## 1.5 ΑΝΑΛΥΣΗ ΤΗΣ ΠΑΡΑΛΛΗΛΙΑΣ ΜΕ ΘΕΩΡΙΑ ΓΡΑΦΗΜΑΤΩΝ

Η έννοια των συστημάτων εργασιών, όπως την χρησιμοποιούν οι Coffman και Denning [6] μας επιτρέπει να εισάγουμε μια τυποποίηση που θα υποστηρίζει την μελέτη μας σχετικά με τον παραλληλισμό των αλγορίθμων. Ενας αλγόριθμος μπορεί να διαχωρισθεί σ'ένα σύνολο απο εργασίες. Η μελέτη της αλληλοεξάρτησης των εργασιών σχετικά με τις μεταβλητές εισόδου και εξόδου μας επιτρέπει να κατασκευάσουμε ένα σύστημα προτεραιότητας το οποίο να ερμηνεύει τον εσωτερικό παραλληλισμό του αλγορίθμου. Σ'ένα τέτοιο σύστημα μπορεί να υπάρχουν περιορισμοί της προτεραιότητας στη σειρά εκτέλεσης των εργασιών και στη περίπτωση αυτή η εκτέλεσή τους είναι ακολουθιακή. Στην αντίθετη περίπτωση, οι εργασίες είναι μη αλληλοεξαρτώμενες, οπότε η εκτέλεσή τους μπορεί να πραγματοποιηθεί παράλληλα. Στη συνέχεια παρουσιάζουμε ορισμένα στοιχεία από τη θεωρία δρομολόγησης (scheduling theory), τα οποία θα χρησιμοποιηθούν στα επόμενα.

Μια **εργασία (task)** είναι μια αδιαίρετη ενότητα (μονάδα) επεξεργασίας που είναι μοναδικά ορισμένη ως πρός την εξωτερική της συμπεριφορά: είσοδοι (I), έξοδοι (O), απεικόνιση (f) και χρόνοι εκτέλεσης (t). Συνεπώς μια εργασία θα παρίσταται με μια τετράδα $(I_T, O_T, f_T, t_T)$ όπου:

$I_T$: είναι το σύνολο των εισόδων της T.
$O_T$: είναι το σύνολο των εξόδων της T.
$f_T$: είναι μια απεικόνιση απο το $I_T$ στο $O_T$, δηλαδή μια σειρά στοιχειωδών πράξεων από το $I_T$ στο $O_T$.
$t_T$: είναι μια παράσταση του χρόνου εκτέλεσης της εργασίας T.

**Ο χρόνος (ή διάρκεια) της εκτέλεσης** μιας εργασίας είναι γενικά το άθροισμα δύο όρων: του **χρόνου υπολογισμού** που αντιστοιχεί στον αριθμό των στοιχειωδών πράξεων που γίνονται στην εργασία και του **χρόνου επικοινωνίας** που οφείλεται στον αριθμό των μεταφορών δεδομένων μεταξύ της κοινής μνήμης και του επεξεργαστή που εκτελεί την εργασία. Στη συνέχεια υποθέτουμε ότι μια εργασία είναι αδιάσπαστη, δηλαδή από τη στιγμή που θα αρχίσει να εκτελείται από ένα επεξεργαστή δεν μπορεί να διακοπεί η επεξεργασία της για να εκτελεσθεί κάποια άλλη εργασία. Ενας **ακολουθιακός αλγόριθμος** είναι ένα σύνολο εργασιών εφοδιασμένο με μια σχέση ολικής διάταξης: $A = (T_1, \ldots, T_n, \rightarrow)$. Η σχέση $T_i \rightarrow T_k$ $(i \neq k)$ σημαίνει ότι η εκτέλεση της εργασίας $T_i$ πρέπει να τελειώσει πριν ξεκινήσει η εκτέλεση της $T_k$. Ενας ακολουθιακός αλγόριθμος μπορεί να θεωρηθεί επίσης σαν μια μοναδική εργασία $(I_A, O_A, f_A, t_A)$.

Σύμφωνα με τον προηγούμενο ορισμό, μια εργασία είναι ένας ακολουθιακός αλγόριθμος και αντίστροφα ένας ακολουθιακός αλγόριθμος είναι μια εργασία στην περίπτωση που γνωρίζουμε τον χρόνο εκτέλεσης του. Τέτοιοι αλγόριθμοι είναι οι κυριότεροι αλγόριθμοι της Αριθμητικής Γραμμικής Αλγεβρας.

Ενας ακολουθιακός αλγόριθμος μπορεί να έχει πολλές διαφορετικές διασπάσεις σε επιμέρους εργασίες. Το **μέγεθος** μιας διάσπασης είναι ένα χονδρικό μέτρο του λόγου του μέσου χρόνου εκτέλεσης μιας εργασίας προς τον συνολικό χρόνο του αλγορίθμου. Η εκλογή της καλύτερης διάσπασης είναι ένα δύσκολο πρόβλημα που σχετίζεται με τους χρόνους εκτέλεσης των εργασιών και την αρχιτεκτονική. Οι κυριότεροι παράγοντες για την εκλογή αυτή είναι ο αριθμός των επεξεργαστών, ο λόγος της μονάδας χρόνου επικοινωνίας και της μονάδας χρόνου υπολογισμού, οι προσπελάσεις στη μνήμη. Παρατηρούμε ότι ο χρήστης έχει την ανάγκη να χρησιμοποιήσει μια γλώσσα προγραμματισμού (Occam, parallel C) που να του επιτρέπει να ορίσει τις εργασίες και τις μεταξύ τους επικοινωνίες.

Ενα **σύστημα εργασιών** $S = (T_1, \ldots, T_n, \ll)$ είναι ένα σύνολο εργασιών εφοδιασμένο με μια σχέση μερικής διάταξης συμβολιζόμενη με $\ll$, όπου $T_i \ll T_k$ $(i \neq k)$ σημαίνει ότι η εκτέλεση της εργασίας $T_i$ πρέπει να τελειώσει πριν ξεκινήσει η εκτέλεση της $T_k$. Ενας ακολουθιακός αλγόριθμος A είναι ένα σύστημα εργασιών στο οποίο όλες οι εργασίες είναι διατεταγμένες από τη σχέση $\ll$.

Εστω $S = (T_1, \ldots, T_n, \ll)$ ένα σύστημα εργασιών. Δυο εργασίες $T_i$ και $T_k$ είναι **ανεξάρτητες** όταν δεν τροποποιούν καμμιά κοινή μεταβλητή. Η σχέση ανεξαρτησίας εκφράζεται επίσης ως εξής:

$$(O_{T_i} \cap O_{T_k}) = (O_{T_i} \cap I_{T_k}) = (I_{T_i} \cap O_{T_k}) = \emptyset$$

Δηλαδή καμμία έξοδος της μιας εργασίας δεν είναι ούτε είσοδος ούτε έξοδος της άλλης, οπότε επιτρέπεται η ταυτόχρονη εκτέλεσή τους.

Οι εργασίες $T_i$ και $T_k$ είναι **διαδοχικές** όταν $T_i \ll T_k$ και δεν υπάρχει καμμιά άλλη εργασία $T_j$ τέτοια ώστε: $T_i \ll T_j \ll T_k$. $T_i \ll T_k$ σημαίνει ότι υπάρχει τουλάχιστον μια είσοδος της $T_i$ που δεν είναι είσοδος ή έξοδος της $T_k$ και η εκτέλεση της $T_k$ δεν ξεκινά πρίν το τέλος της εκτέλεσης της $T_i$.

Ενα σύστημα εργασιών $S = (T_1, \ldots, T_n, \ll)$ είναι ένα **σύστημα προτεραιότητας** όταν για κάθε i και k ($i \neq k$), ισχύει μια από τις παρακάτω συνθήκες:

1. $T_i \ll T_k$
2. $T_k \ll T_i$
3. $T_i$ και $T_k$ είναι ανεξάρτητες.

Το **γράφημα προτεραιοτήτων** (ή γράφημα των εργασιών) G που αντιστοιχεί σ'ένα σύστημα με προτεραιότητα $S = (T_1, \ldots, T_n, \ll)$ ορίζεται με τον ακόλουθο τρόπο:

- Το σύνολο των κορυφών ή κόμβων του G είναι το σύνολο των εργασιών S.
- Οι κόμβοι $T_i$ και $T_k$ συνδέονται με ένα τόξο αν και μόνο αν οι εργασίες $T_i$ και $T_k$ είναι διαδοχικές

Δοθέντος ενός ακολουθιακού αλγόριθμου $A = (T_1, \ldots, T_n, \rightarrow)$ αποδεικνύεται ότι υπάρχει ένα μοναδικό σύστημα προτεραιότητας $S = (T_1, \ldots, T_n, \ll)$ τέτοιο ώστε η σχέση $\ll$ είναι μια υποδιάταξη της σχέσης $\rightarrow$. Η σχέση $\ll$ είναι η παρακάτω σχέση:

$$T_i \ll T_k \iff (T_i \rightarrow T_k \text{ και } (O_{T_i} \cap O_{T_k}) \cup (O_{T_i} \cap I_{T_k}) \cup (I_{T_i} \cap O_{T_k}) \neq \emptyset)$$

Για να παραλληλίσουμε ένα ακολουθιακό αλγόριθμο ξεκινούμε πάντα να υπολογίσουμε τη σχέση $\ll$ από τη σχέση $\rightarrow$. Στη συνέχεια, θα υποθέτουμε ότι έχει γίνει αυτή η φάση.

**Παράδειγμα:** Εστω ένα σύνολο 8 εργασιών που υπόκεινται στους παρακάτω χρονικούς περιορισμούς:

$T_1 \ll T_3$ $\quad$ $T_1 \ll T_4$ $\quad$ $T_1 \ll T_5$ $\quad$ $T_1 \ll T_8$

$T_2 \ll T_3$ $\quad$ $T_2 \ll T_4$ $\quad$ $T_2 \ll T_5$ $\quad$ $T_2 \ll T_7$

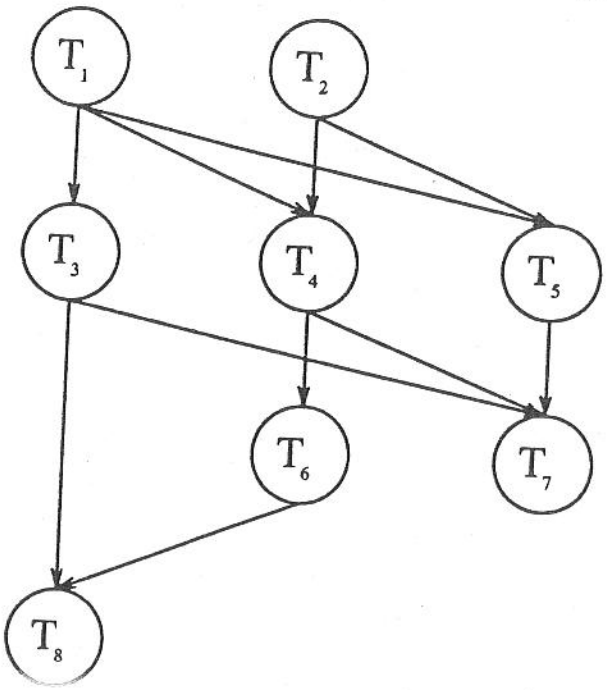
$T_3 \ll T_7$ $\quad$ $T_3 \ll T_8$

$T_4 \ll T_6$ $\quad$ $T_4 \ll T_7$ $\quad$ $T_4 \ll T_8$

$T_5 \ll T_7$

$T_6 \ll T_8$

Αν απαλείψουμε τους περιορισμούς που πλεονάζουν προκύπτει το παρακάτω γράφημα εργασιών:




Μια **διάταξη** συμβιβαστή με ένα σύστημα προτεραιότητας $S = (T_1, \ldots, T_n, \ll)$ είναι μια απεικόνιση

Ord: $\{T_1, \ldots, T_n\} \rightarrow \{1, \ldots, p\} \times \mathrm{N}$ (όπου $p$ ο αριθμός των επεξεργαστών)

$$T_k \rightarrow Ord\,(T_k) = (prc\,(T_k), tps\,(T_k)),$$

έτσι ώστε:

(i) $T_i \ll T_k$ σημαίνει $tps\,(T_i) + t\,(T_i) \leq tps\,(T_k)$

(ii) $prc\,(T_i) = prc\,(T_k)$ σημαίνει $(tps\,(T_i) + t\,(T_i) \leq tps\,(T_k)$ ή $tps\,(T_k) + t\,(T_k) \leq tps\,(T_i))$

όπου $prc\,(T_k)$ είναι ο επεξεργαστής που θα εκτελέσει την εργασία $T_k$ και $tps\,(T_k)$ είναι η χρονική στιγμή που αρχίζει η εκτέλεση της $T_k$. Η συνθήκη (i) σημαίνει απλά ότι αν οι εργασίες $T_i$ και $T_k$ δεν είναι ανεξάρτητες τότε η εκτέλεση της $T_k$ δεν μπορεί να αρχίσει πριν να τελειώσει η εκτέλεση της $T_i$. Η συνθήκη (ii) σημαίνει ότι ένας επεξεργαστής δεν μπορεί να εκτελέσει δυό διαφορετικές εργασίες την ίδια χρονική στιγμή.

Ενας **παράλληλος αλγόριθμος** που αντιστοιχεί σ'ένα ακολουθιακό αλγόριθμο $A = (T_1, \ldots, T_n, \rightarrow$ ) είναι ένα ζεύγος (S,Ord) που αποτελείται από ένα σύστημα προτεραιότητας S που αντιστοιχεί στον A και από μια διάταξη Ord συμβιβαστή με το S.

Ο χρόνος εκτέλεσης του παράλληλου αλγορίθμου $T_{PA}$ είναι η χρονική στιγμή του τέλους της εκτέλεσης της τελευταίας εκτελούμενης εργασίας. Ετσι λοιπόν είναι:

$$T_{PA} = \max_{k=1(1)n} (tps\,(T_k) + t\,(T_k))$$

Ενας παράλληλος αλγόριθμος που αντιστοιχεί σ'ένα ακολουθιακό αλγόριθμο A είναι **βέλτιστος** (optimum), όταν δεν υπάρχει κανένας άλλος παράλληλος αλγόριθμος αντίστοιχος του A με μικρότερο

χρόνο εκτέλεσης. Για λόγους απλότητας, προσπαθούμε να κατασκευάσουμε ασυμπτωτικά βέλτιστους αλγόριθμους.

Ενας παράλληλος αλγόριθμος είναι **ασυμπτωτικά βέλτιστος** για $p$ επεξεργαστές, όταν ο λόγος του χρόνου εκτέλεσής του $T_p$ και του $T_{opt}$ τείνει στο 1, καθώς το $n$ (η τάξη του προβλήματος) τείνει στο άπειρο. Δηλαδή, όταν η διαφορά μεταξύ των δυο χρόνων είναι αμελητέα.

Αν A είναι ένας ακολουθιακός αλγόριθμος και $p$ είναι ο αριθμός των επεξεργαστών, τότε στο πλαίσιο του παραλληλισμού των αλγορίθμων μελετώνται ορισμένα βασικά προβλήματα, τα οποία διατυπώνονται ως εξής:

(1) Αν υποθέσουμε ότι διαθέτουμε απεριόριστο αριθμό επεξεργαστών, ποιός είναι ο χρόνος $T_{opt}$ ενός βέλτιστου παράλληλου αλγόριθμου; Αν $T_{opt}(p)$ είναι ο χρόνος εκτέλεσης ενός βέλτιστου παράλληλου αλγόριθμου με p επεξεργαστές, τότε ορίζεται:

$$T_{opt} = \min_{p=1(1)n} (T_{opt}(p))$$

(2) Ποιός είναι ο ελάχιστος αριθμός των επεξεργαστών $p_{opt}$ που επιτρέπει να υλοποιηθεί ένας αλγόριθμος με χρόνο εκτέλεσης $T_{opt}$ ; Ο $p_{opt}$ ορίζεται ως εξής:

$$p_{opt} = \min_{p=1(1)n} (p : \ T_{opt}(p) = T_{opt})$$

(3) Οταν δίνονται $p$ επεξεργαστές, ποιά είναι η βέλτιστη τιμή της αποδοτικότητας $E_{opt}(p)$ σε σχέση με την τάξη του προβλήματος;

(4) Υπάρχει κάποιος από τους παραλληλους αλγορίθμους που να εκτελείται με $p$ επεξεργαστές και να έχει την βέλτιστη αποδοτικότητα $E_{opt}(p)$;

(5) Ποιός είναι ο αριθμός των επεξεργαστών που μεγιστοποιούν την αποδοτικότητα;.

Για μια λεπτομερή μελέτη της διάταξης ενός συστήματος προτεραιότητας παραπέμπουμε τον αναγνώστη στα συγγράμματα των Coffman και Denning [5] και Coffman [4]. Το πρόβλημα αυτό δεν έχει λυθεί γενικά παρά μόνο στις δυο περιπτώσεις: (i) όταν ο αριθμός των επεξεργαστών είναι 2 και όλες οι εργασίες έχουν τον ίδιο χρόνο εκτέλεσης και (ii) όταν το γράφημα των εργασιών είναι ένα δενδρο και όλες οι εργασίες έχουν τον ίδιο χρόνο εκτέλεσης. Ακόμα και για την περίπτωση που έχουμε μια σχέση χωρίς προτεραιότητες (όλες οι εργασίες είναι ανεξάρτητες), δεν είναι γνωστό ποιός είναι ο βέλτιστος αλγόριθμος. Υπενθυμίζουμε ότι ο χρόνος εκτέλεσης ενός μονοπατιού (δρόμου) στο γράφημα των εργασιών είναι το άθροισμα των χρόνων εκτέλεσης των εργασιών που το αποτελούν. Το μακρύτερο μονοπάτι στο γράφημα εργασιών είναι εκείνο με το μεγαλύτερο χρόνο εκτέλεσης.

Το **ύψος** H(G) ενός γραφήματος εργασιών είναι ο αριθμός των εργασιών του μακρύτερου μονοπατιού.

Οι **προηγούμενες** εργασίες της $T_k$ είναι οι εργασίες $T_i$ τέτοιες ώστε $T_i \ll T_k$. Η $T_i$ είναι ένας **προκάτοχος** της $T_k$ και η $T_k$ είναι ένας **διάδοχος** της $T_i$ αν οι $T_i$ και $T_k$ είναι διαδοχικές.

Ονομάζουμε **διαχωρισμό σε επίπεδα** του γραφήματος των εργασιών $D(G) = \{L_1, \ldots, L_{H(G)}\}$ μια διαμέριση σε H(G) υποσύνολα (επίπεδα) από κόμβους του γραφήματος αυτού, τα οποία ικανοποιούν τις ακόλουθες συνθήκες.

Το επίπεδο 1 αποτελείται από τις εργασίες που δεν έχουν κανένα προκάτοχο. Το επίπεδο k αποτελείται από τις εργασίες των οποίων όλοι οι προκάτοχοι ανήκουν σε κατώτερα επίπεδα και όλοι οι διάδοχοι ανήκουν σε ανώτερα επίπεδα. Το επίπεδο H(G) αποτελείται από τις εργασίες που δεν έχουν κανένα διάδοχο.

Επειδή το H(G) είναι ο αριθμός των εργασιών του μακρύτερου μονοπατιού παρατηρούμε ότι κάθε επίπεδο δεν είναι κενό, δηλαδή περιέχει μια εργασία από καθένα απο τα μακρύτερα μονοπάτια. Επίσης παρατηρούμε ότι το ίδιο γράφημα επιδέχεται περισσότερους από ένα διαχωρισμούς σε επίπεδα.

Δυο ειδικοί διαχωρισμοί είναι ο **διαχωρισμός σε προκατόχους** όταν κάθε επίπεδο περιέχει όλους τους δυνατούς προκατόχους μιας εργασίας και ο **διαχωρισμός σε διαδόχους**. Ο πρώτος διαχωρισμός ονομάζεται επίσης **διαχωρισμός ως προς το πλέον ενωρίτερο** και ο δεύτερος **διαχωρισμός ως προς το πλέον αργότερο**, επειδή μια διάταξη σε επίπεδα προϋποθέτει τη χρονική στιγμή της έναρξης της ελάχιστης εκτέλεσης στην πρώτη περίπτωση και της μεγίστης στη δεύτερη περίπτωση για καθεμιά από τις εργασίες.

**Πλάτος** ενός διαχωρισμού $D(G) = \{L_1, L_2, \ldots, L_{H(G)}\}$ ονομάζεται η μέγιστη απο τις τάξεις των επιπέδων, δηλαδή:

$$L(D) = \max_{k=1(1)H(G)} (\text{τάξη } (L_k))$$

**Πλάτος L(G)** του γραφήματος των εργασιών ονομάζεται το ελάχιστο από τα πλάτη των διαχωρισμών του G, δηλαδή:

$$L(G) = \min_{D(G)} L(D) = \min_{D(G)} \max_{k=1(1)H(G)} (\text{τάξη } (L_k))$$

**Παράδειγμα** Στο γράφημα των εργασιών του προηγούμενου παραδείγματος, το ύψος είναι 4 και οι δυνατοί διαχωρισμοί ανά επίπεδα είναι οι παρακάτω:

1. $L_1 = \{1,2\}$ $L_2 = \{3,4,5\}$ $L_3 = \{6,7\}$ $L_4 = \{8\}$
2. $L_1 = \{1,2\}$ $L_2 = \{3,4\}$ $L_3 = \{5,6\}$ $L_4 = \{7,8\}$
3. $L_1 = \{1,2\}$ $L_2 = \{4,5\}$ $L_3 = \{3,6\}$ $L_4 = \{7,8\}$
4. $L_1 = \{1,2\}$ $L_2 = \{4\}$ $L_3 = \{3,5,6\}$ $L_4 = \{7,8\}$

Το γράφημα έχει πλάτος 2.

Ολοι οι παραπάνω ορισμοί οδηγούν στο ακόλουθο αποτέλεσμα:

Αν $A = (T_1, \ldots, T_n, \rightarrow)$ είναι ένας ακολουθιακός αλγόριθμος του οποίου όλες οι εργασίες στο ίδιο επίπεδο $L_k$ έχουν τον ίδιο χρόνο εκτέλεσης $t_k$ τότε ο χρόνος $T_{opt}$ ενός βέλτιστου παράλληλου αλγορίθμου είναι ίσος με τον χρόνο εκτέλεσης του μακρύτερου μονοπατιού στο γράφημα των εργασιών. Δηλαδή είναι:

$$T_{opt} = \sum_{k=1}^{H(G)} t_k$$

*Είναι δυνατόν να έχουμε διαφορετικά γραφήματα διάταξης για τον ίδιο σειριακό αλγόριθμο, το οποίο οφείλεται σε διαφόρους λόγους όπως είναι το περιεχόμενο μιας εργασίας, ο διπλασιασμός(αντιγραφή) μερικών δεδομένων και συνεπώς ένας ακολουθιακός αλγόριθμος μπορεί να οδηγήσει σε πολλές παράλληλες μορφές.* Για να ορίσουμε ένα παράλληλο αλγόριθμο ακολουθούμε την παρακάτω πορεία:

1. Διαχωρίζουμε τον ακολουθιακό αλγόριθμο σε εργασίες.
2. Ορίζουμε τη σχέση προτεραιότητας ως πρός τη σειρά εκέλεσης των εργασιών και κατασκευάζουμε το γράφημα των εργασιών.

3. Αναθέτουμε σε δεδομένη στιγμή την εκτέλεση μιας δεδομένης εργασίας στον αντίστοιχο επεξεργαστή σύμφωνα με τους περιορισμούς προτεραιότητας.

Ο παρακάτω βρόχος (loop) B:

B: Do i=1(1)n
$T_i$
End

μπορεί να εκτελεσθεί παράλληλα με πολλούς τρόπους σύμφωνα με τον αριθμό των επεξεργαστών και τους περιορισμούς προτεραιότητας. Αν υποθέσουμε ότι διατίθενται $p$ επεξεργαστές τότε οι διαφορετικοί τρόποι παράλληλης εκτέλεσης του βρόχου B βρίσκονται ανάμεσα στις δύο παρακάτω ακραίες περιπτώσεις παράλληλης εκτέλεσης.

### 1. Βρόχος τελείως παράλληλος

Η εκκίνηση της εκτέλεσης των εργασιών γίνεται ταυτόχρονα, δηλαδή έχουμε $tps(T_i) = tps(T_{i+1})$ για κάθε $i = 1, 2, \ldots, n-1$. Δεν υπάρχει κανείς περιορισμός προτεραιότητας μεταξύ των εργασιών. Επομένως όλες οι εργασίες βρίσκονται στι ίδιο επίπεδο και συνεπώς επιτρέπεται η ταυτόχρονη εκτέλεση όλων των εργασιών. Το πλάτος του διαχωρισμού είναι $L(D) = n$ και το μακρύτερο μονοπάτι αποτελείται από τη μεγαλύτερη εργασία. Αν ο αριθμός p των επεξεργαστών είναι μικρότερος από το $n$ $(p < n)$ τότε υπάρχει ένας τουλάχιστον επεξεργαστής που εκτελεί περισσότερες από μια εργασίες. Αντίθετα, αν $p \geq n$ τότε ο κάθε επεξεργαστής εκτελεί το πολύ μία εργασία και ο απαιτούμενος χρόνος είναι ο $T_{opt}$.

