Διάλεξη 15: Ατομική ΚΚΜ Εγγραφής/Ανάγνωσης με Γρήγορες Λειτουργίες

ΕΠΛ 432: Κατανεμημένοι Αλγόριθμοι

Τι θα δούμε σήμερα

- Γρήγορες Λειτουργίες
- Συστήματα Απαρτίας
- Αλγόριθμος SLIQ – Χρήση Quorum Views

Γρήγορες Λειτουργίες

- Μέχρι τώρα είδαμε αλγορίθμους όπου
  - Ανάγνωση χρειάζεται 2 φάσεις
  - Εγγραφή χρειάζεται 1 ή 2 φάσεις

- **Γρήγορη Λειτουργία**: Είναι η λειτουργία εγγραφής ή ανάγνωσης η οποία χρειάζεται μόνο 1 φάση ή ένα επικοινωνιακό γύρο για να ολοκληρωθεί.
  - Π.χ. Όλες οι εγγραφές στον αλγόριθμο ABD είναι γρήγορες

- **Γρήγορος Αλγόριθμος:** Είναι αυτός όπου σε κάθε εκτέλεσή του όλες οι λειτουργίες είναι γρήγορες

Συστήματα Απαρτίας

- Παρατήρηση:
  - Η επικοινωνία με την πλειοψηφία των αντιγράφων διασφαλίζει ότι για κάθε ζεύγος λειτουργιών υπάρχει τουλάχιστον ένας διαχειριστής αντιγράφου που λαμβάνει μήνυμα και από τις δύο λειτουργίες
  - Έπεται από το γεγονός ότι η τομή δύο συνόλων πλειοψηφίας δεν είναι κενή

- Ιδέα Συστημάτων Απαρτίας:
  - Οργάνωσε τους διαχειριστές αντιγράφων σε σύνολα (όχι απαραίτητα σύνολα πλειοψηφίας) έτσι ώστε κάθε δύο τέτοια σύνολα να έχουν μη κενή τομή
  - Τα σύνολα ονομάζονται **απαρτίες**

Ορισμός Συστημάτων Απαρτίας

- Quorum System $: \mathbf{Q}$

$$\mathbf{Q} = \{Q : Q \subseteq S\}\, s.t. \forall Q_i, Q_j \in \mathbf{Q} : Q_i \cap Q_j \neq \emptyset$$

- Απενεργοποιημένη Απαρτία: Μια απαρτία Q είναι απενεργοποιημένη σε μια εκτέλεση α εαν

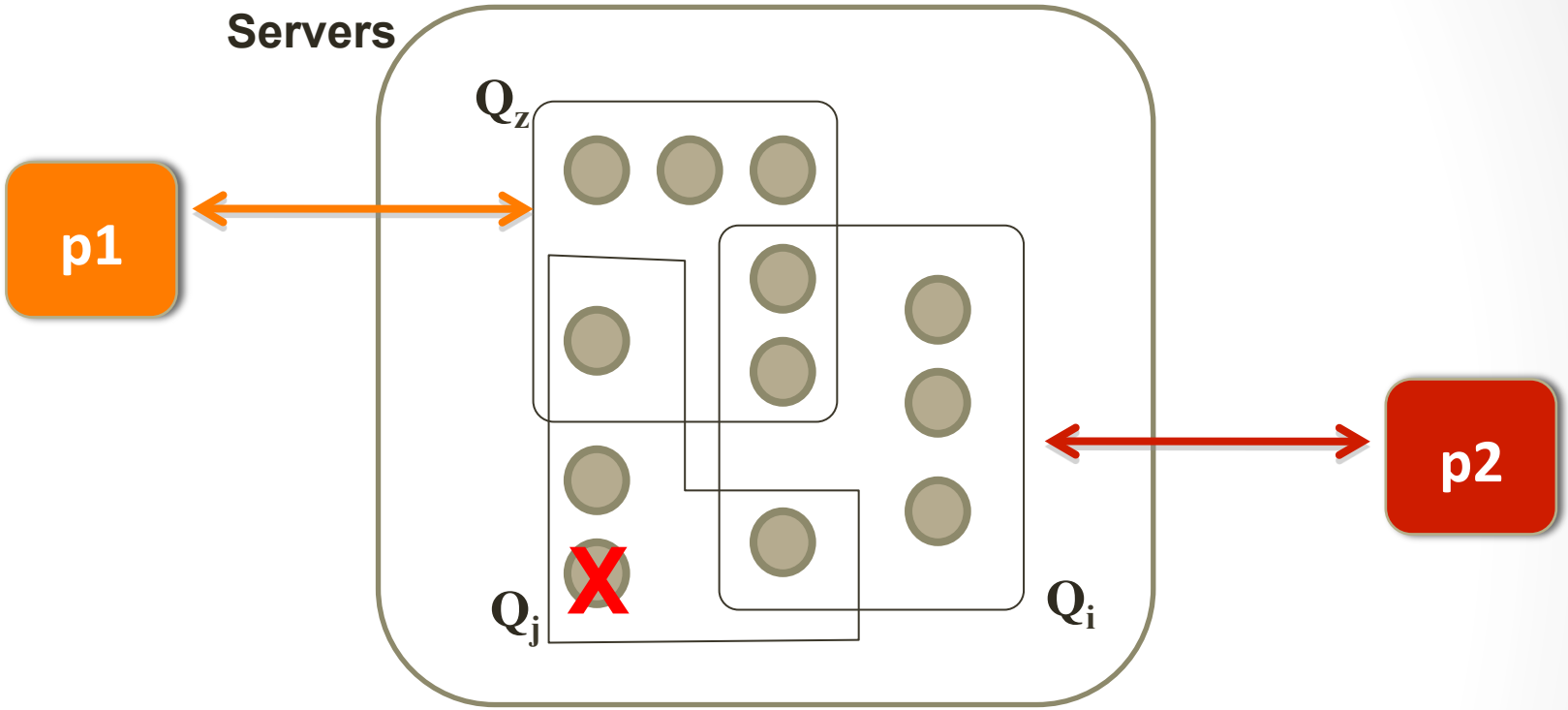
$$\exists s \in Q \text{ s.t. } fail_s \in \alpha$$

