλων ζευγών

κάθε υποδέντρου γραμμής. Για την διεξαγωγή της θα χρησιμοποιηθεί η έννοια της επικάλυψης (overlap) ενός κόμβου j , η οποία μας πληροφορεί για το πλήθος των υποδέντρων γραμμής στα οποία ο j είναι ο ελάχιστος κοινός πρόγονος δύο κόμβων, το μη μηδενικό πρότυπο της j - στην στήλη του πίνακα skeleton , skeleton_j , το οποίο αντιπροσωπεύει τα φύλλα που δεν βρίσκονται σε κανένα παιδί του j και ο όρος D_j για ολόκληρη τη στήλη j .

Η επικάλυψη o_j μετράται ως εξής:

- αν ο κόμβος j δεν ανήκει στο i -στο υποδέντρο γραμμής T^i , τότε δεν ανήκει στο μη μηδενικό πρότυπο L_j και η γραμμή i δεν συμβάλλει στην επικάλυψη o_j .



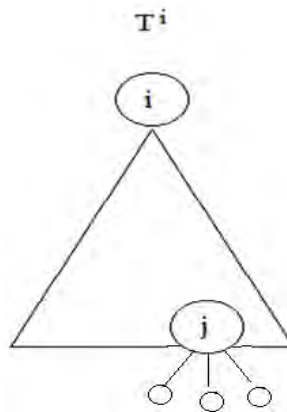
Σχήμα 3.20 Ο κόμβος $j \notin T^i$

- αν ο κόμβος j ανήκει στο i -στο υποδέντρο γραμμής T^i , τότε:
 - αν αποτελεί φύλλο, ισχύει ότι $a_{ij} \neq 0$ στον πίνακα skeleton και η γραμμή i δεν συμβάλλει στην επικάλυψη o_j , γιατί δεν υπάρχουν παιδιά μέσα στο T^i και δεν μπορεί να είναι ελάχιστος κοινός πρόγονος. Ισχύει:

$$o_j = 0$$

$$D_j = |\text{skeleton}_j| + 1,$$

Εφόσον είναι φύλλο βρίσκεται στον πίνακα skeleton , οπότε παίρνουμε το πλήθος των εγγραφών του στην j -στη στήλη του skeleton αθροίζοντας το με την μονάδα για την εγγραφή της διαγωνίου.

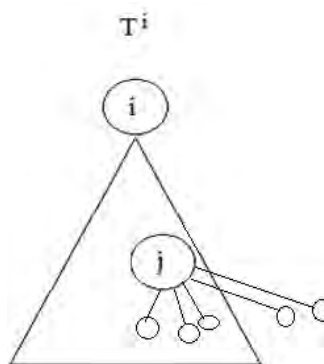


Σχήμα 3.21 Ο κόμβος j αποτελεί φύλλο του T^i

- αν δεν αποτελεί φύλλο, τότε έχει παιδιά στο T^i , τα οποία είναι υποσύνολο των παιδιών του στο δέντρο απαλοιφής T . Έστω d_{ij} και e_j , το πλήθος παιδιών στο T^i και T , αντίστοιχα. Η γραμμή i συμβάλλει στην επικάλυψη o_j κατά $d_{ij}-1$, γιατί θα αποτελεί τον ελάχιστο κοινό πρόγονο για $d-1$ συνεχόμενα ζεύγη φύλλων στο T^i . Από αυτό προκύπτει ότι αν ο j έχει μόνο ένα παιδί στο T^i η γραμμή i εμφανίζεται μόνο σε αυτό και έτσι δεν συμβάλλει στην επικάλυψη ($o_j = 1-1 = 0$).

$$o_j = d_{ij} - 1$$

$$D_j = |\text{skeleton}_j| - e_j - o_j$$



Σήμα 3.22 Ο κόμβος j αποτελεί ενδιάμεσο κόμβο στο T^i

Η εύρεση του πλήθους στήλης υπολογίζεται για κάθε περίπτωση από την σχέση:

$$c_j = D_j + \sum_{j = \text{parent}(s)} c_s \quad (3.6)$$

που είναι το άθροισμα ολόκληρης της στήλης του j με το συνολικό άθροισμα των μεγεθών κάθε παιδιού του.

Εφαρμόζουμε την μέθοδο σε ένα μικρό παράδειγμα:

έστω για την στήλη $j = 4$ του σχήματος 3.15: δεν είναι φύλλο του δέντρου απαλοιφής T , αλλά ενδιάμεσος κόμβος και γονέας των κόμβων 2 και 3. Επομένως, ο πίνακας skeleton στην στήλη $j = 4$ είναι κενός, $\text{skeleton}_4 = 0$, $e_4 = 2$, η επικάλυψη $o_4 = 2$, γιατί εμφανίζεται ως ελάχιστος κοινός πρόγονος των κόμβων 1 και 3, 2 και 3 στα υποδέντρα γραμμής T_4 και T_{11} , αντίστοιχα, και $D_4 = |\text{skeleton}_4| - e_4 - o_4 = -4$.

$c_4 = D_4 + c_2 + c_3$ (1), τα c_2 και c_3 υπολογίζονται:

Για το c_2 : $j = 2$ ενδιάμεσος κόμβος και γονέας του κόμβου 1. Άρα, $\text{skeleton}_2 = 2$, $e_2 = 1$ και $o_2 = 0$, $D_2 = 2 - 1 - 0 = 1$, $c_2 = D_2 + c_1$ (2)

Για το c_1 : $j = 1$, φύλλο στο δέντρο απαλοιφής άρα: $c_1 = |\text{skeleton}_1| + 1 = 3$
Οπότε η (2) γίνεται: $c_2 = 1 + 3 = 4$

Για το c_3 : $j=3$, αποτελεί φύλλο στο δέντρο απαλοιφής, επομένως, $c_3 = |\text{skeleton}_3| + 1 = 2 + 1 = 3$ Τέλος, η σχέση (1) υπολογίζεται : $c_4 = -4 + 4 + 3 = 3$.

Η εύρεση του πλήθους γραμμής και στήλης του πίνακα L στη φάση της συμβολικής ανάλυσης, είναι σημαντική για την μείωση του χρόνου υπολογισμού της αριθμητικής παραγοντοποίησης, διότι πριν προχωρήσουμε σε αυτή γνωρίζουμε εκ των προτέρων σε ποιες θέσεις του πίνακα L θα τοποθετηθούν οι προκύπτουσες αριθμητικές τιμές.

4.

Παραγοντοποίηση Cholesky

Όταν ο πίνακας A είναι αραιός, τετραγωνικός, συμμετρικός και θετικά ορισμένος, η επίλυση του αραιού γραμμικού συστήματος $Ax = b$, πραγματοποιείται μέσω της παραγοντοποίησης Cholesky. Αρχικά, μεταθέτουμε τον πίνακα A με έναν πίνακα μετάθεσης P , χρησιμοποιώντας μία μέθοδο διάταξης fill – reducing έτσι ώστε $LL^T = PAP^T$, όπου L είναι ένας κάτω τριγωνικός πίνακας με θετικά στοιχεία στην διαγώνιο. Έπειτα, μέσω της συμβολικής ανάλυσης δημιουργούμε δομή δεδομένων για τον πίνακα L , με τα επιμέρους βήματα της εύρεσης του δένδρου απαλοιφής T , την μεταδιάταξη αυτού και τον υπολογισμό του πλήθους στήλης του πίνακα L . Ο υπολογισμός της αριθμητικής παραγοντοποίησης Cholesky $LL^T = PAP^T$ μπορεί να υπολογιστεί από κάποιους αραιούς αλγορίθμους που παρουσιάζονται στο κεφάλαιο αυτό και προηγείται των φάσεων μπρος και πίσω αντικατάστασης για την επίλυση του αρχικού αραιού συστήματος $Ax = b$.

4.1 Up – Looking Cholesky

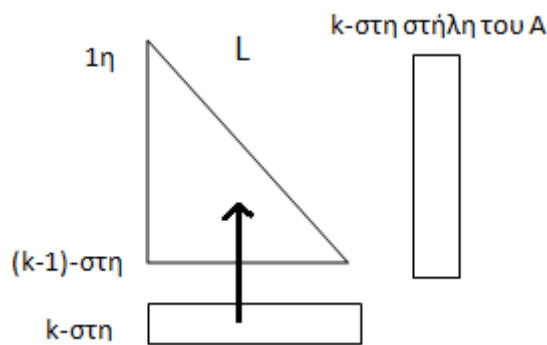
Υποθέτουμε πως «κομματιάζουμε» τους πίνακες A και L , όπως το σύστημα της Παραγράφου 3.1 :

$$\begin{bmatrix} L_{11} & 0 \\ l_{12}^T & l_{22} \end{bmatrix} \begin{bmatrix} L_{11}^T & l_{12} \\ 0 & l_{22} \end{bmatrix} = \begin{bmatrix} A_{11} & a_{12} \\ a_{12}^T & a_{22} \end{bmatrix}$$

Από το οποίο προκύπτουν οι εξισώσεις (3.1) – (3.3) που αναφέραμε:

- $L_{11} L_{11}^T = A_{11}$ (3.1) , επιλύεται αναδρομικά ως προς τον L_{11} με παραγοντοποίηση Cholesky.
- $L_{11} l_{12}^T = a_{12}$ (3.2) είναι η τριγωνική λύση ως προς το l_{12} , γνωρίζοντας τον L_{11} .
- $l_{12}^T l_{12} + l_{22} = a_{22}$ (3.3) , με $a_{22} > l_{12}^T l_{12}$ και η σχέση γίνεται $l_{22} = \sqrt{(a_{22} - l_{12}^T l_{12})}$.

Η μέθοδος up – looking ^[7], που εφαρμόζεται στην (3.1), κατασκευάζει κάθε k-στη γραμμή του L , «κοιτάζοντας» τις k-1 υπολογισμένες γραμμές πάνω από αυτή με την βοήθεια της k-στης στήλης του A, για αυτό φέρει το συγκεκριμένο όνομα.



Σχήμα 4.1 Κατασκευή k-στης γραμμής του πίνακα L

Αναλυτικότερα, υπολογίζεται αναδρομικά το πάνω τριγωνικό κομμάτι του πίνακα L και μόλις αυτό αποκτήσει όλες του τις τιμές , υπολογίζεται η επόμενη γραμμή του πίνακα, η οποία μόλις υπολογιστεί θα προστεθεί στο γνωστό άνω τριγωνικό κομμάτι για τον υπολογισμό της επόμενης γραμμής. Η διαδικασία τερματίζει, μόλις υπολογιστούν όλες οι τιμές της τελευταίας γραμμής. Οι όροι των σχέσεων (3.1) - (3.3) είναι αντιπροσωπευτικοί και αλλάζουν σε κάθε βήμα υπολογισμού.

Η up-looking Cholesky αποτελείται από 3 βασικά βήματα:

- την εύρεση του μη μηδενικού προτύπου της k-στης γραμμής του πίνακα L, μέσω της αναζήτησης κατά βάθος στο δένδρο απαλοιφής.
- την επίλυση του τριγωνικού συστήματος μορφής $Lx = b$, η οποία καθορίζει τις αντίστοιχες αριθμητικές τιμές της k-στης γραμμής του L. Απεικονίζεται με την εξίσωση (3.2)
- τον υπολογισμό της διαγώνιας εγγραφής l_{kk} , που αντιστοιχεί στην εξίσωση (3.3)

Παραθέτουμε ενδεικτικό ψευδοκώδικα υπολογισμού με την μέθοδο:

Up – looking Cholesky:

$$l_{11} = \sqrt{a_{11}}$$

for each row k such that $0 < k < n$ do

//-----Compute the nonzero pattern of L-----//

set L is empty

for each i for which $a_{ik} \neq 0$ do

$$\text{padding-left: 80px;} L_{ki} = \text{Reach}_L(i)$$

end for

// -----Triangular Solve-----//

$l = A * k$ //l is k-th column of matrix A.

$$\text{padding-left: 40px;} d = a_{kk}$$

for each column j for which $l_j \neq 0$ do

$$\text{padding-left: 80px;} l_{kj} = l_j / l_{jj} \quad // \text{ computes the } L(k,j) = l(i) / L(i,i)$$

for each row i > j for which $l_{ij} \neq 0$ do

$$\text{padding-left: 80px;} l_{ki} = l_i - l_{ij} l_{kj}$$

end for

$$\text{padding-left: 40px;} d = d - l_{kj} * l_{kj}$$

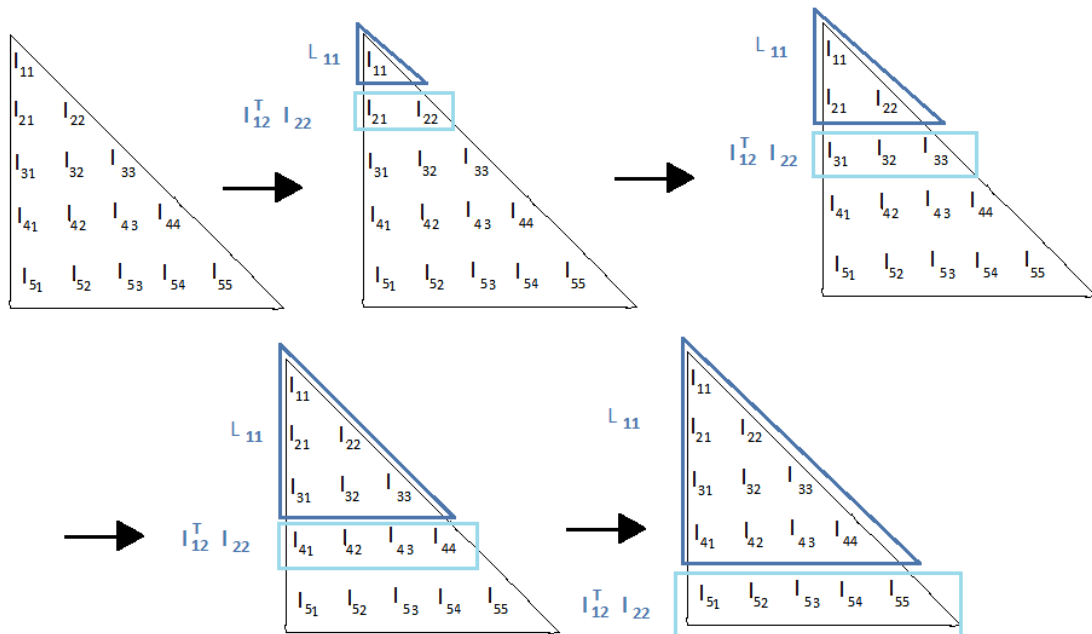
end for

//-----Compute the diagonal entry l_{kk} -----//

$$l_{kk} = \sqrt{(d)}$$

Αρχικά, βρίσκουμε το μη μηδενικό πρότυπο του πίνακα L με ένα πέρασμα μέσω της συνάρτησης $\text{Reach}_L(i)$, η οποία επιστρέφει το σύνολο των κόμβων που είναι προσπελάσιμοι από τον κόμβο i μέσω των μονοπατιών του δένδρου απαλοιφής. Στο κομμάτι της τριγωνικής λύσης θέτουμε το διάνυσμα l ίσο με την k -στη στήλη του πίνακα A , A_{*k} και για κάθε μη μηδενική τιμή του υπολογίζουμε ανά στήλη τις τιμές της k -στης γραμμής του L , καθώς με την μεταβλητή d , υπολογίζουμε το υπόριζο της σχέσης (3.4), που μας χρειάζεται στον τελευταίο υπολογισμό για το διαγώνιο στοιχείο l_{kk} .