Στην πρώτη περίπτωση, όπου είναι $p < n$ και αν οι εργασίες έχουν τον ίδιο χρόνο εκτέλεσης, δεν είναι δυνατόν να βρεθεί ένας παράλληλος αλγόριθμος που να εκτελείται σε χρόνο $T_{opt}$.

### 2. Βρόχος τελείως ακολουθιακός

Η εκκίνηση της εκτέλεσης μιας εργασίας γίνεται αφού έχει τελειώσει η εκτέλεση κάποιας άλλης. Δηλαδή, η εκτέλεση μιας οποιασδήποτε εργασίας $T_i$ $(i \neq 1)$ εξαρτάται αναγκαστικά από μία τουλάχιστον άλλη εργασία. Αν υποθέσουμε ότι αυτή είναι η εργασία $T_{i+1}$, συνεπώς ισχύει $tps(T_{i+1}) = tps(T_i) + t(T_i)$ για κάθε $i = 1, 2, \ldots, n-1$, δηλαδή οι εργασίες εκτελούνται ακολουθιακά. Το γράφημα προτεραιοτήτων αποτελείται από n επίπεδα όπου κάθε εργασία αποτελεί ένα επίπεδο. Το πλάτος διαχωρισμού είναι 1. Το μακρύτερο μονοπάτι αποτελείται από όλες τις εργασίες και το μήκος του είναι ο χρόνος $T_{opt}$. Για τη περίπτωση αυτή δεν υπάρχει κανένας παράλληλος αλγόριθμος.

## 1.6 ΕΝΑ ΘΕΩΡΗΤΙΚΟ ΜΟΝΤΕΛΟ

Στη μελέτη αυτή δεν μας ενδιαφέρουν τα ακριβή χαρακτηριστικά της αρχιτεκτονικής πάνω στη οποία προγραμματίζονται οι αλγόριθμοι. Θεωρούμε απλά ένα σύστημα το οποίο να υποστηρίζει την πολλαπλή ροή εντολών που εκτελούνται ανεξάρτητα πάνω σε πολλαπλή ροή δεδομένων και κοινή μνήμη [26],[48]. Υποθέτουμε ότι υπάρχουν οι κατάλληλοι μηχανισμοί που συγχρονίζουν την πορεία της εκτέλεσης, όπως είναι η υλοποίηση επί της αρχιτεκτονικής της τήρησης των χρονικών περιορισμών προτεραιότητας όπως επιβάλονται από τη φύση των αλγορίθμων. Το κόστος των μηχανισμών συγχρονισμού είναι αμελητέο σε σχέση με το κόστος των αριθμητικών πράξεων. Επίσης θεωρείται αμελητέος ο χρόνος προσπέλασης στη κεντρική μνήμη για την ανάγνωση ή αποθήκευση ενός δεδομένου. Οι παραπάνω προϋποθέσεις για να προσεγγίζουν την πραγματικότητα, πρέπει να συνοδεύονται από τους ακόλουθους περιορισμούς:

1. Η επικοινωνία μεταξύ των επεξεργαστών γίνεται πολύ ταχύτερη μέσω μιας διαμοιραζόμενης μνήμης απ'ότι με μια τοπική αρτηρία (bus).

2. Για ένα πρόβλημα τάξης n, ο αριθμός p των επεξεργαστών περιορίζεται στην τάξη $O(n)$. Θα υποθέσουμε ότι είναι $p = \alpha n$ με $\alpha < 1$ οπότε διευκολύνεται η εκτίμηση της αποδοτικότητας ενός παράλληλου αλγορίθμου. Για την εκτίμηση αυτή θεωρούμε αμελητέους τους χρόνους προσπέλασης στην κεντρική μνήμη, τους χρόνους αποθήκευσης και εναλλαγής δεδομένων.

Αν θεωρήσουμε αμελητέο τον χρόνο επικοινωνίας, επιβάλλουμε ένα συγκεκριμμένο τρόπο προσπέλασης στα δεδομένα. Υποθέτουμε επίσης ότι η προσπέλαση στα στοιχεία ενός πίνακα γίνεται κατα στήλες. Τότε οι δυο δυνατές λειτουργίες μεταφοράς είναι η μεταφορά μιας στήλης του A από το μέσο αποθήκευσης πρός το μέσο διαχείρισης και η αντίστροφη λειτουργία. Ο διπλασιασμός (αντίγραφο) μιας στήλης θα έχει μηδενικό κόστος, δηλαδή υποθέτουμε ότι είναι δυνατόν να μεταφερθεί το ίδιο δεδομένο ταυτόχρονα σε πολλούς επεξεργαστές. Στην περίπτωση αυτή κανένας επεξεργαστής δεν μπορεί να τροποποιήσει αυτό το δεδομένο. Αντίστροφα, ένας επεξεργαστής δεν μπορεί να τροποποιήσει ένα δεδομένο, αν είναι ο μόνος που κατέχει ένα αντίγραφο. Δεν θα ασχοληθούμε με τους μηχανισμούς που εξασφαλίζουν τη λύση του προβλήματος κατανομής των δεδομένων χωρίς σύγκρουση. Παρατηρούμε απλά ότι οι περιορισμοί προτεραιότητας και ο μηχανισμός που τους διευθύνει εξασφαλίζουν μια ροή του αλγόριθμου χωρίς αμφιβολία και ένα ασφαλή συγχρονισμό.

Για τους επεξεργαστές διανύσματος (vector processors) το κόστος μιας αριθμητικής πράξης εκτιμάται όπως παρακάτω:
Υποθέτουμε ότι ο αθροιστής (αντίστοιχα ο πολλαπλασιαστής) αποτελείται από $e_+$ (αντίστοιχα $e_\times$) τμήματα. Ο χρόνος που χρειάζεται για να εκτελεσθεί μια αριθμητική εντολή πάνω σε δυό διανύσματα μήκους n είναι:

$$(e_i + (n-1))\, t$$

όπου $i \in \{+, \times\}$ και t είναι ο χρόνος διάβασης ενός τμήματος.
Αν είναι δυνατή η σύνδεση των αριθμητικών πράξεων, τότε για παράδειγμα ο συνολικός χρόνος για να εκτελεσθεί μια πρόσθεση που ακολουθεί ένα πολλαπλασιασμό είναι:

$$(e_+ + e_\times + (n-1))\, t$$

Στη συνέχεια υπολογίζουμε μόνο το αριθμητικό κόστος επιλέγοντας σαν χρονική μονάδα τον χρόνο εκτέλεσης μιας από τις τέσσερις αριθμητικές πράξεις. Σύμφωνα με την ανάλυση πολυπλοκότητας που έχει προηγηθεί για την μέθοδο απαλοιφής του Gauss με μερική οδήγηση [30] υποθέτουμε γι'αυτόν τον αλγόριθμο ότι κάθε επεξεργαστής μπορεί να εκτελέσει ένα πολλαπλασιασμό και μια αφαίρεση , ή μια σύγκριση και ένα πολλαπλασιασμό ή ακόμα μια διαίρεση.

Αν θεωρήσουμε ότι ο χρόνος επικοινωνίας είναι αμελητέος τότε απλοποιούνται οι υπολογισμοί για την εκτίμηση της αποδοτικότητας των παράλληλων αλγορίθμων. Για να προσεγγίσουμε καλύτερα την πραγματικότητα πρέπει οπωσδήποτε να υπολογίσουμε και τους χρόνους επικοινωνίας στην εκτίμηση της αποδοτικότητας. Το μοντέλο επεκτείνεται στα κόστη επικοινωνίας σε μια διαμοιραζόμενη μνήμη. Το κόστος μιας εργασίας, που περιέχει q πράξεις εκτιμάται με το άθροισμα του αριθμητικού κόστους:

$$\tau_a = aq + b$$

και του κόστους επικοινωνίας:

$$\tau_c = \alpha q + \beta$$

Οι παράμετροι $a, b, \alpha$ και $\beta$ είναι συναρτήσεις των χαρακτηριστικών της μηχανής. Ο χρόνος εκτέλεσης ενός παράλληλου αλγορίθμου είναι λοιπόν:

$$\tau = \sum_{\text{εργασίες}} (\tau_a + \tau_c)$$

Αν ο αριθμός $p$ των επεξεργαστών είναι αρκετά μικρότερος από την τάξη $n$ του πινακοποιημένου προβλήματος τότε το κόστος επικοινωνίας είναι αμελητέο σε σχέση με το υπολογιστικό κόστος. Στην περίπτωση αυτή, το μοντέλο που έχουμε επιλέξει είναι πολύ κοντά στη πραγματικότητα και τα αποτελέσματα της πολυπλοκότητας δίνουν μια ακριβή ιδέα για τις πραγματικές επιδόσεις του παράλληλου αλγορίθμου.

Στη συνέχεια θα συμβολίζεται με $T_p(n)$ ο χρόνος που απαιτείται για την εκτέλεση ενός παράλληλου αλγορίθμου σε ένα σύστημα που αποτελείται από $p$ επεξεργαστές για την επίλυση ενός προβλήματος μεγέθους $n$. Ο χρόνος εκτέλεσης του ακολουθιακού αλγόριθμου, δηλαδή ο χρόνος εκτέλεσης με ένα μόνο επεξεργαστή θα συμβολίζεται με $T_1(n)$.

Η **επιτάχυνση** ενός παράλληλου αλγορίθμου με $p$ επεξεργαστές είναι το πηλίκο:

$$S_p(n) = \frac{T_1(n)}{T_p(n)}$$

Επίσης η **αποδοτικότητα** ενός παράλληλου αλγορίθμου με $p$ επεξεργαστές είναι το πηλίκο:

$$E_p(n) = \frac{S_p(n)}{p} = \frac{T_1(n)}{p\,T_p(n)}$$

Η αποδοτικότητα ενός αλγορίθμου στην πραγματικότητα είναι ένα μέτρο εκτίμησης του ποσοστού απασχόλησης των επεξεργαστών. Επειδή η εκτέλεση ενός παράλληλου αλγορίθμου μπορεί πάντοτε να εξομοιωθεί με την εκτέλεση του ακολουθιακά σε χρόνο $T_1$, ίσο με τον χρόνο $T_p$ επί τον αριθμό $p$ των επεξεργαστών, θα μπορούσε να αναμένει κανείς ότι ισχύει $S_p = p$ και $E_p = 1$.

Ομως αυτό δεν συμβαίνει στην πράξη, διότι υπάρχουν διάφοροι παράγοντες που δεν επιτρέπουν τη βέλτιστη εκμετάλλευση των διαθέσιμων επεξεργαστών. Ενας σημαντικός παράγοντας που μειώνει την αποδοτικότητα ενός αλγορίθμου είναι το γεγονός ότι κατά την εκτέλεσή του μπορεί ορισμένοι επεξεργαστές να μένουν ανενεργοί (idle), ενώ οι υπόλοιποι να λειτουργούν. Αυτό συνήθως οφείλεται στους περιορισμούς προτεραιότητας που επιβάλλουν οι αλληλοεξαρτήσεις των εργασιών του αλγορίθμου. Δεύτερος λόγος είναι το κόστος επικοινωνίας που εισάγεται λόγω της παραλληλίας.

Ετσι λοιπόν συνήθως είναι $E_p < 1$ και ο παραλληλισμός ενός ακολουθιακού αλγόριθμου είναι τόσο καλύτερος (αποτελεσματικότερος) όσο το $E_p$ πλησιάζει τη τιμή 1. (Θεωρητικά μπορεί να είναι και $E_p > 1$).

Στο Κεφάλαιο 1 παρουσιάζεται μια ταξινόμηση των παραλλήλων αρχιτεκτονικών και δίνεται ιδιαίτερη έμφαση στους MIMD Υπολογιστές με διαμοιραζόμενη και με κατανεμημένη μνήμη. Στη συνέχεια δίνονται ορισμένα βασικά στοιχεία για την επικοινωνία μεταξύ των επεξεργαστών μιας παράλληλης αρχιτεκτονικής για ένα σύστημα με κοινή διαμοιραζόμενη μνήμη. Ενας αλγόριθμος μπορεί να διαχωρισθεί σ'ένα σύνολο από εργασίες. Η μελέτη της αλληλοεξάρτησης των εργασιών σχετικά με τις μεταβλητές των εισόδων και εξόδων τους μας επιτρέπει να κατασκευάσουμε ένα σύστημα προτεραιότητας, το οποίο ερμηνεύει τον εσωτερικό παραλληλισμό του αλγορίθμου. Σ'ένα τέτοιο σύστημα μπορεί να υπάρχουν περιορισμοί προτεραιότητας στη σειρά εκτέλεσης των εργασιών και στη περίπτωση αυτή η εκτέλεσή τους είναι ακολουθιακή. Στην αντίθετη περίπτωση, οι εργασίες είναι μη αλληλοεξαρτώμενες, οπότε η εκτέλεσή τους μπορεί να πραγματοποιηθεί παράλληλα. Ορίζουμε λοιπόν στη συνέχεια ορισμένες έννοιες, οι οποίες είναι απαραίτητες για τη δημιουργία ενός παράλληλου αλγορίθμου καθώς και την εκτίμηση της αποτελεσματικότητάς του.

Στο Κεφάλαιο 2 περιγράφεται η ακολουθιακή μέθοδος απαλοιφής του Gauss(GE) και μελετώνται οι παράλληλες μορφές της με $O(n)$ και $O(n^2)$ επεξεργαστές. Πιο συγκεκριμένα για κάθε παράλληλη μορφή κατασκευάζεται το γράφημα αλληλοεξάρτησης των εργασιών, προτείνεται μια βέλτιστη δρομολόγηση των επεξεργαστών και στη συνέχεια γίνεται ασυμπτωτική μελέτη της επιτάχυνσης και της αποδοτικότητάς της.

Αν ο αριθμός $p$ των επεξεργαστών είναι αρκετά μικρότερος από την τάξη $n$ του πινακοποιημένου προβλήματος τότε το κόστος επικοινωνίας είναι αμελητέο σε σχέση με το υπολογιστικό κόστος. Στην περίπτωση αυτή, το μοντέλο που έχουμε επιλέξει είναι πολύ κοντά στη πραγματικότητα και τα αποτελέσματα της πολυπλοκότητας δίνουν μια ακριβή ιδέα για τις πραγματικές επιδόσεις του παράλληλου αλγορίθμου.

Στη συνέχεια θα συμβολίζεται με $T_p(n)$ ο χρόνος που απαιτείται για την εκτέλεση ενός παράλληλου αλγορίθμου σε ένα σύστημα που αποτελείται από $p$ επεξεργαστές για την επίλυση ενός προβλήματος μεγέθους $n$. Ο χρόνος εκτέλεσης του ακολουθιακού αλγόριθμου, δηλαδή ο χρόνος εκτέλεσης με ένα μόνο επεξεργαστή θα συμβολίζεται με $T_1(n)$.

Η **επιτάχυνση** ενός παράλληλου αλγορίθμου με $p$ επεξεργαστές είναι το πηλίκο:

$$S_p(n) = \frac{T_1(n)}{T_p(n)}$$

Επίσης η **αποδοτικότητα** ενός παράλληλου αλγορίθμου με $p$ επεξεργαστές είναι το πηλίκο:

$$E_p(n) = \frac{S_p(n)}{p} = \frac{T_1(n)}{p\,T_p(n)}$$

Η αποδοτικότητα ενός αλγορίθμου στην πραγματικότητα είναι ένα μέτρο εκτίμησης του ποσοστού απασχόλησης των επεξεργαστών. Επειδή η εκτέλεση ενός παράλληλου αλγορίθμου μπορεί πάντοτε να εξομοιωθεί με την εκτέλεση του ακολουθιακά σε χρόνο $T_1$, ίσο με τον χρόνο $T_p$ επί τον αριθμό $p$ των επεξεργαστών, θα μπορούσε να αναμένει κανείς ότι ισχύει $S_p = p$ και $E_p = 1$.

Ομως αυτό δεν συμβαίνει στην πράξη, διότι υπάρχουν διάφοροι παράγοντες που δεν επιτρέπουν τη βέλτιστη εκμετάλλευση των διαθέσιμων επεξεργαστών. Ενας σημαντικός παράγοντας που μειώνει την αποδοτικότητα ενός αλγορίθμου είναι το γεγονός ότι κατά την εκτέλεσή του μπορεί ορισμένοι επεξεργαστές να μένουν ανενεργοί (idle), ενώ οι υπόλοιποι να λειτουργούν. Αυτό συνήθως οφείλεται στους περιορισμούς προτεραιότητας που επιβάλλουν οι αλληλοεξαρτήσεις των εργασιών του αλγορίθμου. Δεύτερος λόγος είναι το κόστος επικοινωνίας που εισάγεται λόγω της παραλληλίας.

Ετσι λοιπόν συνήθως είναι $E_p < 1$ και ο παραλληλισμός ενός ακολουθιακού αλγόριθμου είναι τόσο καλύτερος (αποτελεσματικότερος) όσο το $E_p$ πλησιάζει τη τιμή 1. (Θεωρητικά μπορεί να είναι και $E_p > 1$).

# 3 ΠΑΡΑΛΛΗΛΕΣ ΜΟΡΦΕΣ ΤΗΣ ΜΕΘΟΔΟΥ ΑΠΑΛΟΙΦΗΣ ΤΩΝ GAUSS-JORDAN

## 3.1 ΕΙΣΑΓΩΓΗ

Στο κεφάλαιο αυτό εξετάζεται η άμεση μέθοδος απαλοιφής των Gauss-Jordan [9],[19], [24] για την αριθμητική επίλυση συστημάτων της μορφής

$$Ax = b \tag{3.1.1}$$

όπου A είναι μη ιδιάζων πίνακας.

Η μέθοδος αυτή μετασχηματίζει το αρχικό σύστημα $Ax = b$ σ'ενα ισοδύναμο $Dx = c$, όπου D είναι ένας διαγώνιος πίνακας και έτσι η λύση του συστήματος προκύπτει άμεσα. Η βασική διαφορά της μεθόδου αυτής σε σχέση με τη μέθοδο απαλοιφής του Gauss είναι ότι αποφεύγεται η λύση ενός τριγωνικού συστήματος που είναι ένα πρόβλημα με μικρό βαθμό παραλληλίας στην εκτέλεση των απαιτουμένων υπολογιστικών πράξεων.

Η διαφορά αυτή ως προς τον παραλληλισμό των υπολογιστικών πράξεων μεταξύ των δύο μεθόδων γίνεται πλέον σημαντική στην περίπτωση που έχουμε να λύσουμε ένα μεγάλο αριθμό γραμμικών συστημάτων με τον ίδιο πίνακα $A$. Μία ειδική περίπτωση είναι ο υπολογισμός του αντιστρόφου ενός πίνακα που ανάγεται στην λύση των ακόλουθων γραμμικών συστημάτων

$$Ax_k = e_k,\ k = 1, 2, ..., n$$

όπου $e_k = (0, 0, ..., 0, \underset{\text{k-θέση}}{1}, 0, ..., 0)^T$.

Στο κεφάλαιο αυτό εισάγονται και μελετώνται οι παράλληλες μορφές της μεθόδου απαλοιφής του Gauss-Jordan και Huard με $O(n^2)$ επεξεργαστές.

## 3.2 ΜΕΘΟΔΟΣ ΑΠΑΛΟΙΦΗΣ ΤΩΝ GAUSS-JORDAN (GJ)

Η αρχή της ακολουθιακής μορφής της μεθόδου απαλοιφής των Gauss-Jordan (GJ) συνίσταται στο να απαλείφει κατά στήλες το ένα μετά το άλλο τα στοιχεία του πίνακα A των συντελεστών των αγνώστων. Ετσι είναι δυνατόν να μετασχηματισθεί ο A σε ένα διαγώνιο πίνακα. Η μορφή αυτή του πίνακα είναι απλούστερη απο εκείνη του Gauss καθόσον ο υπολογισμός του αγνώστου $x_i$ μπορεί να προκύψει με μια απλή διαίρεση. Θεωρώντας το αρχικό γραμμικό σύστημα:

$$A^{(1)}x = b^{(1)} \quad \text{όπου} \quad A^{(1)} = A \quad \text{και} \quad b^{(1)} = b$$

το πρώτο βήμα της απαλοιφής GJ είναι ακριβώς το ίδιο με εκείνο της μεθόδου του Gauss. Ετσι έχουμε μετά την εκτέλεση του πρώτου βήματος το ισοδύναμο σύστημα:

$$\begin{array}{ccccccccccc}
\alpha_{11}^{(1)}x_1 & + & \alpha_{12}^{(1)}x_2 & + & \alpha_{13}^{(1)}x_3 & + & \cdots & + & \alpha_{1n}^{(1)}x_n & = & b_1^{(1)} \\
 & & \alpha_{22}^{(2)}x_2 & + & \alpha_{23}^{(2)}x_3 & + & \cdots & + & \alpha_{2n}^{(2)}x_n & = & b_2^{(2)} \\
 & & \alpha_{32}^{(2)}x_2 & + & \alpha_{33}^{(2)}x_3 & + & \cdots & + & \alpha_{3n}^{(2)}x_n & = & b_3^{(2)} \\
 & & \vdots & & & & & & & & \\
 & & \alpha_{n2}^{(2)}x_2 & + & \alpha_{n3}^{(2)}x_3 & + & \cdots & + & \alpha_{nn}^{(2)}x_n & = & b_n^{(2)}
\end{array} \tag{3.2.1}$$

όπου παρατηρούμε ότι ο $x_1$ έχει απαλειφθεί από τις $n-1$ τελευταίες εξισώσεις οπότε έχουμε πάλι το σύστημα:

$$A^{(2)}x = b^{(2)} \tag{3.2.2}$$

Ομως στο δεύτερο βήμα της μεθόδου απαλοιφής GJ ο $x_2$ απαλείφεται όχι μονο από τις $n-2$ τελευταίες εξισώσεις (όπως συμβαίνει στη μέθοδο απαλοιφής του Gauss), αλλά συγχρόνως και από την πρώτη εξίσωση. Ετσι προκύπτει το ισοδύναμο σύστημα:

$$\begin{aligned}
\alpha_{11}^{(1)}x_1 \quad &+ \quad \alpha_{13}^{(3)}x_3 \quad + \quad \cdots \quad + \quad \alpha_{1n}^{(3)}x_n \quad = \quad b_1^{(3)} \\
\alpha_{22}^{(2)}x_2 \quad &+ \quad \alpha_{23}^{(2)}x_3 \quad + \quad \cdots \quad + \quad \alpha_{2n}^{(2)}x_n \quad = \quad b_2^{(2)} \\
\alpha_{33}^{(2)}x_3 \quad &+ \quad \cdots \quad + \quad \alpha_{3n}^{(2)}x_n \quad = \quad b_3^{(2)} \\
&\vdots \\
\alpha_{n3}^{(2)}x_3 \quad &+ \quad \cdots \quad + \quad \alpha_{nn}^{(2)}x_n \quad = \quad b_n^{(2)}
\end{aligned} \tag{3.2.3}$$

ή

$$A^{(3)}x = b^{(3)} \tag{3.2.4}$$

Για λόγους ομοιομορφίας στον συμβολισμό υποθέτουμε ότι αλλάζουμε στο (3.2.3) τους πάνω δείκτες των συντελεστών της δεύτερης γραμμής εκτός του πρώτου. Μετά από $k-1$ τέτοια βήματα προκύπτει το ισοδύναμο σύστημα:

$$\begin{aligned}
\alpha_{11}^{(1)}x_1 \quad &+ \quad \alpha_{1k}^{(k)}x_k \quad + \quad \cdots \quad + \quad \alpha_{1n}^{(k)}x_n \quad = \quad b_1^{(k)} \\
\alpha_{22}^{(2)}x_2 \quad &+ \quad \alpha_{2k}^{(k)}x_k \quad + \quad \cdots \quad + \quad \alpha_{2n}^{(k)}x_n \quad = \quad b_2^{(k)} \\
&\ddots \quad \vdots \qquad\qquad\qquad\quad \vdots \qquad\quad \vdots \\
\alpha_{k-1,k-1}^{(k-1)}x_{k-1} \quad &+ \quad \alpha_{k-1,k}^{(k)}x_k \quad + \quad \cdots \quad + \quad \alpha_{k-1,n}^{(k)}x_n \quad = \quad b_{k-1}^{(k)} \\
&\qquad \alpha_{kk}^{(k)}x_k \quad + \quad \cdots \quad + \quad \alpha_{kn}^{(k)}x_n \quad = \quad b_k^{(k)} \\
&\qquad \vdots \qquad\qquad\qquad\quad \vdots \qquad\quad \vdots \\
&\qquad \alpha_{nk}^{(k)}x_k \quad + \quad \cdots \quad + \quad \alpha_{nn}^{(k)}x_n \quad = \quad b_n^{(k)}
\end{aligned} \tag{3.2.5}$$

ή

$$A^{(k)}x = b^{(k)} \tag{3.2.6}$$

όπου για λόγους ομοιομορφίας αλλάζουμε τους επάνω δείκτες των συντελεστών της $k-1$ γραμμής εκτός του πρώτου.

