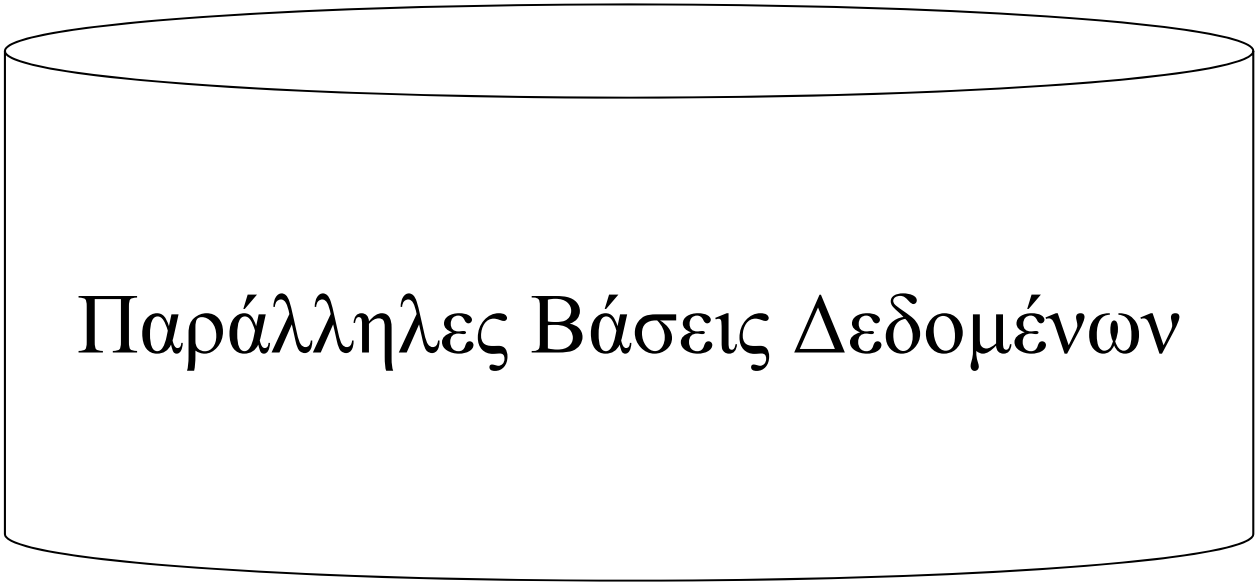


- Εισαγωγικά στοιχεία για παραλληλισμό και ΒΔ
- Μοντέλα και αρχιτεκτονικές παραλληλισμού
- Διαμερισμός δεδομένων
- Παράλληλη επεξεργασία ερωτημάτων

Πρόβλημα - κίνητρο

- Οι ΒΔ γίνονται όλο και πιο μεγάλες.
  - Τα μεγέθη πλέον μετρούνται σε TB (10^3 GB) και PB (10^6 GB).
  - Όλο και περισσοτερα δεδομένα δημοσιεύονται στον Παγκόσμιο Ιστό.
  - Νέα επιστημονικά πειράματα παράγουν τεράστιους όγκους δεδομένων.
  - Δεδομένα συναλλαγών συλλέγονται και αποθηκεύονται για μελλοντική ανάλυση.
  - Όλο και περισσότερα πολυμεσικά δεδομένα αποθηκεύονται.
- Αντίστοιχα αυξάνουν και οι απαιτήσεις των χρηστών.
- Μεγάλα μεγέθη δεδομένων $\Rightarrow$ χρήση δίσκου ως αποθηκευτικού χώρου και μεγάλης μνήμης, πρόβλημα απόδοσης.
- Η απόδοση περιορίζεται κυρίως από τις πράξεις I/O (bottleneck)
  - Ταχύτητα δίσκου << Ταχύτητα RAM << Ταχύτητα CPU
  - Υπάρχουν όμως περιπτώσεις που η ταχύτητα CPU (και η μνήμη) είναι καθοριστικός παράγοντας.