- Απενεργοποιημένο Σύστημα Απαρτίας: Ένα σύστημα απαρτίας **Q** είναι απενεργοποιημένο εαν

$$\forall Q \in \mathbb{Q}, Q \text{ is faulty}$$

- **Υπόθεση Σφάλματος:** Μια απαρτία παραμένει ενεργοποιημένη

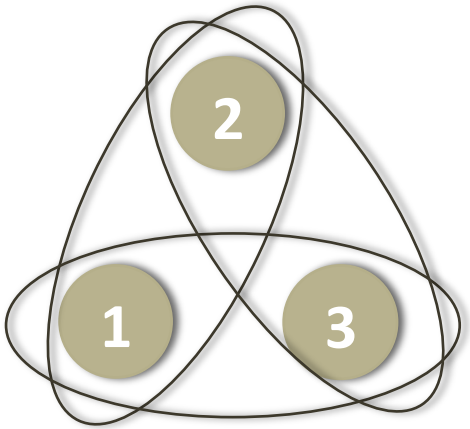
Συστήματα Απαρτίας Σχηματικά



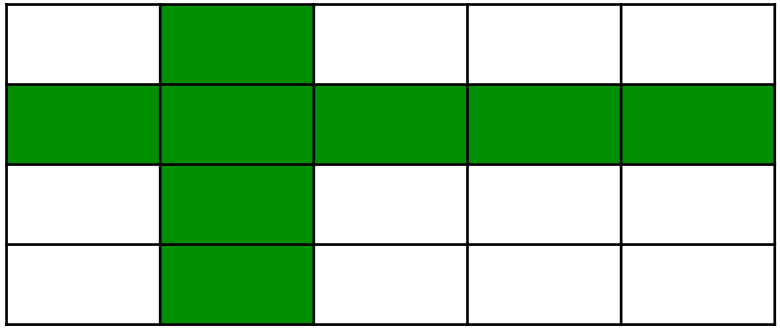
- Q_i, Q_j, Q_z είναι **απαρτίες**
- **Σύστημα Απαρτίας** είναι το σύνολο {Q_i, Q_j, Q_z}
  - Ιδιότητα: κάθε ζεύγος απαρτιών τέμνονται
- Κάθε λειτουργία Εγγραφής/Ανάγνωσης επικοινωνεί με μια απαρτία
- Εσφαλμένη Απαρτία: Αυτή που περιέχει μια εσφαλμένη διεργασία