Τελικά μετα από $n$ τέτοια βήματα προκύπτει το ισοδύναμο διαγώνιο σύστημα:

$$\begin{aligned}
a_{11}^{(1)}x_1 \quad &= \quad b_1^{(n+1)} \\
a_{22}^{(2)}x_2 \quad &= \quad b_2^{(n+1)} \\
\ddots \quad &\vdots \\
a_{nn}^{(n)}x_n \quad &= \quad b_n^{(n+1)}
\end{aligned} \tag{3.2.7}$$

ή

$$A^{(n)}x = b^{(n+1)} \tag{3.2.8}$$

όπου ο $A^{(n)}$ είναι ένας διαγώνιος πίνακας.

Η λύση του (3.2.8) είναι η:

$$x_i = \frac{b_i^{(n+1)}}{a_{ii}^{(i)}}, \quad i = 1(1)n \tag{3.2.9}$$

εφ'όσον

$$a_{ii}^{(i)} \neq 0, \quad i = 1(1)n$$

Αναλυτικώτερα χρησιμοποιούμε πάλι τους συμβολισμούς

$$a_{ij}^{(1)} = a_{ij}, \qquad i,j = 1(1)n$$

$$b_i^{(1)} = b_i, \qquad i = 1(1)n$$

και αν $a_{11}^{(1)} \neq 0$ τότε ορίζουμε τους πολλαπλασιαστές

$$m_{i1} = -\frac{a_{i1}^{(1)}}{a_{11}^{(1)}}, \quad i = 2(1)n$$

Τότε για να απαλειφθεί ο $x_1$ από την i-οστή εξίσωση (i=2(1)n), αρκεί να προστεθεί στην i-οστή εξίσωση η πρώτη εξίσωση πολλαπλασιασμένη με $m_{i1}$, οπότε προκύπτει το σύστημα (3.2.1), όπου

$$\begin{aligned} & a_{ij}^{(2)} = a_{ij}^{(1)} + m_{i1}a_{1j}^{(1)}, && i = 1(1)n, && i \neq 1 \\ & && j = 1(1)n && \\ \text{και} \quad & b_i^{(2)} = b_i^{(1)} + m_{i1}b_1^{(1)}, && i = 1(1)n, && i \neq 1 \end{aligned}$$

Στο σημείο αυτό παρατηρούμε ότι πριν ακολουθήσει το δεύτερο βήμα θα πρέπει για λόγους ομοιομορφίας να κάνουμε τις αντικαταστάσεις

$$\begin{aligned} a_{1j}^{(2)} &= a_{1j}^{(1)}, \qquad j = 2(1)n \\ b_1^{(2)} &= b_1^{(1)} \end{aligned}$$

Στο δεύτερο βήμα τώρα απαλείφεται ο άγνωστος $x_2$ τόσο από τις τελευταίες $n-2$ εξισώσεις όσο και από την πρώτη εξίσωση οπότε προκύπτει το σύστημα (3.2.3), όπου

$$\begin{aligned} & a_{ij}^{(3)} = a_{ij}^{(2)} + m_{i2}a_{2j}^{(2)}, && i = 1(1)n, && i \neq 2 \\ & && j = 2(1)n && \\ \text{και} \quad & b_i^{(3)} = b_i^{(2)} + m_{i2}b_2^{(2)}, && i = 1(1)n, && i \neq 2 \end{aligned}$$

και για λόγους ομοιομορφίας θέτουμε:

$$\begin{aligned} a_{2j}^{(3)} &= a_{2j}^{(2)}, \qquad j = 3(1)n \\ b_2^{(3)} &= b_2^{(2)} \end{aligned}$$

Συνεχίζοντας καταλήγουμε στο διαγώνιο σύστημα (3.2.7)

Ο μετασχηματισμός του A σε ένα διαγώνιο πίνακα της ίδιας τάξης αποτελείται απο $n$ κύρια βήματα τα οποία αντοιστοιχούν σε $n$ πράξεις των γραμμών του πίνακα.

Υπο μορφή πινάκων η μέθοδος GJ αναζητεί μη ιδιάζοντα πίνακα M έτσι ώστε:

$$MAx = Mb \tag{3.2.10}$$

με

$$MA = I \tag{3.2.11}$$

Ας υποθέσουμε ότι το αρχικό σύστημα είναι το $A^{(1)}x = b^{(1)}$ τότε δημιουργούνται οι ακολουθίες πινάκων:

$$\begin{aligned} A^{(k+1)} &= M^{(k)}A^{(k)} \\ b^{(k+1)} &= M^{(k)}b^{(k)} \end{aligned} \quad , \quad k = 1(1)n \tag{3.2.12}$$

ώστε το γραμμικό σύστημα $A^{(k+1)}x = b^{(k+1)}$ να είναι ισοδύναμο με το $A^{(k)}x = b^{(k)}$ [32].
Μετά από $n$ βήματα οι αναδρομικοί τύποι (3.2.12) οδηγούν σ'ένα γραμμικό σύστημα της μορφής:

$$A^{(n)}x = b^{(n)} \tag{3.2.13}$$

όπου ο

$$A^{(n)} = M^{(n)}M^{(n-1)} \dots M^{(2)}M^{(1)} \tag{3.2.14}$$

είναι ένας διαγώνιος πίνακας.

Η παραπάνω διαδικασία αποτελεί "τη φάση διαγωνοποίησης" του γραμμικού συστήματος. Στο $k$-βήμα της φάσης αυτής εκτελείται ο πολλαπλασιασμός του πίνακα $M^{(k)}$ επί τον $A^{(k)}$ όπου:

$$M^{(k)} = \begin{bmatrix} & & \mu_{1k} & \\ & & \mu_{2k} & \\ I_{k-1} & & \vdots & 0 \\ & & \mu_{k-1,k} & \\ 0 \cdots 0 & & \mu_{k,k} & 0 \cdots 0 \\ & & \mu_{k+1,k} & \\ 0 & & \vdots & I_{n-k} \\ & & \mu_{n,k} & \end{bmatrix},$$

$$A^{(k)} = \begin{bmatrix} a_{11}^{(1)} & & & & a_{1k}^{(1)} & \cdots & a_{1n}^{(1)} \\ & a_{22}^{(2)} & & 0 & a_{2k}^{(2)} & \cdots & a_{2n}^{(2)} \\ & & \ddots & & \vdots & & \vdots \\ & 0 & & a_{k-1,k-1}^{(k-1)} & a_{k-1,k}^{(k-1)} & \cdots & a_{k-1,n}^{(k-1)} \\ & & & & a_{kk}^{(k)} & \cdots & a_{kn}^{(k)} \\ & & 0 & & a_{k+1,k}^{(k)} & \cdots & a_{k+1,n}^{(k)} \\ & & & & a_{nk}^{(k)} & \cdots & a_{nn}^{(k)} \end{bmatrix} \tag{3.2.15}$$

$$\mu_{ik} = \begin{cases} \dfrac{1}{a_{kk}^{(k)}}, & i = k \\ -\dfrac{a_{ik}^{(k)}}{a_{kk}^{(k)}}, & i \neq k \qquad (a_{kk}^{(k)} \neq 0) \end{cases} \tag{3.2.16}$$

Επίσης εκτελείται ο πολλαπλασιασμός του πίνακα $A^{(k)}$ επί τον $b^{(k)}$ και καταλήγουμε στο σύστημα:

$$A^{(k+1)}x = b^{(k+1)} \tag{3.2.17}$$

με τις ακόλουθες τροποποιήσεις στοιχείων των πινάκων $A^{(k)}$ και $b^{(k)}$:

$$a_{ij}^{(k+1)} = a_{ij}^{(k)} + \mu_{ik} \cdot a_{kj}^{(k)}, \quad i = 1(1)n \text{ και } i \neq k, \quad j = k + 1(1)n + 1 \tag{3.2.18}$$

$$b_i^{(k+1)} = b_i^{(k)} + \mu_{ik} \cdot b_k^{(k)}, \quad i = 1(1)n \text{ και } i \neq k \tag{3.2.19}$$

Συνήθως θέτουμε $a_{i,n+1}^{(k)} = b_i^{(n+1)}$ για λόγους ομοιομορφίας. Ετσι ο αλγόριθμος της ακολουθιακής μορφής της μεθόδου απαλοιφής GJ είναι αυτός του σχήματος 3.1).

(* **Διαγωνοποίηση** *)

Για $k = 1(1)n$ επανάλαβε
  Εύρεση δείκτη $\ell$ τέτοιου ώστε:
  $|\alpha_{\ell k}| = max \ \{|\alpha_{kk}|, \ldots, |\alpha_{nk}|\}$
  $piv(k) := \ell$ {οδηγός γραμμή}
  $\alpha_{piv(k),k} \leftrightarrow \alpha_{kk}$
  για $j = k + 1(1)n + 1$ επανάλαβε
    $\alpha_{piv(k),j} \leftrightarrow \alpha_{kj}$
    $\alpha_{kj} := \alpha_{kj} \ / \ \alpha_{kk}$
    για $i = 1(1)n, \quad i \neq k$ επανάλαβε
      $\alpha_{ij} := \alpha_{ij} - \alpha_{ik} * \alpha_{kj}$

**Σχήμα 3.1** Αλγόριθμος της ακολουθιακής μορφής της μεθόδου απαλοιφής των Gauss-Jordan με μερική οδήγηση

## 3.3 ΠΑΡΑΛΛΗΛΕΣ ΜΟΡΦΕΣ ΤΗΣ ΜΕΘΟΔΟΥ ΑΠΑΛΟΙΦΗΣ ΤΩΝ GAUSS-JORDAN ΜΕ O(n) ΕΠΕΞΕΡΓΑΣΤΕΣ

Υποθέτουμε ότι ο αριθμός $p$ των επεξεργαστών που είναι διαθέσιμοι είναι μεταξύ 1 και $O(n)$.

Αν στον προηγούμενο αλγόριθμο (Σχ. 3.1) αλλάξουμε τη σειρά των δεικτών $(k, j, i)$ των βρόχων (do loops) προκύπτουν οι ακόλουθες πιθανές παράλληλες μορφές της μεθόδου.

1. Μορφή GJ KJI
2. Μορφή GJ KIJ

3. Μορφή GJ JKI

4. Μορφή GJ IKJ

5. Μορφή GJ IJK

6. Μορφή GJ JIK

Στις επόμενες παραγράφους μελετώνται οι παραπάνω παράλληλες μορφές της μεθόδου απαλοιφής των GJ. Πιο συγκεκριμένα κατασκευάζεται το γράφημα αλληλοεξάρτησης των εργασιών και επιχειρείται η εύρεση μιας βέλτιστης δρομολόγησης των επεξεργαστών.

### 3.3.1 Παραλληλία κατά στήλες

Στην παράγραφο αυτή θα εξετάσουμε τις παράλληλες μορφές κατά στήλες της μεθόδου GJ που είναι οι εξής:

1. Μορφή GJ KJI

2. Μορφή GJ JKI

#### Μορφή KJI-GJ

Στη μορφή αυτή ακολουθείται η εξής σειρά εργασιών: Στο $k$-βήμα της μεθόδου εκτελούνται οι εργασίες $T_{kj}$, $i = k+1(1)n+1$ (βλ. σχήμα 3.1.1) που περιλαμβάνουν τις διαιρέσεις των στοιχείων $a_{kj}^{k}$ δια του διαγωνίου στοιχείου $a_{kk}^{(k)}$ και τις τροποποιήσεις όλων των στοιχείων $a_{ij}^{(k)}$ του $n(n-k+1)$ δεξιού υποπίνακα του $A^{(k)}$ με εφαρμογή του τύπου:

$$a_{ij} = a_{ij} - a_{ik} * a_{kj}, \quad 1 \leq i \leq n, \quad i \neq k, \quad k+1 \leq j \leq n+1.$$

Πιο συγκεκριμένα στο κάθε $k$-βήμα η εργασία $T_{kj}$ περιλαμβάνει πράξεις που τροποποιούν τα στοιχεία $a_{ij}$ της $j$-στήλης του $A^{(k)}$ εκτός του στοιχείου $a_{kk}$ της κυρίας διαγωνίου.

Ετσι μετά από $n$ βήματα ολοκληρώνεται η διαγωνοποίηση του πίνακα $A$ και προκύπτει η λύση του γραμμικού συστήματος $x_i = a_{i,n+1}, \quad i = 1(1)n$.

Αν τώρα απαιτείται να χρησιμοποιήσουμε την τεχνική της μερικής οδήγησης, πρέπει στο $k$-βήμα της μεθόδου να προηγηθεί μια επιπρόσθετη εργασία $T_{kk}$ η οποία να παριλαμβάνει (i) την αναζήτηση του οδηγού στοιχείου $a_{\ell k}$ μεταξύ των $n-k+1$ στοιχείων $a_{ik}, \quad k \leq i \leq n$ και την επιλογή, σαν οδηγού στοιχείου, του μεγίστου κατ'απόλυτο τιμή και (ii) την εναλλαγή των στοιχείων $a_{\ell k}$ και $a_{kk}$ (βλ. σχήμα 3.1.2)

Επίσης πρέπει να συμπεριληφθεί στις εργασίες $T_{kj}$ η κατάλληλη εναλλαγή των στοιχείων της οδηγού γραμμής $\ell$ με την γραμμή $k$. Στα σχήματα 3.1.1 και 3.1.2 παρουσιάζονται οι αλγόριθμοι της μορφής KJI-GJ χωρίς οδήγηση και με μερική οδήγηση, αντίστοιχα.

**Χωρίς οδήγηση**

Για $k = 1(1)n$ επανάλαβε
Για $j = k + 1(1)n + 1$ επανάλαβε

$$\textcircled{T_{kj}} : \begin{cases} a_{kj} := a_{kj} \,/\, a_{kk} \\ \text{για } i = 1(1)n, \quad i \neq k \text{ επανάλαβε} \\ \qquad a_{ij} := a_{ij} - a_{ik} * a_{kj} \end{cases}$$

**Σχήμα 3.1.1** Ορισμός των εργασιών για την μορφή KJI-GJ χωρίς οδήγηση.

**Με μερική οδήγηση**

Για $k = 1(1)n$ επανάλαβε

$$\textcircled{T_{kk}} : \begin{cases} \textit{Εύρεση δείκτη } \ell \textit{ τέτοιου ώστε :} \\ |a_{\ell k}| := max\{|a_{kk}|, \ldots, |a_{nk}|\} \\ piv(k) := \ell \\ a_{piv(k),k} \leftrightarrow a_{kk} \end{cases}$$

Για $j = k + 1(1)n + 1$ επανάλαβε

$$\textcircled{T_{kj}} : \begin{cases} a_{piv(k),j} \leftrightarrow a_{kj} \\ a_{kj} := a_{kj} \,/\, a_{kk} \\ \text{για i=1(1)n,} \quad i \neq k \text{ επανέλαβε} \\ \qquad a_{ij} := a_{ij} - a_{ik} * a_{kj} \end{cases}$$

**Σχήμα 3.1.2** Ορισμός των εργασιών για την μορφή KJI-GJ με μερική οδήγηση

Στη συνέχεια παρουσιάζονται δύο εφαρμογές της μορφής KJI χωρίς οδήγηση και με μερική οδήγηση προκειμένου να προσδιοριστούν οι συνθήκες εξάρτησης των εργασιών όπως αυτές έχουν ορισθεί στα σχήματα 3.1.1 και 3.1.2 αντοίστοιχα. Αντικειμενικός σκοπός της εργασίας αυτής είναι η κατασκευή του μέγιστου παράλληλου γραφήματος.

**Εφαρμογή 1** Μορφή KJI-GJ χωρίς οδήγηση ($n = 4$)

$$A = \left[\begin{array}{cccc|c} a_{11} & a_{12} & a_{13} & a_{14} & a_{15} \\ a_{21} & a_{22} & a_{23} & a_{24} & a_{25} \\ a_{31} & a_{32} & a_{33} & a_{34} & a_{35} \\ a_{41} & a_{42} & a_{43} & a_{44} & a_{45} \end{array}\right]$$

**k = 1**

**j = 2**

$$(T_{12}): \begin{cases} & a_{12} = a_{12} \,/\, a_{11} \\ i=2 & a_{22} = a_{22} - a_{21} * a_{12} \\ i=3 & a_{32} = a_{32} - a_{31} * a_{12} \\ i=4 & a_{42} = a_{42} - a_{41} * a_{12} \end{cases}$$

**j = 3**

$$(T_{13}): \begin{cases} & a_{13} = a_{13} \,/\, a_{11} \\ i=2 & a_{23} = a_{23} - a_{21} * a_{13} \\ i=3 & a_{33} = a_{33} - a_{31} * a_{13} \\ i=4 & a_{43} = a_{43} - a_{41} * a_{13} \end{cases}$$

**j = 4**

$$(T_{14}): \begin{cases} & a_{14} = a_{14} \,/\, a_{11} \\ i=2 & a_{24} = a_{24} - a_{21} * a_{14} \\ i=3 & a_{34} = a_{34} - a_{31} * a_{14} \\ i=4 & a_{44} = a_{44} - a_{41} * a_{14} \end{cases}$$

**j = 5**

$$(T_{15}): \begin{cases} & a_{15} = a_{15} \,/\, a_{11} \\ i=2 & a_{25} = a_{25} - a_{21} * a_{15} \\ i=3 & a_{35} = a_{35} - a_{31} * a_{15} \\ i=4 & a_{45} = a_{45} - a_{41} * a_{15} \end{cases}$$

**k = 2**

**j = 3**

$$(T_{23}): \begin{cases} & a_{23} = a_{23} \,/\, a_{22} \\ i=1 & a_{13} = a_{13} - a_{12} * a_{23} \\ i=3 & a_{33} = a_{33} - a_{32} * a_{23} \\ i=4 & a_{43} = a_{43} - a_{42} * a_{23} \end{cases}$$

**j = 4**

$$(T_{24}): \begin{cases} & a_{24} = a_{24} \,/\, a_{22} \\ i=1 & a_{14} = a_{14} - a_{12} * a_{24} \\ i=3 & a_{34} = a_{34} - a_{32} * a_{24} \\ i=4 & a_{44} = a_{44} - a_{42} * a_{24} \end{cases}$$

**j = 5**

$$(T_{25}): \begin{cases} & a_{25} = a_{25} \,/\, a_{22} \\ i=1 & a_{15} = a_{15} - a_{12} * a_{25} \\ i=3 & a_{35} = a_{35} - a_{32} * a_{25} \\ i=4 & a_{45} = a_{45} - a_{42} * a_{25} \end{cases}$$

**k = 3**

**j = 4**

$$(T_{34}): \begin{cases} & a_{34} = a_{34} \,/\, a_{33} \\ i=1 & a_{14} = a_{14} - a_{13} * a_{34} \\ i=2 & a_{24} = a_{24} - a_{23} * a_{34} \\ i=4 & a_{44} = a_{44} - a_{43} * a_{34} \end{cases}$$

**j = 5**

$$(T_{35}): \begin{cases} & a_{35} = a_{35} \,/\, a_{33} \\ i=1 & a_{15} = a_{15} - a_{13} * a_{35} \\ i=2 & a_{25} = a_{25} - a_{23} * a_{35} \\ i=4 & a_{45} = a_{45} - a_{43} * a_{35} \end{cases}$$

**k = 4**

**j = 5**

$$(T_{45}): \begin{cases} & a_{45} = a_{45} \,/\, a_{44} \\ i=1 & a_{15} = a_{15} - a_{14} * a_{45} \\ i=2 & a_{25} = a_{25} - a_{24} * a_{45} \\ i=3 & a_{35} = a_{35} - a_{34} * a_{45} \end{cases}$$

**Εφαρμογή 2** Μορφή KJI-GJ με μερική οδήγηση ($n = 4$)

**k = 1**

$$(T_{11}): \begin{cases} |a_{\ell 1}| = max\{|a_{11}|, |a_{21}|, |a_{31}|, |a_{41}|\} \\ piv(1) = \ell \\ a_{piv(1),1} \leftrightarrow a_{11} \end{cases}$$

**j = 2**

$$(T_{12}): \begin{cases} a_{piv(1),2} \leftrightarrow a_{12} \\ a_{12} = a_{12} \,/\, a_{11} \\ \underline{i=2} \quad a_{22} = a_{22} - a_{21} * a_{12} \\ \underline{i=3} \quad a_{32} = a_{32} - a_{31} * a_{12} \\ \underline{i=4} \quad a_{42} = a_{42} - a_{41} * a_{12} \end{cases}$$

**j = 3**

$$(T_{13}): \begin{cases} a_{piv(1),3} \leftrightarrow a_{13} \\ a_{13} = a_{13} \,/\, a_{11} \\ \underline{i=2} \quad a_{23} = a_{23} - a_{21} * a_{13} \\ \underline{i=3} \quad a_{33} = a_{33} - a_{31} * a_{13} \\ \underline{i=4} \quad a_{43} = a_{43} - a_{41} * a_{13} \end{cases}$$

**j = 4**

$$(T_{14}): \begin{cases} a_{piv(1),4} \leftrightarrow a_{14} \\ a_{14} = a_{14} \,/\, a_{11} \\ \underline{i=2} \quad a_{24} = a_{24} - a_{21} * a_{14} \\ \underline{i=3} \quad a_{34} = a_{34} - a_{31} * a_{14} \\ \underline{i=4} \quad a_{44} = a_{44} - a_{41} * a_{14} \end{cases}$$

**j = 5**

$$(T_{15}): \begin{cases} a_{piv(1),5} \leftrightarrow a_{15} \\ a_{15} = a_{15} \,/\, a_{11} \\ \underline{i=2} \quad a_{25} = a_{25} - a_{21} * a_{15} \\ \underline{i=3} \quad a_{35} = a_{35} - a_{31} * a_{15} \\ \underline{i=4} \quad a_{45} = a_{45} - a_{41} * a_{15} \end{cases}$$

**k = 2**

$$(T_{22}): \begin{cases} |a_{\ell 2}| = max\{|a_{22}|, |a_{32}|, |a_{42}|\} \\ piv(2) = \ell \\ a_{piv(2),2} \leftrightarrow a_{22} \end{cases}$$

**j = 3**

$$(T_{23}): \begin{cases} a_{piv(2),3} \leftrightarrow a_{23} \\ a_{23} = a_{23} \,/\, a_{22} \\ \underline{i=1} \quad a_{13} = a_{13} - a_{12} * a_{23} \\ \underline{i=3} \quad a_{33} = a_{33} - a_{32} * a_{23} \\ \underline{i=4} \quad a_{43} = a_{43} - a_{42} * a_{23} \end{cases}$$

**j = 4**

$$(T_{24}): \begin{cases} a_{piv(2),4} \leftrightarrow a_{24} \\ a_{24} = a_{24} \,/\, a_{22} \\ \underline{i=1} \quad a_{14} = a_{14} - a_{12} * a_{24} \\ \underline{i=3} \quad a_{34} = a_{34} - a_{32} * a_{24} \\ \underline{i=4} \quad a_{44} = a_{44} - a_{42} * a_{24} \end{cases}$$

**j = 5**

$$(T_{25}): \begin{cases} a_{piv(2),5} \leftrightarrow a_{25} \\ a_{25} = a_{25} \,/\, a_{22} \\ \underline{i=1} \quad a_{15} = a_{15} - a_{12} * a_{25} \\ \underline{i=3} \quad a_{35} = a_{35} - a_{32} * a_{25} \\ \underline{i=4} \quad a_{45} = a_{45} - a_{42} * a_{25} \end{cases}$$

**k = 3**

$$(T_{33}) : \begin{cases} |a_{\ell 3}| = max\{|a_{33}|, |a_{43}|\} \\ piv(3) = \ell \\ a_{piv(3),3} \leftrightarrow a_{33} \end{cases}$$

**j = 4**

$$(T_{34}) : \begin{cases} a_{piv(3),4} \leftrightarrow a_{34} \\ a_{34} = a_{34} \,/\, a_{33} \\ \underline{i = 1} \quad a_{14} = a_{14} - a_{13} * a_{34} \\ \underline{i = 2} \quad a_{24} = a_{24} - a_{23} * a_{34} \\ \underline{i = 4} \quad a_{44} = a_{44} - a_{43} * a_{34} \end{cases}$$

**j = 5**

$$(T_{35}) : \begin{cases} a_{piv(3),5} \leftrightarrow a_{35} \\ a_{35} = a_{35} \,/\, a_{33} \\ \underline{i = 1} \quad a_{15} = a_{15} - a_{13} * a_{35} \\ \underline{i = 2} \quad a_{25} = a_{25} - a_{23} * a_{35} \\ \underline{i = 4} \quad a_{45} = a_{45} - a_{43} * a_{35} \end{cases}$$

**k = 4**

$$(T_{44}) : \begin{cases} |a_{\ell 4}| = max\{|a_{44}|\} \\ piv(4) = \ell \\ a_{piv(4),4} \leftrightarrow a_{44} \end{cases}$$