Βασικό Σκεπτικό

- Το βασικό σκεπτικό είναι να αυξηθούν οι διαθέσιμοι πόροι ενός ΣΒΔ μέσω παράλληλης αρχιτεκτονικής και επεξεργασίας ερωτημάτων.
  - Για λόγους απόδοσης αλλά και διαθεσιμότητας/αξιοπιστίας.
  - Το σχεσιακό μοντέλο έχει παίξει σημαντικό ρόλο.
- Τα παράλληλα μηχανήματα είναι πλέον συνηθισμένα και προσιτά σε τιμή.
  - Η τιμή των μικροεπεξεργαστών, της μνήμης και του αποθηκευτικού σώρου έχουν μειωθεί απότομα.
  - Δεν απαιτείται εξειδικευμένο υλικό.
- Η χρήση ευρείας κλίμακας παράλληλων συστημάτων ΒΔ συνεχώς αυξάνεται:
  - Για την αποθήκευση μεγάλων ποσοτήτων δεδομένων.
  - Για την επεξεργασία χρονοβόρων ερωτημάτων.
  - Για την επεξεργασία πολλών ταυτόχρονων συναλλαγών (high throughput) .

Σύντομη ιστορική αναδρομή

- Οι πρώτες προσπάθειες στόχευαν στην δημιουργία ειδικού υλικού.
- Χονδρικά, τοποθετούνται χρονικά μέχρι τα μέσα της δεκαετίας του ‘80.
- Όλες σχεδόν απέτυχαν.
- Επόμενες προσπάθειες που χρησιμοποιούσαν κοινό υλικό στέφθηκαν με επιτυχία.
- Πολλά επιτυχημένα παράλληλα ΣΔΒΔ (ιδίως από τη δεκαετία του ’90 μέχρι σήμερα)

Παραλληλισμός σε ΒΔ

- Τα δεδομένα μπορούν να αποθηκευτούν σε πολλαπλούς δίσκους για παράλληλες πράξεις I/O.
- Οι σχεσιακοί τελεστές είναι εύκολα παραλληλοποιήσιμοι.
  - Ο κάθε επεξεργαστής μπορεί να επεξεργάζεται δεδομένα ανεξάρτητα από τους άλλους.
- Τα ερωτήματα δεν αλλάζουν, όσον αφορά τη σύνταξη.
  - Εκφράζονται δηλ. σε SQL.
  - Ο βελτιστοποιητής και ο επεξεργαστής είναι υπεύθυνοι για παραλληλισμό.
  - Πολύ μεγάλη διευκόλυνση για τον χρήστη.
- Επίσης, διαφορετικά ερωτήματα μπορούν και αυτά να εκτελούνται παράλληλα.
  - Απαιτείται πιο περίπλοκος έλεγχος ταυτοχρονισμού.

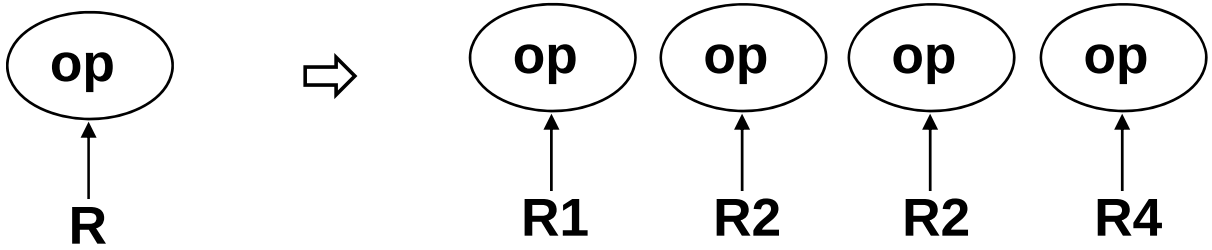
Τύποι Παραλληλισμού

Δύο βασικές –αλληλοσυμπληρωτματικές- κατηγορίες:

1. Παραλληλισμός ανάμεσα σε πολλαπλά ερωτήματα (inter-query parallelism).
   – Βασικός στόχος η αύξηση του ρυθμού ολοκλήρωσης συναλλαγών.
2. Παραλληλισμός σε ένα ερώτημα (intra-query parallelism).
   – Βασικός στόχος η επιτάχυνση της εκτέλεσης του ερωτήματος.
   – Τρεις διαφορετικές μορφές:
     - Intra-operator
     - Inter-operator
     - Ανεξάρτητος

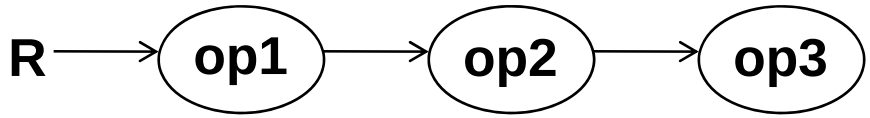
Παραλληλισμός ενός τελεστή

- Ένας τελεστής (ή μια ομάδα τελεστών) του πλάνου εκτέλεσης αντιγράφεται πολλαπλές φορές.
- Κάθε ένα στιγμιότυπο αυτού του τελεστή εκτελείται
  - Σε διαφορετικό επεξεργαστή.
  - Σε διαφορετικό τμήμα (partition) των δεδομένων προς εκτέλεση.
- Η εφαρμογή του μπορεί να επιφέρει τη μεγαλύτερη αύξηση της απόδοσης συγκριτικά με τις άλλες δύο μορφές παραλληλισμού ενός ερωτήματος.
- Ονομάζεται intra-operator ή διαμερισμένος (partitioned) παραλληλισμός



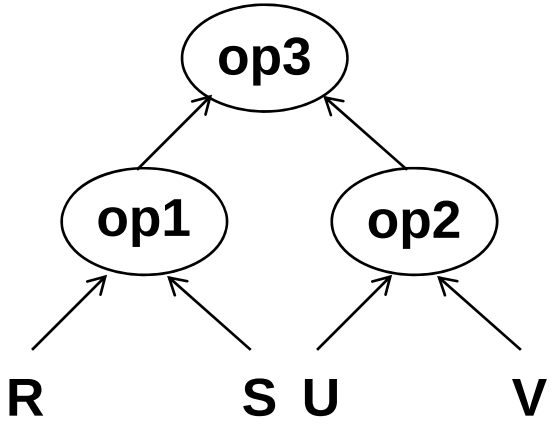
Παραλληλισμός ανάμεσα σε τελεστές

- Πολλοί συνδεδεμένοι τελεστές του πλάνου εκτέλεσης εκτελούνται παράλληλα.
- Ονομάζεται inter-operator ή pipelined παραλληλισμός.
- Το αποτέλεσμα ενός τελεστή δίνεται κατευθείαν ως είσοδο στον επόμενο τελεστή στο πλάνο εκτέλεσης.
- Εξαρτάται και από την υλοποίηση του λογικού τελεστή.
- Δεν επιφέρει τόσο μεγάλες μειώσεις στο χρόνο εκτέλεσης
  - Κάποιοι τελεστές πρέπει να καταναλώσουν όλη την είσοδο πριν αρχίσουν να παράγουν πλειάδες εξόδου.
  - Καλύτερα να συνδυάζεται με διαμερισμένο παραλληλισμό.