Παραδείγματα Συστημάτων Απαρτίας

- Πλειοψηφία

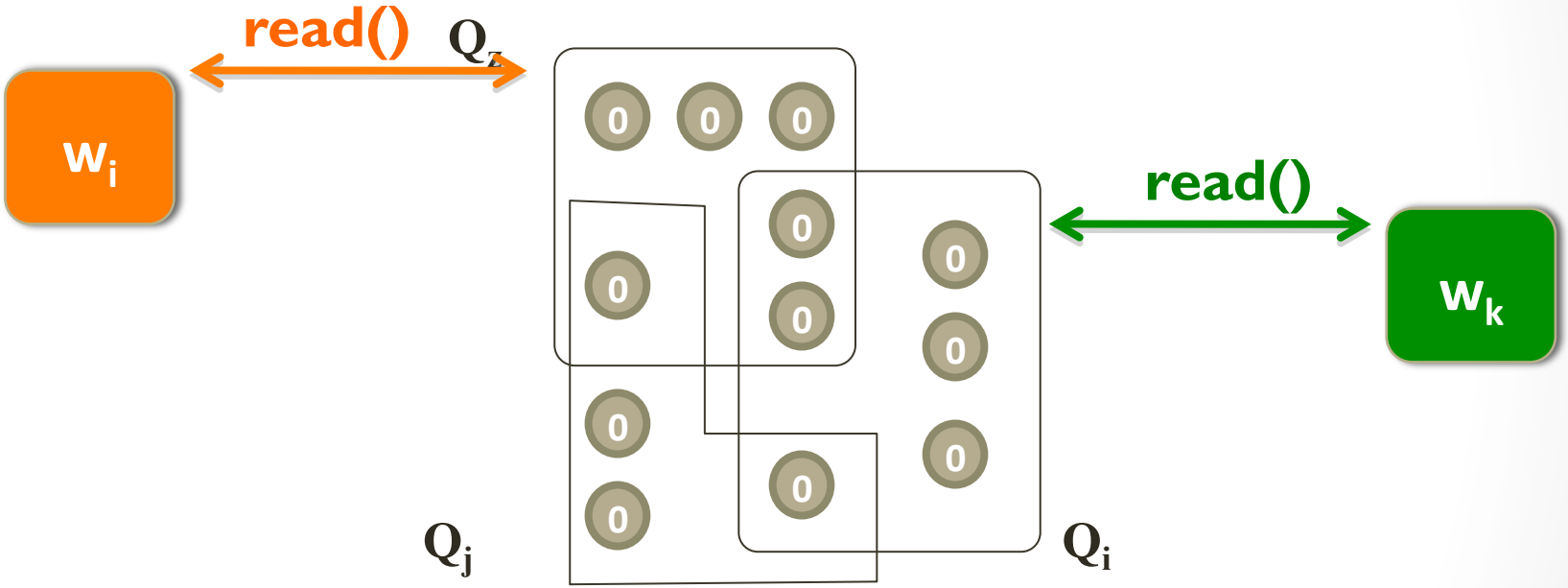


- Απαρτίες σε Πλέγμα: Κάθε απαρτία μια γραμμή και μια στήλη



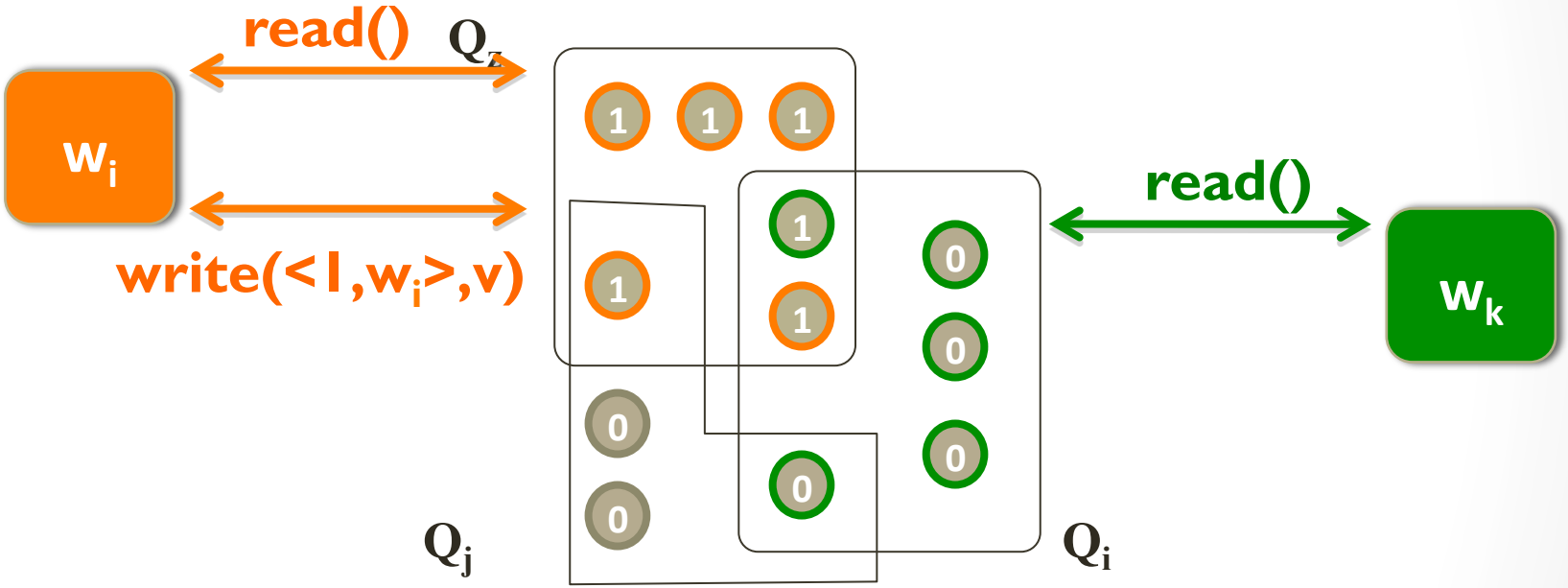
Αλγόριθμος MWMR με Απαρτίες

- Υποθέτουμε ότι $w_i > w_k$



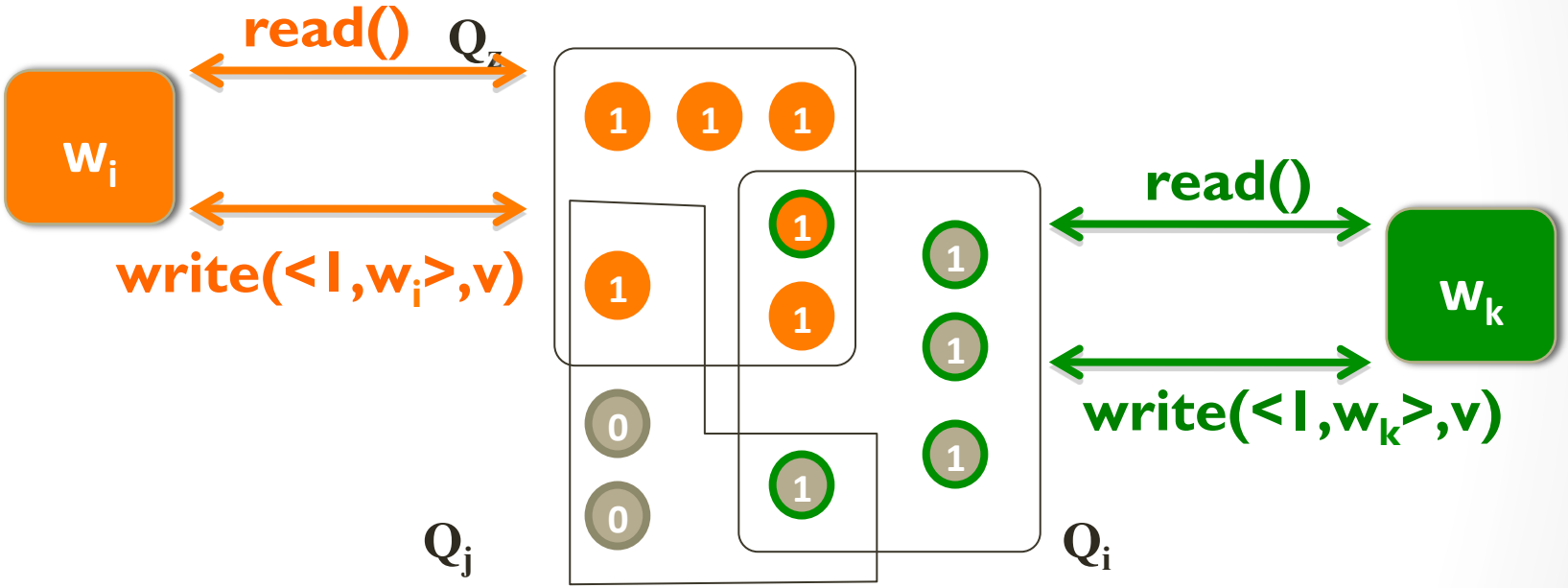
Αλγόριθμος MWMR με Απαρτίες

- Υποθέτουμε ότι $w_i > w_k$



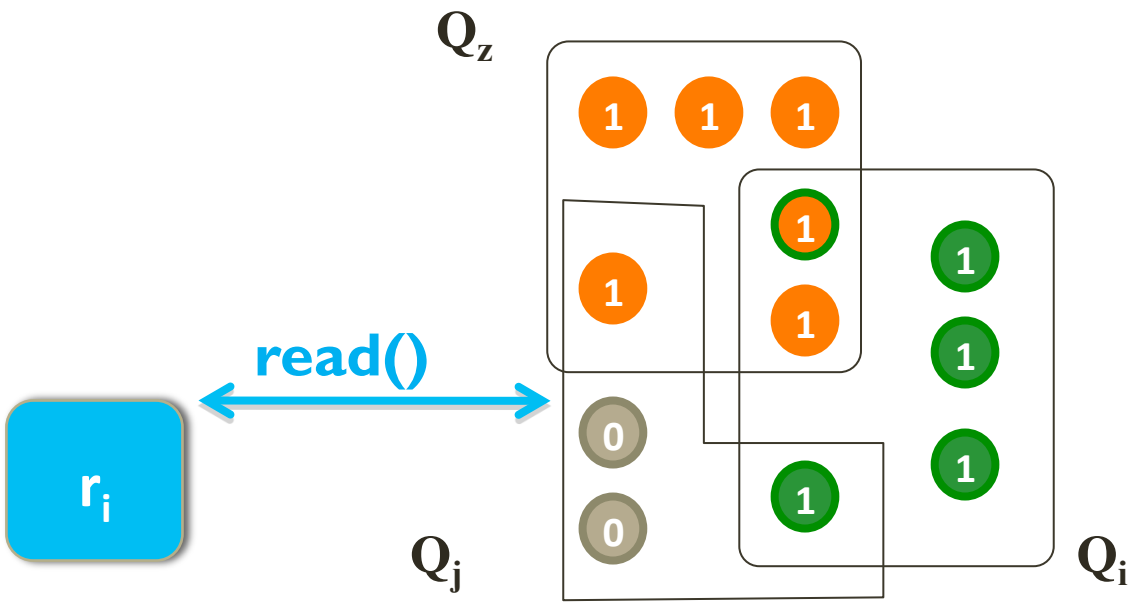
Αλγόριθμος MWMR με Απαρτίες

- Υποθέτουμε ότι $w_i > w_k$



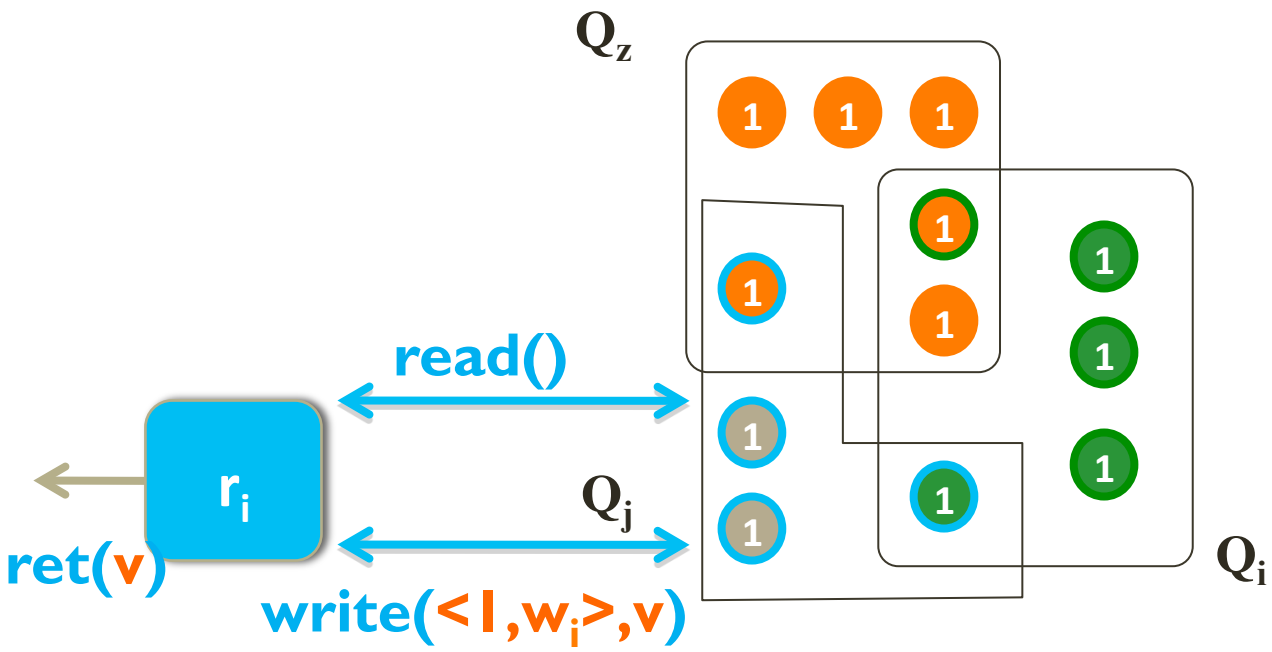
Αλγόριθμος MWMR με Απαρτίες