**j = 5**

$$(T_{45}) : \begin{cases} a_{piv(4),5} \leftrightarrow a_{45} \\ a_{45} = a_{45} \,/\, a_{44} \\ \underline{i = 1} \quad a_{15} = a_{15} - a_{14} * a_{45} \\ \underline{i = 2} \quad a_{25} = a_{25} - a_{24} * a_{45} \\ \underline{i = 3} \quad a_{35} = a_{35} - a_{34} * a_{45} \end{cases}$$

Οπως φαίνεται στα σχήματα 3.1.1 και 3.1.2 το σύνολο των εργασιών για τη διαγωνοποίηση του γραμμικού συστήματος είναι το:

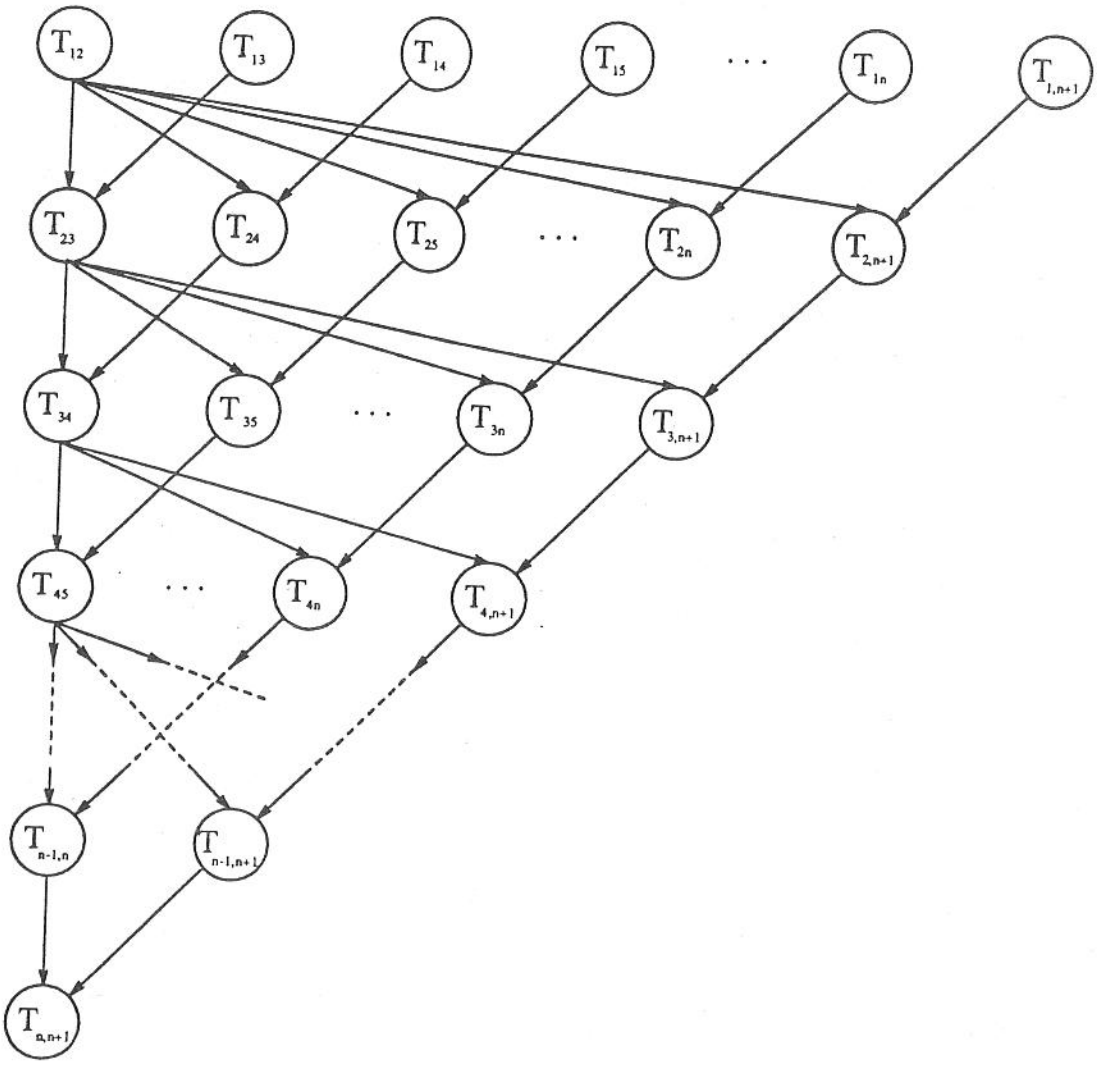
$$J = \{T_{kj} \,/\, 1 \leq k \leq n, \quad k \leq j \leq n\}$$

Οι περιορισμοί προτεραιότητας των εργασιών που επιβάλλονται από τον ακολουθιακό αλγόριθμο είναι:

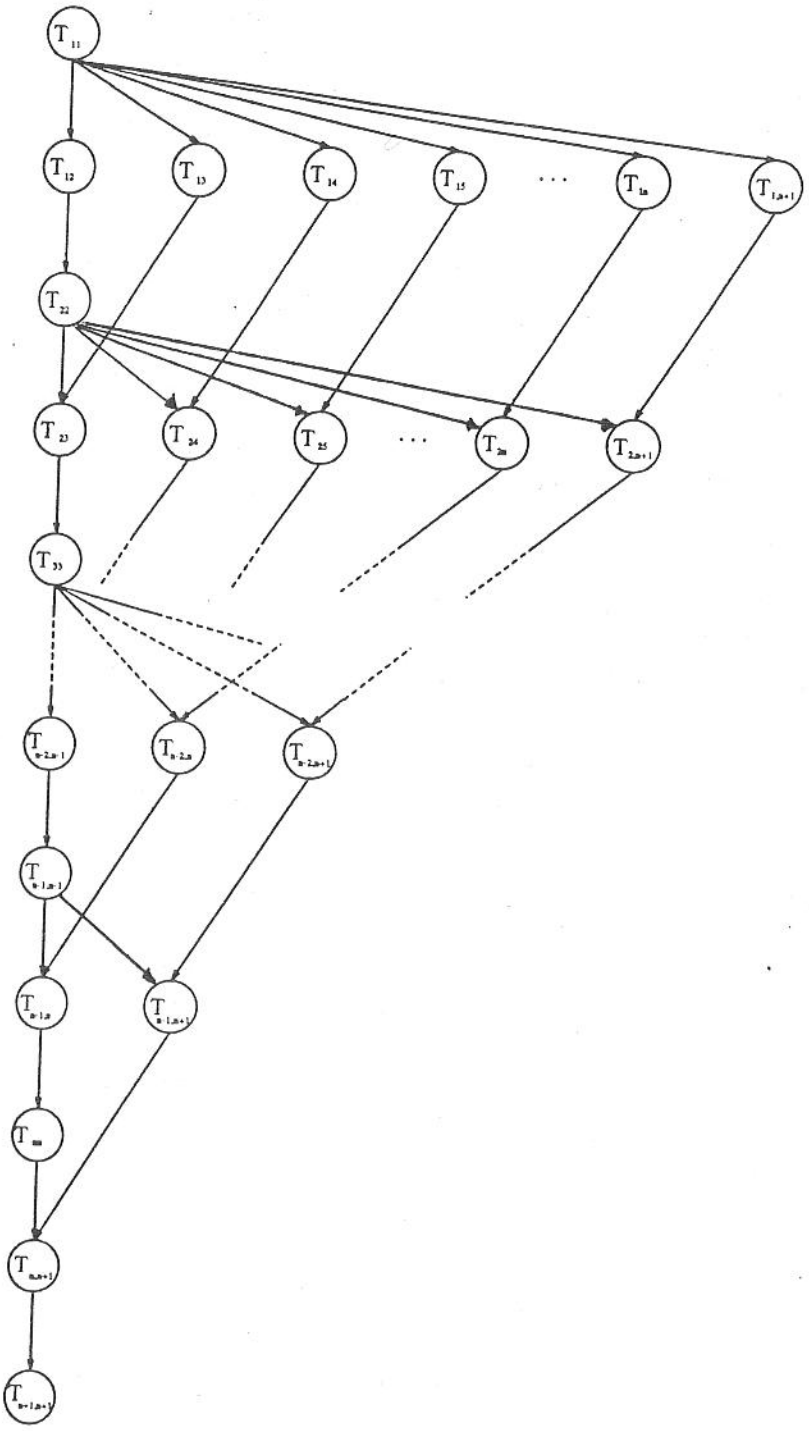
$$T_{k,k+1} \ll T_{k+1,j}, \quad k+2 \leq j \leq n+1, \quad 1 \leq k \leq n-1$$

$$T_{k,j} \ll T_{k+1,j}, \quad k+1 \leq j \leq n+1, \quad 1 \leq k \leq n-1$$

Το σύνολο J εφοδιασμένο με τους παραπάνω περιορισμούς αποτελεί ένα σύστημα εργασιών, έστω το $C = (J, \ll)$. Το αντίστοιχο μέγιστο παράλληλο γράφημα αλληλεξάρτησης των εργασιών του C φαίνεται στα σχήματα 3.1.3 και 3.1.4.




**Σχήμα 3.1.3** Γράφημα του μέγιστου παράλληλου συστήματος εργασιών C για τη διαγωνοποίηση της μορφής KJI-GJ χωρίς οδήγηση.



**Σχήμα 3.1.4** Γράφημα του μέγιστου παράλληλου συστήματος εργασιών C για τη διαγωνοποίηση της μορφής KJI-GJ με μερική οδήγηση.

Αν υποθέσουμε ότι κάθε αριθμητική πράξη ή μια σύγκριση αποτελεί μια χρονική μονάδα και ότι όλες οι αριθμητικές πράξεις απαιτούν για την εκτέλεση τους τον ίδιο αριθμό χρονικών μονάδων, τότε οι χρόνοι εκτέλεσης των εργασιών που ορίσαμε είναι:

$$W(T_{kk}) = n - k, \quad 1 \leq k \leq n-1$$

$$W(T_{kj}) = 2n - 1, \quad k+1 \leq j \leq n, \quad 1 \leq k \leq n-1$$

Ο συνολικός χρόνος εκτέλεσης των εργασιών του C είναι:

$$T_1 = \begin{cases} \sum_{k=1}^{n} \sum_{j=k+1}^{n+1} W(T_{kj}) & = \sum_{k=1}^{n}(n-k+1)(2n-1) & = n^3 + \frac{1}{2}n^2 - \frac{1}{2}n \\ & & \textit{χωρίς οδήγηση} \\ \sum_{k=1}^{n}\left[\sum_{j=k+1}^{n+1} W(T_{kj}) + W(T_{kk})\right] & = \sum_{k=1}^{n}[(n-k+1)(2n-1)+n-k] & = n^3 + n^2 - n \\ & & \textit{με μερική οδήγηση} \end{cases}$$

Επομένως και για τις δύο περιπτώσεις είναι $T_1 = n^3 + O(n^2)$.

Αν θεωρήσουμε τώρα τα γραφήματα (βλ. σχήμα 3.1.3 και 3.1.4) με βάρη των κόμβων τους χρόνους εκτέλεσής τους, τότε τα μακρύτερα μονοπάτια είναι για τις δύο περιπτώσεις, αντίστοιχα:

$$s_1 = (T_{12}, T_{23}, T_{34}, \ldots, T_{n-1,n}, T_{n,n+1})$$

*και*

$$s_1^{piv} = (T_{11}, T_{12}, T_{22}, T_{23}, \ldots, T_{n-1,n-1}, T_{n-1,n}, T_{nn}, T_{n,n+1})$$

Ο συνολικός χρόνος για την εκτέλεση των εργασιών που ανήκουν στα $s_1$ και $s_1^{piv}$ είναι:

$$T_p = L(s_1) = \sum_{k=1}^{n} W(T_{k,k+1}) = \sum_{k=1}^{n}(2n-1) = 2n^2 - n$$

*και*

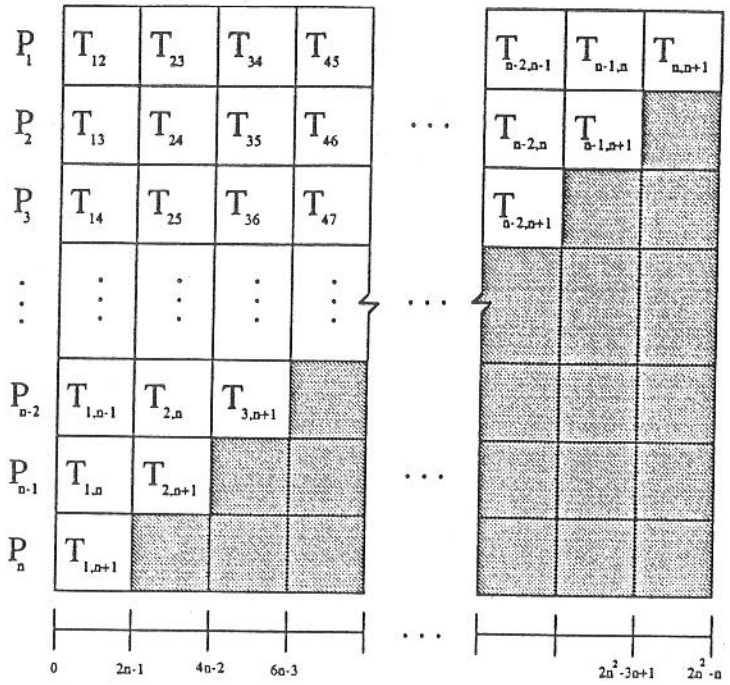
$$T_p^{piv} = L(s_1^{piv}) = \sum_{k=1}^{n}[W(T_{k,k}) + W(T_{k,k+1})] = \sum_{k=1}^{n}(n-k+2n-1) = \frac{5}{2}n^2 - \frac{3n}{2}$$

Οι χρόνοι αυτοί είναι οι ελάχιστοι χρόνοι για την εκτέλεση των εργασιών στα γραφήματα των σχ. 3.1.3 και 3.1.4, αντίστοιχα.

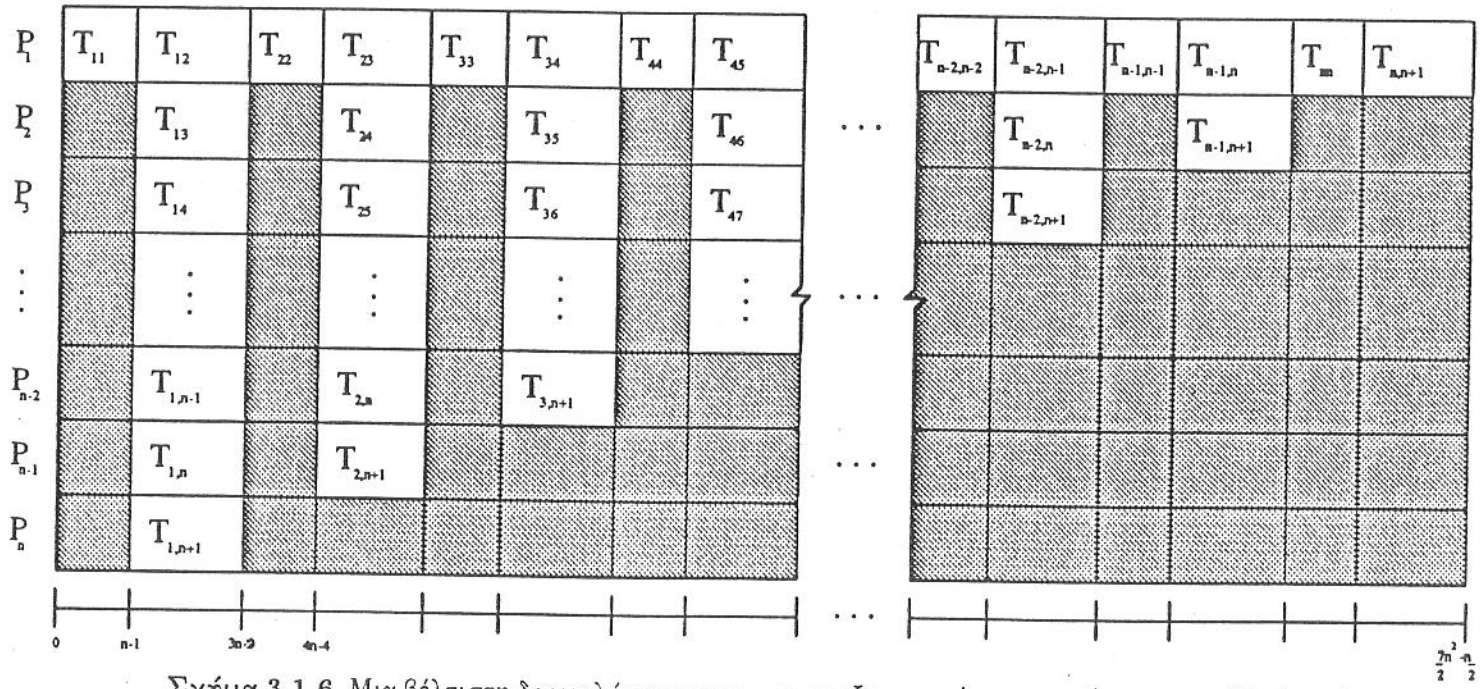
Ο μέγιστος αριθμός των εργασιών που μπορούν να εκτελεσθούν παράλληλα είναι n και οι εργασίες αυτές είναι οι:

$$T_{12}, T_{13}, T_{14}, \ldots, T_{1,n+1}$$

Αρα ο μέγιστος αριθμός των επεξεργαστών που απαιτείται ώστε το σύστημα εργασιών C να εκτελεσθεί σε χρόνο: $T_p = L(s_1)$ ή $T_p = L(s_1^{piv})$ είναι $p = n$.
Στη συνέχεια παρουσιάζεται μια βέλτιστη δρομολόγηση για τα μέγιστα παράλληλα συστήματα εργασιών C των σχ. 3.1.3 και 3.1.4 με $p = n$ επεξεργαστές (βλ. σχ. 3.1.5 και 3.1.6).




**Σχήμα 3.1.5** Μια βέλτιστη δρομολόγηση με $p = n$ επεξεργαστές για το μέγιστο παράλληλο σύστημα εργασιών C της μορφής KJI-GJ χωρίς οδήγηση.




**Σχήμα 3.1.6** Μια βέλτιστη δρομολόγηση με $p = n$ επεξεργαστές για το μέγιστο παράλληλο σύστημα εργασιών C της μορφής KJI-GJ με μερική οδήγηση.

Η επιτάχυνση $S_p$ και η αποδοτικότητα $E_p$ του αλγορίθμου χρησιμοποιώντας την παραπάνω δρομολόγηση είναι :

$$S_p = \frac{T_1}{T_p} = \begin{cases} \dfrac{n^3 + \frac{1}{2}n^2 + O(n)}{2n^2 - n} \simeq \dfrac{1}{2}n, & \textit{χωρίς οδήγηση} \\ \dfrac{n^3 + n^2 - n}{\frac{5}{2}n^2 - \frac{3n}{2}} \simeq \dfrac{2}{5}n, & \textit{με μερική οδήγηση} \end{cases}$$

$$E_p = \frac{S_p}{p} \simeq \begin{cases} \dfrac{\frac{1}{2}n}{n} = \dfrac{1}{2} = 0.5 & , \quad \textit{χωρίς οδήγηση} \\ \dfrac{\frac{2}{7}n}{n} = \dfrac{2}{7} \simeq 0.4 & , \quad \textit{με μερική οδήγηση} \end{cases}$$

Παρατηρούμε λοιπόν μια σημαντική μείωση της αποδοτικότητας του παράλληλου αλγορίθμου της μεθόδου GJ (μορφή KJI) όταν εφαρμόσουμε μερική οδήγηση σε σύγκριση με εκείνη της ίδιας μεθόδου χωρίς οδήγηση. Το φαινόμενο αυτό είναι αναμενόμενο καθόσο κατά τη διάρκεια εκτέλεσης των εργασιών $T_{kk}$ (μερική οδήγηση) οι υπόλοιποι $n-1$ επεξεργαστές παραμένουν αδρανείς με αποτέλεσμα την πτώση της απόδοσης του αλγορίθμου.

### Μορφή JKI-GJ

Στη μορφή αυτή ακολουθείται η εξής σειρά εργασιών: Στην αρχή εκτελείται η εργασία $T_{11}$ (βλ. σχήμα 3.1.7) που περιλαμβάνει τις διαιρέσεις όλων των στοιχείων της πρώτης στήλης του $A^{(1)}$ εκτός της κυρίας διαγωνίου δια του διαγωνίου στοιχείου $a_{11}^{(1)}$. Στη συνέχεια στο k βήμα της μεθόδου $(1 \leq k \leq n-1)$ εκτελούνται οι εργασίες $T_{kj}$ που περιλαμβάνουν τις τροποποιήσεις όλων των στοιχείων της j στήλης του $A^{(k)}$, $k+1 \leq j \leq n+1$ με εφαρμογή του τύπου:

$$a_{ij} = a_{ij} - a_{ik} * a_{kj}, \quad 1 \leq i \leq n, \quad i \neq k$$

Αμέσως μετά ακολουθεί η εργασία $T_{k+1,k+1}$ που περιλαμβάνει τις διαιρέσεις όλων των στοιχείων της j στήλης του $A^{(k)}$, εκτός του διαγωνίου δια του διαγωνίου στοιχείου $a_{k+1,k+1}^{(k)}$. Στο τέλος εκτελείται η εργασία $T_{n+1,n+1}$ που περιλαμβάνει τις διαιρέσεις των στοιχείων $a_{i,n+1}$, $1 \leq i \leq n$ δια των αντιστοίχων διαγωνίων στοιχείων $a_{ii}$. Έτσι η λύση του συστήματος είναι $x_i = a_{i,n+1}$, $i = 1(1)n$.

Αν τώρα απαιτείται να χρησιμοποιήσουμε την τεχνική της μερικής οδήγησης, αρκεί να συμπεριληφθεί στην εργασία $T_{jj}$ , $1 \leq j \leq n$ η αναζήτηση του οδηγού στοιχείου $a_{\ell j}$ μεταξύ των $n-j+1$ στοιχείων $a_{ij}$ , $i = j(1)n$, και η επιλογή, σαν οδηγού στοιχείου, του μεγίστου κατ'απόλυτο τιμή. Επίσης πρέπει να συμπεριληφθεί στην εργασία $T_{jj}$ η εναλλαγή των στοιχείων των γραμμών $j$ και $piv(j)$.

### Χωρίς οδήγηση

Για $j = 1(1)n$ επανάλαβε
  Για $k = 1(1)j-1$ επανάλαβε

$$(T_{kj}) : \begin{cases} \textit{Για i=1(1)n,} \quad i \neq k \textit{ επανάλαβε} \\ a_{ij} = a_{ij} - a_{ik} * a_{kj} \end{cases}$$

$$\textcircled{T_{jj}} : \begin{cases} \textit{Για}\ \text{i=1(1)n}, \quad i \neq j\ \textit{επανάλαβε} \\ \qquad a_{ij} = a_{ij} \,/\, a_{jj} \end{cases}$$

Για $k = 1(1)n$ επανάλαβε

$$\textcircled{T_{\substack{k,\\ n+1}}} : \begin{cases} \textit{για}\ \text{i=1(1)n},\ i \neq k\ \textit{επανάλαβε} \\ \qquad a_{i,n+1} := a_{i.n+1} - a_{ik} * a_{k,n+1} \end{cases}$$

$$\textcircled{T_{\substack{n+1,\\ n+1}}} : \begin{cases} \textit{για}\ \text{i=1(1)n}\ \textit{επανάλαβε} \\ \qquad a_{i,n+1} := a_{i,n+1} \,/\, a_{ii} \end{cases}$$

**Σχήμα 3.1.7** Ορισμός των εργασιών για τη μορφή JKI-GJ χωρίς οδήγηση.

**Με μερική οδήγηση**

Για $j = 1(1)n$ επανάλαβε

Για $k = 1(1)j-1$ επανάλαβε

$$\textcircled{T_{kj}} : \begin{cases} \textit{για}\ i = 1(1)n, \quad i \neq k\ \textit{επανάλαβε} \\ \qquad a_{ij} := a_{ij} - a_{ik} * a_{kj} \end{cases}$$

Εύρεση δείκτη $\ell$ τέτοιου ώστε:

$$\textcircled{T_{jj}} : \begin{cases} |a_{\ell j}| := max\{|a_{jj}|, |a_{j+1,j}|, \ldots, |a_{nj}|\} \\ piv(j) := \ell \\ a_{\ell j} \leftrightarrow a_{jj} \\ \textit{για}\ i = 1(1)n+1, \quad i \neq j\ \textit{επανάλαβε} \\ \qquad a_{piv(j),i} \leftrightarrow a_{ji} \\ \textit{για}\ i = 1(1)n, \quad i \neq j\ \textit{επανάλαβε} \\ \qquad a_{ij} := a_{ij} \,/\, a_{jj} \end{cases}$$

Για $k = 1(1)n$ επανάλαβε

$$\textcircled{T_{\substack{k,\\ n+1}}} : \begin{cases} \textit{για}\ i = 1(1)n, \quad i \neq k\ \textit{επανάλαβε} \\ \qquad a_{i,n+1} := a_{i.n+1} - a_{ik} * a_{k,n+1} \end{cases}$$

$$\textcircled{T_{\substack{n+1,\\ n+1}}} : \begin{cases} \textit{για}\ i = 1(1)n\ \textit{επανέλαβε} \\ \qquad a_{i,n+1} = a_{i,n+1} \,/\, a_{ii} \end{cases}$$

**Σχήμα 3.1.8** Ορισμός των εργασιών για τη μορφή JKI-GJ με μερική οδήγηση.

Στη συνέχεια παρουσιάζονται δύο εφαρμογές της μορφής JKI-GJ χωρίς οδήγηση και με μερική οδήγηση (βλ. σχήμα 3.1.8) προκειμένου να προσδιοριστούν οι συνθήκες εξάρτησης των εργασιών όπως αυτές έχουν ορισθεί στα σχήματα 3.1.7 και 3.1.8 αντίστοιχα. Στη συνέχεια κατασκευάζεται το μέγιστο παράλληλο γράφημα των εργασιών για κάθε περίπτωση.