Στο Σχήμα 4.2 απεικονίζεται η διαδικασία σχηματικά :



Σχήμα 4.2 Παράδειγμα Εκτέλεσης της Μεθόδου

Ξεκινάμε με τον πίνακα L_{11} ως στοιχείο l_{11} με μπλε πλαίσιο και υπολογίζουμε την επόμενη γραμμή του για τα στοιχεία l_{12} και l_{22} που αντιστοιχούν στα l_{12}^T και l_{22} , των σχέσεων (3.2) και (3.3), με γαλάζιο πλαίσιο. Στην επόμενη επανάληψη, ο πίνακας L_{11} είναι το γνωστό άνω τριγωνικό τμήμα αποτελούμενο από τα στοιχεία l_{11} , l_{12} και l_{22} και υπολογίζουμε την επόμενη γραμμή, η οποία αντιστοιχεί στα διανύσματα

γραμμής στους l_{12}^T και l_{22} , μέσω των σχέσεων (3.2) και (3.3). Αυτό που αλλάζει είναι η ονομασία της απεικόνισης των στοιχείων του πίνακα στην διάρκεια των υπολογισμών της αναδρομής .

4.2 Left – Looking Cholesky

Η left – looking ^{[8][9]} είναι πιο διαδεδομένη μέθοδος από την up-looking, με το σύστημα παραγοντοποίησης να έχει την μορφή:

$$\begin{bmatrix} L_{11} & 0 & \\ l_{12}^T & l_{22} & \\ L_{31} & l_{32} & L_{33} \end{bmatrix} \begin{bmatrix} L_{11}^T & l_{12} & L_{31}^T \\ 0 & l_{22} & l_{32}^T \\ & & L_{33}^T \end{bmatrix} = \begin{bmatrix} A_{11} & a_{12} & A_{31}^T \\ a_{12}^T & a_{22} & a_{32}^T \\ A_{31} & a_{32} & A_{33} \end{bmatrix}$$

Όπου, οι A_{11} και L_{11} είναι πίνακες διαστάσεων $(k-1) \times (k-1)$, τα $[l_{12}^T \ l_{22}]$ και $[a_{12}^T \ a_{22} \ a_{32}^T]$, $[l_{12} \ l_{22}]^T$ και $[a_{12} \ a_{22} \ a_{32}]^T$ είναι διανύσματα γραμμής και στήλης αντίστοιχα, ο A_{33} είναι πίνακας μεγέθους $(n-k) \times (n-k)$.

Στη μέθοδο αυτή ^[8], υποθέτουμε ότι οι πρώτες $k-1$ στήλες και γραμμές του πίνακα L είναι γνωστές, δηλαδή οι $[L_{11} \ l_{12}^T \ L_{31}]^T$, $[L_{11}^T \ l_{12} \ L_{31}^T]$ και υπολογίζουμε κάθε φορά την k -στη στήλη, η οποία αντιστοιχεί στην μεσαία στήλη του L . Επομένως, οι δύο εξισώσεις που προκύπτουν για το σύστημα είναι:

- $l_{12}^T l_{12} + l_{22}^2 = a_{22}$, επιλύεται ως προς το l_{22} , επομένως γίνεται:

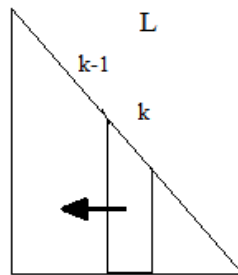
$$l_{22} = \sqrt{a_{22} - l_{12}^T l_{12}} \quad (3.7)$$

- $L_{31} l_{12} + l_{32} l_{22} = a_{32}$, ομοίως ως προς το l_{32} και έχουμε :

$$l_{32} = (a_{32} - L_{31} l_{12}) / l_{22} \quad (3.8)$$

όπου οι l_{22} και l_{32} είναι βαθμωτά μεγέθη, l_{12} και a_{32} είναι διανύσματα και προκύπτει ότι η ποσότητα $(L_{31} l_{12})$ της (3.8) είναι πολλαπλασιασμός πίνακα επί διάνυσμα.

Συγκεκριμένα, υπολογίζουμε σε κάθε βήμα την k -στη στήλη του L , από τις δεξιότερες $k-1$ γνωστές στήλες:



Σχήμα 4.3 Υπολογισμός της k -στης στήλης του πίνακα L .

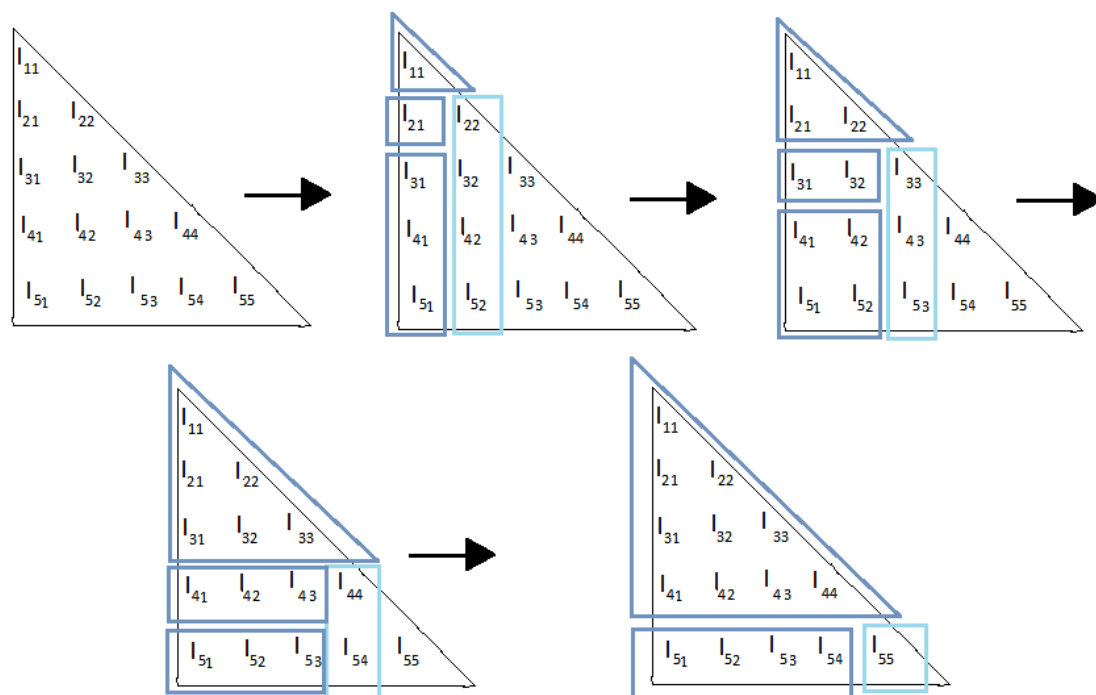
Ο υπολογισμός της παραγοντοποίησης πραγματοποιείται αναδρομικά, μόλις υπολογίσουμε όλα τα στοιχεία που ανήκουν στο δεξιότερο κομμάτι μπορούμε να υπολογίσουμε τις ποσότητες της k -στης στήλης. Απαιτείται η εύρεση του μη μηδενικού προτύπου των στηλών του L αλλά και οι αντίστοιχες αριθμητικές τους τιμές, πριν την αριθμητική παραγοντοποίηση.

Ο αντίστοιχος ψευδοκώδικας είναι ο εξής:

Left - Looking Cholesky:

```
for each column k which is  $1 \leq k \leq n$  do
    for each row i = k until  $i \leq n$ 
         $x_i = a_{ik}$  // x is a vector which keeps entries of  $A^*_{*k}$ 
    end for
     $L_{k*} = \text{Reach}_L(i)$  // the nonzero pattern of the k-th row of L.
    for each j in  $L_{k*}$ 
        for each row i = k until  $i \leq n$ 
             $x_i = x_i - l_{ij} * l_{kj}$  //  $a_{ik} = a_{ik} - l_{ij} * l_{kj}$ 
        end for
    end for
     $l_{kk} = \sqrt{x_k}$  // computes (3.7)
    for each row i = k+1 until  $i \leq n$ 
         $l_{i,k} = x_i / l_{kk}$  // computes (3.8)
    end for
end for
```

Αρχικά, κρατάμε σε ένα διάνυσμα στήλη την αντίστοιχη k-στη στήλη του πίνακα A , A^*_{*k} , ώστε οι απαραίτητες πράξεις για τον υπολογισμό των επιμέρους όρων των σχέσεων να μην αλλοιώσουν τις τιμές του αρχικού αραιού πίνακα. Έπειτα, λαμβάνουμε τις θέσεις του πίνακα L που θα συμμετάσχουν στους υπολογισμούς μέσω της εύρεσης του μη μηδενικού προτύπου της k-στης γραμμής του πίνακα L από το υποδέντρο γραμμής T^k . Τέλος, υπολογίζονται οι εξισώσεις (3.7) και (3.8). Σχηματικά, μπορούμε να απεικονίσουμε την μέθοδο left-looking ως εξής, με την βοήθεια του Σχήματος 4.4:



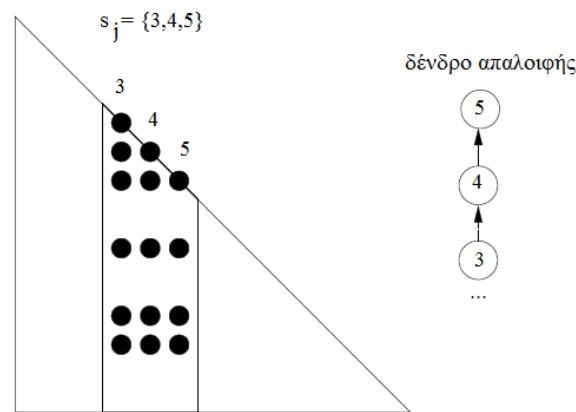
Σχήμα 4.4 Παράδειγμα Εκτέλεσης της Μεθόδου

Ομοίως, με τον $up - looking$ κατά την αναδρομή τα στοιχεία των εξισώσεων (3.7) και (3.8), συμβολίζουν στοιχεία του πίνακα, τα οποία αλλάζουν με την πάροδο της εκτέλεσης της μεθόδου. Με σκούρο μπλε απεικονίζονται τα στοιχεία του πίνακα που είναι ήδη υπολογισμένα, δηλαδή οι πρώτες $k-1$ στήλες και με γαλάζιο η τρέχουσα k -στη στήλη προς υπολογισμό.

4.3 Supernodal Cholesky

Η μέθοδος left-looking αποτελεί την βάση για την διεξαγωγή της supernodal μεθόδου. Έστω ότι έχουμε το ίδιο σύστημα παραγοντοποίησης με αυτό της μεθόδου left – looking και πως γνωρίζουμε της $k-1$ πρώτες στήλες του L , με την μόνη διαφορά ότι όλες ομαδοποιούνται σε σύνολα στηλών με όμοιο μη μηδενικό πρότυπο. Αναλυτικότερα, έστω ότι η μεσαία γραμμή και στήλη του συστήματος

και των τριών πινάκων αντιπροσωπεύεται από ένα block γραμμών και στηλών $s_j \geq 1$, στο οποίο ανήκουν όλες οι στήλες που έχουν ταυτόσημα μη μηδενικά πρότυπα με εξαίρεση το διαγώνιο block L_{22} . Ο πίνακας διαστάσεων $|L_f| \times s_j$ αποτελεί τον υπερκόμβο (supernode) j , ο οποίος περιέχει τις στήλες του block s_j και L_f είναι η δεξιότερη στήλη του πίνακα L , η οποία ανήκει στον υπερκόμβο. Σχηματικά, απεικονίζεται το σύνολο s_j στο Σχήμα 4.5 ^[8], με την αντίστοιχη μορφή του δένδρου απαλοιφής.



Σχήμα 4.5 Σύνολο Στηλών και Δένδρο Απαλοιφής

Παρατηρούμε, ότι το δένδρο απαλοιφής έχει μορφή αλυσίδας και ότι μπορούμε να εκμεταλλευτούμε ιδιότητες πυκνών υποπινάκων, σε κάθε block στηλών s_j .

Οι αντίστοιχες εξισώσεις του συστήματος έχουν μορφή^[8]:

- $l_{12}^T l_{12} + l_{22} l_{22}^T = a_{22}$, όπου ο όρος a_{22} είναι υποπίνακας επομένως λόγω σημειογραφίας θα τον απεικονίσουμε ως A_{22} και ο όρος l_{22} ομοίως είναι υποπίνακας πλέον για αυτό τον πολλαπλασιάζουμε με τον ανάστροφό του. Επομένως, η σχέση γίνεται:

$$L_{22} L_{22}^T = A_{22} - l_{12} l_{12}^T \quad (3.9)$$

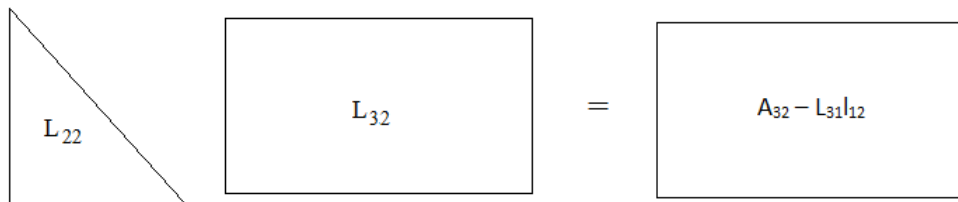
και αποτελεί τον πυκνό υποπίνακα (s_j) του L όπως στο σχήμα 3.26

- $L_{31} l_{12} + l_{32} l_{22}^T = a_{32}$, ομοίως παίρνει την μορφή:

$$L_{32} L_{22}^T = A_{32} - L_{31} l_{12} \quad (3.10)$$

και είναι η τριγωνική λύση του συστήματος μορφής $Lx = b^T$, όπου L αντιστοιχεί στον υποπίνακα L_{22} (η διαγώνια είσοδος-πίνακας του υπερκόμβου j), x στον πυκνό υποπίνακα L_{32} (ο υπερκόμβος κάτω από την διαγώνιο) και ο όρος b στον πυκνό πίνακα $A_{32} - L_{31} l_{12}$.

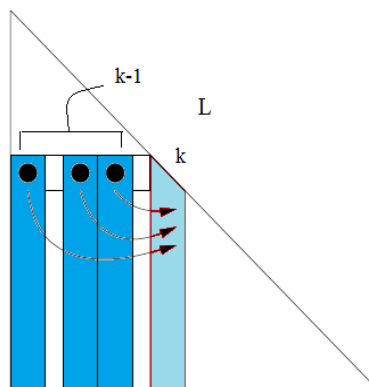
(Να υπενθυμίσουμε στο σημείο αυτό ότι τα διανύσματα και τα βαθμωτά μεγέθη σημειώνονται με πεζά και οι υποπίνακες – πίνακες με κεφαλαία.)