- Υποθέτουμε ότι $w_i > w_k$



Αλγόριθμος MWMR με Απαρτίες

- Υποθέτουμε ότι $w_i > w_k$



11/9/12

Operation Ordering: w_k -> w_i -> r_i

Quorum Views

Ιδέα:

- Προσπαθούμε να προσδιορίσουμε την κατάσταση της τελευταίας λειτουργίας εγγραφής

17/3/2011

- Αν η κατάσταση της εγγραφής κατά τον πρώτο γύρο της ανάγνωσης μπορεί να

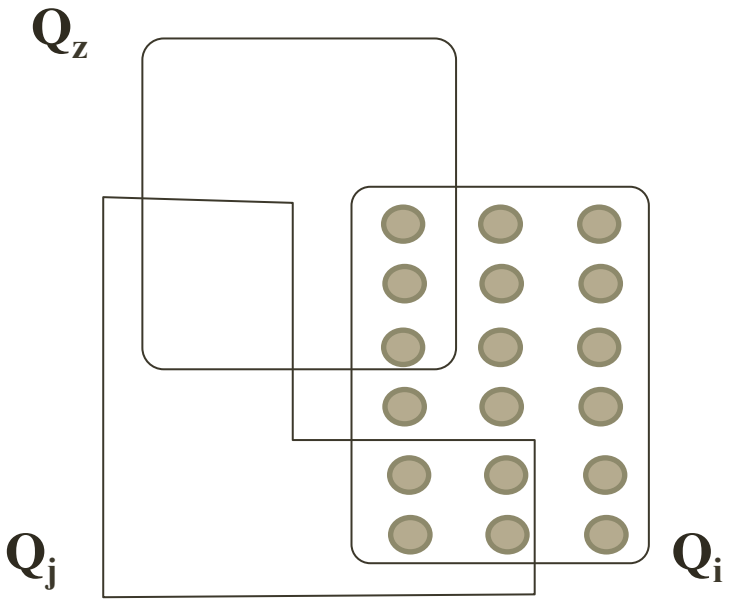
  Προσδιοριστεί => Η Ανάγνωση είναι Γρήγορη

  Δεν προσδιορίζεται => Η Ανάγνωση είναι Αργή

Προσδιοριστέα Εγγραφή - Qview(1)