**Εφαρμογή 1** Μορφή JKI-GJ χωρίς οδήγηση ($n = 4$)

$$A = \left[\begin{array}{cccc|c} a_{11} & a_{12} & a_{13} & a_{14} & a_{15} \\ a_{21} & a_{22} & a_{23} & a_{24} & a_{25} \\ a_{31} & a_{32} & a_{33} & a_{34} & a_{35} \\ a_{41} & a_{42} & a_{43} & a_{44} & a_{45} \end{array}\right]$$

**j = 1**

$$(T_{11}) : \begin{cases} \underline{i=2} & a_{21} = a_{21} \,/\, a_{11} \\ \underline{i=3} & a_{31} = a_{31} \,/\, a_{11} \\ \underline{i=4} & a_{41} = a_{41} \,/\, a_{11} \end{cases}$$

**j = 2**

k = 1

$$(T_{12}) : \begin{cases} \underline{i=2} & a_{22} = a_{22} - a_{21} * a_{12} \\ \underline{i=3} & a_{32} = a_{32} - a_{31} * a_{12} \\ \underline{i=4} & a_{42} = a_{42} - a_{41} * a_{12} \end{cases}$$

$$(T_{22}) : \begin{cases} \underline{i=1} & a_{12} = a_{12} \,/\, a_{22} \\ \underline{i=3} & a_{32} = a_{32} \,/\, a_{22} \\ \underline{i=4} & a_{42} = a_{42} \,/\, a_{22} \end{cases}$$

**j = 3**

k = 1

$$(T_{13}) : \begin{cases} \underline{i=2} & a_{23} = a_{23} - a_{21} * a_{13} \\ \underline{i=3} & a_{33} = a_{33} - a_{31} * a_{13} \\ \underline{i=4} & a_{43} = a_{43} - a_{41} * a_{13} \end{cases}$$

k = 2

$$(T_{23}) : \begin{cases} \underline{i=1} & a_{13} = a_{13} - a_{12} * a_{23} \\ \underline{i=3} & a_{33} = a_{33} - a_{32} * a_{23} \\ \underline{i=4} & a_{43} = a_{43} - a_{42} * a_{23} \end{cases}$$

$$(T_{33}) : \begin{cases} \underline{i=1} & a_{13} = a_{13} \,/\, a_{33} \\ \underline{i=2} & a_{23} = a_{23} \,/\, a_{33} \\ \underline{i=4} & a_{43} = a_{43} \,/\, a_{33} \end{cases}$$

**j = 4**

k = 1

$$(T_{14}) : \begin{cases} \underline{i=2} & a_{24} = a_{24} - a_{21} * a_{14} \\ \underline{i=3} & a_{34} = a_{34} - a_{31} * a_{14} \\ \underline{i=4} & a_{44} = a_{44} - a_{41} * a_{14} \end{cases}$$

k = 2

$$(T_{24}) : \begin{cases} \underline{i=1} & a_{14} = a_{14} - a_{12} * a_{24} \\ \underline{i=3} & a_{34} = a_{34} - a_{32} * a_{24} \\ \underline{i=4} & a_{44} = a_{44} - a_{42} * a_{24} \end{cases}$$

k = 3

$$(T_{34}) : \begin{cases} \underline{i=1} & a_{14} = a_{14} - a_{13} * a_{34} \\ \underline{i=2} & a_{24} = a_{24} - a_{33} * a_{34} \\ \underline{i=4} & a_{44} = a_{44} - a_{43} * a_{34} \end{cases}$$

$$(T_{44}) : \begin{cases} \underline{i=1} & a_{14} = a_{14} \,/\, a_{44} \\ \underline{i=2} & a_{24} = a_{24} \,/\, a_{44} \\ \underline{i=3} & a_{34} = a_{34} \,/\, a_{44} \end{cases}$$

$\underline{\mathbf{k = 1}}$

$$(T_{15}) : \begin{cases} \underline{i = 2} & a_{25} = a_{25} - a_{21} * a_{15} \\ \underline{i = 3} & a_{35} = a_{35} - a_{31} * a_{15} \\ \underline{i = 4} & a_{45} = a_{45} - a_{41} * a_{15} \end{cases}$$

$\underline{\mathbf{k = 2}}$

$$(T_{25}) : \begin{cases} \underline{i = 1} & a_{15} = a_{15} - a_{12} * a_{25} \\ \underline{i = 3} & a_{35} = a_{35} - a_{32} * a_{25} \\ \underline{i = 4} & a_{45} = a_{45} - a_{42} * a_{25} \end{cases}$$

$\underline{\mathbf{k = 3}}$

$$(T_{35}) : \begin{cases} \underline{i = 1} & a_{15} = a_{15} - a_{13} * a_{35} \\ \underline{i = 2} & a_{25} = a_{25} - a_{23} * a_{35} \\ \underline{i = 4} & a_{45} = a_{45} - a_{43} * a_{35} \end{cases}$$

$\underline{\mathbf{k = 4}}$

$$(T_{45}) : \begin{cases} \underline{i = 1} & a_{15} = a_{15} - a_{14} * a_{45} \\ \underline{i = 2} & a_{25} = a_{25} - a_{24} * a_{45} \\ \underline{i = 3} & a_{35} = a_{35} - a_{34} * a_{45} \end{cases}$$

$$(T_{55}) : \begin{cases} \underline{i = 1} & a_{15} = a_{15} \,/\, a_{11} \\ \underline{i = 2} & a_{25} = a_{25} \,/\, a_{22} \\ \underline{i = 3} & a_{35} = a_{35} \,/\, a_{33} \\ \underline{i = 4} & a_{45} = a_{45} \,/\, a_{44} \end{cases}$$

Οπως φαίνεται στα σχήματα 3.1.7 και 3.1.8 το σύνολο των εργασιών για την διαγωνοποίηση του γραμμικού συστήματος είναι το :

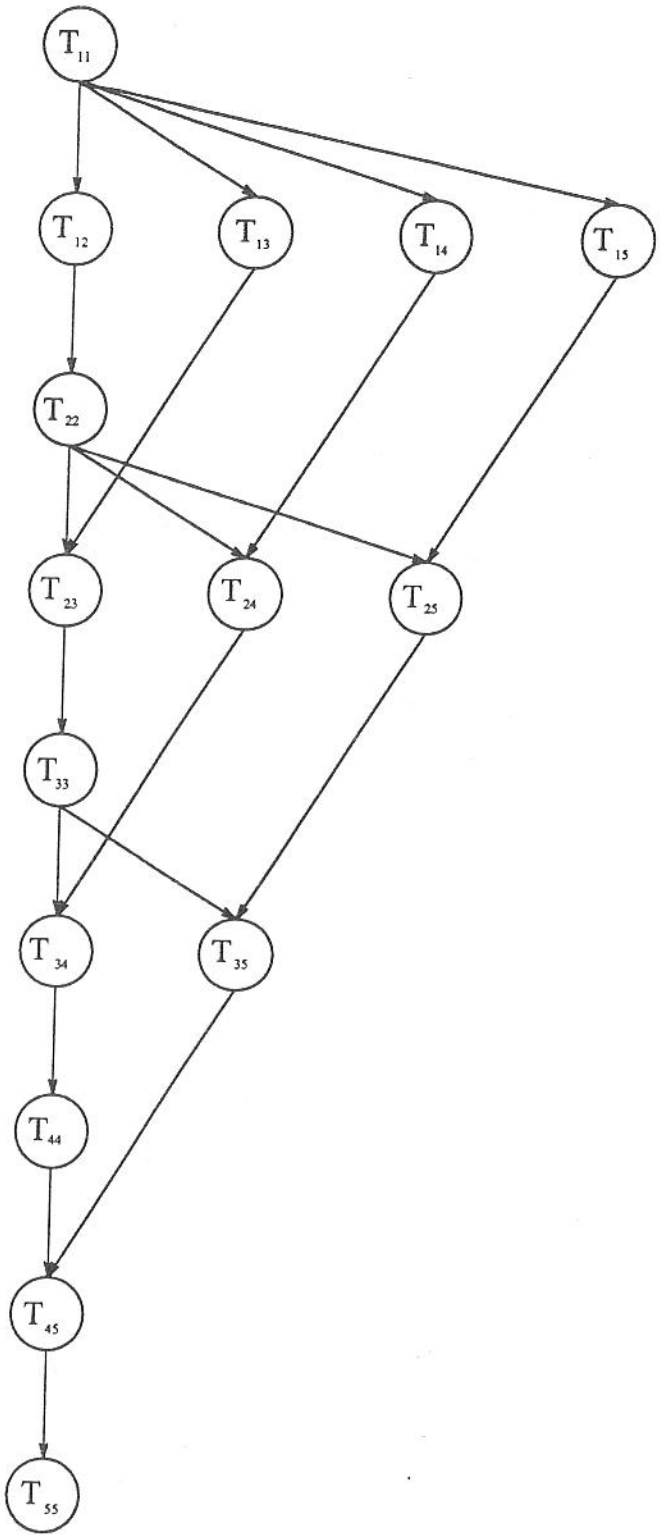
$$J = \{T_{kj} \,/\, 1 \leq k \leq n, \quad k \leq j \leq n+1\}$$

Οι περιορισμοί προτεραιότητας στην εκτέλεση των εργασιών που επιβάλλονται απο τον ακολουθιακό αλγόριθμο είναι:

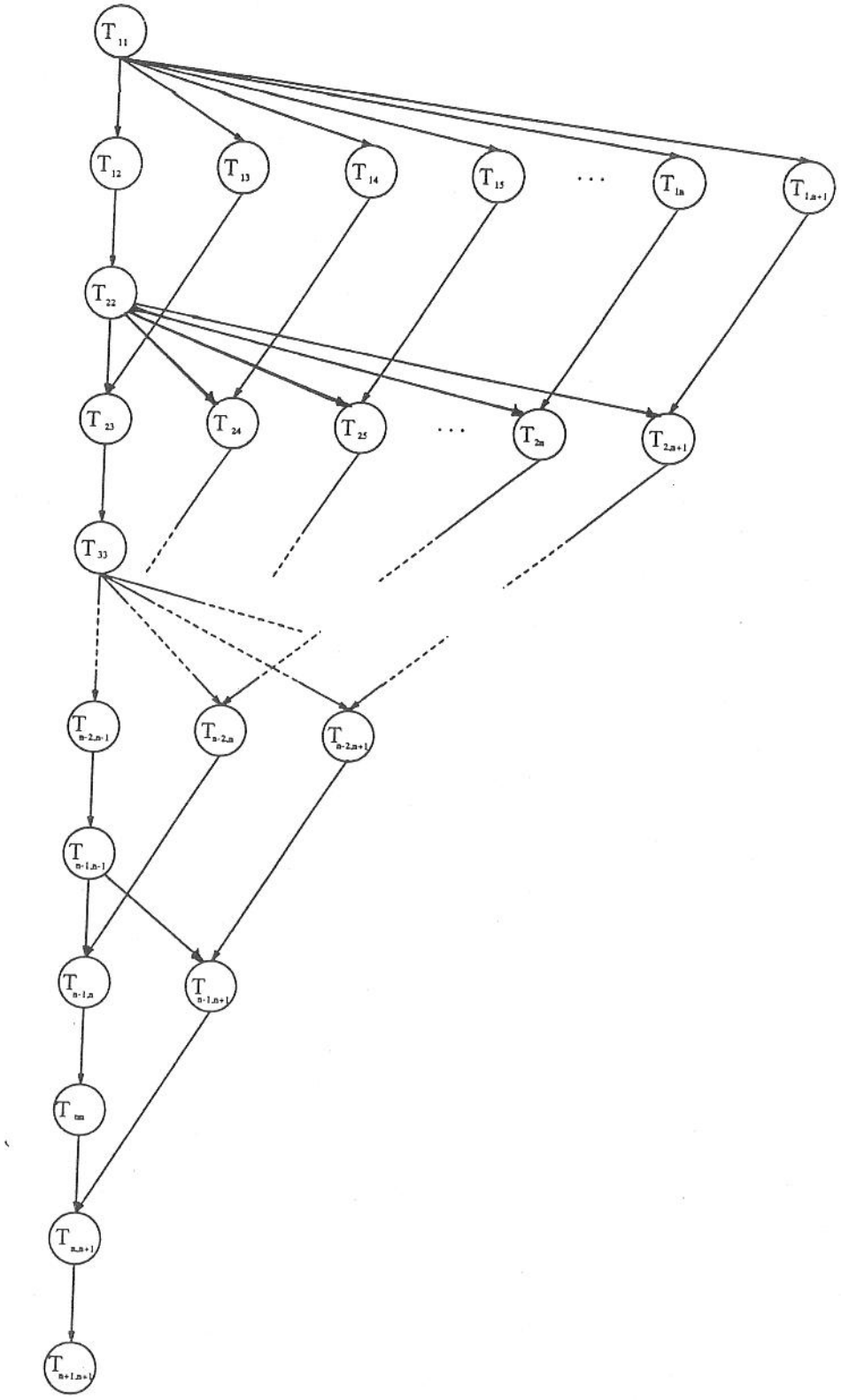
$$T_{kj} \ll T_{k+1,j}, \quad 1 \leq k \leq n, \quad k \leq j \leq n+1$$

$$T_{jj} \ll T_{jk}, \quad 1 \leq k \leq n, \quad k+1 \leq j \leq n+1$$

Το σύνολο J εφοδιασμένο με τους παραπάνω περιορισμούς αποτελεί ένα σύστημα εργασιών, έστω το $C = (J, \ll)$. Το αντίστοιχο μέγιστο παράλληλο γράφημα αλληλοεξάρτησης των εργασιών του C φαίνεται στα σχήματα 3.1.9 και 3.1.10.




**Σχήμα 3.1.9** Γράφημα του μέγιστου παράλληλου συστήματος εργασιών $C_1$ για τη διαγωνοποίηση της μορφής JKI-GJ χωρίς οδήγηση ($n = 4$).




**Σχήμα 3.1.10** Γράφημα του μέγιστου παράλληλου συστήματος εργασιών $C_1$ για τη διαγωνοποίηση της μορφής JKI-GJ.

Οι χρόνοι εκτέλεσης των εργασιών που ορίσαμε είναι:

$$W(T_{kj}) = 2(n-1)$$

$$W(T_{jj}) = \begin{cases} n-1 & \text{, χωρίς οδήγηση} \\ 2n-j-1 & \text{, με μερική οδήγηση} \end{cases} \qquad j = 1, 2, \ldots, n$$

$$W(T_{n+1,n+1}) = n$$

Αν θεωρήσουμε τώρα τό γράφημα (βλ. σχήμα 3.1.10) με βάρη των κόμβων τους χρόνους εκτέλεσής τους, τότε μακρύτερο μονοπάτι είναι το:

$$s_1 = (T_{11}, T_{12}, T_{22}, T_{23}, \ldots, T_{nn}, T_{n,n+1}, T_{n+1,n+1})$$

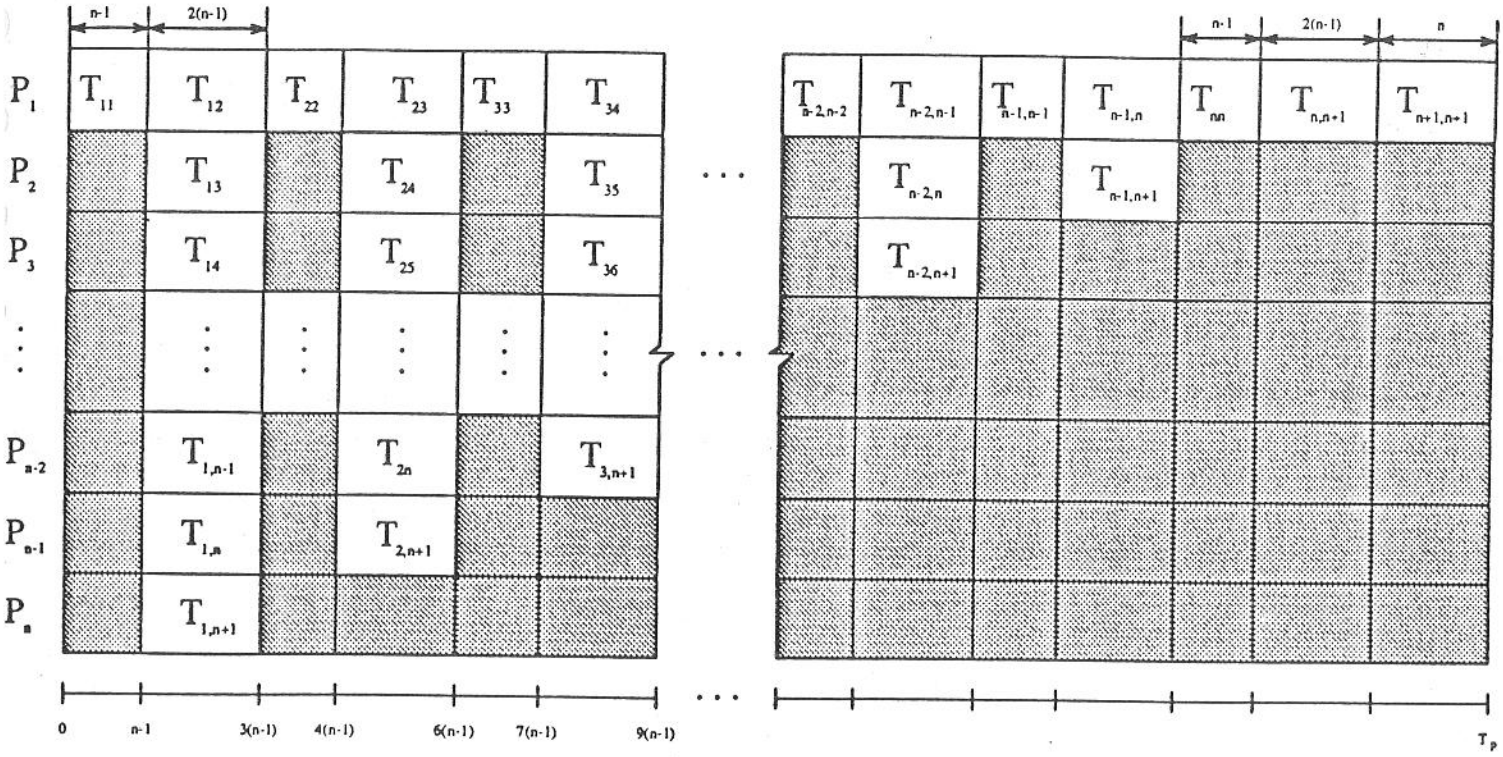
Ο χρόνος εκτέλεσης των εργασιών που ανήκουν στο $S_1$ είναι:

$$\begin{aligned} T_p &= L(s_1) = \sum_{j=1}^{n} [W(T_{jj}) + W(T_{j,j+1})] + W(T_{n+1,n+1}) \\ &= \begin{cases} \sum_{j=1}^{n} [n-1+2(n-1)] + n & = \frac{3}{2}n^2 - \frac{1}{2}n \quad \text{, χωρίς οδήγηση} \\ \sum_{j=1}^{n} [2n-j-1+2(n-1)] + n & = \frac{7}{2}n^2 - \frac{5}{2}n \quad \text{, με μερική οδήγηση} \end{cases} \end{aligned}$$

Ο χρόνος $T_p$ είναι ο ελάχιστος χρόνος για την εκτέλεση των εργασιών στο γράφημα του σχήματος 3.1.10. Επίσης ο μέγιστος αριθμός των εργασιών που μπορούν να εκτελεσθούν παράλληλα είναι n και αυτές οι εργασίες είναι οι:

$$T_{12}, T_{13}, T_{14}, \ldots, T_{1,n+1}$$

Αρα ο μέγιστος αριθμός των επεξεργαστών που απαιτείται ώστε το σύστημα εργασιών C να εκτελεσθεί σε χρόνο $T_p$ είναι $p = n$. Παρατηρώντας το παραπάνω γράφημα διαπιστώνουμε ότι έχει την ίδια μορφή με το γράφημα της μορφής KJI με μερική οδήγηση (βλ. σχήμα 3.1.4 και 3.1.10). Η μόνη σημαντική διαφορά είναι ότι το τελευταίο γράφημα περιέχει μία επιπλέον εργασία την $T_{n+1,n+1}$. Πρέπει να τονίσουμε όμως ότι οι εργασίες που αποτελούν τα δύο γραφήματα δεν είναι ίδιες και έχουν διαφορετικούς χρόνους εκτέλεσης. Στη συνέχεια παρουσιάζεται μια βέλτιστη δρομολόγηση για το μέγιστο παράλληλο σύστημα εργασιών C με $p = n$ επεξεργαστές (βλ. σχήμα 3.1.11). Η δρομολόγηση αυτή είναι παρόμοια εκείνης του σχήματος 3.1.6.




**Σχήμα 3.1.11** Μια βέλτιστη δρομολόγηση με $p = n$ επεξεργαστές για το μέγιστο παράλληλο σύστημα εργασιών C της μορφής JKI-GJ.

Οπως φαίνεται στο σχήμα 3.1.11 οι εργασίες που αποτελούν το μακρύτερο μονοπάτι $s_1$ στο προηγούμενο γράφημα δίνονται για εκτέλεση στον επεξεργαστή $P_1$, ενώ οι υπόλοιπες εργασίες δίνονται στους $n-1$ άλλους επεξεργαστές ($P_j$ , $j = 2(1)n$). Πιο συγκεκριμένα ο επεξεργαστής $P_j$ αναλαμβάνει την εκτέλεση των εργασιών:

$$T_{1,j+1}, T_{2,j+2}, \ldots, T_{n-j+1,n+1}$$

Επειδή αυτή η δρομολόγηση έχει μήκος $T_p = L(s_1)$ που είναι το μήκος του μακρύτερου μονοπατιού, είναι μια βέλτιστη δρομολόγηση με n επεξεργαστές. Η επιτάχυνση $S_p$ και απόδοση $E_p$ του αλγορίθμου χρησιμοποιώντας την παραπάνω δρομολόγηση είναι:

$$\left.\begin{aligned} S_p &= \frac{T_1}{T_p} = \frac{n^3 + O(n^2)}{\frac{3}{2}n^2 - \frac{1}{2}n} \simeq \frac{2}{3}n \\ E_p &= \lim_{n\to\infty} \frac{T_1}{pT_p} = \lim_{n\to\infty} \frac{n^3 + 0(n^2)}{n(\frac{3}{2}n^2 - \frac{1}{2}n)} = \frac{2}{3} = 0.666 \end{aligned}\right\} \text{ χωρίς οδήγηση}$$

$$\left.\begin{aligned} S_p^{piv} &= \frac{T_1}{T_p} = \frac{n^3 + 0(n^2)}{\frac{7}{2}n^2 - \frac{5}{2}n} \simeq \frac{2}{7}n \\ E_p^{piv} &= \lim_{n\to\infty} \frac{T_1}{pT_p^{piv}} = \lim_{n\to\infty} \frac{n^3 + 0(n^2)}{n(\frac{7}{2}n^2 - \frac{5}{2}n)} = \frac{2}{7} = 0.286 \end{aligned}\right\} \text{με μερική οδήγηση}$$

Παρατηρούμε λοιπόν μια σημαντική μείωση της αποδοτικότητας του παράλληλου αλγόριθμου της μορφής JKI-GJ όταν εφαρμόσουμε μερική οδήγηση σε σύγκριση με την αποδοτικότητα της ίδιας μεθόδου χωρίς οδήγηση. Η αιτία του φαινομένου αυτού έχει ήδη εξηγηθεί στην προηγούμενη μορφή.