Σχήμα 4.6 Τριγωνικό Σύστημα Μορφής $Lx = b^T$

Τα βασικά βήματα υπολογισμού κατά την supernodal ^[8] μέθοδο και τον υπολογισμό του k -στου block στηλών από τα δεξιότερα $k-1$ blocks είναι τα παρακάτω:

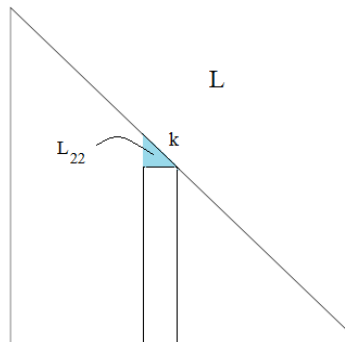
- 1) ο πολλαπλασιασμός των αραιών δεξιότερων blocks : $l_{12} l_{12}^T$



Σχήμα 4.7 Υπολογισμός k -στης στήλης

Όπου, με μαύρο κύκλο συμβολίζεται ο υποπίνακας τον οποίο πολλαπλασιάζουμε με τον αντίστοιχο πίνακα-στήλη που βρίσκεται ακριβώς από κάτω (πολλαπλασιασμός πίνακα επί πίνακα) και το γινόμενο θα το τοποθετήσουμε στην k-στη στήλη με γαλάζιο χρώμα. Με μπλε χρώμα απεικονίζεται ο κάθε υπολογισμένος υπερκόμβος του πίνακα L. Η ίδια διαδικασία επαναλαμβάνεται για κάθε υπερκόμβο με την κατάληξή του στην k-στη στήλη.

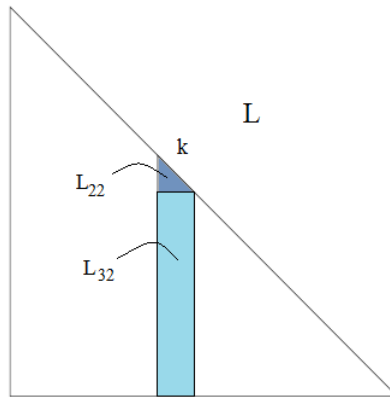
2) Υπολογισμός της πυκνής παραγοντοποίησης Cholesky, για την εύρεση του L_{22} .⁴



Σχήμα 4.8 Υπολογισμός L_{kk}

3) ο υπολογισμός της κάτω τριγωνικής λύσης του συστήματος, που αντιστοιχεί στη σχέση (3.10), διαιρώντας το με το k-στο διαγώνιο block του πίνακα L.

⁴ Μπορούμε να διεξάγουμε τον υπολογισμό αυτό μέσω της συνάρτησης του Matlab chol(A): η οποία χρησιμοποιεί μόνο το διαγώνιο και άνω τριγωνικό τμήμα του πίνακα, ο οποίος αν είναι θετικά ορισμένος επιστρέφεται ο πίνακας $R : RR^T = A$.



Σχήμα 4.9 Υπολογισμός Κάτω Τριγωνικής Λύσης

Ακολουθεί ο ψευδοκώδικας για την μέθοδο supernodal :

Supernodal Cholesky

```

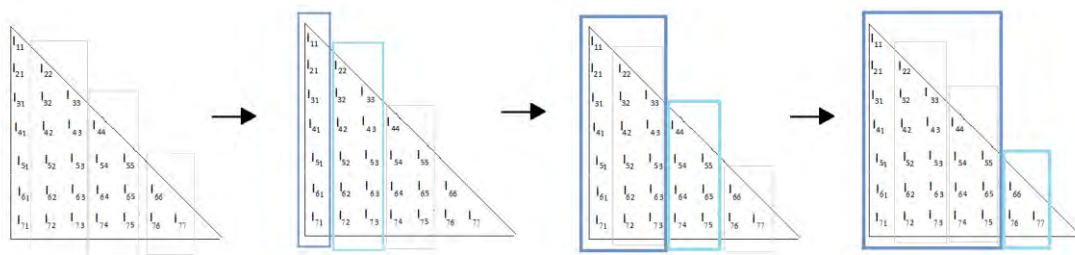
for each supernode j
    start = start(j)    //start of supernode j
    end =start(j+1)    //end of supernode j
    for each k = start until k = end-1
        for each column c = 1 until c ≤ start -1
            product1 = product1 + Lk,c*Lk,cT // l12*l12T
        end for
        Lkk = cholesky_dense(Akk-product1) //computes (3.9)
        for each row r = end until r ≤ n
            for each column c = 1 until c ≤ start - 1
                product2 = product2 + Lr,c*Lk,cT //L31*l12
            end for
            Lrk = (Ar,k - product2) / LkkT //computes (3.10)
        end for
    end for
end for

```

Υπολογίζουμε αναδρομικά για κάθε υπερκόμβο j, τα όρια του και τις σχέσεις (3.9) και (3.10). Για την διεξαγωγή της πρώτης χρησιμοποιούμε την συνάρτηση παραγοντοποίησης Cholesky για πυκνούς πίνακες, η οποία λαμβάνει ως παράμετρο έναν θετικά ορισμένο πίνακα A και επιστρέφει έναν άνω τριγωνικό πίνακα L, για τον οποίο

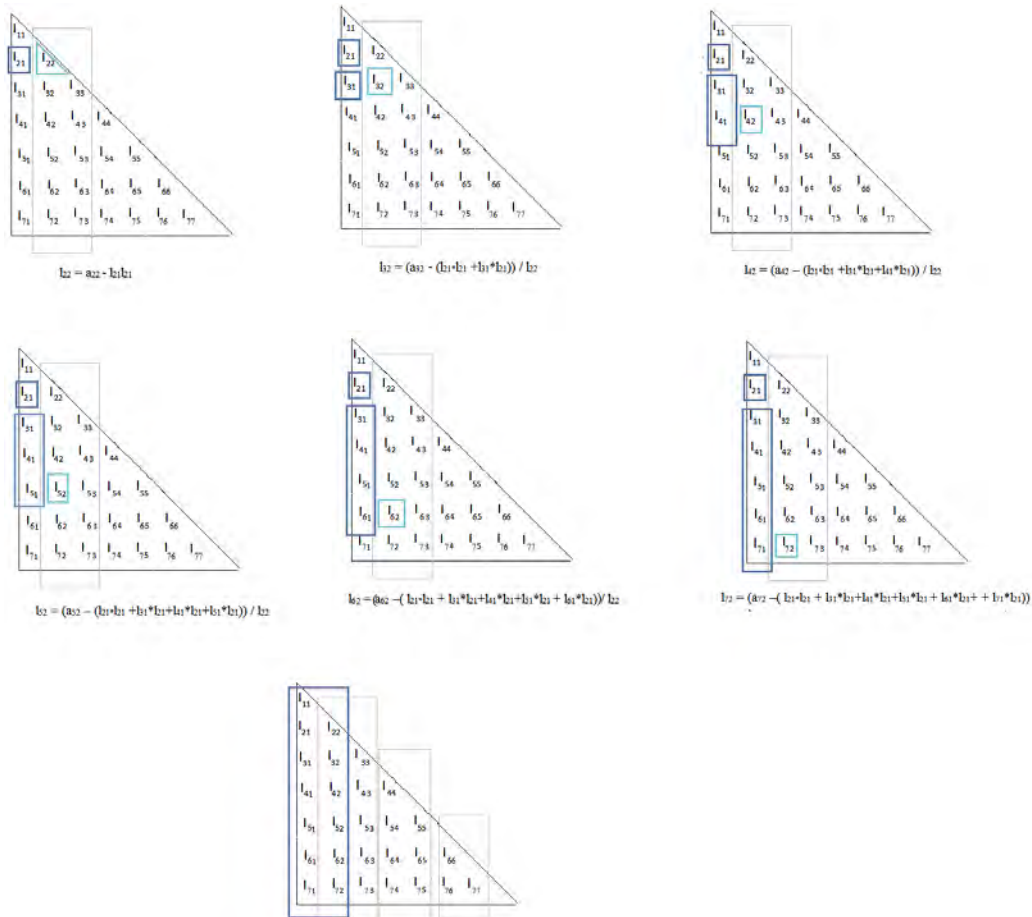
ισχύει ότι $LL^T = A$. Ουσιαστικά, ο κώδικας μέχρι και την εντολή για την εξίσωση (3.9) υπολογίζει τον διαγώνιο υποπίνακα του k -στου συνόλου στηλών του Σχήματος 4.8. Με τις υπόλοιπες εντολές υπολογίζεται ο πυκνός υποπίνακας του Σχήματος 4.9.

Στο Σχήμα 4.10 απεικονίζεται σχετικό παράδειγμα, όπου οι στήλες σε γκρι πλαίσιο δηλώνουν τα σύνολα s_j . Κατά την εκτέλεση της μεθόδου υπολογίζονται αναδρομικά τα στοιχεία της k -στης στήλης του j -στου υπερκόμβου από τις δεξιότερες γνωστές $k-1$ στήλες. Οι ονομασίες των στοιχείων και σε αυτή την μέθοδο είναι συμβολικές, καθώς προχωράμε μόνο όταν οι πίνακες έχουν υπολογισμένα όλα τα στοιχεία τους.



Σχήμα 4.10 Παράδειγμα Εκτέλεσης της Μεθόδου

Στο Σχήμα 4.11, απεικονίζεται αναλυτικότερα τα βήματα υπολογισμού : υποθέτοντας ότι η πρώτη στήλη είναι γνωστή : τα όρια του υπερκόμβου είναι $start = 2, end = 3$. Σε γαλάζιο πλαίσιο εμφανίζεται το στοιχείο προς υπολογισμό και με μπλε πλαίσιο τα στοιχεία που λαμβάνουν μέρος στις πράξεις. Για κάθε k -στη στήλη που υπολογίζουμε, να υπενθυμίσουμε ότι χρησιμοποιούμε και την αντίστοιχη k -στη στήλη του A , η οποία δεν εμφανίζεται στο σχήμα. Τέλος, το κάθε βήμα συνοδεύεται από την αντίστοιχη σχέση (3.10).



Σχήμα 4.11 Αναλυτικό Παράδειγμα Εφαρμογής της Μεθόδου

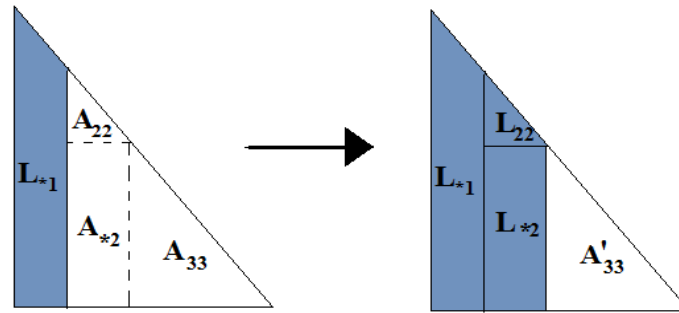
Παρατηρούμε, ότι ο αντίστοιχος υπολογισμός του κάθε στοιχείου l_{ik} γίνεται αναδρομικά από το συνολικό άθροισμα του γινομένου του στοιχείου $l_{k,k-1}$ με τα στοιχεία της υπόλοιπης στήλης l_{ik-1} που αντιστοιχούν στο ύψος της γραμμής i . Για παράδειγμα, το l_{52} υπολογίζεται από το $l_{21}*(l_{21} + l_{31} + l_{41} + l_{51})$.

Στον τελευταίο πίνακα, έχουμε ήδη υπολογίσει την πρώτη στήλη του υπερκόμβου και μπορούμε να συνεχίσουμε στην επόμενη ($k = 3$).

4.4 Right-Looking Cholesky

Αποτελεί μία υποπερίπτωση της right - looking LU παραγοντοποίησης που θα δούμε στο επόμενο κεφάλαιο, για την

παραγοντοποίηση ενός θετικά ορισμένου πίνακα A , από έναν κάτω τριγωνικό πίνακα L . Η ονομασία του οφείλεται στην τροποποίηση του δεξιότερου πίνακα A κατά τον υπολογισμό της k -στης στήλης του πίνακα L , ο οποίος χρησιμοποιείται στον υπολογισμό της $k+1$ -της στήλης. Σχηματικά, απεικονίζεται ένα στιγμιότυπο στο Σχήμα 4.12^[10] :



Σχήμα 4.12 Υπολογισμός της k -στης στήλης του L

όπου βλέπουμε ότι έχοντας υπολογίσει ήδη την πρώτη στήλη του L , προχωράμε στην εύρεση της δεύτερης στήλης L_{*2} με την βοήθεια της δεύτερης στήλης του πίνακα A , A_{*2} . Μετά το πέρας της διαδικασίας, το δεξιότερο κομμάτι του πίνακα A έχει τροποποιηθεί σε A_{33}' .

Το σύστημα παραγοντοποίησης^[11] που χρησιμοποιούμε για την διεξαγωγή της μεθόδου είναι το παρακάτω:

$$\begin{bmatrix} l_{11} & 0 \\ l_{21} & L_{22} \end{bmatrix} \begin{bmatrix} l_{11}^T & l_{21}^T \\ 0 & L_{22}^T \end{bmatrix} = \begin{bmatrix} a_{11} & a_{21}^T \\ a_{21} & A_{22} \end{bmatrix}$$

Όπου, οι όροι l_{11} , a_{11} είναι βαθμωτά μεγέθη, οι L_{22} , A_{22} είναι υποπίνακες και οι υπόλοιποι όροι είναι διανύσματα στήλης ή γραμμής. Οι αντίστοιχες εξισώσεις που προκύπτουν:

- $l_{11} l_{11}^T = a_{11}$, ως προς l_{11} και επειδή είναι βαθμωτό μέγεθος γίνεται:

$$l_{11} = \sqrt{a_{11}} \quad (3.11)$$

- $l_{21} l_{11}^T = a_{21}$, ως προς l_{21} διαιρώντας το διάνυσμα a_{21} με τον l_{11} .
(3.12)

- $l_{21} l_{21}^T + L_{22} L_{22}^T = A_{22}$, ως προς τον L_{22} : $L_{22} L_{22}^T = A_{22} - l_{21} l_{21}^T$
(3.13)

Όπως και στις προηγούμενες μεθόδους έτσι και σε αυτή οι όροι του συστήματος είναι αντιπροσωπευτικοί.

Ο αντίστοιχος κώδικας της μεθόδου είναι ο εξής:

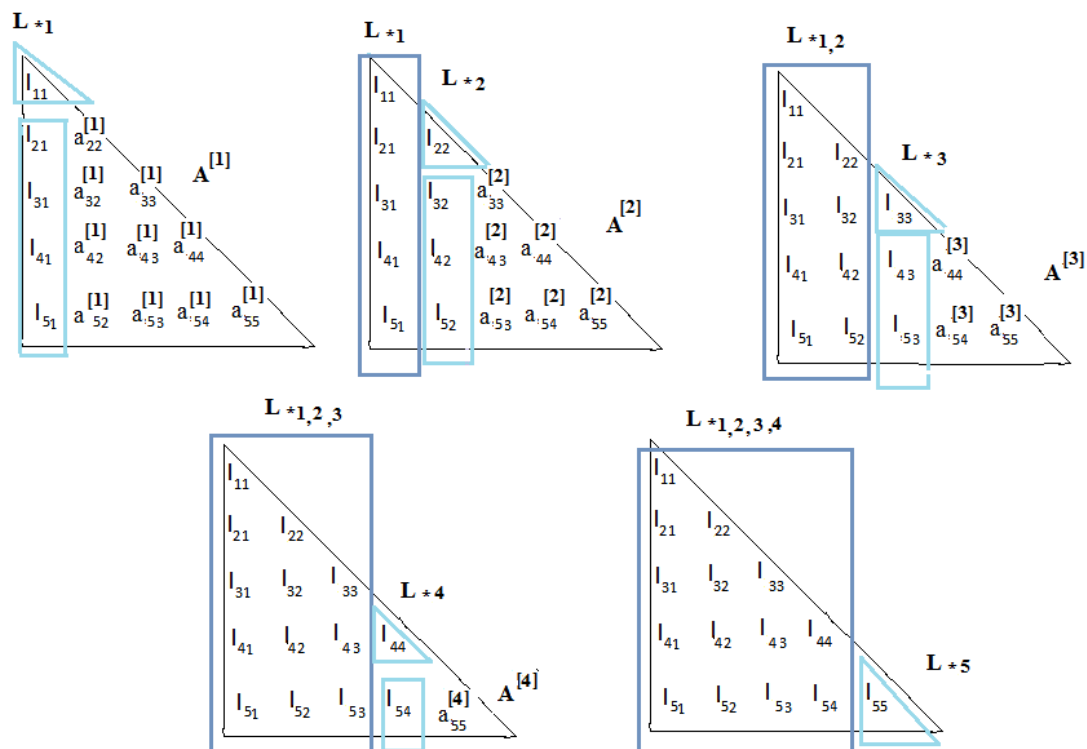
Right – looking Cholesky

```
for column k = 1 until k ≤ n do
     $l_{kk} = \sqrt{a_{kk}}$ 
    for row i = k+1 until i ≤ n do
         $l_{ik} = a_{ik} / l_{kk}$ 
    end for
    for row i = k+1 until i ≤ n do
        for column j = k+1 until j ≤ n do
             $a_{ij} = a_{ij} - l_{ik} * l_{jk}^T$  //entries of A must be modified.
        end for
    end for
end for
```

Αρχικά , υπολογίζουμε την διαγώνια ποσότητα του πίνακα L , έπειτα στον βρόγχο υπολογίζουμε τα υπόλοιπα στοιχεία της τρέχουσας στήλης και τέλος τροποποιούμε τα στοιχεία του δεξιού κομματιού του πίνακα A , τα οποία θα χρησιμοποιηθούν στην επόμενη επανάληψη.