- Όλα οι εξυπηρετητές της απαρτίας απαντούν με την ίδια χρονοσφραγίδα (ίση με την μέγιστη χρονοσφραγίδα που παρατηρεί ο Αναγνώστης)

$$[qView(1)] : \forall s \in Q_i : s.ts = \max TS$$

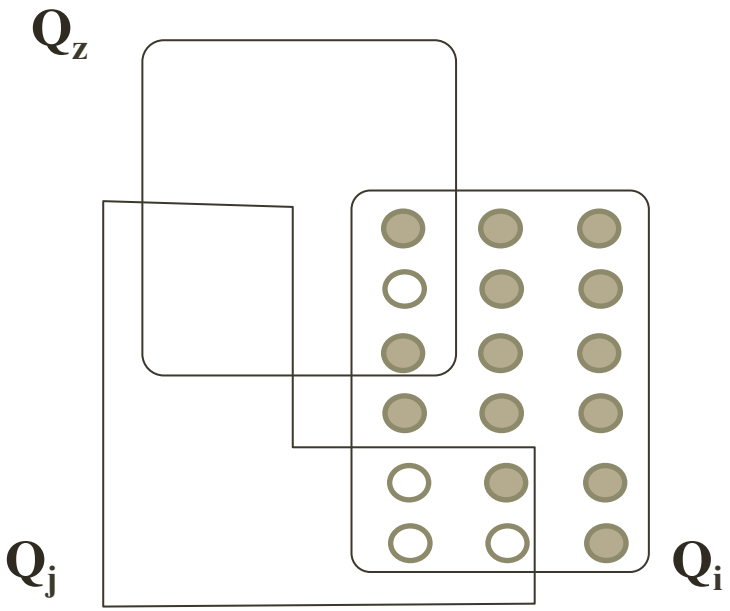


(Πιθανότατα) Η Εγγραφή έχει Τερματιστεί

Προσδιοριστέα Εγγραφή - Qview(2)

- Κάθε τομή περιέχει ένα μέλος με χρονοσφραγίδα μικρότερη από τη μέγιστη

$$[qView(2)]: \forall j \neq i, \exists A \subseteq Q_i \cap Q_j \; s.t. \; A \neq \varnothing \text{ and } \forall s \in A : s.ts < \max TS$$