### 3.3.2 Παραλληλία κατά γραμμές

Στην παράγραφο αυτή θα εξετάσουμε τις πιθανές παράλληλες μορφές **κατά γραμμές** της μεθόδου GJ που είναι οι εξής:

1. Μορφή KIJ
2. Μορφή IKJ

#### Μορφή KIJ-GJ

Στη μορφή αυτή ακολουθείται η εξής σειρά εργασιών: Στο $k$ βήμα της μεθόδου εκτελείται πρώτα η εργασία $T_{kk}$ που περιλαμβάνει τις διαιρέσεις των στοιχείων $a_{kj}^{(k)}$, $k+1 \leq j \leq n+1$ της $k$ γραμμής δια του διαγωνίου στοιχείου $a_{kk}^{(k)}$ (βλ. σχήμα 3.2.1). Στη συνέχεια εκτελούνται παράλληλα οι ανεξάρτητες εργασίες $T_{ki}$, $1 \leq i \leq n$ και $i \neq k$, όπου η καθεμιά περιλαμβάνει τις τροποποιήσεις όλων των στοιχείων $a_{ij}^{(k)}$ της $i$ γραμμής $A^{(k)}$ με εφαρμογή του τύπου:

$$a_{ij} = a_{ij} - a_{ik} * a_{kj}, \quad k+1 \leq j \leq n+1$$

Αν τώρα απαιτείται να χρησιμοποιήσουμε την τεχνική της μερικής οδήγησης, αρκεί να συμπεριληφθεί στην εργασία $T_{kk}$, $1 \leq k \leq n$ η αναζήτηση του οδηγού στοιχείου $a_{\ell k}$ μεταξύ των n-k+1 στοιχείων $a_{ik}$, $i = k(1)n$ και η επιλογή, σαν οδηγού στοιχείου, του μεγίστου κατ'απόλυτο τιμή (βλ. σχήμα 3.2.2). Επίσης πρέπει να συμπεριληφθεί στην εργασία $T_{kk}$ και η εναλλαγή των στοιχείων των γραμμών k και piv(k).

Στη συνέχεια δίνονται στα σχήματα 3.2.1 και 3.2.2 οι αλγόριθμοι της μορφής KIJ-GJ χωρίς οδήγηση και με μερική οδήγηση, αντίστοιχα.

**Χωρίς οδήγηση**

*Για* k=1(1)n *επανάλαβε*

$$\textcircled{T_{kk}} : \begin{cases} \textit{Για } j = k+1(1)n+1, \quad i \neq k \textit{ επανάλαβε} \\ \quad a_{kj} := a_{kj} \,/\, a_{kk} \end{cases}$$

*Για* i=1(1)n, $i \neq k$ *επανάλαβε*

$$\textcircled{T_{ki}} : \begin{cases} \textit{Για } j = k+1(1)n+1 \textit{ επανάλαβε} \\ \quad a_{ij} := a_{ij} - a_{ik} * a_{kj} \end{cases}$$

**Σχήμα 3.2.1** Ορισμός των εργασιών για την μορφή KIJ-GJ χωρίς οδήγηση.

**Με μερική οδήγηση**

*Για* $k = 1(1)n$ *επανάλαβε*

$$\textcircled{T_{kk}} : \begin{cases} \textit{Εύρεση δείκτη } \ell \textit{ τέτοιου ώστε :} \\ |a_{\ell k}| := max\{|a_{kk}|, \ldots, |a_{nk}|\} \\ piv(k) := \ell \\ a_{piv(k),k} \leftrightarrow a_{kk} \\ \textit{Για } j = k+1(1)n+1 \textit{ επανάλαβε} \\ \quad apiv(k), j \leftrightarrow a_{kj} \\ \quad a_{kj} := a_{kj} \,/\, a_{kk} \end{cases}$$

$$\textcircled{T_{kj}} : \begin{cases} \textit{Για } i = 1(1)n, \; i \neq k \textit{ επανάλαβε} \\ \quad \textit{για } j = k+1(1)n+1 \textit{ επανάλαβε} \\ \quad\quad a_{ij} := a_{ij} - a_{ik} * a_{kj} \end{cases}$$

**Σχήμα 3.2.2** Ορισμός των εργασιών για την μορφή KIJ-GJ με μερική οδήγηση.

Στη συνέχεια παρουσιάζονται δύο εφαρμογές της μορφής KIJ-GJ, χωρίς οδήγηση (βλ. σχήμα 3.2.1) και με μερική οδήγηση (βλ. σχήμα 3.2.2), προκειμένου να προσδιοριστούν οι συνθήκες εξάρτησης των εργασιών όπως αυτές έχουν ορισθεί στα σχήματα 3.2.1 και 3.2.2, αντίστοιχα. Αντικειμενικός σκοπός της εργασίας αυτής είναι η κατασκευή του μέγιστου παράλληλου γραφήματος.

**Εφαρμογή 1** Μορφή KIJ-GJ χωρίς οδήγηση ($n = 4$)

$$A = \left[\begin{array}{cccc|c} a_{11} & a_{12} & a_{13} & a_{14} & a_{15} \\ a_{21} & a_{22} & a_{23} & a_{24} & a_{25} \\ a_{31} & a_{32} & a_{33} & a_{34} & a_{35} \\ a_{41} & a_{42} & a_{43} & a_{44} & a_{45} \end{array}\right]$$

**k = 1**

$$(T_{11}) : \begin{cases} \underline{j=2} & a_{12} = a_{12} \,/\, a_{11} \\ \underline{j=3} & a_{13} = a_{13} \,/\, a_{11} \\ \underline{j=4} & a_{14} = a_{14} \,/\, a_{11} \\ \underline{j=5} & a_{15} = a_{15} \,/\, a_{11} \end{cases}$$

**i = 2**

$$(T_{12}) : \begin{cases} \underline{j=2} & a_{22} = a_{22} - a_{21} * a_{12} \\ \underline{j=3} & a_{23} = a_{23} - a_{21} * a_{13} \\ \underline{j=4} & a_{24} = a_{24} - a_{21} * a_{14} \\ \underline{j=5} & a_{25} = a_{25} - a_{21} * a_{15} \end{cases}$$

**i = 3**

$$(T_{13}) : \begin{cases} \underline{j=2} & a_{32} = a_{32} - a_{31} * a_{12} \\ \underline{j=3} & a_{33} = a_{33} - a_{31} * a_{13} \\ \underline{j=4} & a_{34} = a_{34} - a_{31} * a_{14} \\ \underline{j=5} & a_{35} = a_{35} - a_{31} * a_{15} \end{cases}$$

**i = 4**

$$(T_{14}) : \begin{cases} \underline{j=2} & a_{42} = a_{42} - a_{41} * a_{12} \\ \underline{j=3} & a_{43} = a_{43} - a_{41} * a_{13} \\ \underline{j=4} & a_{44} = a_{44} - a_{41} * a_{14} \\ \underline{j=5} & a_{45} = a_{45} - a_{41} * a_{15} \end{cases}$$

**k = 2**

$$(T_{22}) : \begin{cases} \underline{j=3} & a_{23} = a_{23} \,/\, a_{22} \\ \underline{j=4} & a_{24} = a_{24} \,/\, a_{22} \\ \underline{j=5} & a_{25} = a_{25} \,/\, a_{22} \end{cases}$$

**i = 1**

$$(T_{21}) : \begin{cases} \underline{j=3} & a_{13} = a_{13} - a_{12} * a_{23} \\ \underline{j=4} & a_{14} = a_{14} - a_{12} * a_{24} \\ \underline{j=5} & a_{15} = a_{15} - a_{12} * a_{25} \end{cases}$$

**i = 3**

$$(T_{23}) : \begin{cases} \underline{j=3} & a_{33} = a_{33} - a_{32} * a_{23} \\ \underline{j=4} & a_{34} = a_{34} - a_{32} * a_{24} \\ \underline{j=5} & a_{35} = a_{35} - a_{32} * a_{25} \end{cases}$$

**i = 4**

$$(T_{24}) : \begin{cases} \underline{j=3} & a_{43} = a_{43} - a_{42} * a_{23} \\ \underline{j=4} & a_{44} = a_{44} - a_{42} * a_{24} \\ \underline{j=5} & a_{45} = a_{45} - a_{42} * a_{25} \end{cases}$$

k = 3

$(T_{33})$ : $\begin{cases} \underline{j=4} & a_{34} = a_{34} \,/\, a_{33} \\ \underline{j=5} & a_{35} = a_{35} \,/\, a_{33} \end{cases}$

i = 1

$(T_{31})$ : $\begin{cases} \underline{j=4} & a_{14} = a_{14} - a_{13} * a_{34} \\ \underline{j=5} & a_{15} = a_{15} - a_{13} * a_{35} \end{cases}$

i = 2

$(T_{32})$ : $\begin{cases} \underline{j=4} & a_{24} = a_{24} - a_{23} * a_{34} \\ \underline{j=5} & a_{25} = a_{25} - a_{23} * a_{35} \end{cases}$

i = 4

$(T_{34})$ : $\begin{cases} \underline{j=4} & a_{44} = a_{44} - a_{43} * a_{34} \\ \underline{j=5} & a_{45} = a_{45} - a_{43} * a_{35} \end{cases}$

k = 4

$(T_{44})$ : $\begin{cases} \underline{j=5} & a_{45} = a_{45} \,/\, a_{44} \end{cases}$

i = 1

$(T_{41})$ : $\begin{cases} \underline{j=5} & a_{15} = a_{15} - a_{14} * a_{45} \end{cases}$

i = 2

$(T_{42})$ : $\begin{cases} \underline{j=5} & a_{25} = a_{25} - a_{24} * a_{45} \end{cases}$

i = 3

$(T_{43})$ : $\begin{cases} \underline{j=5} & a_{35} = a_{35} - a_{34} * a_{45} \end{cases}$

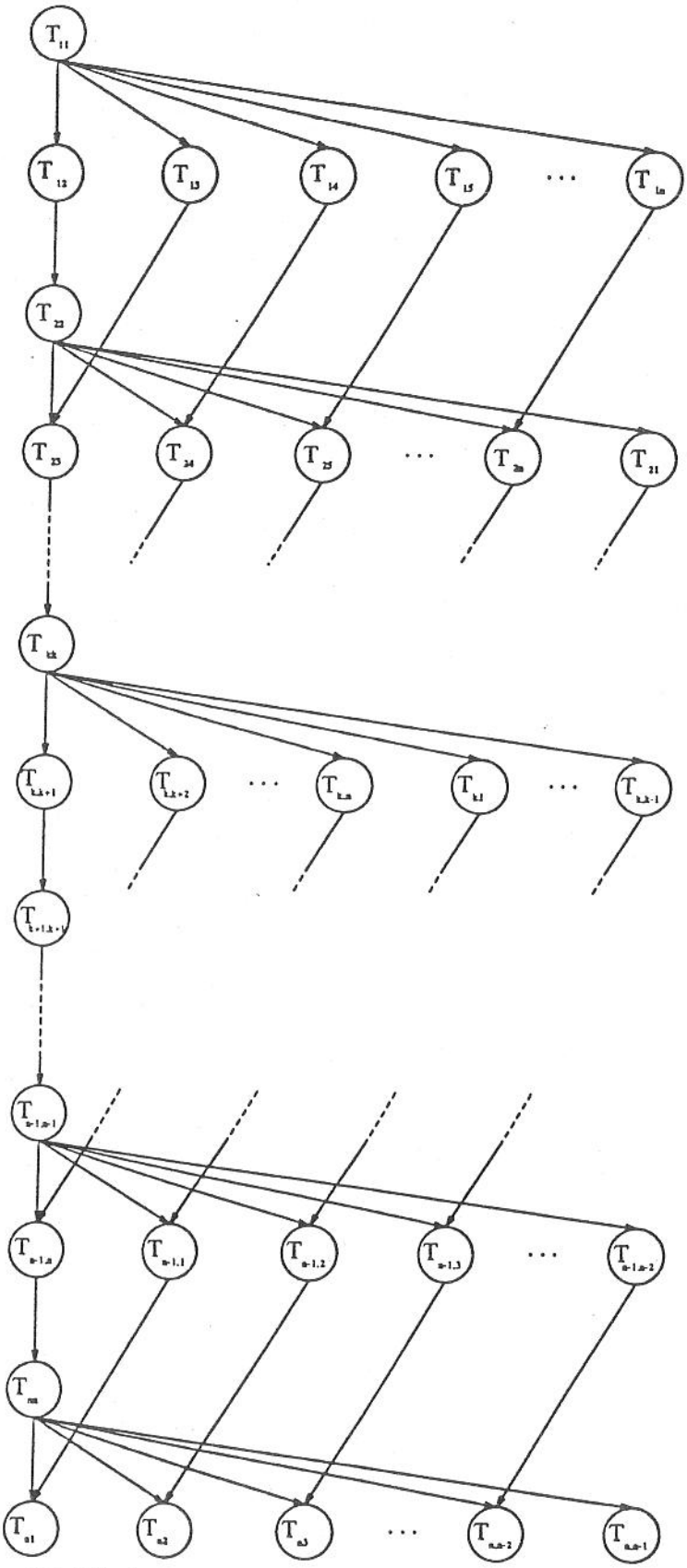
Οπως φαίνεται στα σχήματα 3.2.1 και 3.2.2 το σύνολο των εργασιών για τη διαγωνοποίηση του γραμμικού συστήματος είναι το:

$$J = \{T_{ki} \,/\, 1 \leq k \leq n, \quad k \leq i \leq n\}$$

Οι περιορισμοί προτεραιότητας στη σειρά εκτέλεσης των εργασιών που επιβάλλονται από τον ακολουθιακό αλγόριθμο είναι:

$$T_{k,k} \ll T_{k,i}, \quad 1 \leq i \leq n, \quad 1 \leq k \leq n$$
$$T_{k,i} \ll T_{k+1,i}, \quad 1 \leq i \leq n, \quad 1 \leq k \leq n-1$$

Το σύνολο J εφοδιασμένο με τους παραπάνω περιορισμούς αποτελεί ένα σύστημα εργασιών, έστω το $C = (J, \ll)$. Το αντίστοιχο μέγιστο παράλληλο γράφημα αλληλεξάρτησης των εργασιών του C φαίνεται στο σχήμα 3.2.3.



**Σχήμα 3.2.3** Γράφημα του μέγιστου παράλληλου συστήματος εργασιών C για τη διαγωνοποίηση της μορφής KIJ-GJ χωρίς οδήγηση.

Αν υποθέσουμε ότι κάθε αριθμητική πράξη ή μια σύγκριση αποτελεί μια χρονική μονάδα και ότι όλες οι αριθμητικές πράξεις απαιτούν για την εκτέλεση τους τον ίδιο αριθμό χρονικών μονάδων, τότε οι χρόνοι εκτέλεσης των εργασιών που ορίσαμε είναι:

$$W(T_{kk}) = \begin{cases} n-k+1 & \text{, χωρίς οδήγηση} \\ 2(n-k)+1 & \text{, με μερική οδήγηση} \end{cases}$$

$$W(T_{ki}) = 2(n-k+1)$$

Αν θεωρήσουμε τώρα τό γράφημα (βλ. σχήμα 3.2.3) με βάρη των κόμβων τους χρόνους εκτέλεσης τους, τότε μακρύτερο μονοπάτι είναι το:

$$s_1 = (T_{11}, T_{12}, T_{22}, T_{23}, \ldots, T_{n-1,n-1}, T_{n-1,n}, T_{n,n}, T_{n,1})$$

Ο χρόνος εκτέλεσης των εργασιών που ανήκουν στο $s_1$ είναι:

$$T_p = L(s_1) = \sum_{k=1}^{n-1} [W(T_{kk}) + W(T_{k,k+1})] + W(T_{nn}) + W(T_{n1})$$

$$= \begin{cases} \sum_{k=1}^{n-1} [3(n-k+1)] + 3, & \text{χωρίς οδήγηση} \\ \sum_{k=1}^{n-1} [4(n-k)+3] + 3, & \text{με μερική οδήγηση} \end{cases}$$

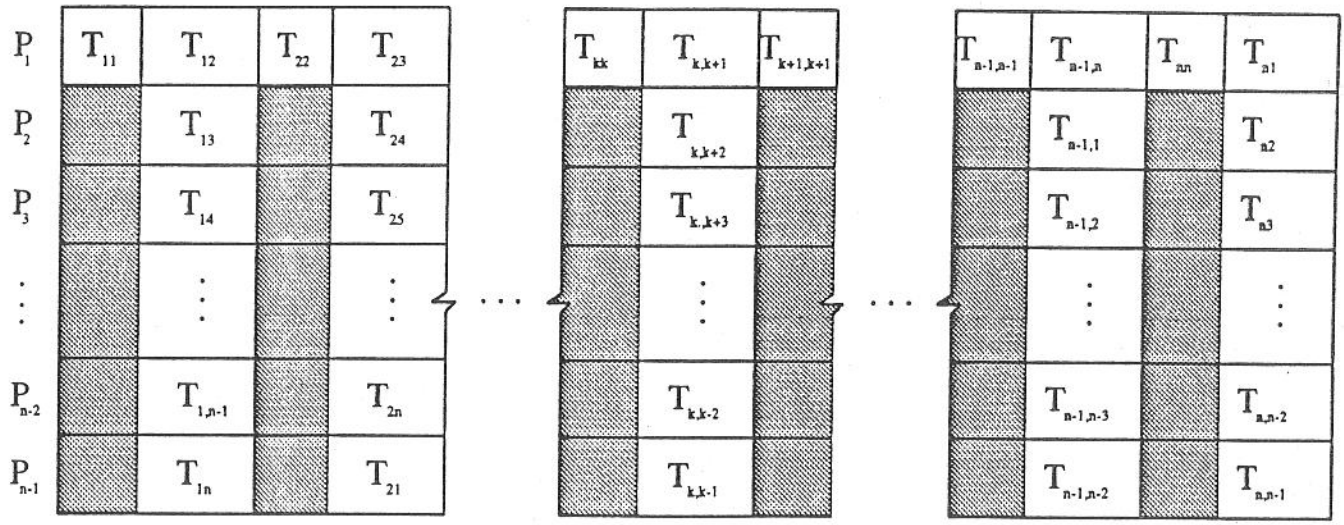
Αρα είναι:

$$T_p = \begin{cases} \frac{3}{2}n^2 + \frac{3}{2}n, & \text{χωρίς οδήγηση} \\ 2n^2 + n, & \text{με μερική οδήγηση} \end{cases}$$

Ο χρόνος αυτός είναι ο ελάχιστος χρόνος για την εκτέλεση των εργασιών στο γράφημα του σχήματος 3.2.3, για κάθε περίπτωση, αντίστοιχα. Επίσης ο μέγιστος αριθμός των εργασιών του μπορούν να εκτελεσθούν παράλληλα είναι $n-1$ και είναι οι εργασίες:

$$T_{k,k+1}, T_{k,k+2}, \ldots, T_{kn}, T_{k1}, \ldots, T_{k,k-1}$$

Αρα ο μέγιστος αριθμός των επεξεργαστών που απαιτείται ώστε το σύστημα εργασιών C να εκτελεσθεί σε χρόνο $T_p$ είναι $p = n-1$. Στη συνέχεια κατασκευάζουμε μια βέλτιστη δρομολόγηση για το μέγιστο παράλληλο σύστημα εργασιών C του σχήματος 3.2.3 με $p = n-1$ επεξεργαστές (βλ. σχήμα 3.2.4).




**Σχήμα 3.2.4** Μια βέλτιστη δρομολόγηση με $p = n-1$ επεξεργαστές για τη μορφή KIJ-GJ χωρίς οδήγηση και με μερική οδήγηση.

Οπως φαίνεται στο σχήμα 3.2.4 οι εργασίες που αποτελούν το μακρύτερο μονοπάτι στο προηγούμενο γράφημα (σχ. 3.2.3) δίνονται για εκτέλεση στον επεξεργαστή $P_1$, ενώ οι υπόλοιπες δίνονται στους $n-2$ άλλους επεξεργαστές $P_j$, $j = 2(1)n-1$ έτσι ώστε ο επεξεργαστής $P_j$ αναλαμβάνει να εκτελέσει τις εργασίες:

$$T_{1,j+1}, T_{2,j+2}, \ldots, T_{n-j,n}, T_{n-1,j-1}, T_{n,j}$$

Επειδή αυτή η δρομολόγηση έχει μήκος $L(s_1)$, που είναι το μήκος του μακρύτερου μονοπατιού, είναι μία βέλτιστη δρομολόγηση με $n-1$ επεξεργαστές. Η επιτάχυνση $S_p$ του αλγορίθμου χρησιμοποιώντας την παραπάνω δρομολόγηση είναι:

$$S_p = \frac{T_1}{T_p} = \begin{cases} \dfrac{n^3 + O(n^2)}{\frac{3}{2}n^2 + \frac{3}{2}n} \simeq \frac{2}{3}n, & \textit{χωρίς οδήγηση} \\ \dfrac{n^3 + 0(n^2)}{2n^2 + n} \simeq \frac{1}{2}n, & \textit{με μερική οδήγηση} \end{cases}$$

και η αποδοτικότητά του είναι:

$$E_p = \lim_{n\to\infty} \frac{T_1}{pT_p} = \begin{cases} \lim\limits_{n\to\infty} \dfrac{n^3 + 0(n^2)}{(n-1)(\frac{3}{2}n^2 + \frac{3}{2}n)} = \frac{2}{3} \simeq 0.666, & \textit{χωρίς οδήγηση} \\ \lim\limits_{n\to\infty} \dfrac{n^3 + 0(n^2)}{(n-1)(2n^2 + n)} = \frac{1}{2} \simeq 0.5, & \textit{με μερική οδήγηση} \end{cases}$$

Παρατηρούμε ότι η μείωση της αποδοτικότητας του παράλληλου αλγορίθμου της μεθόδου KIJ-GJ όταν εφαρμόσουμε μερική οδήγηση είναι λιγότερο σημαντική σε σύγκριση με εκείνη που παρατηρήθηκε στην μορφή KJI. Αυτό οφείλεται στο γεγονός ότι η προσθήκη της μερικής οδήγησης δεν αλλάζει τη μορφή του γραφήματος. Απλά αυξάνεται ο χρόνος των εργασιών $T_{kk}$, $k = 1, 2, \ldots, n$. Ας σημειωθεί ότι το γράφημα της μορφής αυτής δεν είναι "τριγωνικό" αλλά "ορθογώνιο". Ετσι όλοι

οι επεξεργαστές είναι απασχολημένοι συνεχώς μέχρι το τέλος με εξαίρεση τα χρονικά διαστήματα που εκτελούνται οι εργασίες $T_{kk}$. Επομένως, η αύξηση της αποδοτικότητας του αλγορίθμου ήταν αναμενόμενη.

### 3.3.3 Παραλληλία κατά πίνακες

Στην παράγραφο αυτή θα εξετάσουμε τις παράλληλες μορφές μιας μεθόδου διαγωνοποίησης του πίνακα A που προκύπτει με μία τροποποίηση της μεθόδου απαλοιφής του Gauss και η οποία οδηγεί μετά από n βήματα στο ίδιο διαγώνιο σύστημα που μας οδηγεί και η μέθοδος GJ. Η μέθοδος αυτή είναι γνωστή σαν μέθοδος του Huard (HU) [8],[9],[28] και μπορεί να θεωρηθεί σαν μια παραλλαγή της μεθόδου απαλοιφής GJ.