Στο Σχήμα 4.13 απεικονίζεται ένα στιγμιότυπο της μεθόδου , κατά το οποίο με γαλάζιο πλαίσιο είναι οι εγγραφές του πίνακα L , της τρέχουσας στήλης που μόλις έχουν υπολογιστεί στο βήμα αυτό και με μπλε οι ήδη υπολογισμένες $k-1$ πρώτες υπολογισμένες στήλες. Επίσης, σε κάθε βήμα τροποποιούμε τις τιμές του πίνακα A , ο οποίος συμβολίζεται

με τις εγγραφές του να φέρουν έναν δείκτη που μας πληροφορεί για την τάξη της τροποποίησης.



Σχήμα 4.13 Παράδειγμα Εφαρμογής της Μεθόδου

Η right-looking μέθοδος αποτελεί την βάση της μεθόδου multifrontal Cholesky , μίας ειδικής περίπτωσης της multifrontal LU, που θα δούμε στο επόμενο κεφάλαιο, για έναν θετικά ορισμένο πίνακα A .

Θα περιγράψουμε μία μικρή εισαγωγή της μεθόδου στην επόμενη παράγραφο.

4.5 Multifrontal Cholesky

Βασίζεται στον διαχωρισμό των πινάκων L και A ,που δημιουργεί το σύστημα της right –looking Cholesky. Το σκεπτικό είναι παρόμοιο με αυτό της supernodal Cholesky, κατά την οποία ομαδοποιούμε στήλες με όμοιο μη μηδενικό πρότυπο, με την διαφορά πως στην multifrontal^[11] οι

ομαδοποιημένες στήλες του πίνακα L αντιπροσωπεύονται από έναν πυκνό πίνακα απεικόνισης, frontal matrix, με διαστάσεις $|L_f| \times |L_f|$, όπου L_f είναι το μη μηδενικό πρότυπο της πρώτης στήλης του πίνακα απεικόνισης.

Ο αλγόριθμος της multifrontal θα αναλυθεί στο επόμενο κεφάλαιο για τις μορφές παραγοντοποίησης LU, επειδή είναι παρόμοια με την Cholesky που είναι για συμμετρικό θετικά ορισμένο πίνακα.

4.6 Χρήση της Παραγοντοποίησης Cholesky

Το αραιό γραμμικό σύστημα $Ax = b$, με A αραιό συμμετρικό θετικά ορισμένο πίνακα, b το διάνυσμα του δεξιού μέλους της εξίσωσης και x το διάνυσμα της λύσης, επιλύεται μέσω της παραγοντοποίησης Cholesky, όπως αναφέραμε και στην αρχή του κεφαλαίου.

Έστω ο πίνακας μετάθεσης P που προκύπτει από κάποια μέθοδο fill-reducing, με τον αρχικό πίνακα A να γίνεται $A = PAP^T$. Μέσω της συμβολικής ανάλυσης και κάποιας από τις μεθόδους παραγοντοποίησης Cholesky βρίσκουμε τον παράγοντα L , για τον οποίο ισχύει ότι $LL^T = PAP^T$.

Το αρχικό σύστημα παίρνει την μορφή : $PAP^T Px = Pb$, έχοντας μεταθέσει το διάνυσμα b και λύνοντας ως προς x :

$Ly = Pb$, ως προς το διάνυσμα y με την μέθοδο εμπρός αντικατάστασης^B.

$L^T z = y$, ως προς το διάνυσμα z , με την μέθοδο προς τα πίσω αντικατάστασης^C.

και τέλος $x = P^T z$, από όπου παίρνουμε την τελική λύση.

5.

Παραγοντοποίηση LU

Η παραγοντοποίηση LU είναι παλαιότερη από την Cholesky και βασίζεται στην απαλοιφή του Gauss. Για το γραμμικό σύστημα $Ax = b$, με A αραιό μη συμμετρικό τετραγωνικό πίνακα, μεταθέτουμε τον A με κάποια μέθοδο fill-reducing, κάνουμε συμβολική ανάλυση για τον παράγοντα L , όπως στην Cholesky με την διαφορά ότι μας ενδιαφέρουν οι αριθμητικές τιμές των εγγραφών του A , γιατί θέλουμε η διαγώνιος να φέρει τις μεγαλύτερες αυτών. Για το λόγο αυτό κάνουμε μερική οδήγηση. Έπειτα, ο πίνακας A διασπάται στην μορφή $A = LU$, όπου L είναι κάτω τριγωνικός και U άνω τριγωνικός πίνακας και τέλος το σύστημα επιλύεται με μεθόδους αντικατάστασης. Στο κεφάλαιο αυτό θα παρουσιάσουμε κάποιες από τις μεθόδους της παραγοντοποίησης LU.

5.1 Μερική Οδήγηση

Κατά την διαδικασία της μερικής οδήγησης, στο k -στο βήμα της τριγωνοποίησης ενός πίνακα, έστω $A^{[k]}$ ο τροποποιημένος μέχρι το τρέχον βήμα πίνακας A που τριγωνοποιείται, εξετάζουμε όλα τα στοιχεία της k -στης υποστήλης $A_{*k}^{[k]}$, βρίσκουμε ένα από αυτά με την μεγαλύτερη απόλυτη τιμή και με εναλλαγή γραμμών το φέρνουμε στην διαγώνιο, όπου είναι και η αντίστοιχη θέση του οδηγού. Η μερική οδήγηση συναντάται στην απαλοιφή του Gauss ^[12], η οποία είναι και η βάση για την παραγοντοποίηση LU με μερική οδήγηση, όπου στην δεύτερη ο πίνακας A αναλύεται σε γινόμενο παραγόντων $PA = LU \Rightarrow A = P^{-1}LU$, με P τον πίνακα μετάθεσης που καταγράφει τις εναλλαγές γραμμών, L ο κάτω τριγωνικός πίνακας με μονάδες στην διαγώνιο και

τους πολλαπλασιαστές της απαλοιφής και U ο άνω τριγωνικός πίνακας που φέρει το τελικό προϊόν της τριγωνοποίησης. Η μέθοδος αυτή θα μας βοηθήσει στην διεξαγωγή των μεθόδων LU που θα περιγράψουμε στην συνέχεια.

5.2 Left - Looking LU

Στον αλγόριθμο της left – looking LU ^[13] υπολογίζουμε κάθε φορά μία στήλη των L και U από αριστερά προς τα δεξιά. Στο k-στο βήμα , διαπερνάμε τις k-1 πρώτες στήλες του L και την k-στη στήλη του πίνακα A, για την εύρεση της k-στης στήλης των L και U. Αν αγνοήσουμε την μερική οδήγηση μπορούμε να απεικονίσουμε την left-looking από το παρακάτω σύστημα , στο οποίο οι πίνακες, A ,L ,U κομματιάζονται σε 3x3 block πινάκων, παρόμοια με του left-looking Cholesky.

$$\begin{bmatrix} L_{11} & & \\ l_{21} & 1 & \\ L_{31} & l_{32} & L_{33} \end{bmatrix} \begin{bmatrix} U_{11} & u_{12} & U_{13} \\ & u_{22} & u_{23} \\ & & U_{33} \end{bmatrix} = \begin{bmatrix} A_{11} & a_{12} & A_{13} \\ a_{21} & a_{22} & a_{23} \\ A_{31} & a_{32} & A_{33} \end{bmatrix}$$

Θεωρούμε ότι ο πίνακας L έχει μοναδιαία διαγώνιο και πως η προς υπολογισμό k-στη γραμμή και στήλη είναι η μεσαία γραμμή και στήλη των πινάκων. Οι τρεις εξισώσεις που προκύπτουν είναι:

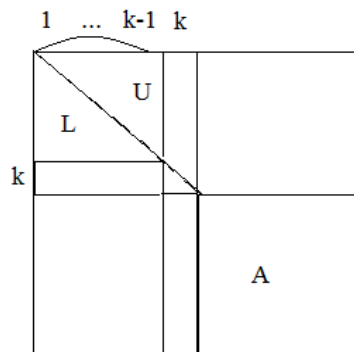
$L_{11}u_{12} = a_{12}$ (5.1), το τριγωνικό σύστημα που επιλύεται ως προς το u_{12} .

$l_{21}u_{12} + u_{22} = a_{22}$ (5.2), επιλύεται ως προς την εγγραφή οδήγησης u_{22}

$L_{32}u_{12} + l_{32}u_{22} = a_{32}$ (5.3), ομοίως ως προς την l_{32} .

Ομοίως, με το προηγούμενο κεφάλαιο με πεζά αντιπροσωπεύουμε διανύσματα στηλών και γραμμών ή στοιχεία πίνακα και με κεφαλαία,

υποπίνακες των αρχικών πινάκων. Στο Σχήμα 5.1 ^[13] απεικονίζεται το k-στο βήμα υπολογισμού της αντίστοιχης στήλης των πινάκων L, U με την χρήση των τιμών της k-στης στήλης του πίνακα A και των k-1 ήδη υπολογισμένων στηλών των L και U.



Σχήμα 5.1 Υπολογισμός της k-στης στήλης των πινάκων L και U

Παρατηρούμε πως αν θέσουμε:

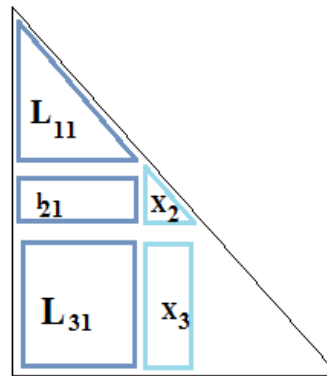
- $x_1 = u_{12}$
- $x_2 = u_{22}$
- $x_3 = l_{32}u_{22}$

μπορούμε να μετατρέψουμε τις εξισώσεις σε μία αραιή τριγωνική λύση της μορφής:

$$\begin{bmatrix} L_{11} & & \\ I_{21} & 1 & \\ L_{31} & 0 & I \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} a_{11} \\ a_{21} \\ a_{31} \end{bmatrix}$$

Όπου η πρώτη στήλη του πίνακα L αντιπροσωπεύει τις πρώτες k-1 γνωστές στήλες του , το διάνυσμα με τις εγγραφές του πίνακα A είναι η k-στη στήλη που χρησιμοποιείται σε κάθε βήμα των υπολογισμών και η εύρεση του διανύσματος x επιλύει το σύστημα των τριών εξισώσεων.

Μέχρι τώρα έχουμε αγνοήσει την μερική οδήγηση, λόγω των ανεξάρτητων υπολογισμών του διανύσματος x από την μετάθεση γραμμών σε κάθε k -στη οδηγό – στήλη. Σχηματικά, ένα στιγμιότυπο του υπολογισμού του διανύσματος x φαίνεται το Σχήμα 5.2^[13].



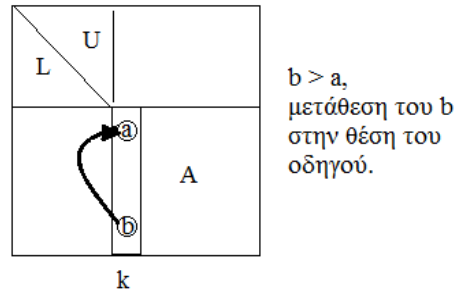
Σχήμα 5.2 Στιγμιότυπο Υπολογισμού του διανύσματος x

Η βασική ιδέα της μερικής οδήγησης είναι πως ο παράγοντας της οδήγησης μπορεί να μεταφέρεται πριν ξεκινήσει ο πίνακας να παραγοντοποιείται, αυτό προκύπτει από το γεγονός ότι η διάταξη των γραμμών στα τμήματα b_1 και L_{31} (σχήμα 5.2) δεν εξαρτάται από την μέχρι τώρα εκτέλεση. Ακόμα και να ανταλλάσσαμε κάποιες από τις γραμμές των τμημάτων αυτών, οι τιμές θα παρέμεναν ίδιες με την μόνη διαφορά ότι θα βρίσκονταν σε διαφορετικές θέσεις. Επομένως, τα x_2 και x_3 μπορούν να μετατεθούν τυχαία, αποθηκεύοντας πάντα το είδος της μετάθεσης.

Αντίθετα, στο κομμάτι του υποπίνακα L_{11} η μετάθεση είναι σταθερή (δεν αλλάζει) και έχει σημασία γιατί υπάρχει διάταξη και είναι τριγωνική λύση, καθώς περιέχει τις οδηγούς γραμμές.

Στο Σχήμα 5.3 απεικονίζεται το k -στο βήμα, κατά το οποίο παίρνουμε την k -στη στήλη του πίνακα A που αντιστοιχεί στην στήλη των x_2, x_3 του Σχήματος 5.2, βρίσκουμε την μεγαλύτερη τιμή της και την τοποθετήσουμε στην διαγώνιο με μετάθεση γραμμών, κατά την

οποία ο πίνακας U δεν επηρεάζεται. Ουσιαστικά, μας ενδιαφέρει η γραμμή i που θα γίνει η k -στη γραμμή οδήγησης, η οποία δεν μπορεί να επιλεγεί πάνω από μια φορά.



Σχήμα 5.3 k -στο Βήμα Υπολογισμού με Μερική Οδήγηση

Από τα παραπάνω συνοψίζουμε για τους πίνακες τα εξής:

- ο πίνακας A προσπελάζεται από αριστερά προς τα δεξιά ανά μία στήλη την φορά.
- ο U φέρει τις $k-1$ γραμμές οδηγούς και προσπελάζεται από αριστερά προς τα δεξιά.
- ο πίνακας L είναι δυναμικός, διότι προσπελάζεται μία στήλη την φορά και επειδή ο αλγόριθμος είναι *left – looking*, πηγαίνουμε σε κάθε βήμα πίσω στις $k-1$ γνωστές στήλες για την εκτέλεση της τριγωνικής λύσης.