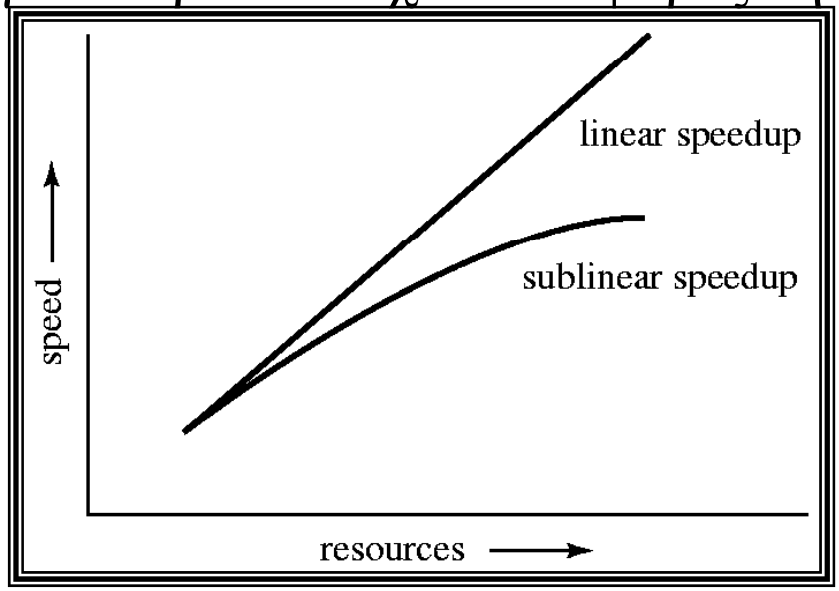
Ανεξάρτητος παραλληλισμός

- Εφαρμόζεται όταν το πλάνο εκτέλεσης του ερωτήματος, που αναπαριστάται ως κατευθυνόμενος γράφος, περιέχει υπογράφους ανεξάρτητους μεταξύ τους.
- Παράδειγμα: οι op1 και op2 στο παρακάτω πλάνο



Μετρικές - Επιτάχυνση

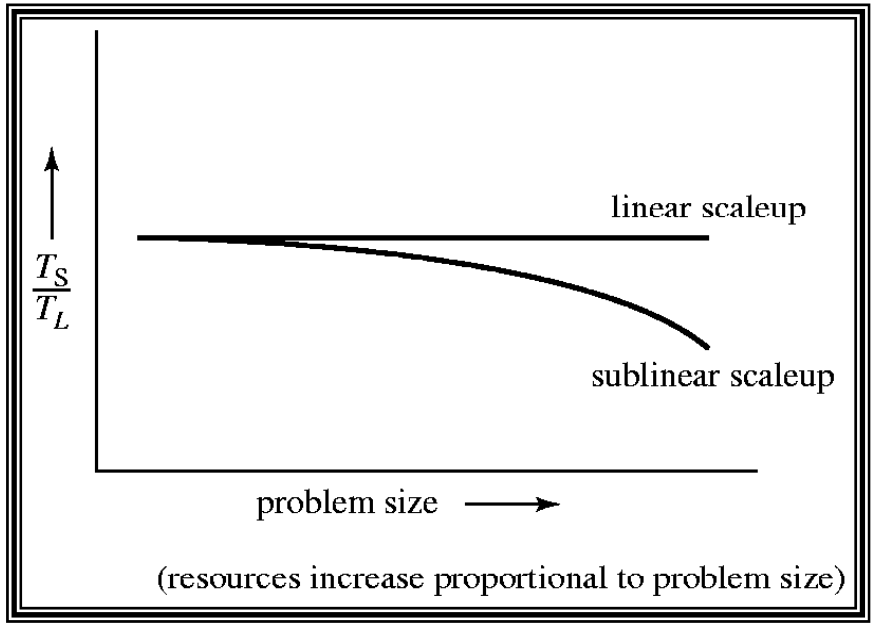
- Επιτάχυνση (speed-up).
  - Ο λόγος του χρόνου εκτέλεσης σε ένα μικρότερο/φθηνότερο σύστημα προς το χρόνο εκτέλεσης σε ένα μεγαλύτερο/ακριβότερο σύστημα.
  - Ιδανική κατάσταση: γραμμική επιτάχυνση, που σημαίνει ότι για το ίδιο πρόβλημα (μέγεθος δεδομένων) ένα σύστημα *N* φορές μεγαλύτερο επιταχύνει *N* φορές την εκτέλεση.