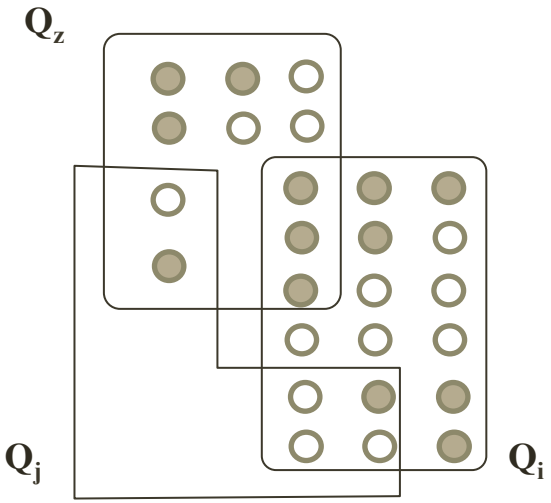


(Σίγουρα) Η Εγγραφή <maxTag,v> ΔΕΝ τερματίστηκε

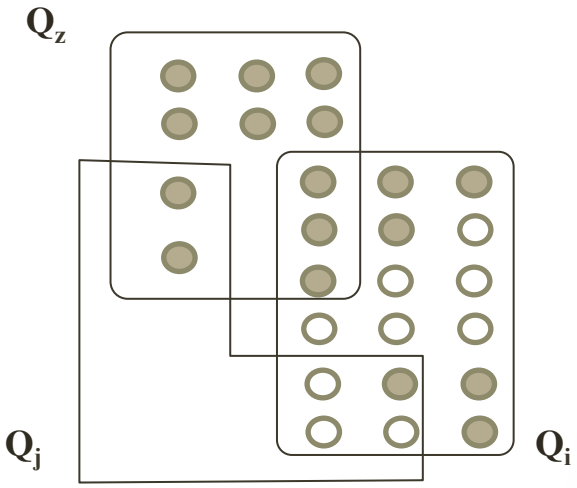
Απροσδιόριστη Εγγραφή - Qview(3)

- Όλα τα μέλη της τομής της απαρτίας που μας απάντησε και κάποιας άλλης απαρτίας στο σύστημα περιέχουν τη μέγιστη χρονοσφραγίδα.

$$[qView(3)]: \exists s \in Q_i \; s.t. \; s.ts < \max Ts \;\; and \;\; \exists j \neq i \; s.t. \forall s \in Q_i \cap Q_j : s.ts = \max TS$$



qV(3) και Μη Τερματισμένη Εγγραφή



qV(3) και Τερματισμένη Εγγραφή

Απροσδιόριστη => Δύο Φάσεις

Αλγόριθμος SLIQ

Write Protocol: one round

- P1: Αύξησε την χρονοσφραγίδα σου ts και στείλε μήνυμα write(<ts,v>) σε όλους και περίμενε απαντήσεις από μια απαρτία
  - Μόλις λάβεις απαντήσεις επέστρεψε ack και τερμάτισε

Read Potocol: one or two rounds

- P1: Στείλε μήνυμα read(<ts,v>) όπου <ts,v> η τελευταία τιμή που επέστρεψε ο αναγνώστης σε όλα τα αντίγραφα και περίμενε απάντηση από μια απαρτία Q
  - $QView_Q(1)$ – **Γρήγορη** και επέστρεψε την τιμή του maxTS
  - $QView_Q(2)$ – **Γρήγορη** και επέστρεψε την τιμή του maxTS-1
  - $QView_Q(3)$ – **Αργή** προχώρα στη φάση P2 και ακολούθως επεστρεψε maxTS
- P2: προώθησε <maxTS,v> σε μια απαρτία και μετά επέστρεψε v

Server Protocol: passive role

- Παρέλαβε αιτήσεις read(<ts,v>) και write(<ts,v>)
- Εαν local.ts<msg.ts ενημέρωσε το τοπικό σου αντίγραφο
- Απάντησε με reply(<ts,v>)

17/3/2011

Ορθότητα Αλγορίθμου

- **Ζωτικότητα:** Σύμφωνα με το μοντέλο σφαλμάτων πάντα υπάρχει μια απαρτία να απαντήσει σε κάθε λειτουργία

- **Ατομικότητα**: Πρέπει να δείξουμε ότι
  - Κάθε Ανάγνωση επιστρέφει τουλάχιστον την τιμή που γράφτηκε από την τελευταία Εγγραφή (Νόμιμη Ακολουθία Λειτουργιών)
  - Αν μία Ανάγνωση ρ1 προηγείται μιας ανάγνωσης ρ2, τότε η ρ2 επιστρέφει την ίδια ή νεότερη τιμή από αυτή που επιστρέφει η ρ1.

Μονοτονικότητα Χρονοσφραγίδας

- **Λήμμα 1**: Αν κάποιος εξυπηρετητής λάβει μήνυμα με <ts,v> τότε κάθε μεταγενέστερη απάνησή του περιλαμβάνει ζεύγος <ts', v'> τ.ω. ts' ≥ ts.

- Απόδειξη:
  - Κάθε φορά που ένας εξυπηρετητής παραλάβει μήνυμα read(<ts,v>) και write(<ts,v>) κάνει τον εξής έλεγχο
    - Αν local.ts<msg.ts τότε local.ts=msg.ts και local.v = msg.v
    - Αλλιώς δεν κάνει τίποτα
  - Αρα μετά την παραλαβή μηνύματος local.ts ≥ msg.ts
  - Επομένως σε κάθε reply(<ts', v'>) που έπεται οποιασδήποτε από τις πιο πάνω παραλαβές ισχύει ότι ts'=local.ts ≥ msg.ts =ts

Ορθότητα Αλγορίθμου

- **Λήμμα 2**: Αν μια Ανάγνωση ρ1 έπεται μιας Εγγραφής τότε επιστρέφει τουλάχιστον την τιμή που γράφτηκε