Ο αντίστοιχος ψευδοκώδικας:

Left – looking LU:

```

for each column k which is  $1 \leq k \leq n$  do
    row_piv = k
    for each row  $k \leq i \leq n$ 
         $x_i = a_{ik}$  // x is a vector which keeps entries of  $A^*_{*k}$ 
    end for
     $X = \text{Reach}_L(i)$  // the nonzero pattern of the k-th row of L.
//-----Triangular solve -----//
    for each j which is  $x_j \neq 0$ 
         $x_j = x_j / l_{kj}$ 
        for each row  $i > j$  for which  $l_{ij} \neq 0$ 
             $x_i = x_i - l_{ij}x_j$ 
        end for
    end for
//-----Find pivot -----//
    pivot =  $x_{kk}$ 
    for each row  $k \leq i \leq n$ 
        if ( $|x_i| > \text{pivot}$ ) then
            pivot =  $x[i]$ 
            row_piv = i
        end if
    end for
    if ( $k \neq \text{row\_piv}$ ) then
        swap( $L_{k*}, L_{\text{row\_piv}*}$ )
    end if
//----- Finds the k-th column of L and U-----//
    for each row  $1 \leq i \leq k-1$ 
         $u_{ik} = x[i]$  // the k-th column of U
    end for
     $u_{kk} = \text{pivot}$ 
     $l_{kk} = 1$ 
    for each row  $k+1 \leq i \leq n$ 
         $l_{i,k} = x_i / u_{kk}$  //the k-th column of L
    end for
end for

```

Ο κώδικας αυτός χωρίζεται σε τμήματα υπολογισμών για την καλύτερη κατανόηση του. Συγκεκριμένα:

- την εύρεση του μη μηδενικού προτύπου της k -στης γραμμής του L ,
- τον υπολογισμό της τριγωνικής λύσης,
- την εύρεση του οδηγού στοιχείου της k -στης γραμμής,
- και την εύρεση των τιμών της k -στης στήλης των L και U .

Αρχικά, θέτουμε την μεταβλητή της γραμμής οδηγού , row_pivot , ίση με την k -στη ,η οποία αντιστοιχεί στην στήλη που υπολογίζουμε.

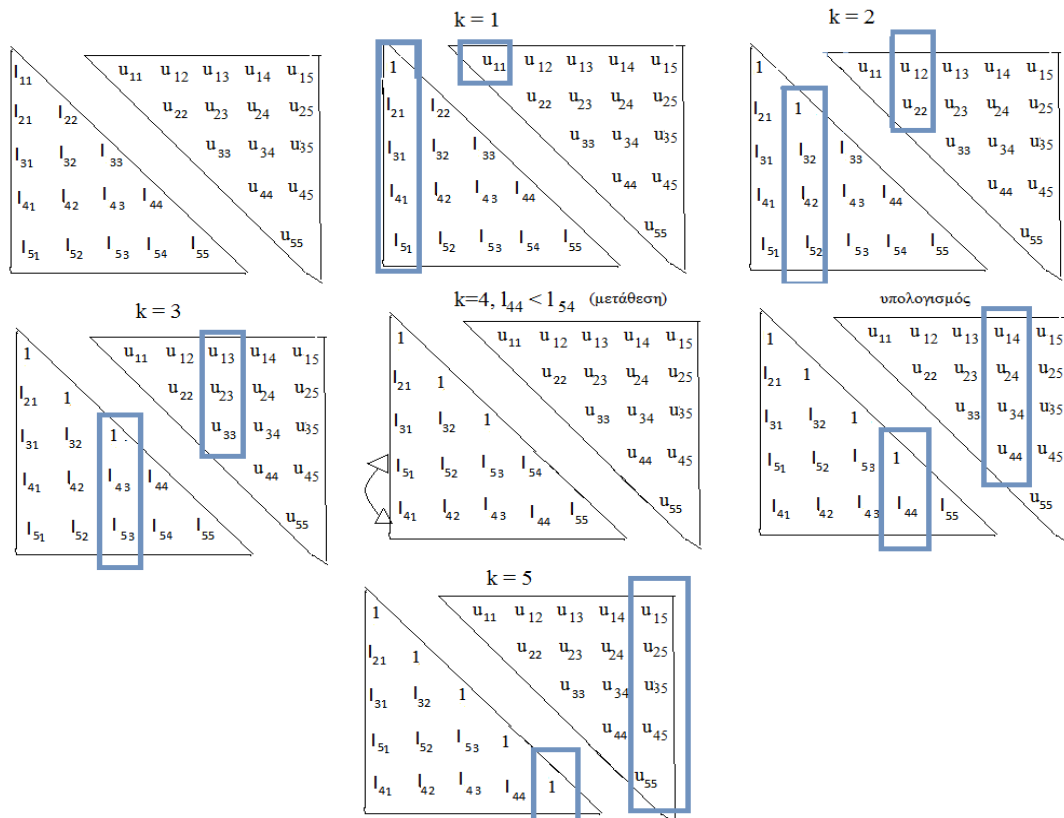
Το κομμάτι της εύρεσης του μη μηδενικού προτύπου και της τριγωνικής λύσης τα έχουμε ξανασυναντήσει στο προηγούμενο κεφάλαιο.

Η εύρεση του οδηγού στοιχείου πραγματοποιείται αν πάρουμε την απόλυτη τιμή του κάθε στοιχείου k -στης στήλης και το συγκρίνουμε με την τιμή της μεταβλητής row_pivot , η οποία φέρει τον αντίστοιχο οδηγό και αρχικοποιείται σε κάθε επανάληψη του αλγορίθμου με το k -στο διαγώνιο στοιχείο. Έπειτα, έχοντας καθορίσει τον οδηγό, αν η γραμμή που βρίσκεται δεν είναι η ίδια με την k -στη ,ανταλλάσσουμε τις δύο γραμμές (μετάθεση γραμμών) μέσω της συνάρτησης $swap$. Διαφορετικά, το οδηγό στοιχείο είναι στην k -στη γραμμή και άρα δεν χρειάζεται μετάθεση γραμμών. Η μετάθεση των γραμμών γίνεται στον πίνακα L , γιατί όπως περιγράψαμε παραπάνω δεν μας απασχολεί η μετάθεση γραμμών κάτω από τον υποπίνακα L_{11} .

Στο τελευταίο κομμάτι, έχουμε τον υπολογισμό της k -στης στήλης του U πίνακα, U_{*k} , μέσω του διανύσματος x που φέρει τις τροποποιημένες τιμές του πίνακα A και το διαγώνιο στοιχείο του u_{kk} θα γίνει ίσο με τον k -στο οδηγό. Για τον πίνακα L , φροντίζουμε το διαγώνιο

στοιχείο l_{kk} να είναι ίσο με την μονάδα και υπολογίζουμε την υπόλοιπη k -στη στήλη L_{*k} με την διαίρεση κάθε στοιχείου της αντίστοιχης γραμμής του πίνακα U με τον οδηγό, γιατί ο L φέρει τους πολλαπλασιαστές της παραγοντοποίησης.

Ένα ενδεικτικό παράδειγμα των υπολογισμών φαίνεται στο Σχήμα 5.4, όπου με μπλε πλαίσιο χρωματίζονται τα τμήματα των L και U που έχουν υπολογιστεί στο k -στο βήμα. Συγκεκριμένα, η k -στη στήλη του L φέρει την μονάδα στο διαγώνιο στοιχείο και οι υπόλοιπες τιμές που βρίσκονται κάτω από αυτό υπολογίζονται από την σχέση $l_{ik} = x_i / u_{kk}$, για την αντίστοιχη στήλη του U υπολογίζονται ομοίως οι τιμές από την σχέση $u_{ik} = x_i$. Μόλις, βρεθούν όλα τα στοιχεία, συγχωνεύονται με τις ήδη υπολογισμένες και προχωράμε στον υπολογισμό της επόμενης ($k+1$ -ης) γραμμής και στήλης. Στο πέμπτο βήμα, μετατίθενται οι γραμμές 4, 5 γιατί το διαγώνιο στοιχείο l_{44} είναι μικρότερο από το l_{55} που ανήκει στα $[k + 1, n]$ στοιχεία της στήλης. Φυσικά, δεν θα υπήρχε κάποια διαφορά στο αποτέλεσμα, όπως αναφέραμε παραπάνω για τον L_{11} που περιέχει τις γραμμές οδηγούς και τους l_{21}, L_{31} που μπορούν να ανταλλάξουν τις γραμμές τους κατά τον υπολογισμό της οδηγού γραμμής.



Σχήμα 5.4 Παράδειγμα Υπολογισμού της Μεθόδου

Ο αλγόριθμος αυτός είναι ασυμπτωτικά τέλειος και χρησιμοποιείται μόνο για την περίπτωση που ο πίνακας A είναι αραιός μη συμμετρικός και θετικά ορισμένος πίνακας.

5.3 Right - Looking LU

Η απαλοιφή του Gauss αποτελεί μία περίπτωση της μεθόδου right-looking LU ^[14]. Στο κομμάτι αυτό θα κάνουμε μια εισαγωγή στην μέθοδο αυτή, γιατί αποτελεί την βάση για την multifrontal που θα παρουσιάσουμε αναλυτικά στην επόμενη παράγραφο.

Η right-looking μοιάζει σε σκεπτικό με την right - looking Cholesky, στο σημείο υπολογισμού της k -στης στήλης των πινάκων L , U και στην τροποποίηση του δεξιότερου κομματιού του πίνακα A , το οποίο

χρησιμοποιείται σε κάθε βήμα των υπολογισμών. Το σύστημα, που χρησιμοποιούμε για να την εκφράσουμε είναι το εξής:

$$\begin{bmatrix} l_{11} & 0 \\ l_{21} & L_{22} \end{bmatrix} \begin{bmatrix} u_{11} & u_{12} \\ 0 & U_{22} \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & A_{22} \end{bmatrix}$$

Όπου ο $l_{11} = 1$ είναι βαθμωτό στοιχείο και οι υπόλοιποι όροι είναι τετράγωνοι πίνακες, με όμοιο διαχωρισμό στηλών και γραμμών. Υπάρχουν και άλλοι τρόποι επιλογής της τιμής του l_{11} . Ωστόσο, αυτός οδηγεί σε ένα κάτω τριγωνικό πίνακα L με τις τέσσερις εξισώσεις:

- $u_{11} = a_{11} \quad (5.4)$

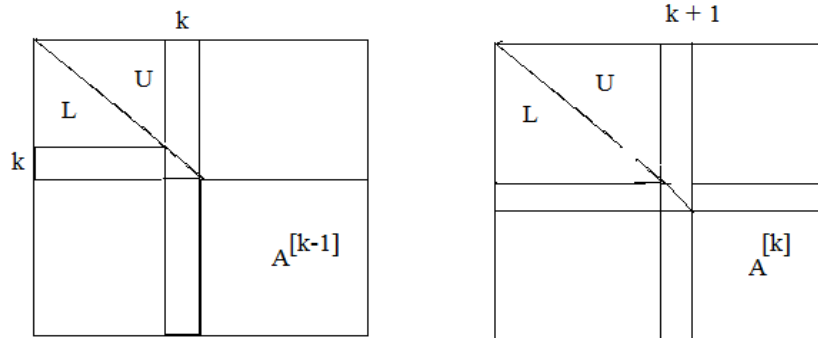
- $u_{12} = a_{12} \quad (5.5)$

- $l_{21} u_{11} = a_{21} \quad (5.6)$

- $l_{21} u_{12} + L_{22} U_{22} = A_{22} \quad (5.7)$

Σε κάθε βήμα του αλγορίθμου, το εξωτερικό γινόμενο της στήλης και της γραμμής οδηγού αφαιρείται από τον κάτω δεξί υποπίνακα A . Χρησιμοποιώντας μερική οδήγηση, οι γραμμές του A μετατίθενται, ώστε η διαγώνια εγγραφή $|u_{kk}|$ να είναι η μέγιστη στο κάθε βήμα.

Ένα στιγμιότυπο της μεθόδου απεικονίζεται στο Σχήμα 5.5, όπου έχουμε υπολογίσει μέχρι το $k-1$ ο βήμα τους L και U κατά στήλες και ο πίνακας A έχει τροποποιηθεί $k-1$ φορές, όπως φαίνεται και από την τάξη τροποποίησης. Στο επόμενο $k + 1$ ο βήμα, ομοίως υπολογίζουμε την αντίστοιχη στήλη με την χρήση του τροποποιημένου στα δεξιά πίνακα A .

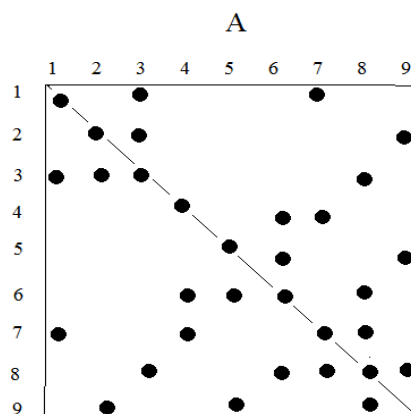


Σχήμα 5.5 Υπολογισμός k-στης στήλης των πινάκων L και U.

Η right-looking LU είναι πιο πολύπλοκη από την left – looking που περιγράψαμε στην προηγούμενη παράγραφο και είναι πιο δύσκολη η εφαρμογή της. Το πλεονέκτημά της σε σχέση με την left-looking LU είναι πως μπορεί να επιλέξει μία αραιή γραμμή οδήγησης και να θυμάται το μη μηδενικό πρότυπο του υποπίνακα $A^{[k]}$, για τον προσδιορισμό του πλήθους των μη μηδενικών στην γραμμή οδήγησης.

5.4 Multifrontal LU

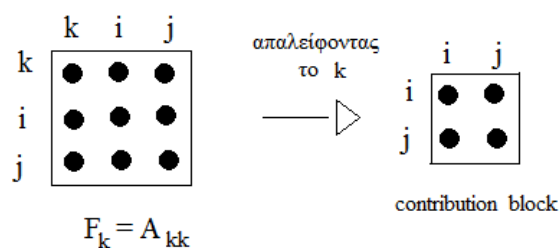
Η αραιή μέθοδος multifrontal LU ^[14] στηρίζεται στην right – looking LU. Για την επίλυση του αραιού γραμμικού συστήματος $Ax = b$, όπου ο πίνακας A είναι αραιός μη συμμετρικός και με συμμετρικό μη μηδενικό πρότυπο, όπως στο Σχήμα 5.6^[4].



Σχήμα 5.6 Ο Πίνακας A Μη Συμμετρικός με Συμμετρικό Μη Μηδενικό Πρότυπο

Αρχικά, καθορίζουμε το δένδρο απαλοιφής, στο οποίο ο κάθε κόμβος του αντιστοιχεί σε έναν μετωπικό πίνακα, frontal. Ο πίνακας frontal F_k του k -στου κόμβου είναι ένας $|L_k| \times |L_k|$ πυκνός πίνακας, ο οποίος περιέχει τις πραγματικές εγγραφές του πίνακα A στην k -στη γραμμή και στήλη.