Μετρικές - Κλιμάκωση(scale-up)

- Ο λόγος του χρόνου εκτέλεσης ενός μικρότερου προβλήματος σε ένα μικρότερο/φθηνότερο σύστημα προς το χρόνο εκτέλεσης ενός μεγαλύτερου προβλήματος σε ένα μεγαλύτερο/ακριβότερο σύστημα.

- Ιδανική κατάσταση: γραμμική κλιμάκωση, που σημαίνει ότι μεγαλώνοντας το σύστημα N φορές και το μέγεθος του προβλήματος επίσης N φορές (N φορές περισσότερα δεδομένα), ο χρόνος για την εκτέλεση παραμένει σταθερός.



Περιπτώσεις Κλιμάκωσης

- Batch :
  - Αναφέρεται σε ένα μεγάλο ερώτημα, όπου απαιτούνται *N* περισσότεροι πόροι για επεξεργασία *N* περισσοτέρων δεδομένων.
- Συναλλαγές:
  - Αναφέρεται σε περιπτώσεις που πολλαπλοί χρήστες υποβάλλουν ταυτόχρονα ερωτήματα και οι πόροι του συστήματος αυξάνουν *N* φορές όταν αυξάνουν *N* φορές τόσο οι χρήστες όσο και το μέγεθος της ΒΔ.

Εμπόδια στον αποδοτικό παραλληλισμό

- Κόστος εκκίνησης (startup)
  - Το κόστος εκκίνησης μίας παράλληλης πράξης μπορεί να είναι μεγαλύτερο από τα οφέλη του παραλληλισμού.
- Παρεμβολές (interference)
  - Πολλές παράλληλες διεργασίες προσπαθούν να προσπελάσουν ταυτόχρονα κοινούς πόρους $\Rightarrow$ προξένηση καθυστερήσεων.
- Ανομοιόμορφη Κατανομή (skew)
  - Ο χρόνος ολοκλήρωσης μία παραλληλοποιημένης πράξεις είναι ο μεγαλύτερος από τους επιμέρους χρόνους ολοκλήρωσης.

Αρχιτεκτονικές

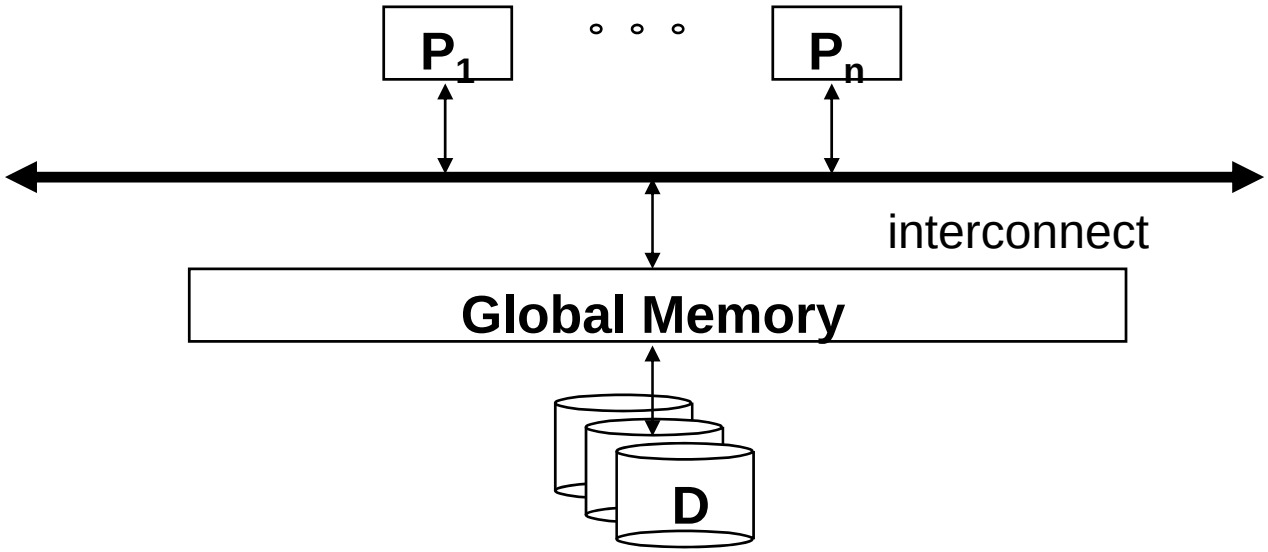
- Εναλλακτικές λύσεις:
  - Κοινόχρηστης μνήμης (όλοι οι πόροι κοινόχρηστοι) - Shared memory (shared everything)
  - Κοινόχρηστου δίσκου - Shared disk
  - Χωρίς κοινόχρηστους πόρους - Shared nothing (message-passing)
- Υβριδικές αρχιτεκτονικές (Hybrid architectures)
  - Ιεραρχική (cluster)
  - Non-Uniform Memory Architecture (NUMA)