- Απόδειξη:
  - Κάθε εγγραφή επικοινωνεί με μια απαρτία πριν ολοκληρωθεί
  - Έστω w μια εγγραφή που έλαβε μηνύματα από την απαρτία Q_i και έγραψε την τιμή <ts, v>
  - Έστω ρ είναι μια ανάγνωση που έπεται της πιο πάνω εγγραφής και λαμβάνει απαντήσεις από την απαρτία Q_j
  - Έχουμε δύο περιπτώσεις: 1) $Q_i = Q_j$, και 2) $Q_i \neq Q_j$

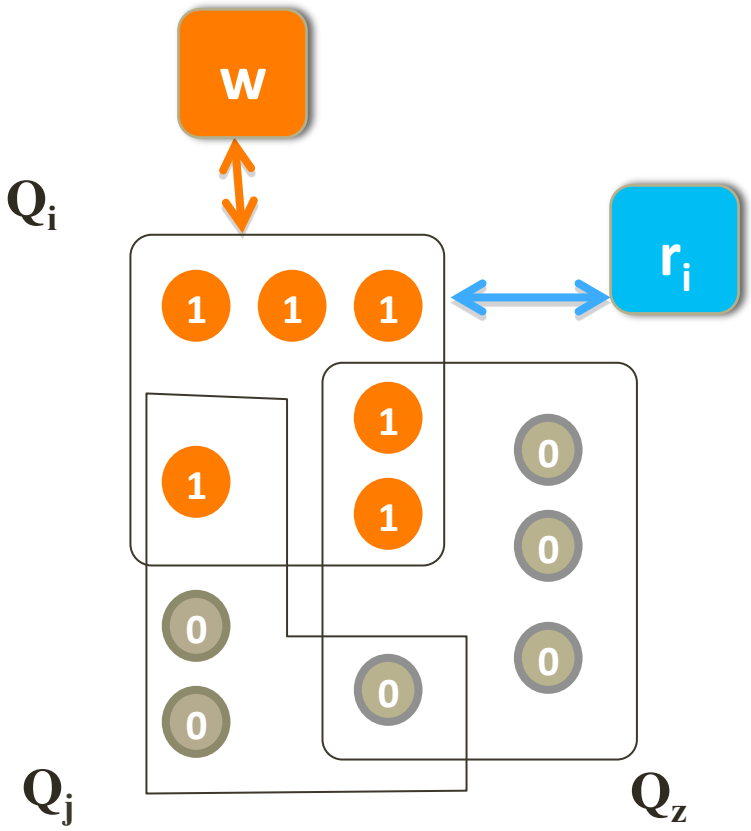
Λήμμα 2: Περίπτωση 1 ($Q_i = Q_j$)

- Όλοι οι εξυπηρετητές στο Q_i έλαβαν μήνυμα write(<ts, v>) από τον w πριν απαντήσουν στην ανάγνωση ρ
- Από το Λήμμα 1 θα στείλουν reply(<ts', v') στον ρ τ.ω. ts' ≥ ts
- Αν ts' = ts και maxTS = ts τότε ο ρ θα παρατηρήσει qView(1) και θα επιστρέψει ν
  - Όλοι απαντούν με την ίδια χρονοσφραγίδα ts
- Αν ts' > ts τότε maxTS = ts' και ο ρ θα επιστρέψει τουλάχιστον την τιμή που αντιστοιχεί στην χρονοσφραγίδα maxTS-1 ≥ ts
  - Αρα νεότερη ή ίση με ν

Λήμμα 2: Περίπτωση 2 ($Q_i \neq Q_j$)

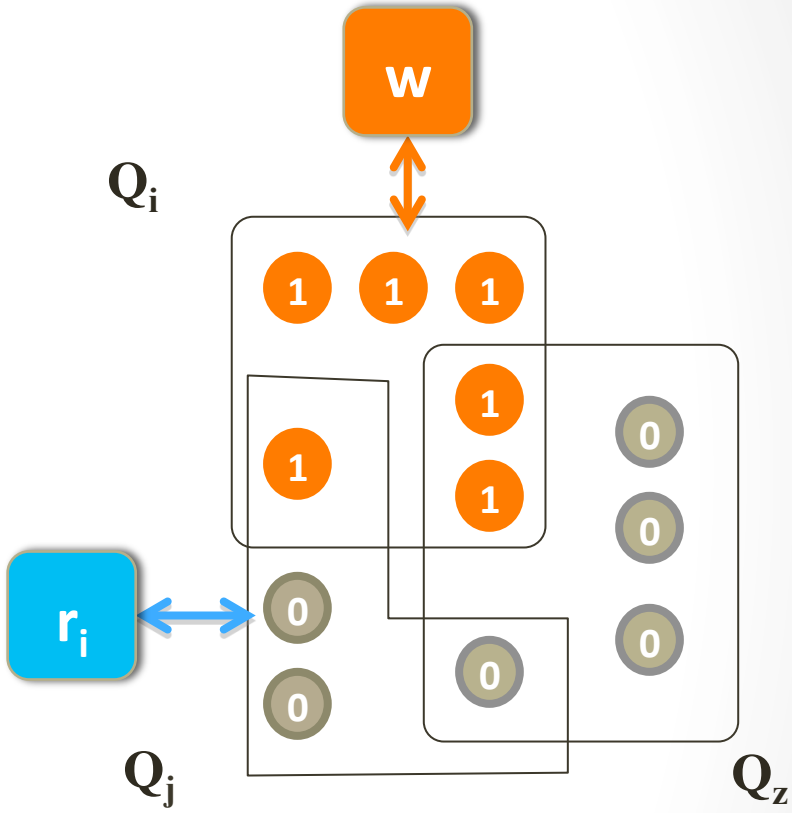
- Όλοι οι εξυπηρετητές στο $Q_i \cap Q_j$ έλαβαν μήνυμα write(<ts, v>) από τον w πριν απαντήσουν στην ανάγνωση ρ
- Από το Λήμμα 1 θα στείλουν reply(<ts', v') στον ρ τ.ω. ts' ≥ ts
- Αν ts' = ts και maxTS = ts τότε ο ρ θα παρατηρήσει qView(3) και θα επιστρέψει ν αφού πρώτα εκτελέσει $2^{η}$ φάση
  - Όλοι σε μια τομή απαντούν με την ίδια χρονοσφραγίδα ts
- Αν ts' > ts τότε maxTS = ts' και ο ρ θα επιστρέψει τουλάχιστον την τιμή που αντιστοιχεί στην χρονοσφραγίδα maxTS-1 ≥ ts
  - Αρα νεότερη ή ίση με ν