Οι πίνακες frontal όλων των κόμβων, μπορούν να συσχετιστούν μεταξύ τους σε ένα δένδρο συγκρότησης (assembly tree) που μοιάζει με το δέντρο απαλοιφής αλλά είναι πιο πλατύ. Έστω δύο κόμβοι c , p , με τον πρώτο να είναι το μοναδικό παιδί του δεύτερου. Αν έχουν το ίδιο μη μηδενικό πρότυπο ($L_p = L_c$), τότε μπορούν να συγχωνευτούν σε έναν μεγαλύτερο frontal πίνακα που να απεικονίζει και τους δύο, με αποτέλεσμα τη συγχώνευση του δένδρου συγκρότησης. Αν ο κόμβος p έχει πάνω από ένα παιδιά, τότε οι εγγραφές του πίνακα F_p προστίθενται με τα blocks συνεισφοράς των παιδιών του. Το block συνεισφοράς (contribution block) ενός κόμβου k , είναι ο υποπίνακας που προκύπτει από τον F_k αν απαλείψουμε την k -στη γραμμή και στήλη, όπως φαίνεται στο Σχήμα 5.7:



Σχήμα 5.7 Πίνακας frontal και block συνεισφοράς

Συνοπτικά, η διαδικασία υπολογισμού των πινάκων L και U μέσω της multifrontal LU είναι η παρακάτω:

Για κάθε κόμβο k στο δένδρο απαλοιφής T , από τα φύλλα έως την ρίζα:

- προσθέτουμε τον k -στο frontal πίνακα F_k , μαζί με το block συνεισφοράς των παιδιών του.
- Εξαλείφουμε την k -στη μεταβλητή, ώστε να πάρουμε το block συνεισφοράς.
- Περνάμε το block συνεισφοράς στον γονέα του k μόλις τον συναντήσουμε.

Ο αντίστοιχος ψευδοκώδικας για την κατασκευή των πινάκων F_k και block συνεισφοράς U_k κάθε κόμβου k είναι:

```

for each node  $k$  of  $T$  from leaves to root
    if ( $k == \text{leaf}$ ) then
         $F_k = A_{kk}$ 
    else
         $F_k = A_{kk} + \sum_c U_{\text{children}}$ 
    end if
     $U_k = F_k - \{k\}$ 
end for

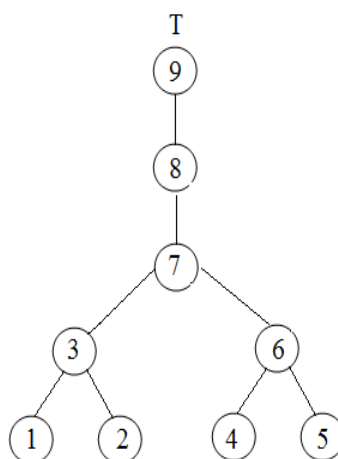
```

Στον οποίο αν ο κόμβος k είναι φύλλο ο F_k είναι ίσος με τον πίνακα A_{kk} που περιέχει μόνο τις εγγραφές για την k -στη γραμμή και στήλη. Αλλιώς, είναι ενδιάμεσος κόμβος, άρα αποτελεί γονέας κάποιων κόμβων και ο F_k είναι το άθροισμα των στοιχείων στον A_{kk} για την και των block συνεισφοράς των παιδιών του.

Παρατηρούμε πως ο πίνακας F_k , που προκύπτει και στις δύο περιπτώσεις, περιέχει γεμίσματα (fill – in) τα οποία είναι εγγραφές που θα γεμίσουν τις αντίστοιχες θέσεις στους παράγοντες L και U του πίνακα A και αποτελούν μέρος του block συνεισφοράς που αποστέλλεται στον

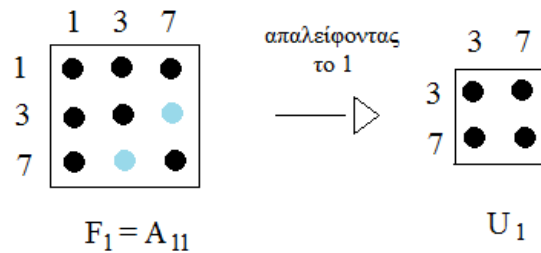
γονέα. Με την μερική οδήγηση, επιλέγουμε ως οδηγό μία γραμμή από τις ήδη συγκροτημένες γραμμές και στήλες του πίνακα απεικόνισης. Έστω, το στοιχείο a_{kk} του παραπάνω σχήματος, αν αυτό είναι αριθμητικά μη αποδεκτό, τότε είναι πιθανό να επιλεγούν τα στοιχεία a_{ki} και a_{ik} ως οι επόμενοι δύο οδηγοί αντί των στοιχείων a_{kk} και a_{ii} , λόγω της ανταλλαγής των γραμμών i, k , εφόσον $a_{kk} < a_{ki}$. Αν κάτι τέτοιο είναι αδύνατο και άρα η οδηγός γραμμή βρίσκεται παρακάτω έστω ότι είναι η j -στη του σχήματος, τότε ο πίνακας F_k θα είναι μεγαλύτερος από ό,τι περιμέναμε και όταν τον συγχωνεύουμε στον πατέρα του k , έστω p , τότε και ο πίνακας F_p θα είναι και αυτός μεγαλύτερος. Σε αυτό το σημείο καταλήγουμε ότι στον πίνακα απεικόνισης ενός κόμβου γονέα υπάρχουν όλοι οι οδηγοί που απέτυχαν από τα παιδιά του ή οποιονδήποτε απόγονο. Αν όλοι οι οδηγοί αυτοί είναι αριθμητικά αποδεκτοί (δηλαδή βρίσκονται στην θέση τους και δεν χρειάζονται μεταθέσεις γραμμών) το block συγχώνευσης θα είναι αναμενόμενου μεγέθους.

Παραθέτουμε ένα παράδειγμα εκτέλεσης της multifrontal LU στον πίνακα A του Σχήματος 5.6. Το αντίστοιχο δένδρο απαλοιφής T απεικονίζεται στο Σχήμα 5.8.:



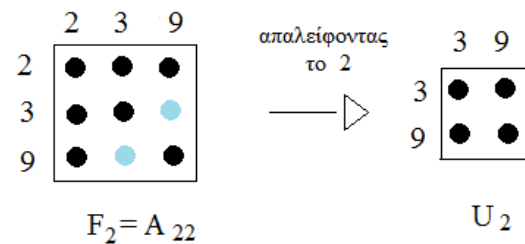
Σχήμα 5.8 Δένδρο Απαλοιφής του πίνακα A

Ξεκινάμε από τον $k = 1$ κόμβο, που είναι φύλλο και φτιάχνουμε τον πίνακα απεικόνισης F_1 , ο οποίος περιέχει τις πραγματικές εγγραφές του πίνακα A_{11} για την 1η γραμμή και στήλη. Απαλείφουμε τον 1 και λαμβάνουμε τον ανανεωμένο πίνακα U_1 , όπου θα περαστεί στον γονέα του $k = 1$, μόλις τον συναντήσουμε.

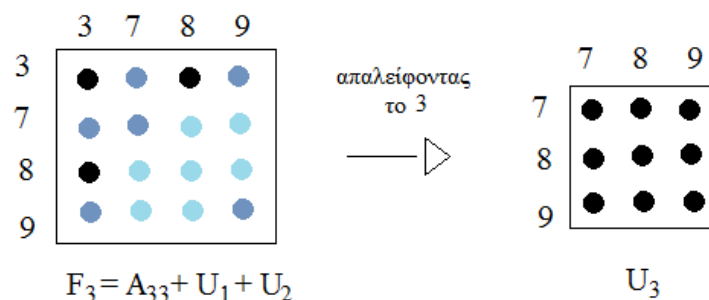


Με γαλάζιο χρώμα απεικονίζονται οι εγγραφές – γεμίσματα.

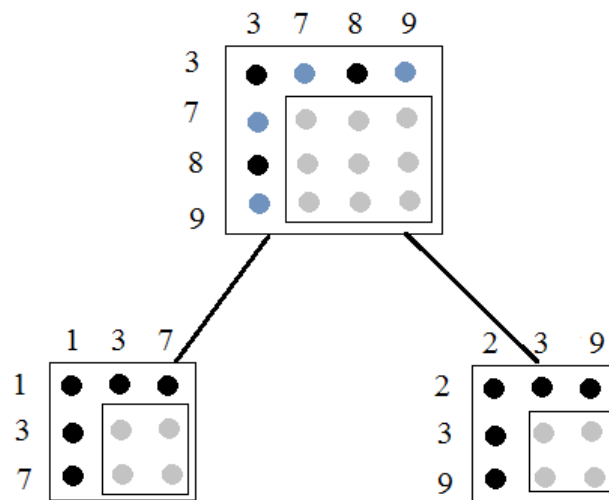
Για τον κόμβο $k = 2$, ομοίως ο πίνακας απεικόνισης frontal και το block συνεισφοράς είναι:



Για τον κόμβο $k = 3$: είναι ενδιάμεσος κόμβος, άρα είναι γονέας. Τα παιδιά του είναι οι κόμβοι 1 και 2, τα οποία προσθέτουν τα block συνεισφοράς τους στον A_{33} . Στο σχήμα, με μαύρο χρώμα απεικονίζονται τα πραγματικά στοιχεία του A_{33} , μπλε τα στοιχεία που προήλθαν από τους 1,2 και με γαλάζιο τα νέα στοιχεία (γεμίσματα).



Στο Σχήμα 5.8 φαίνεται το δένδρο συγκρότησης μέχρι τώρα :

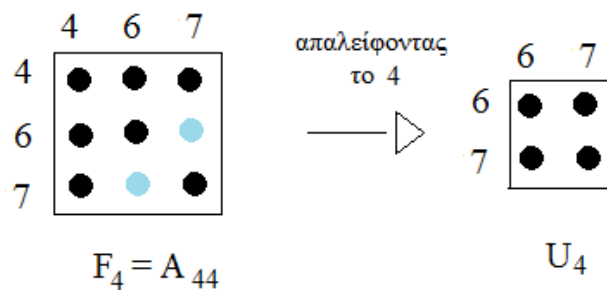


Σχήμα 5.8 Υποδέντρο Συγκρότησης για τους κόμβους 1, 2, 3

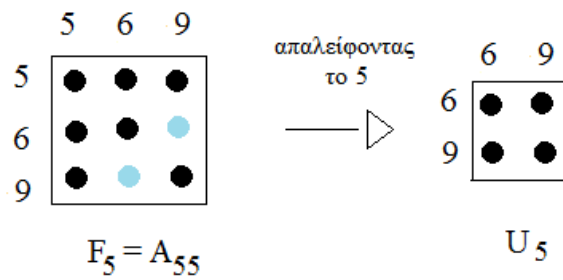
Όπου με γκρι εμφανίζονται τα στοιχεία που θα δοθούν στον γονέα.

Ομοίως, για $k = 4$ και $k = 5$ που αποτελούν φύλλα του δένδρου απαλοιφής και για $k = 6$ που είναι γονέας των δύο προηγούμενων. Οι αντίστοιχοι frontal πίνακες ,blocks συνεισφοράς απεικονίζονται παρακάτω.

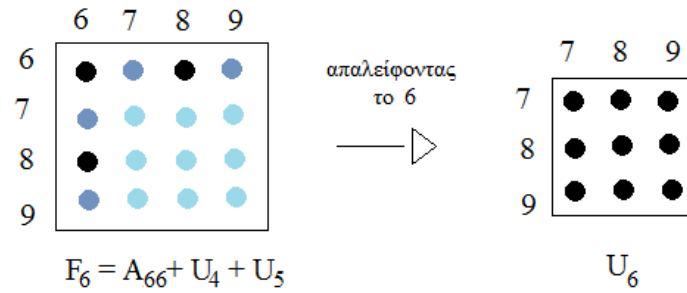
$k = 4$:



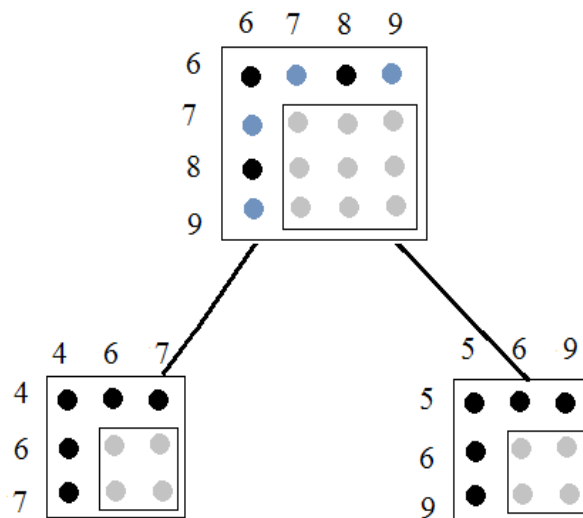
$k = 5$:



$k = 6$:

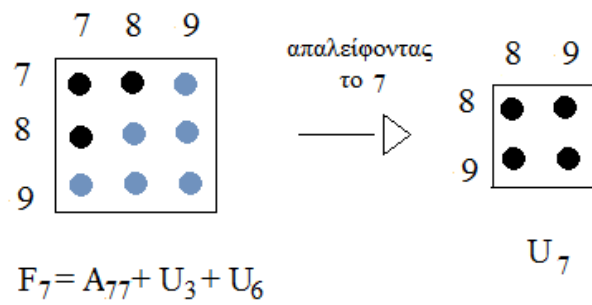


Το δένδρο συγκρότησης μέχρι τώρα αποτελείται από το υποδέντρο του Σχήματος 5.8 και του Σχήματος 5.9:

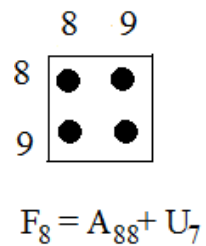


Σχήμα 5.9 Υποδέντρο Συγκρότησης για τους κόμβους 4,5,6

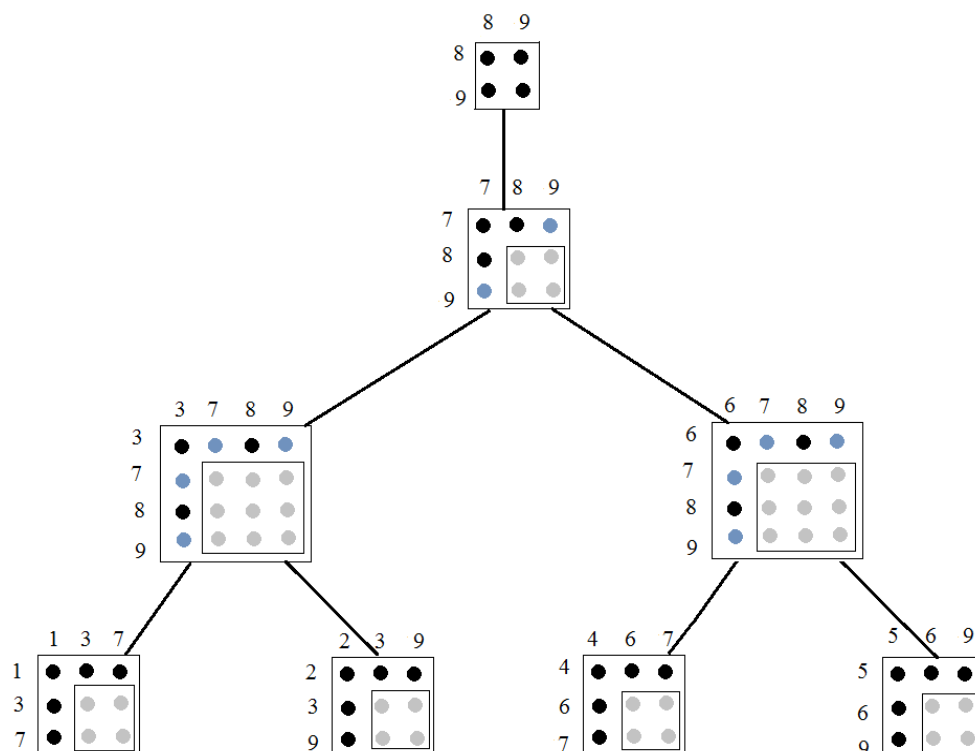
Για τους κόμβους $k = 7$ είναι γονέας των κόμβων 3, 6 και ο δεύτερος του 7. Επομένως, έχουμε:



Ο κόμβος $k = 8$ είναι γονέας του κόμβου 7 και ο $F_8 = U_8$:

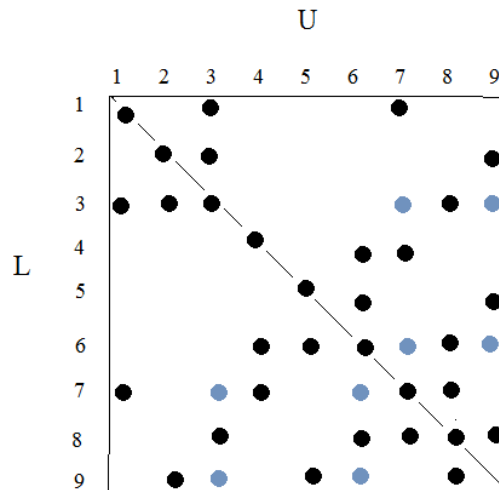


Το ολοκληρωμένο δένδρο συγκρότησης απεικονίζεται στο Σχήμα 5.10:



Σχήμα 5.10 Δένδρο Συγκρότησης του πίνακα A

Οι πίνακες L και U που προκύπτουν απεικονίζονται στο Σχήμα 5.11^[4], όπου με μπλε εμφανίζονται οι εγγραφές γεμίσματα που προέκυψαν κατά την διαδικασία των υπολογισμών.