Θεωρώντας το αρχικό γραμμικό σύστημα:

$$A^{(1)}x = b^{(1)} \tag{3.3.1}$$

όπου $A^{(1)} = A$ και $b^{(1)} = b$ μετά την εκτέλεση του πρώτου βήματος έχουμε το ισοδύναμο σύστημα:

$$\begin{array}{ccccccccccc}
x_1 & + & a_{12}^{(2)}x_2 & + & a_{13}^{(2)}x_3 & + & \cdots & + & a_{1n}^{(2)}x_n & = & b_1^{(2)} \\
a_{21}^{(1)}x_1 & + & a_{22}^{(1)}x_2 & + & a_{23}^{(1)}x_3 & + & \cdots & + & a_{2n}^{(1)}x_n & = & b_2^{(1)} \\
a_{31}^{(1)}x_1 & + & a_{32}^{(1)}x_2 & + & a_{33}^{(1)}x_3 & + & \cdots & + & a_{3n}^{(1)}x_n & = & b_3^{(1)} \\
\vdots & & \vdots & & \vdots & & & & \vdots & & \vdots \\
a_{n1}^{(1)}x_1 & + & a_{n2}^{(1)}x_2 & + & a_{n3}^{(1)}x_3 & + & \cdots & + & a_{nn}^{(1)}x_n & = & b_n^{(1)}
\end{array} \tag{3.3.2}$$

όπου παρατηρούμε ότι οι συντελεστές των αγνώστων της 1ης εξίσωσης έχουν διαιρεθεί με το οδηγό στοιχείο $a_{11}^{(1)}$, οπότε έχουμε το σύστημα:

$$A^{(2)}x = b^{(2)} \tag{3.3.3}$$

Στο δεύτερο βήμα της μεθόδου απαλείφεται ο άγνωστος $x_1$ από τη δεύτερη εξίσωση και ο άγνωστος $x_2$ από την πρώτη γραμμή. Ετσι προκύπτει το σύστημα:

$$\begin{array}{ccccccccccc}
x_1 & + & & + & a_{13}^{(3)}x_3 & + & \cdots & + & a_{1n}^{(3)}x_n & = & b_1^{(3)} \\
 & & x_2 & + & a_{23}^{(3)}x_3 & + & \cdots & + & a_{2n}^{(3)}x_n & = & b_2^{(3)} \\
a_{31}^{(1)}x_1 & + & a_{32}^{(1)}x_2 & + & a_{33}^{(1)}x_3 & + & \cdots & + & a_{3n}^{(1)}x_n & = & b_3^{(1)} \\
\vdots & & \vdots & & \vdots & & & & \vdots & & \vdots \\
a_{n1}^{(1)}x_1 & + & a_{n2}^{(1)}x_2 & + & a_{n3}^{(1)}x_3 & + & \cdots & + & a_{nn}^{(1)}x_n & = & b_n^{(1)}
\end{array} \tag{3.3.4}$$

ή

$$A^{(3)}x = b^{(3)} \tag{3.3.5}$$

Στο τρίτο βήμα της μεθόδου απαλείφονται οι άγνωστοι $x_1$ και $x_2$ από την 3η εξίσωση και ο άγνωστος $x_3$ από την 1η και 2η γραμμή. Ετσι προκύπτει το σύστημα:

$$\begin{array}{rcl}
x_1 + a_{14}^{(4)}x_4 + \cdots + a_{1n}^{(4)}x_n &=& b_1^{(4)} \\
x_2 + a_{24}^{(4)}x_4 + \cdots + a_{2n}^{(4)}x_n &=& b_2^{(4)} \\
x_3 + a_{34}^{(4)}x_4 + \cdots + a_{3n}^{(4)}x_n &=& b_3^{(4)} \\
a_{41}^{(1)}x_1 + a_{42}^{(1)}x_2 + a_{43}^{(1)}x_3 + a_{44}^{(1)}x_4 + \cdots + a_{4n}^{(1)}x_n &=& b_4^{(1)} \\
\vdots & & \\
a_{n1}^{(1)}x_1 + a_{n2}^{(1)}x_2 + a_{n3}^{(1)}x_3 + a_{n4}^{(1)}x_4 + \cdots + a_{nn}^{(1)}x_n &=& b_n^{(1)}
\end{array} \tag{3.3.6}$$

ή

$$A^{(4)}x = b^{(4)} \tag{3.3.7}$$

Μετά από $k-1$ τέτοια βήματα προκύπτει το ισοδύναμο σύστημα:

$$\begin{array}{rcl}
x_1 + a_{1k}^{(k)}x_k + \cdots + a_{1n}^{(k)}x_n &=& b_1^{(k)} \\
x_2 + a_{2k}^{(k)}x_k + \cdots + a_{2n}^{(k)}x_n &=& b_2^{(k)} \\
\ddots \quad \vdots \qquad\qquad \vdots & & \vdots \\
x_{k-1} + a_{k-1,k}^{(k)}x_k + \cdots + a_{k-1,n}^{(k)}x_n &=& b_{k-1}^{(k)} \\
a_{k1}^{(1)}x_1 + a_{k2}^{(1)}x_2 + \cdots + a_{k,k-1}^{(1)}x_{k-1} + a_{kk}^{(1)}x_k + \cdots + a_{k,n}^{(1)}x_n &=& b_k^{(1)} \\
\vdots & & \\
a_{n1}^{(1)}x_1 + a_{n2}^{(1)}x_2 + \cdots + a_{n,k-1}^{(1)}x_{k-1} + a_{nk}^{(1)}x_k + \cdots + a_{nn}^{(1)}x_n &=& b_n^{(1)}
\end{array} \tag{3.3.8}$$

ή

$$A^{(k)}x = b^{(k)} \tag{3.3.9}$$

Τελικά μετά από $n-1$ τέτοια βήματα προκύπτει το ισοδύναμο διαγώνιο σύστημα:

$$\begin{array}{rcl}
x_1 &=& b_1^{(n)} \\
x_2 &=& b_2^{(n)} \\
\ddots \quad \vdots & & \\
x_n &=& b_n^{(n)}
\end{array} \tag{3.3.10}$$

ή

$$x = b^{(n)} \tag{3.3.11}$$

η οποία είναι η λύση του αρχικού συστήματος.

Αναλυτικώτερα χρησιμοποιούμε πάλι τους συμβολισμούς

$$a_{ij}^{(1)} = a_{ij}, \qquad i,j = 1(1)n$$

$$a_{i,n+1}^{(1)} = b_i, \qquad i = 1(1)n$$

Στο πρώτο βήμα διαιρείται η πρώτη εξίσωση με το οδηγό στοιχείο $a_{11}^{(1)}$, οπότε προκύπτει το σύστημα (3.3.2) όπου:

$$a_{11}^{(2)} = 1$$

και

$$a_{1j}^{(2)} = \frac{a_{1j}^{(1)}}{a_{11}^{(1)}}, \quad j = 2(1)n+1$$

Στο δεύτερο βήμα ορίζουμε τον πολλαπλασιαστή:

$$h_{21} = -a_{21}^{(1)}$$

τότε για να απαλειφθεί ο $x_1$ από την 2η εξίσωση, αρκεί να προστεθεί σ'αυτή η πρώτη εξίσωση πολλαπλασιασμένη με $h_{21}$, οπότε έχουμε:

$$a_{2j}^{(2)} = a_{2j}^{(1)} + h_{21} \cdot a_{1j}^{(2)}, \quad j = 2(1)n+1$$

Στη συνέχεια διαιρείται η 2η εξίσωση με το οδηγό στοιχείο $a_{22}^{(2)}$, οπότε έχουμε:

$$a_{22}^{(3)} = 1$$

$$a_{2j}^{(3)} = \frac{a_{2j}^{(2)}}{a_{22}^{(2)}}, \quad j = 3(1)n+1$$

Ορίζουμε τον πολλαπλασιαστή:

$$g_{12} = -a_{12}^{(2)}$$

τότε για να απαλειφθεί ο $x_2$ από την πρώτη εξίσωση, αρκεί να προστεθεί σ'αυτή η δεύτερη εξίσωση πολλαπλασιασμένη με $g_{12}$, οπότε έχουμε:

$$a_{1j}^{(3)} = a_{1j}^{(2)} + g_{12} \cdot a_{2j}^{(3)}, \quad j = 3(1)n+1$$

έτσι καταλήγουμε στο σύστημα (3.3.4). Τελικά, μετά από n παρόμοια βήματα καταλήγουμε στο διαγώνιο σύστημα (3.3.10) που δίνει άμεσα τη λύση του αρχικού συστήματος.

Υπό μορφή πινάκων η μέθοδος HU αναζητά ένα μη ιδιάζοντα πίνακα H τέτοιο ώστε:

$$HAx = Hb \tag{3.3.12}$$

με

$$HA = I \tag{3.3.13}$$

Ας υποθέσουμε ότι το αρχικό σύστημα είναι το $A^{(1)}x = b^{(1)}$ τότε δημιουργούνται οι ακολουθίες πινάκων:

$$\begin{aligned} A^{(k+1)} &= G^{(k)}H^{(k)}A^{(k)} \\ b^{(k+1)} &= G^{(k)}H^{(k)}b^{(k)} \end{aligned}, \quad k = 1(1)n-1 \tag{3.3.14}$$

όπου

$$H^{(k)} = I_n + \sum_{i=1}^{k-1} h_{ki} E_{ki}$$

$$G^{(k)} = I_n + \sum_{i=1}^{k-1} g_{ik} E_{ik}$$

$$\left.\begin{array}{l} h_{ki} = -a_{ki} \\ \\ g_{ik} = -a_{ik} \end{array}\right\}, \quad i = 1(1)k-1$$

$$E_{ki} = \begin{array}{c} \\ \\ \\ k \mapsto \\ \\ \\ \end{array} \overset{\overset{i}{\downarrow}}{\begin{bmatrix} 0 & 0 & \cdots & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & \cdots & 0 \\ \vdots & \vdots & & \vdots & & \vdots \\ 0 & 0 & \cdots & 1 & \cdots & 0 \\ \vdots & \vdots & & \vdots & & \vdots \\ 0 & 0 & \cdots & 0 & \cdots & 0 \end{bmatrix}}, \quad E_{ik} = E_{ki}^T$$

ώστε το γραμμικό σύστημα $A^{(k+1)}x = b^{(k+1)}$ να είναι ισοδύναμο με το $A^{(k)}x = b^{(k)}$.

Μετά από $n$ βήματα οι αναδρομικοί τύποι (3.3.14) οδηγούν σ'ένα γραμμικό σύστημα της μορφής:

$$A^{(n)}x = b^{(n)} \tag{3.3.15}$$

όπου

$$A^{(n)} = (G^{(n)}H^{(n)})(G^{(n-1)}H^{(n-1)})\cdots(G^{(1)}H^{(1)})$$

είναι ένας διαγώνιος πίνακας.

Η μέθοδος Huard περιλαμβάνει στο $k$ βήμα τις παρακάτω δύο φάσεις εργασιών:

1. **Φάση καθόδου**

   Περιλαμβάνει το μηδενισμό των $k-1$ πρώτων στοιχείων της k γραμμής του πίνακα A και τις αντίστοιχες τροποποιήσεις των $n-k+2$ επομένων στοιχείων της.

2. **Φάση ανόδου**

   Περιλαμβάνει το μηδενισμό των $k-1$ πρώτων στοιχείων της k στήλης του πίνακα A και τις αντίστοιχες τροποποιήσεις των $n-k+1$ τελευταίων στοιχείων των $k-1$ πρώτων γραμμών.

Η σειρά προσπέλασης στα στοιχεία του πίνακα A μας οδηγεί σε δύο διαφορετικούς τρόπους εκτέλεσης των εργασιών σε καθεμιά από τις παραπάνω φάσεις: κατά γραμμές ή κατά στήλες. Ετσι προκύπτουν 4 διαφορετικοί αλγόριθμοι που είναι οι παρακάτω:

**(*Κάθοδος κατά γραμμές*)**
*Για* $i := 1(1)k-1$ *επανάλαβε*
  *για* $j := k(1)n+1$ *επανάλαβε*
    $a_{kj} := a_{kj} - a_{ki} * a_{ij}$

**(*Κάθοδος κατά στήλες*)**
*Για* $j := k(1)n+1$ *επανάλαβε*
  *για* $i := 1(1)k-1$ *επανάλαβε*
    $a_{kj} := a_{kj} - a_{ki} * a_{ij}$

**(*Άνοδος κατά γραμμές*)**
*Για* $i := 1(1)k-1$ *επανάλαβε*
  *για* $j := k+1(1)n+1$ *επανάλαβε*
    $a_{ij} := a_{ij} - a_{ik} * a_{kj}$

**(*Άνοδος κατά στήλες*)**
*Για* $j := k+1(1)n+1$ *επανάλαβε*
  *για* $i := 1(1)k-1$ *επανάλαβε*
    $a_{ij} := a_{ij} - a_{ik} * a_{kj}$

Η κάθοδος κατά γραμμές δεν μπορεί να παραλληλισθεί, γιατί για κάθε τιμή του $i$, τροποποιούνται τα $n+2-k$ στοιχεία $a_{kj}$. Κατά συνέπεια η παράλληλη τροποποίηση αυτών των στοιχείων αντίκειται στη βασική προϋπόθεση ότι ένα συγκεκριμμένο στοιχείο δεν μπορεί να τροποποιηθεί ταυτόχρονα από διαφορετικούς επεξεργαστές. Ετσι απομένουν μόνο δύο παράλληλες μορφές της μεθόδου, που είναι οι παρακάτω:

1. **Στήλες-γραμμές** (Κάθοδος κατά στήλες-άνοδος κατά γραμμές)
2. **Στήλες-στήλες** (Κάθοδος κατά στήλες-άνοδος κατά στήλες)

**Μορφή Στήλες-γραμμές**

(*Κάθοδος κατά στήλες-άνοδος κατά γραμμές*)

*Για* $k = 1(1)n$ *επανάλαβε*

(*Κάθοδος κατά στήλες*)
  *Για* $j = k(1)n+1$ *επανάλαβε*
  $(D_{kj})$ : { *Για* i=1(1)k-1 *επανάλαβε*
    $a_{kj} := a_{kj} - a_{ki} * a_{ij}$

(*Προετοιμασία της k γραμμής*)
  *Για* $j = k+1(1)n+1$ *επανάλαβε*
  $(C_{kj})$ : { $a_{kj} := a_{kj} / a_{kk}$

(*Άνοδος κατά στήλες*)
  *Για* $i = 1(1)k-1$ *επανάλαβε*
  $(R_{ik})$ : { *Για* j=k+1(1)n+1 *επανάλαβε*
    $a_{ij} := a_{ij} - a_{ik} * a_{kj}$

**Σχήμα 3.3.1** Ορισμός των εργασιών για τη μορφή στήλες-γραμμές HU.

Στη συνέχεια παρουσιάζεται μια εφαρμογή της παραπάνω μορφής προκειμένου να προσδιοριστούν οι συνθήκες εξάρτησης των εργασιών όπως αυτές έχουν ορισθεί στο σχήμα 3.3.1.

**Εφαρμογή** Μορφή στήλες-γραμμές HU ($n = 4$)

$$A = \left[\begin{array}{cccc|c} a_{11} & a_{12} & a_{13} & a_{14} & a_{15} \\ a_{21} & a_{22} & a_{23} & a_{24} & a_{25} \\ a_{31} & a_{32} & a_{33} & a_{34} & a_{35} \\ a_{41} & a_{42} & a_{43} & a_{44} & a_{45} \end{array}\right]$$

**k = 1**

$C_{12}$ : $j = 2$ $a_{12} = a_{12} / a_{11}$

$C_{13}$ : $j = 3$ $a_{13} = a_{13} / a_{11}$

$C_{14}$ : $j = 4$ $a_{14} = a_{14} / a_{11}$

$C_{15}$ : $j = 5$ $a_{15} = a_{15} / a_{11}$

**k = 2**

**j = 2**

$D_{22}$ : $i = 1$ $a_{22} = a_{22} - a_{21} * a_{12}$

**j = 3**

$D_{23}$ : $i = 1$ $a_{23} = a_{23} - a_{21} * a_{13}$

**j = 4**

$D_{24}$ : $i = 1$ $a_{24} = a_{24} - a_{21} * a_{14}$

**j = 5**

$D_{25}$ : $i = 1$ $a_{25} = a_{25} - a_{21} * a_{15}$

$C_{23}$ : $j = 3$ $a_{23} = a_{23} / a_{22}$

$C_{24}$ : $j = 4$ $a_{24} = a_{24} / a_{22}$

$C_{25}$ : $j = 5$ $a_{25} = a_{25} / a_{22}$

$i = 1$

$$R_{12} : \begin{cases} j = 3 & a_{13} = a_{13} - a_{12} * a_{23} \\ j = 4 & a_{14} = a_{14} - a_{12} * a_{24} \\ j = 5 & a_{15} = a_{15} - a_{12} * a_{25} \end{cases}$$

**k = 3**

**j = 3**

$$D_{33} : \begin{cases} i = 1 & a_{33} = a_{33} - a_{31} * a_{13} \\ i = 2 & a_{33} = a_{33} - a_{32} * a_{23} \end{cases}$$

**j = 4**

$$D_{34} : \begin{cases} i = 1 & a_{34} = a_{34} - a_{31} * a_{14} \\ i = 2 & a_{34} = a_{34} - a_{32} * a_{24} \end{cases}$$

**j = 5**

$$D_{35} : \begin{cases} i = 1 & a_{35} = a_{35} - a_{31} * a_{15} \\ i = 2 & a_{35} = a_{35} - a_{32} * a_{25} \end{cases}$$

$C_{34}$ : $j = 4$ $a_{34} = a_{34} / a_{33}$

$C_{35}$ : $j = 5$ $a_{35} = a_{35} / a_{33}$

**i = 1**

$$R_{13} : \begin{cases} j = 4 & a_{14} = a_{14} - a_{13} * a_{34} \\ j = 5 & a_{15} = a_{15} - a_{13} * a_{35} \end{cases}$$

**i = 2**

$$R_{23} : \begin{cases} j = 4 & a_{24} = a_{24} - a_{23} * a_{34} \\ j = 5 & a_{25} = a_{25} - a_{23} * a_{35} \end{cases}$$

**k = 4**

**j = 4**

$$(D_{44}) : \begin{cases} \underline{j = 1} & a_{44} = a_{44} - a_{41} * a_{14} \\ \underline{j = 2} & a_{44} = a_{44} - a_{42} * a_{24} \\ \underline{j = 3} & a_{44} = a_{44} - a_{43} * a_{34} \end{cases}$$

**j = 5**

$$(D_{45}) : \begin{cases} \underline{j = 1} & a_{45} = a_{45} - a_{41} * a_{15} \\ \underline{j = 2} & a_{45} = a_{45} - a_{42} * a_{25} \\ \underline{j = 3} & a_{45} = a_{45} - a_{43} * a_{35} \end{cases}$$

$$(C_{45}) : \quad \{ \ \underline{j = 5} \quad a_{45} = a_{45} \ / \ a_{44}$$

**i = 1**

$$(R_{14}) : \{ \ \underline{j = 5} \quad a_{15} = a_{15} - a_{14} * a_{45}$$

**i = 2**

$$(R_{24}) : \{ \ \underline{j = 5} \quad a_{25} = a_{25} - a_{24} * a_{45}$$

**i = 3**

$$(R_{34}) : \{ \ \underline{j = 5} \quad a_{35} = a_{35} - a_{34} * a_{45}$$

Οπως φαίνεται στο σχήμα 3.3.1 το σύνολο των εργασιών για την διαγωνοποίηση του γραμμικού συστήματος είναι το :

$$J = \{D_{kj} \,/\, 1 \leq k \leq n, \quad k \leq j \leq n+1\} \cup \{C_{kj} \,/\, 1 \leq k \leq n, \quad k+1 \leq j \leq n+1\} \\ \cup \{R_{ik} \,/\, 1 \leq k \leq n, \quad 1 \leq i \leq k-1\}$$

Οι περιορισμοί προτεραιότητας στην σειρά εκτέλεσης των εργασιών που επιβάλλονται απο τον ακολουθιακό αλγόριθμο είναι:

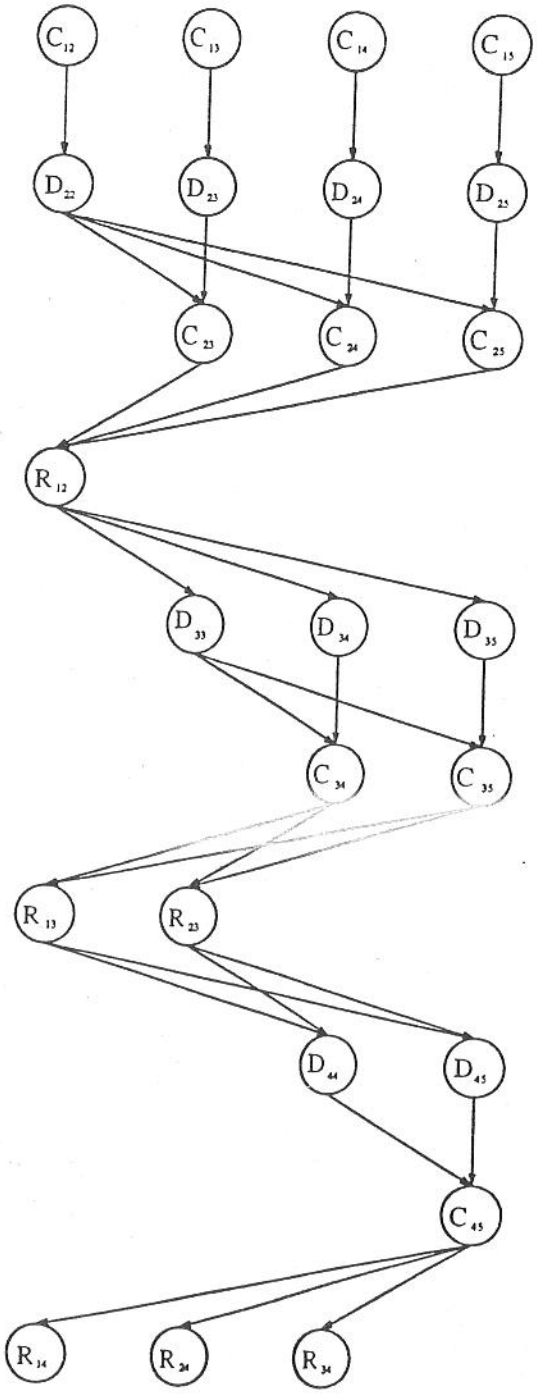
$$C_{kj} \ll R_{ik}, \quad 1 \leq k \leq n, \quad k+1 \leq j \leq n+1, \quad 1 \leq i \leq k-1$$

$$D_{kk} \ll C_{kj}, \quad 2 \leq k \leq n, \quad k+1 \leq j \leq n+1$$

$$D_{kj} \ll C_{kj}, \quad 2 \leq k \leq n, \quad k+1 \leq j \leq n+1$$

$$R_{ik} \ll D_{k+1,j}, \quad 2 \leq k \leq n, \quad 1 \leq i \leq k-1, \quad k+1 \leq j \leq n+1$$

Το σύνολο J εφοδιασμένο με τους παραπάνω περιορισμούς αποτελεί ένα σύστημα εργασιών, έστω το $C = (J, \ll)$. Το αντίστοιχο μέγιστο παράλληλο γράφημα αλληλοεξάρτησης των εργασιών του C φαίνεται στο σχήμα 3.3.2.




**Σχήμα 3.3.2** Γράφημα του μέγιστου παράλληλου συστήματος εργασιών C για τη διαγωνοποίηση της μορφής στήλες-γραμμές HU.

Οι χρόνοι εκτέλεσης των εργασιών που ορίσαμε είναι:

$$W(D_{kj}) = 2(k-1)$$

$$W(C_{kj}) = 1$$

$$W(R_{ik}) = 2(n-k+1)$$

Αν θεωρήσουμε τώρα τό γράφημα (βλ. σχήμα 3.3.2) με βάρη των κόμβων τους χρόνους εκτέλεσης τους, τότε μακρύτερο μονοπάτι είναι το:

$$s_1 = (C_{12}, D_{22}, C_{23}, R_{12}, D_{33}, C_{34}, R_{13}, \ldots, D_{nn}, C_{n,n+1}, R_{1n})$$

Ο χρόνος εκτέλέσης των εργασιών που ανήκουν στο $s_1$ είναι:

$$\begin{aligned} T_p = L(s_1) &= W(C_{12}) + \sum_{k=2}^{n} [W(D_{kk}) + W(C_{k,k+1}) + W(R_{1k})] \\ &= 1 + \sum_{k=2}^{n} [2(k-1) + 1 + 2(n-k+1)] \\ &= 1 + \sum_{k=2}^{n} (2n+1) = 1 + (n-1)(2n+1) = 2n^2 - n \end{aligned}$$

Ο χρόνος $T_p$ είναι ο ελάχιστος χρόνος για την εκτέλεση των εργασιών στο γράφημα του σχήματος 3.3.2. Επίσης ο μέγιστος αριθμός των εργασιών του μπορούν να εκτελεσθούν παράλληλα είναι n και είναι οι εργασίες:

$$C_{12}, C_{13}, C_{14}, \ldots, C_{1,n+1}$$

Αρα ο μέγιστος αριθμός των επεξεργαστών που απαιτείται ώστε το σύστημα εργασιών C να εκτελεσθεί σε χρόνο $T_p$ είναι $p = n$. Στη συνέχεια μπορούμε να κατασκευάσουμε μια βέλτιστη δρομολόγηση για το μέγιστο παράλληλο σύστημα εργασιών C με $p = n$ επεξεργαστές (βλ. σχήμα 3.3.3).

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| $P_1$ | $C_{12}$ | $D_{22}$ | $C_{23}$ | $R_{12}$ | $D_{33}$ | $C_{34}$ | $R_{13}$ | | | |
| $P_2$ | $C_{13}$ | $D_{23}$ | $C_{24}$ | | $D_{34}$ | $C_{35}$ | $R_{23}$ | | | |
| $P_3$ | $C_{14}$ | $D_{24}$ | $C_{25}$ | | $D_{35}$ | $C_{36}$ | | | | |
| $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\cdots$ | $\vdots$ | $\vdots$ |
| $P_{n-2}$ | $C_{1,n-1}$ | $D_{2,n-1}$ | $C_{2,n}$ | | $D_{3n}$ | $C_{3,n+1}$ | | | | |
| $P_{n-1}$ | $C_{1,n}$ | $D_{2,n}$ | $C_{2,n+1}$ | | $D_{3,n+1}$ | | | | | |
| $P_{n-1}$ | $C_{1,n+1}$ | $D_{2,n+1}$ | | | | | | | | |

| | | |
|---|---|---|
| $D_{nn}$ | $C_{n,n+1}$ | $R_{1n}$ |
| $D_{n,n+1}$ | | $R_{2n}$ |
| | | $R_{3n}$ |
| $\vdots$ | $\vdots$ | |
| | | $R_{n-2,n}$ |
| | | $R_{n-1,n}$ |
| | | |

**Σχήμα 3.3.3** Μια βέλτιστη δρομολόγηση με $p = n$ επεξεργαστές για το μέγιστο παράλληλο σύστημα εργασιών C της μορφής στήλες-γραμμές HU.