Σχηματικά



$Q_i = Q_j$ => qView(1)

Returns maxTS=1 in one round



$Q_i \neq Q_j$ => qView(3)

Returns maxTS=1 in two rounds

Ορθότητα Αλγορίθμου

- **Λήμμα 3**: Αν μία Ανάγνωση ρ1 προηγείται μιας ανάγνωσης ρ2, τότε η ρ2 επιστρέφει την ίδια ή νεότερη τιμή από αυτή που επιστρέφει η ρ1

- Απόδειξη: Έστω ότι
  - ρ1 διάβασε από Q_i και επέστρεψε <ts, v> και
  - ρ2 διάβασε από Q_j και επέστρεψε <ts', v'>

  - Και οι δύο αναγνώσεις προέρχονται από τον ίδιο αναγνώστη
    - Τότε ts' ≥ ts. (Γιατί;)

  - Αν οι δύο αναγνώσεις προέρχονται από 2 διαφορετικούς αναγνώστες πρέπει να μελετήσουμε τις περιπτώσεις όπου: 1) ρ1 αργή , και 2) ρ1 γρήγορη

Λήμμα 3: Περίπτωση 1 (ρ1 αργή)

- Αν ρ1 είναι αργή σημαίνει ότι πριν ολοκληρωθεί στέλνει read(<ts,v>) στα μέλη μιας απαρτίας (έστω Q_z)
- Από το Λήμμα 1 όλοι οι εξυπηρετητές στη $Q_j \cap Q_z$ θα στείλουν reply(<ts', v') στον ρ τ.ω. ts' ≥ ts
- 'Αρα ο ρ2 θα παρατηρήσει maxTS ≥ ts' με το τέλος του πρώτου γύρου
- Αν maxTS>ts τότε στην χειρότερη περίπτωση ο ρ2 θα επιστρέψει την τιμή που αντιστοιχεί στο maxTS-1 ≥ ts
- Αν maxTS=ts τότε ο ρ2 θα επιστρέψει <maxTS, v>=<ts,v> αφού θα παρατηρήσει είτε
  - qView(1) if $Q_j = Q_z$, ή
  - qView(3) if $Q_j \neq Q_z$ αφού όλοι οι εξυπηρετητές στο $Q_j \cap Q_z$ στέλνουν reply(<ts,v>)

Λήμμα 3: Περίπτωση 2 (ρ1 γρήγορη)

- Αν ρ1 είναι γρήγορη σημαίνει στον πρώτο της γύρο προσεξε
  - 2.1) qView(1) και επέστρεψε maxTS=ts
  - 2.2) qView(2) και επέστρεψε maxTS-1=ts

- Περίπτωση 2.1
  - Παρατήρησε qView(1) => όλα τα μέλη του Q_i απάντησαν με <ts,v>
  - Επομένως από το Λήμμα 1 όλα τα μέλη της $Q_i \cap Q_j$ θα στείλουν reply(<ts', v') στον ρ τ.ω. ts' ≥ ts
  - 'Αρα ο ρ2 θα παρατηρήσει maxTS ≥ ts' με το τέλος του πρώτου γύρου
  - Με την ίδια λογική όπως και στην περίπτωση 1 μπορούμε να δείξουμε ότι ο ρ2 επιστρέφει <ts',v'> s.t. ts' ≥ ts

Λήμμα 3: Περίπτωση 2 (ρ1 γρήγορη)

- Περίπτωση 2.2
  - Παρατήρησε qView(3) => ο ρ1 επέστρεψε maxTS-1=ts
  - Κάθε διεργασία τελειώνει μια λειτουργία πριν μεταβεί στην επόμενη.
  - Επομένως αφού
    - ο ρ1 παρατήρησε την χρονοσφραγίδα maxTS στο σύστημα
    - ο εγγραφέας χρησιμοποιεί μια χρονοσραγίδα ανα εγγραφή,
    - Σε κάθε εγγραφή αυξάνεται η χρονοσφραγίδα κατά 1, και
    - έχουμε μόνο ένα εγγραφέα

    έπεται ότι η εγγραφή που χρησιμοποίησε τη χρονοσφραγίδα maxTS-1 έχει ήδη ολοκληρωθεί πριν ή κατα τη διάρκεια του ρ1.
  - Αφού το ρ1 προηγείται της ρ2 επομένως και η εγγραφή με maxTS-1 προηγείται της ρ2
  - Επομένως από το Λήμμα 2 το ρ2 θα επιστρέψει ts’≥maxTS-1 => ts’ ≥ ts.

Ερωτήσεις;