Σχήμα 5.11 Οι πίνακες L και U

Συμπεραίνουμε από τα παραπάνω ότι η μέθοδος multifrontal LU για συμμετρικό μη μηδενικό πρότυπο του πίνακα A :

- Χρησιμοποιεί υπερκόμβους (super nodes) και όχι απλούς κόμβους.
- Το δένδρο απαλοιφής που έχουμε δημιουργήσει, κρατά ουσιαστικά τις μεταθέσεις των γραμμών που γίνονται κατά την διάρκεια της μερικής οδήγησης.
- Όλες οι αριθμητικές πράξεις λαμβάνουν χώρα σε πυκνούς τετραγωνικούς πίνακες.

5.5 Χρήση της παραγοντοποίησης LU

Στο αραιό γραμμικό σύστημα $Ax = b$, όταν ο πίνακας A είναι γενικά αραιός τετραγωνικός πίνακας χρησιμοποιούμε την παραγοντοποίηση LU. Με την εκτέλεση κάποιας από τις μεθόδους fill-reducing μεταθέτουμε τον πίνακα A έτσι ώστε να μειώσουμε τα μηδενικά του και να διατηρήσουμε την αριθμητική του ακρίβεια : PAQ ,όπου P και

Q οι δύο πίνακες μετάθεσης που επιτυγχάνουν αντίστοιχα τους δύο σκοπούς της μετάθεσης. Στην αραιή left-looking LU ο πίνακας Q επιλέγεται για την μείωση του πλήθους των μη μηδενικών ενώ ο P για την επιλογή των οδηγών – γραμμών. Αν ο πίνακας έχει ως επί το πλείστον συμμετρικό μη μηδενικό πρότυπο επιλέγεται η μετάθεση ελαχίστου βαθμού.

Έπειτα επιλύουμε τα εξής συστήματα με την σειρά:

$$Ly = Pb$$

$$Uz = y, \text{ που μας δίνει την μεταθεμένη λύση}$$

$$x = Qz, \text{ η οποία μας δίνει την τελική λύση του συστήματος.}$$

Η παραγοντοποίηση LU είναι η παλαιότερη μέθοδος παραγοντοποίησης και η πιο γενική, καθώς μπορεί να εφαρμοστεί σε ένα γενικότερο τετραγωνικό αραιό πίνακα. Η συμβολική ανάλυση της γίνεται παράλληλα με την αριθμητική γιατί χρησιμοποιεί μερική οδήγηση και την ενδιαφέρει η αριθμητική τιμή κάθε εγγραφής. Από την άλλη, η παραγοντοποίηση Cholesky είναι νεότερη και μία περίπτωση της LU, γιατί αφορά συμμετρικούς πίνακες. Η συμβολική ανάλυση αυτής, όπως είδαμε γίνεται ξεχωριστά από την αριθμητική παραγοντοποίηση, λόγω της συμμετρίας των εγγραφών και την απουσία οδήγησης.

6.

Πειραματική Μελέτη

Στο κεφάλαιο αυτό θα διεξαχθούν παραδείγματα εκτέλεσης των αραιών μεθόδων παραγοντοποίησης Cholesky και LU, που αναλύθηκαν στα προηγούμενα κεφάλαια, για αραιούς πίνακες μεγάλων μεγεθών. Πριν την παρουσίαση τους θα αναφερθούμε στα πακέτα και τις βιβλιοθήκες που χρησιμοποιήθηκαν για την εκτέλεση των πειραμάτων.

6.1 SuiteSparse

Το SuiteSparse^[16] είναι μια συλλογή βιβλιοθηκών με υπολογισμούς που αφορούν αραιούς πίνακες, υλοποιημένη σε γλώσσα C και περιλαμβάνει τη διαμόρφωση πίνακα, πράξεις μεταξύ πινάκων, πράξεις μεταξύ πίνακα και διανύσματος, την αναδιάταξη πίνακα καθώς και τις μεθόδους παραγοντοποίησης Cholesky και LU. Κάποια από τα πακέτα που περιέχει είναι τα εξής: AMD προσεγγιστική διάταξη ελαχίστου βαθμού, BTF μετάθεση για την αποφυγή τριγωνικής μορφής, COLAMD κατά στήλη προσεγγιστική διάταξη ελαχίστου βαθμού, CCOLAMD κατά στήλη περιορισμένη προσεγγιστική διάταξη ελαχίστου βαθμού, CHOLMOD αραιή παραγοντοποίηση Cholesky, CSparse ένα συνοπτικό πακέτο αραιού πίνακα, CXSparse η επέκταση του προηγούμενου πακέτου, KLU αραιή παραγοντοποίηση LU κυρίως για προσομοίωση σε κυκλώματα, UMFPACK αραιή παραγοντοποίηση LU και κάποια ακόμα. Αναλόγως την έκδοση του SuiteSparse μπορεί να υπάρχουν και άλλες βιβλιοθήκες ή να λείπουν μερικές από τις προαναφερθείσες. Ωστόσο, η νεότερη έκδοσης(4.4.4), την οποία χρησιμοποιήσαμε, περιέχει όλες τις παραπάνω και κάποια επιπλέον πακέτα και αρχεία.

Το SuiteSparse χρησιμοποιείται για τον προγραμματισμό σε γλώσσα C ή C++. Είναι αρκετά εύκολο στην χρήση του και η παροχή των αρχείων README και Makefile βοηθούν στην γρήγορη εξοικείωση του χρήστη με το περιβάλλον και την εύκολη εγκατάσταση του.

6.2 CSparse

Το πακέτο CSparse^[16] περιλαμβάνει αλγορίθμους για αραιούς πίνακες. Οι συναρτήσεις του είναι διαχωρισμένες σε τρεις κατηγορίες: πρωτοβάθμιες, δευτεροβάθμιες και τριτοβάθμιες. Οι πρωτοβάθμιες είναι οι συναρτήσεις εκείνες που χρειάζονται για την δημιουργία ενός πίνακα, την επίλυση του συστήματος $Ax = b$ και την εφαρμογή των βασικών λειτουργιών των πινάκων. Οι δευτεροβάθμιες είναι συναρτήσεις που εκτελούν τις μεθόδους διάταξης και παραγοντοποίησης, τον χειρισμό διανυσμάτων μετάθεσης και άλλων σημαντικών λειτουργιών, που είναι χρήσιμες για την ανάλυση και την διάταξη ενός πίνακα για την μετέπειτα πολλαπλή παραγοντοποίηση του. Τέλος, οι τριτοβάθμιες είναι ρουτίνες που χρησιμοποιούνται σπανίως σε οποιαδήποτε εφαρμογή και εκτελούνται ως συναρτήσεις από το ίδιο το CSparse.

Το κάθε αρχείο του CSparse ξεκινά με την δήλωση `#include "cs.h"`, την οποία πρέπει να έχει κάθε πρόγραμμα που χρησιμοποιεί το πακέτο.

6.3 Πειραματικά Αποτελέσματα

Έχοντας εγκαταστήσει το SuiteSparse και κατ' επέκταση το CSparse, εκτελούμε κώδικα που υλοποιεί την παραγοντοποίηση ενός αραιού πίνακα με τις μεθόδους παραγοντοποίησης Cholesky και LU καθώς εκτυπώνει τον αντίστοιχο χρόνο υπολογισμού κάθε μιας. Αν ο

πίνακας είναι τετραγωνικός, συμμετρικός και θετικά ορισμένος πραγματοποιείται παραγοντοποίηση Cholesky, διαφορετικά μόνο παραγοντοποίηση LU. Υπενθυμίζουμε, πως όταν ένας πίνακας είναι τετραγωνικός, συμμετρικός και θετικά ορισμένος υπάρχει η παραγοντοποίηση Cholesky, ενώ η παραγοντοποίηση LU υπάρχει για γενικό τετραγωνικό πίνακα.

Για την σύγκριση του χρόνου υπολογισμού των δύο μεθόδων χρησιμοποιούμε κυρίως τετραγωνικούς, συμμετρικούς και θετικά ορισμένους πίνακες, τα χαρακτηριστικά των οποίων απεικονίζονται στον Πίνακα 6.1^[17].

Όνομα	Διαστάσεις	Μη μηδενικές Εγγραφές	Συμμετρικότητα	Θετικά Ορισμένος
M1	48x48	400	συμμετρικός	ναι
M2	420x420	7.860	συμμετρικός	ναι
M3	1.806 x 1.806	63.454	συμμετρικός	ναι
M4	2.003 x 2.003	83.883	συμμετρικός	ναι
M5	5.357 x 5.357	207.123	συμμετρικός	ναι
M6	5.489 x 5.489	217.669	συμμετρικός	ναι
M7	5.489 x 5.489	262.411	συμμετρικός	ναι
M8	15440x15440	252.241	συμμετρικός	ναι
M9	10974x10974	428.650	συμμετρικός	ναι
M10	4.929 x 4.929	33005	μη συμμετρικός	όχι
M11	90.450x90.450	2.148.558	συμμετρικός	ναι
M12	30.402x30.402	471601	συμμετρικός	ναι
M13	82.655x82.655	574458	συμμετρικός	ναι
M14	102.158x102.158	711558	συμμετρικός	ναι
M15	16.146x16.146	1.015.516	συμμετρικός	ναι

Πίνακας 6.1 Ιδιότητες Πινάκων που χρησιμοποιήθηκαν

Παρακάτω απεικονίζονται οι πίνακες του Πίνακα 6.1 σε αύξουσα σειρά του πλήθους των μη μηδενικών στοιχείων με τους αντίστοιχους χρόνους υπολογισμού σε δευτερόλεπτα(seconds) για τον καθένα.

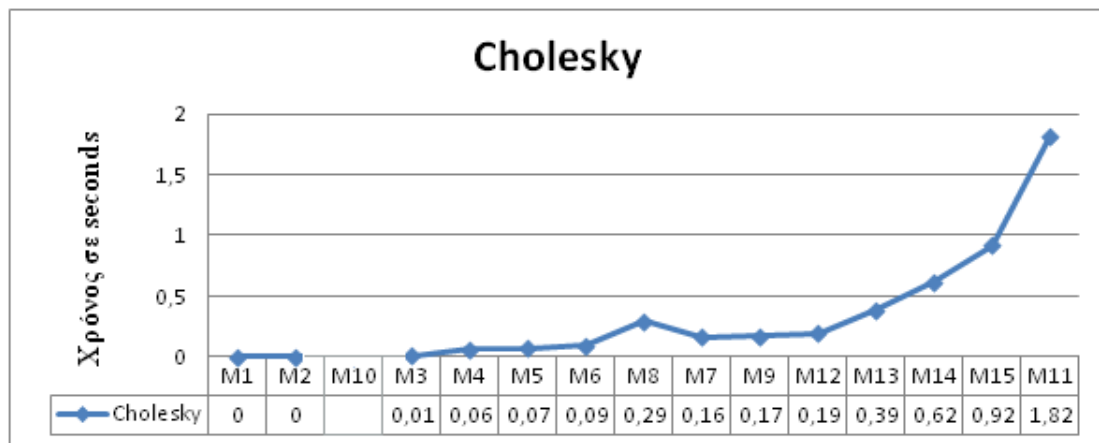
Πίνακας	Μέγεθος	Cholesky	LU
M1	400	0	0
M2	7.860	0	0
M10	33.005		1,84
M3	63.454	0,01	0,04
M4	83.883	0,06	0,18
M5	207.123	0,07	0,18
M6	217.669	0,09	0,26
M8	252.241	0,29	0,92
M7	262.411	0,16	0,5
M9	428.650	0,17	0,46
M12	471.601	0,19	0,52
M13	574.458	0,39	1,09
M14	711.558	0,62	1,81
M15	1.015.516	0,92	2,69
M11	2.148.558	1,82	5,54

Πίνακας 6.2 Χρόνοι Υπολογισμού

Παρατηρούμε ότι ο πίνακας M10 που επιλέξαμε είναι τετραγωνικός αλλά όχι συμμετρικός και η παραγοντοποίηση Cholesky δεν υπάρχει (σαν απόδειξη των όσων προηγήθηκαν). Επίσης, από τον πίνακα των χρόνων παρατηρούμε ότι η Cholesky είναι ταχύτερη της LU για κάθε πίνακα που ελέγχουμε. Ειδικά, σε πίνακες με μεγάλο πλήθος μη μηδενικών εγγραφών, ότι η διαφορά αυτή είναι τουλάχιστον διπλάσια. Η αιτία είναι η εκμετάλλευση της συμμετρίας της δομής και των τιμών των μη μηδενικών στοιχείων του αραιού πίνακα από την παραγοντοποίηση Cholesky.