Οπως φαίνεται στο σχήμα 3.3.3 οι εργασίες που αποτελούν το μακρύτερο μονοπάτι $s_1$ στο προηγούμενο γράφημα (σχ. 3.3.2) δίνονται για εκτέλεση στον επεξεργαστή $P_1$, ενώ οι υπόλοιπες δίνονται στους $n-1$ άλλους επεξεργαστές έτσι ώστε ο επεξεργαστής $P_j$ αναλαμβάνει να εκτελέσει τις εργασίες:

$$C_{1,j+1}, D_{2,j+1}, C_{2,j+2}, D_{3,j+2}, C_{3,j+3}, R_{j,j+1}, \ldots, D_{n-j+1,n}, C_{n-j+1,n+1}, R_{j,n}$$

Επειδή αυτή η δρομολόγηση έχει μήκος $L(s_1)$, που είναι το μήκος του μακρύτερου μονοπατιού, είναι μία βέλτιστη δρομολόγηση με $n-1$ επεξεργαστές. Η επιτάχυνση του αλγορίθμου χρησιμοποιώντας την παραπάνω δρομολόγηση είναι:

$$S_p = \frac{T_1}{T_p} = \frac{\frac{2}{3}n^3 + O(n^2)}{2n^2 - n} \simeq \frac{1}{3}n$$

και η αποδοτικότητά του είναι:

$$E_p = \lim_{n\to\infty} \frac{T_1}{pT_p} = \lim_{n\to\infty} \frac{\frac{2}{3}n^3 + O(n^2)}{n(2n^2 - n)} = \frac{1}{3} \simeq 0.333$$

**Μορφή Στήλες-στήλες**

(*Κάθοδος κατά στήλες-άνοδος κατά στήλες*)

Για $k = 1(1)n$ επανάλαβε
  Για $j = k+1(1)n+1$ επανάλαβε

$$(T_{kj}) : \begin{cases} a_{kj} := a_{kj} \,/\, a_{kk} \\ \Gamma\iota\alpha \;\; i = 1(1)k-1 \;\; \text{επανάλαβε} \\ \qquad a_{ij} := a_{ij} - a_{ik} * a_{kj} \\ \Gamma\iota\alpha \;\; i = 1(1)k, \quad \text{επανάλαβε} \\ \qquad a_{k+1,j} := a_{k+1,j} - a_{k+1,i} * a_{ij} \end{cases}$$

**Σχήμα 3.3.4** Ορισμός των εργασιών για τη μορφή στήλες-στήλες της μεθόδου HU.

Στη συνέχεια παρουσιάζεται μια εφαρμογή της παραπάνω μορφής προκειμένου να προσδιοριστούν οι συνθήκες εξάρτησης των εργασιών όπως αυτές έχουν ορισθεί στο σχήμα 3.3.4.

**Εφαρμογή** Μορφή στήλες-στήλες HU ($n = 4$)

$$A = \left[\begin{array}{cccc|c} a_{11} & a_{12} & a_{13} & a_{14} & a_{15} \\ a_{21} & a_{22} & a_{23} & a_{24} & a_{25} \\ a_{31} & a_{32} & a_{33} & a_{34} & a_{35} \\ a_{41} & a_{42} & a_{43} & a_{44} & a_{45} \end{array}\right]$$

$\underline{\mathbf{k=1}}$

$\underline{\mathbf{j=2}}$

$$(T_{12}): \begin{cases} a_{12} = a_{12} \,/\, a_{11} \\ \underline{i=1} \quad a_{22} = a_{22} - a_{21} * a_{12} \end{cases}$$

$\underline{\mathbf{j=3}}$

$$(T_{13}): \begin{cases} a_{13} = a_{13} \,/\, a_{11} \\ \underline{i=1} \quad a_{23} = a_{23} - a_{21} * a_{13} \end{cases}$$

$\underline{\mathbf{j=4}}$

$$(T_{14}): \begin{cases} a_{14} = a_{14} \,/\, a_{11} \\ \underline{i=1} \quad a_{24} = a_{24} - a_{21} * a_{14} \end{cases}$$

$\underline{\mathbf{j=5}}$

$$(T_{15}): \begin{cases} a_{15} = a_{15} \,/\, a_{11} \\ \underline{i=1} \quad a_{25} = a_{25} - a_{21} * a_{15} \end{cases}$$

$\underline{\mathbf{k=2}}$

$\underline{\mathbf{j=3}}$

$$(T_{23}): \begin{cases} a_{23} = a_{23} \,/\, a_{22} \\ \underline{i=1} \quad a_{13} = a_{13} - a_{12} * a_{23} \\ \underline{i=1} \quad a_{33} = a_{33} - a_{31} * a_{13} \\ \underline{i=2} \quad a_{33} = a_{33} - a_{32} * a_{23} \end{cases}$$

$\underline{\mathbf{j=4}}$

$$(T_{24}): \begin{cases} a_{24} = a_{24} \,/\, a_{22} \\ \underline{i=1} \quad a_{14} = a_{14} - a_{12} * a_{24} \\ \underline{i=1} \quad a_{34} = a_{34} - a_{31} * a_{14} \\ \underline{i=2} \quad a_{34} = a_{34} - a_{32} * a_{24} \end{cases}$$

$\underline{\mathbf{j=5}}$

$$(T_{25}): \begin{cases} a_{25} = a_{25} \,/\, a_{22} \\ \underline{i=1} \quad a_{15} = a_{15} - a_{12} * a_{25} \\ \underline{i=1} \quad a_{35} = a_{35} - a_{31} * a_{15} \\ \underline{i=2} \quad a_{35} = a_{35} - a_{32} * a_{25} \end{cases}$$

$\underline{\mathbf{k=3}}$

$\underline{\mathbf{j=4}}$

$$(T_{34}): \begin{cases} a_{34} = a_{34} \,/\, a_{33} \\ \underline{i=1} \quad a_{14} = a_{14} - a_{13} * a_{34} \\ \underline{i=2} \quad a_{24} = a_{24} - a_{23} * a_{34} \\ \underline{i=1} \quad a_{44} = a_{44} - a_{41} * a_{14} \\ \underline{i=2} \quad a_{44} = a_{44} - a_{42} * a_{24} \\ \underline{i=3} \quad a_{44} = a_{44} - a_{43} * a_{34} \end{cases}$$

$\underline{\mathbf{j=5}}$

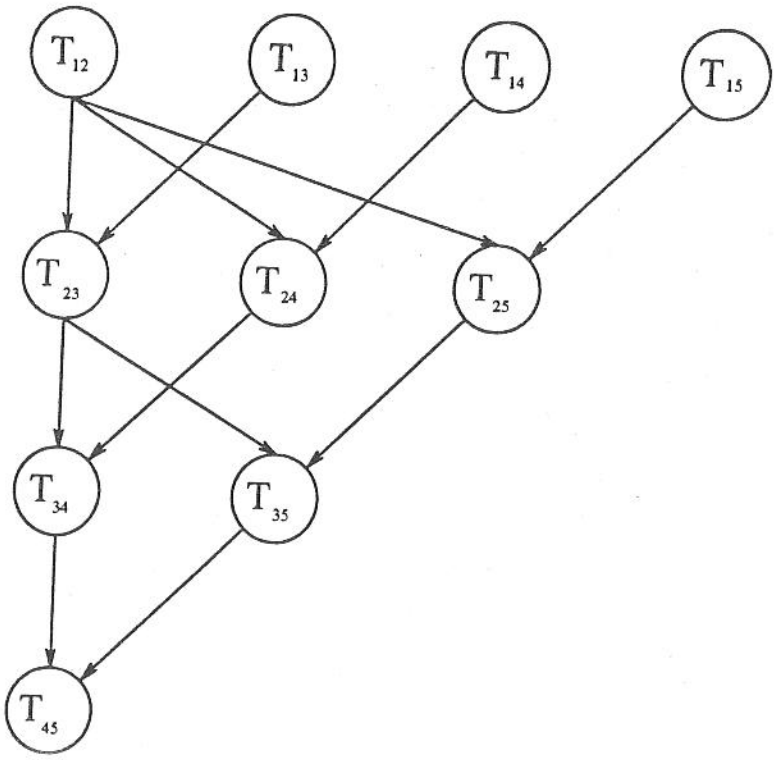
$$(T_{35}): \begin{cases} a_{35} = a_{35} \,/\, a_{33} \\ \underline{i=1} \quad a_{15} = a_{15} - a_{13} * a_{35} \\ \underline{i=2} \quad a_{25} = a_{25} - a_{23} * a_{35} \\ \underline{i=1} \quad a_{45} = a_{45} - a_{41} * a_{15} \\ \underline{i=2} \quad a_{45} = a_{45} - a_{42} * a_{25} \\ \underline{i=3} \quad a_{45} = a_{45} - a_{43} * a_{35} \end{cases}$$

$\underline{\mathbf{k=4}}$

$\underline{\mathbf{j=5}}$

$$(T_{45}): \begin{cases} a_{45} = a_{45} \,/\, a_{22} \\ \underline{i=1} \quad a_{15} = a_{15} - a_{14} * a_{45} \\ \underline{i=2} \quad a_{25} = a_{25} - a_{24} * a_{45} \\ \underline{i=3} \quad a_{35} = a_{35} - a_{34} * a_{45} \end{cases}$$

Ετσι καταλήγουμε στο μέγιστο παράλληλο γράφημα των εργασιών που φαίνεται στο παρακάτω σχήμα 3.3.5.




**Σχήμα 3.3.5** Μέγιστο παράλληλο γράφημα της μορφής στήλες-στήλες HU ($n = 4$).

Οπως φαίνεται στο σχήμα 3.3.4 το σύνολο των εργασιών για τη διαγωνοποίηση του γραμμικού συστήματος είναι το :

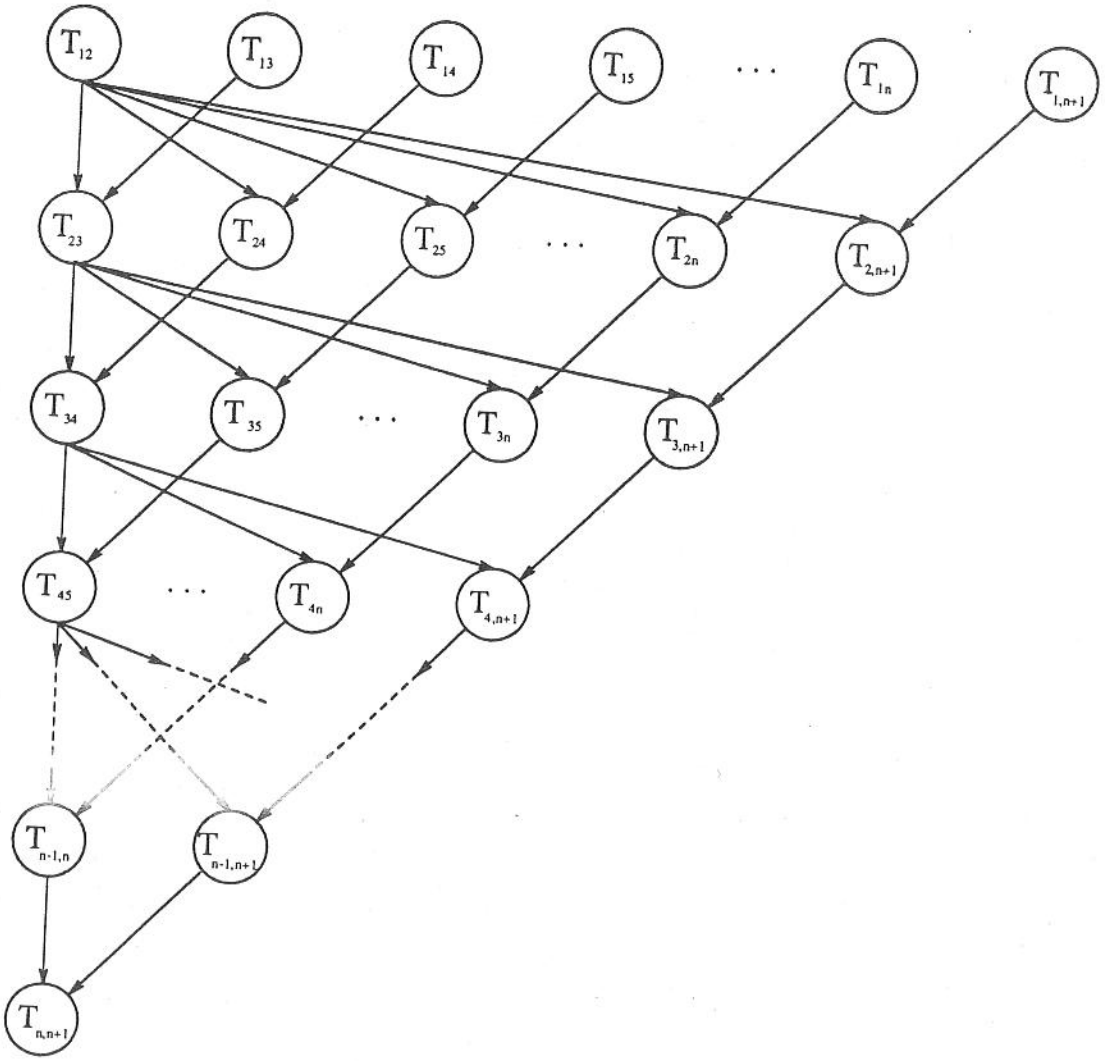
$$J = \{T_{kj} \,/\, 1 \leq k \leq n, \quad k+1 \leq j \leq n+1\}$$

Οι περιορισμοί προτεραιότητας στην σειρά εκτέλεσης των εργασιών που επιβάλλονται από τον ακολουθιακό αλγόριθμο είναι:

$$T_{k,k+1} \ll T_{k+1,j}, \quad 1 \leq k \leq n-1, \quad k+2 \leq j \leq n+1$$

$$T_{k,j} \ll T_{k+1,j}, \quad 1 \leq k \leq n-1, \quad k+2 \leq j \leq n+1$$

Το σύνολο $J$ εφοδιασμένο με τους παραπάνω περιορισμούς αποτελεί ένα σύστημα εργασιών, έστω το $C = (J, \ll)$. Το αντίστοιχο μέγιστο παράλληλο γράφημα αλληλοεξάρτησης των εργασιών του C φαίνεταιστο σχήμα 3.3.6.




**Σχήμα 3.3.6** Μέγιστο παράλληλο γράφημα του συστήματος εργασιών C για τη διαγωνοποίηση της μορφής στήλες-στήλες HU.

Οι χρόνοι εκτέλεσης των εργασιών που ορίσαμε είναι:

$$W(T_{kj}) = 4k - 1$$

Παρατηρούμε ότι για σταθερό $k$ οι εργασίες $T_{kj}$ έχουν τον ίδιο χρόνο εκτέλεσης που είναι $4k-1$.

Αν θεωρήσουμε τώρα τό γράφημα (βλ. σχήμα 3.3.6) με βάρη των κόμβων τους χρόνους εκτέλεσης τους, τότε μακρύτερο μονοπάτι είναι το:

$$s_1 = (T_{12}, T_{23}, T_{34}, \ldots, T_{n-1,n}, T_{n,n+1})$$

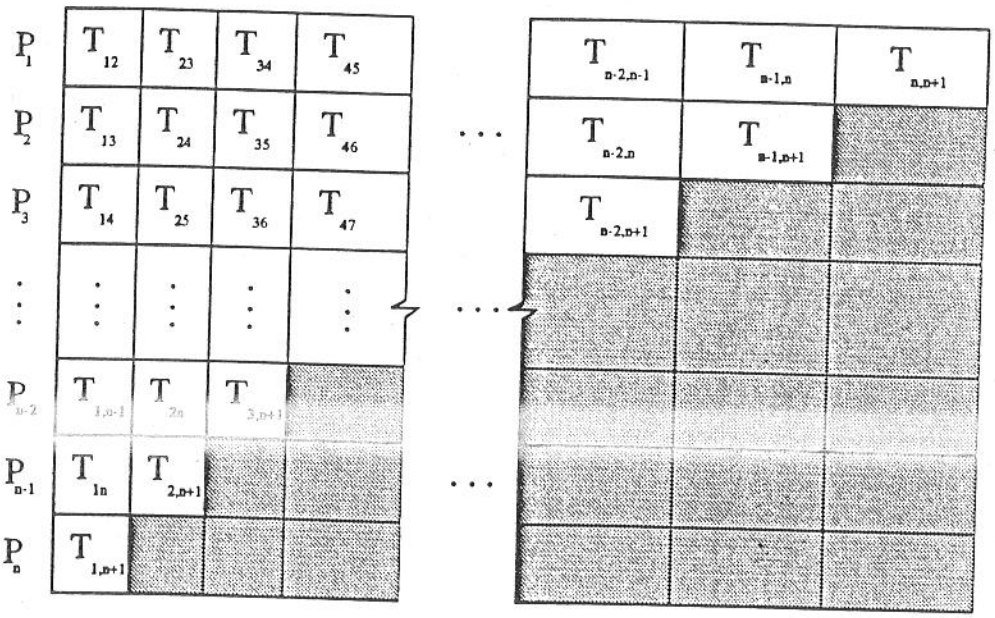
Ο χρόνος εκτέλεσης των εργασιών που ανήκουν στο $s_1$ είναι:

$$T_p = L(s_1) = \sum_{k=1}^{n} W(T_{k,k+1}) = \sum_{k=1}^{n}(4\varkappa - 1) = 4\frac{n(n+1)}{2} - n = 2n^2 + n$$

Ο χρόνος $T_p$ είναι ο ελάχιστος χρόνος για την εκτέλεση των εργασιών στο γράφημα του σχήματος 3.3.6. Επίσης ο μέγιστος αριθμός των εργασιών του μπορούν να εκτελεσθούν παράλληλα είναι n και οι εργασίες αυτές είναι οι:

$$T_{12}, T_{13}, T_{14}, \ldots, T_{1n}, T_{1,n+1}$$

Αρα ο μέγιστος αριθμός των επεξεργαστών που απαιτείται ώστε το σύστημα εργασιών C να εκτελεσθεί σε χρόνο $T_p$ είναι $p = n$. Στη συνέχεια παρουσιάζεται μια βέλτιστη δρομολόγηση για το μέγιστο παράλληλο σύστημα εργασιών C με $p = n$ επεξεργαστές (βλ. σχήμα 3.3.7).




**Σχήμα 3.3.7** Μια βέλτιστη δρομολόγηση με $p = n$ επεξεργαστές για το μέγιστο παράλληλο σύστημα εργασιών C της μορφής στήλες-στήλες HU.

Οπως φαίνεται στο σχήμα 3.3.7 οι εργασίες που αποτελούν το μακρύτερο μονοπάτι στο γράφημα του σχήματος 3.1.3 δίνονται για εκτέλεση στον επεξεργαστή $P_1$, ενώ οι υπόλοιπες δίνονται στους $n-1$ άλλους επεξεργαστές $(P_j, j = 2(1)n)$. Συγκεκριμένα ο επεξεργαστής $P_j$ αναλαμβάνει την εκτέλεση των εργασιών:

$$T_{1,j+1}, T_{2,j+2}, \ldots, T_{n-j+1,n+1}$$

Επειδή αυτή η δρομολόγηση έχει μήκος $L(s_1)$, που είναι το μήκος του μακρύτερου μονοπατιού, είναι μία βέλτιστη δρομολόγηση με $n$ επεξεργαστές. Η επιτάχυνση του αλγορίθμου χρησιμοποιώντας την παραπάνω δρομολόγηση είναι:

$$S_p = \frac{T_1}{T_p} = \frac{\frac{2}{3}n^2 + O(n^2)}{2n^2 + n} \simeq \frac{1}{3}n$$

και η αποδοτικότητά του είναι:

$$E_p = \lim_{n\to\infty} \frac{T_1}{pT_p} = \lim_{n\to\infty} \frac{\frac{2}{3}n^3 + O(n^2)}{n(2n^2 + n)} = \frac{1}{3} \simeq 0.333$$

Παρατηρούμε ότι οι δυό παράλληλες μορφές της μεθόδου Huard που μελετήθηκαν παραπάνω έχουν την ίδια αποδοτικότητα. Επίσης διαπιστώνεται ότι το γράφημα της μορφής στήλες-στήλες της HU

έχει την ίδια μορφή με το γράφημα της μορφής KJI-GJ (βλ. σχ. 3.1.3 και 3.3.6), ενώ οι εργασίες με τον ίδιο συμβολισμό $T_{ij}$ που ανήκουν σε αυτά είναι διαφορετικές.

## 3.4 ΠΑΡΑΛΛΗΛΕΣ ΜΟΡΦΕΣ ΤΗΣ ΜΕΘΟΔΟΥ ΑΠΑΛΟΙΦΗΣ ΤΩΝ GAUSS-JORDAN (GJ) ΜΕ O(n²) ΕΠΕΞΕΡΓΑΣΤΕΣ

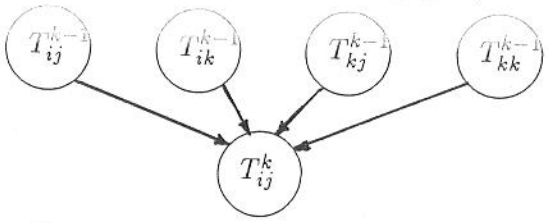
Υποθέτουμε ότι διαθέτουμε ένα μεγάλο αριθμό $p \simeq O(n^2)$ επεξεργαστών και ότι κάθε επεξεργαστής μπορεί να προσπελάσει κάθε στοιχείο του πίνακα $[A \mid b]$ του αρχικού γραμμικού συστήματος. Ο ακολουθιακός αλγόριθμος της μεθόδου GJ είναι ο ακόλουθος :

Για $k = 1(1)n$ επανάλαβε
  Για $j = k+1(1)n+1$ επανάλαβε
    Για $i = 1(1)n, \quad i \neq k$ επανάλαβε
      $a_{ij} := a_{ij} - a_{ik}a_{kj} \,/\, a_{kk}$

Παρατηρούμε ότι ο αλγόριθμος αυτός αν εκτελεσθεί ακολουθιακά είναι ασύμφορος, γιατί επαναλαμβάνει πολλές φορές τις ίδιες πράξεις. Αντίθετα, αν είναι διαθέσιμοι $n(n-1)$ επεξεργαστές, είναι δυνατόν για κάθε τιμή του $k$, να εκτελεστεί παράλληλα η τροποποίηση (ενημέρωση) όλων των συντελεστών $a_{ij}$. Αν ορίσουμε $T_{ij}^k$ την ακόλουθη υπολογιστική εργασία:

$$\left(T_{ij}^k\right) \; : \; a_{ij} := a_{ij} - a_{ik}a_{kj} \,/\, a_{kk},$$

τότε ο παραπάνω ακολουθιακός αλγόριθμος επιβάλλει στο $k$-βήμα η διάταξη των εργασιών να είναι η ακόλουθη:




### 3.4.1 Παραλληλία κατά στήλες

Στην παράγραφο αυτή θα εξετάσουμε τις παράλληλες μορφές **κατά στήλες** της μεθόδου GJ που είναι οι εξής:

1. Μορφή KJI
2. Μορφή JKI

**Μορφή KJI-GJ**

Στη μορφή αυτή ακολουθείται η εξής σειρά εργασιών: Στο $k$-βήμα της μεθόδου εκτελούνται πρώτα οι εργασίες $T_{ik}^k$, $i = 1(1)n$ , $i \neq k$ (βλ. σχήμα 4.1.1) που περιλαμβάνουν τις διαιρέσεις των στοιχείων $a_{ik}^k$ της $k$-στήλης του $A^{(k)}$ (εκτός του διαγωνίου $a_{kk}^{(k)}$) δια του διαγωνίου στοιχείου $a_{kk}^{(k)}$. Στη συνέχεια ακολουθούν οι εργασίες $T_{ij}^k$, $\quad 1 \leq i \leq n, \quad i \neq k, \quad k+1 \leq j \leq n+1$, όπου η καθεμιά περιλαμβάνει την τροποποίηση ενός στοιχείου $a_{ij}^{(k)}$ του $n(n-k+1)$ δεξιού υποπίνακα του $A^{(k)}$ με χρήση του τύπου:

$$a_{ij} := a_{ij} - a_{ik} * a_{kj}, \quad 1 \leq i \leq n, \quad i \neq k, \quad k+1 \leq j \leq n+1.$$

Αν τώρα απαιτείται να χρησιμοποιήσουμε την τεχνική της μερικής οδήγησης (βλ. σχήμα 4.1.2), τότε πρέπει στο $k$-βήμα της μεθόδου να προηγηθεί μια επιπρόσθετη εργασία $P_{kk}$ η οποία να περιλαμβάνει την αναζήτηση του οδηγού στοιχείου $a_{\ell k}$ μεταξύ των $n-k+1$ στοιχείων $a_{ik}$, $k \leq i \leq n$ και