- Ιδανικά, ένα παράλληλο ΣΔΒΔ έχει άπειρη ισχύ επεξεργασίας, άπειρη μνήμη και άπειρο εύρος δικτύου.
  - Ο στόχος των ΠΣΔΒΣ είναι να πλησιάσουν όσο το δυνατόν πιο πολύ στην ιδανική αυτή κατάσταση χρησιμοποιώντας μεγάλο αριθμό πόρων πεπερασμένων δυνατοτήτων.

Αρχιτεκτονική κοινόχρηστης μνήμης

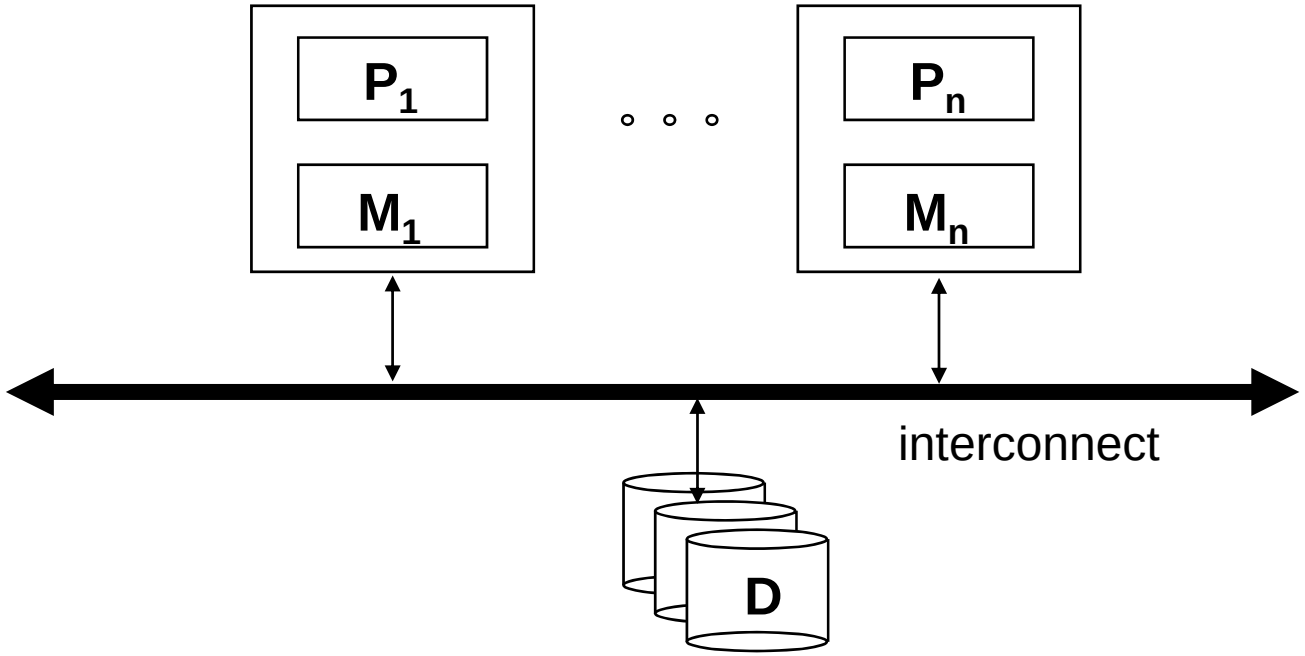
- Οι επεξεργαστές διαμοιράζονται μία κοινή μνήμη.
  + πολύ εύκολη επικοινωνία μεταξύ των επεξεργαστών.
  - όχι ικανοποιητική κλιμάκωση, πρόβλημα με παρεμβολές.