Παρακάτω, απεικονίζεται το Διάγραμμα 6.1 που αντιστοιχεί στους χρόνους υπολογισμού της παραγοντοποίησης των πινάκων, του Πίνακα 6.1, με την παραγοντοποίηση Cholesky. Παρατηρούμε πως όσο

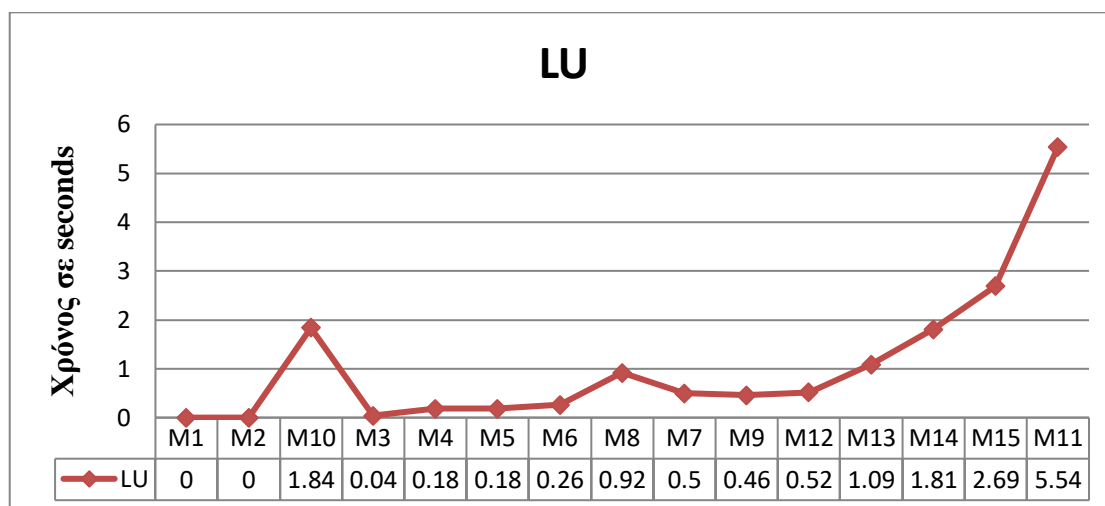
αυξάνεται το πλήθος των μη μηδενικών ενός πίνακα M_i , $i=1,\dots,15$, αυξάνεται και ο χρόνος υπολογισμού της παραγοντοποίησης, γιατί αυξάνονται και οι επαναλήψεις υπολογισμού στον αλγόριθμο. Ωστόσο, πρέπει να προσέξουμε ότι και οι διαστάσεις του πίνακα M_i συντελούν στον χρόνο υπολογισμού. Σαν παράδειγμα μπορούμε να συγκρίνουμε τις πίνακες $M7$ και $M8$, με διαστάσεις 5.489×5.489 και 15.440×15.440 και με πλήθος μη μηδενικών στοιχείων 262.411 και 252.241, αντίστοιχα. Ο πρώτος είναι μικρότερος του δεύτερου, με περισσότερα μη μηδενικά στοιχεία αλλά απαιτεί λιγότερο χρόνο υπολογισμού και για τις δύο παραγοντοποιήσεις.



Διάγραμμα 6.1 Διάγραμμα Παραγοντοποίησης Cholesky

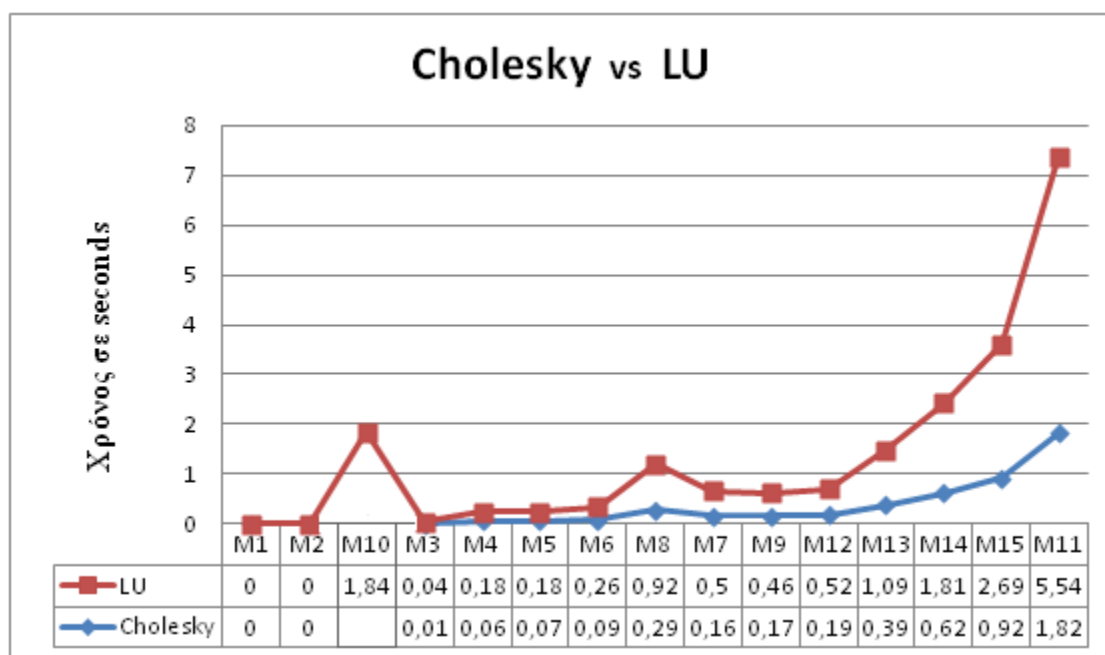
Για τον πίνακα $M10$ δεν υπάρχει η παραγοντοποίηση Cholesky, για αυτό και ο αντίστοιχος χρόνος είναι κενός και η γραφική παράσταση διακόπτεται σε αυτό το σημείο.

Αντιθέτως, στο Διάγραμμα 6.2 της παραγοντοποίησης LU, η γραφική παράσταση είναι μία συνεχόμενη γραμμή για όλους του πίνακες, γιατί όλοι είναι τετραγωνικοί και υπάρχει η αντίστοιχη παραγοντοποίηση LU.



Διάγραμμα 6.2 Διάγραμμα Παραγοντοποίησης LU

Τέλος, στο Διάγραμμα 6.3 που συγκρίνονται οι δύο μέθοδοι παρατηρούμε ότι για πίνακες μικρού πλήθους μη μηδενικών στοιχείων, η χρονική διαφορά δεν είναι αισθητή και οι γραφικές παραστάσεις σχεδόν τέμνονται, ενώ για πίνακες μεγάλου μεγέθους η Cholesky είναι ταχύτερη της LU και οι γραφικές παραστάσεις έχουν αισθητή διαφορά.



Διάγραμμα 6.3 Διάγραμμα Παραγοντοποίησης Cholesky vs LU

7.

Συμπεράσματα και Μελλοντική Εργασία

Στο κεφάλαιο αυτό συνοψίζουμε βασικά συμπεράσματα της παρούσας διπλωματικής και δίνουμε κατευθύνσεις για μελλοντική έρευνα.

7.1 Συμπεράσματα

Στην παρούσα διπλωματική μελετήσαμε τις άμεσες μεθόδους επίλυσης αραιών γραμμικών συστημάτων και διεξαγάγαμε μετρήσεις για αντιπροσωπευτικά μεγέθη προβλημάτων. Πιο συγκεκριμένα, οι μέθοδοι παραγοντοποίησης Cholesky και LU που αναλύθηκαν έχουν υπολογιστικό κόστος της τάξης των μη μηδενικών στοιχείων του αραιού πίνακα, καθώς κατά την διάρκεια των υπολογισμών η δυσκολία του κόστους υπολογισμού αυξάνεται λόγω της εισαγωγής νέων μη μηδενικών στοιχείων (fill-in) σε θέσεις που αρχικά υπήρχαν μηδενικά στοιχεία.

Για τον λόγο αυτό, η αριθμητική παραγοντοποίηση ενός αραιού πίνακα, έπεται της φάσης ελαχιστοποίησης των μη μηδενικών εγγραφών του, όπου εφαρμόζεται κάποια από τις μεθόδους διάταξης fill – reducing και της φάσης της συμβολικής ανάλυσης, η οποία διεξάγει την ρητή αναπαράσταση του μη μηδενικού προτύπου της παραγοντοποίησης. Η πληροφορία της θέσης των μη μηδενικών στοιχείων που παρέχεται από την συμβολική ανάλυση επιταχύνει τους υπολογισμούς της μεθόδου στη φάση της αριθμητικής παραγοντοποίησης.

Καταληκτικά, η παραγοντοποίηση Cholesky είναι μία ειδική περίπτωση της παραγοντοποίησης LU με αποτέλεσμα να πραγματοποιούν τον ίδιο υπολογισμό με παρόμοιο τρόπο και με την διαφορά ότι η πρώτη είναι ταχύτερη της δεύτερης.

7.2 Μελλοντική Εργασία

Με την ολοκλήρωση της εργασίας αυτής, αναδείχθηκαν διάφορα θέματα τα οποία χρήζουν περαιτέρω διερεύνησης και θα μπορούσαν να αποτελέσουν αντικείμενο επόμενων διπλωματικών εργασιών. Αναφέρουμε μερικά από αυτά:

- Μελέτη και ανάλυση του αλγορίθμου Multifrontal Cholesky.
- Μελέτη και ανάλυση του αλγορίθμου Right-looking LU, ο οποίος λειτουργεί ως βάση για τον multifrontal LU που μελετήσαμε αναλυτικά.
- Συνολική παρουσίαση και μελέτη των άμεσων και των επαναληπτικών μεθόδων επίλυσης αραιών γραμμικών συστημάτων και σύγκριση αυτών μέσα από πειράματα.

Βιβλιογραφία

[1]ορθογώνιος πίνακας:

http://el.wikipedia.org/wiki/%CE%9F%CF%81%CE%B8%CE%BF%CE%B3%CF%8E%CE%BD%CE%B9%CE%BF%CF%82_%CF%80%CE%AF%CE%BD%CE%B1%CE%BA%CE%B1%CF%82

[2] Ορισμοί και Πράξεις Αραιών Πινάκων: Chapter 2 Direct Methods For Sparse Linear Systems, Tim Davis.

[3] Διατάξεις Fill-Reducing : Chapter 7 Direct Methods For Sparse Linear Systems, Tim Davis.

[4] *Sparse Matrix Algorithms*, John R. Gilbert

[5] Συμβολική Ανάλυση: Chapter 3 Direct Methods For Sparse Linear Systems, Tim Davis.

[6] θετικά ορισμένος πίνακας : http://edu.eap.gr/pli/pli12/shmeiwseis/tetragwnikes_morfes.pdf

[7] Up – looking Cholesky : Paragraph 4.7 Direct Methods For Sparse Linear Systems, Tim Davis.

[8] Left-looking Cholesky και Supernodal Cholesky: Paragraph 4.8 Direct Methods For Sparse Linear Systems, Tim Davis.

[9]Left - looking Cholesky: <http://www.cs.umd.edu/~yluo1/Documents/Papers/TR-supernodal.pdf>

[10] Right-looking Cholesky : <http://www.netlib.org/utk/papers/factor/node9.html>

[11] Right-looking Multifrontal Cholesky : Paragraph 4.9 Direct Methods For Sparse Linear Systems, Tim Davis.

[12] Μερική Οδήγηση Εισαγωγή Στην Αριθμητική Ανάλυση, 3^η Έκδοση, Γ.Δ. ΑΚΡΙΒΗΣ-Β.Α. ΔΟΥΛΓΑΡΗΣ

[13] Left – looking LU : Paragraph 6.2 Direct Methods For Sparse Linear Systems, Tim Davis.

[14] Right – looking και Multifrontal LU : Paragraph 6.3 Direct Methods For Sparse Linear Systems, Tim Davis.

[15] DFS:

http://el.wikipedia.org/wiki/%CE%91%CE%BD%CE%B1%CE%B6%CE%AE%CF%84%CE%B7%CF%83%CE%B7_%CE%9A%CE%B1%CF%84%CE%AC_%CE%92%CE%AC%CE%B8%CE%BF%CF%82

[16] SuiteSparse & CSparse: <http://faculty.cse.tamu.edu/davis/welcome.html>

[17] Συλλογή Αραιών Πινάκων: <http://www.cise.ufl.edu/research/sparse/matrices/>

[18] https://en.wikipedia.org/?title=LU_decomposition

[19] https://en.wikipedia.org/?title=Cholesky_decomposition

[20] https://en.wikipedia.org/wiki/Gaussian_elimination

[21] https://el.wikipedia.org/wiki/%CE%91%CF%81%CE%B9%CE%B8%CE%BC%CE%B7%CF%84%CE%B9%CE%BA%CE%AE_%CE%B1%CE%BD%CE%AC%CE%BB%CF%85%CF%83%CE%B7#.CE.86.CE.BC.CE.B5.CF.83.CE.B5.CF.82_.CE.BA.CE.B1.CE.B9_.CE.B5.CF.80.CE.B1.CE.BD.CE.B1.CE.BB.CE.B7.CF.80.CF.84.CE.B9.CE.BA.CE.AD.CF.82_.CE.BC.CE.AD.CE.B8.CE.BF.CE.B4.CE.BF.CE.B9

[22] Διαλέξεις : <https://www.youtube.com/user/DrTimothyAldenDavis/videos>

ΠΑΡΑΡΤΗΜΑ Α

^A **Οι επαναληπτικές μέθοδοι** επίλυσης δεν υπολογίζουν την ακριβή λύση ενός γραμμικού συστήματος, αλλά αντιθέτως, προσπαθούν να υπολογίσουν μία καλή προσέγγιση της λύσης. Οι επαναληπτικές μέθοδοι είναι πιο συχνές από τις άμεσες μεθόδους στην αριθμητική ανάλυση.

^B Προς τα εμπρός αντικατάσταση :

Κατά την οποία επιλύουμε το σύστημα από την πρώτη εξίσωση που φέρει τον πρώτο άγνωστο και συνεχίζουμε μέχρι την τελευταία, βρίσκοντας και αντικαθιστώντας τους αγνώστους. Ο αλγόριθμος υπολογισμού προκύπτει εύκολα αν παρατηρήσουμε ότι κάθε φορά ο μοναδικός άγνωστος είναι αυτός που βρίσκεται στην δεξιότερη θέση της τρέχουσας εξίσωσης. Το σύστημα εξισώσεων και οι σχέσεις υπολογισμού του πρώτου αγνώστου x_1 και x_i φαίνονται παρακάτω:

$$\begin{aligned} \ell_{1,1}x_1 &= b_1 \\ \ell_{2,1}x_1 + \ell_{2,2}x_2 &= b_2 \\ &\vdots \\ \ell_{n-1,1}x_1 + \ell_{n-1,2}x_2 + \dots + \ell_{n-1,n-1}x_{n-1} &= b_{n-1} \\ \ell_{n,1}x_1 + \ell_{n,2}x_2 + \dots + \ell_{n,n-1}x_{n-1} + \ell_{n,n}x_n &= b_n \end{aligned}$$

^c Προς τα πίσω αντικατάσταση :

Κατά την οποία επιλύουμε το σύστημα από την τελευταία εξίσωση που φέρει τον τελευταίο άγνωστο και προχωράμε προς την πρώτη που φέρει όλους τους αγνώστους. Ο αλγόριθμος υπολογισμού προκύπτει εύκολα αν παρατηρήσουμε ότι κάθε φορά ο μοναδικός άγνωστος είναι αυτός που βρίσκεται στην βρίσκεται στην διαγώνια θέση της τρέχουσας

εξίσωσης. Το σύστημα εξισώσεων και οι σχέσεις υπολογισμού του τελευταίου αγνώστου x_n και x_i φαίνονται παρακάτω:

$$\begin{array}{lcl} u_{1,1}x_1 + u_{1,2}x_2 + u_{1,3}x_3 + \dots + u_{1,n}x_n & = & b_1 \\ u_{2,2}x_2 + u_{2,3}x_3 + \dots + u_{2,n}x_n & = & b_2 \\ u_{3,3}x_3 + \dots + u_{3,n}x_n & = & b_3 \\ \vdots & & \vdots \\ u_{n-1,n-1}x_{n-1} + u_{n-1,n}x_n & = & b_{n-1} \\ u_{n,n}x_n & = & b_n \end{array} \quad \begin{array}{l} x_n = b_n / u_{n,n} \\ x_i = \left(b_i - \sum_{k=i+1}^n u_{i,k}x_k \right) / u_{i,i}, \quad i = n-1, n-2, \dots, 1 \end{array}$$