  Στην πράξη, μπορεί να χρησιμοποιηθεί αποδοτικά για μικρούς βαθμούς παραλληλισμού.



Αρχιτεκτονική κοινόχρηστου δίσκου

- Οι επεξεργαστές διαμοιράζονται ένα κοινό δίσκο.
  - \+ καλύτερη κλιμάκωση, αλλά και πάλι υπάρχει πρόβλημα με παρεμβολές.
  - \- λιγότερο εύκολη επικοινωνία μεταξύ των επεξεργαστών.

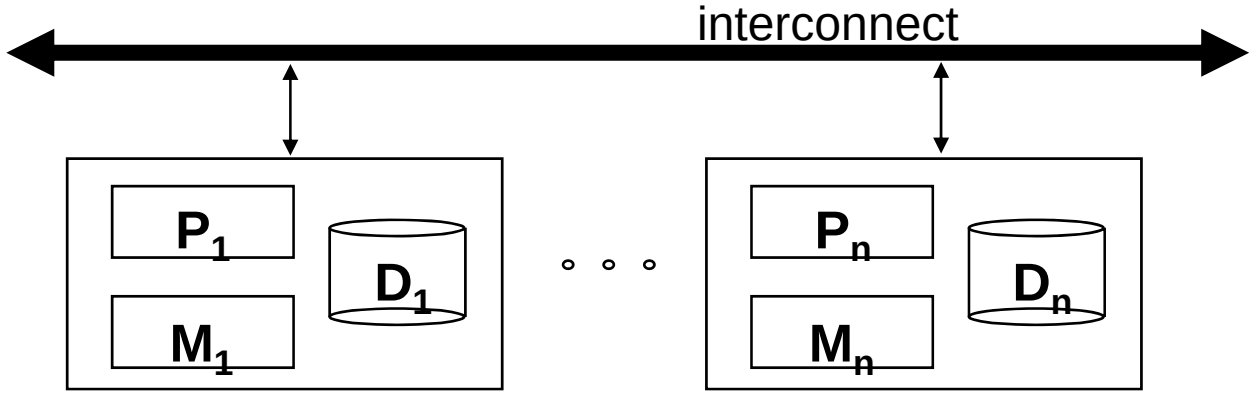


Αντίκτυπος παρεμβολών

- Έστω ότι σε ένα παράλληλο σύστημα που βασίζεται στο διαμοιρασμό πόρων, η προσθήκη ενός επιπλέον επεξεργαστή προξενεί 1% επιβράδυνση στους υπόλοιπους επεξεργαστές.
- Τότε, η μέγιστη επιτάχυνση είναι μικρότερη από 37 (για 100 επεξεργαστές).
- Αν είχαμε 1000 επεξεργαστές, η απόδοσή τους παράλληλλου συστήματος θα ήταν μόλις 4% ενός μη παράλληλου συστήματος!

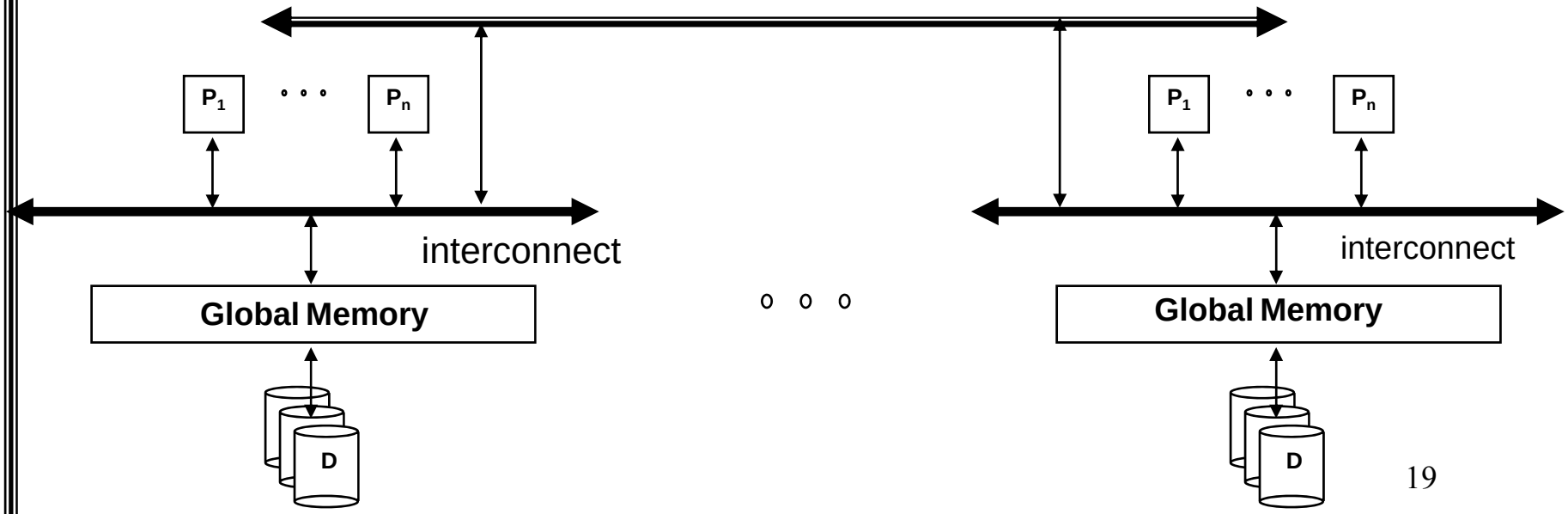
Αρχιτεκτονική χωρίς κοινόχρηστους πόρους

- Οι επεξεργαστές δεν διαμοιράζονται πόρους.
  - \+ εξαλείφεται το πρόβλημα της παρεμβολής.
  - \+ πολύ καλή κλιμάκωση, στις τάξεις των χιλιάδων επεξεργαστών.
  - \+ επίτευξη σχεδόν γραμμικής επιτάχυνσης και κλιμάκωσης.
  - \+ μπορεί να δημιουργηθεί με απλή διασύνδεση συνηθισμένων μηχανημάτων.
  - \- μεγαλύτερο κόστος στην επικοινωνία μεταξύ των επεξεργαστών, αλλά και λιγότερα μηνύματα (ουσιαστικά μεταφέρονται μόνο υποερωτήματα και ενδιάμεσα αποτελέσματα).
  - \- πιο δύσκολη η εξισορρόπηση φόρτου



Ιεραρχική Αρχιτεκτονική

- Συνδυασμός "shared nothing" και "shared memory".
- Στόχος ο συνδυασμός εύκολης εξισορρόπησης φόρτου και δυνατότητας κλιμάκωσης.
- Παρόμοιος είναι και ο στόχος των αρχιτεκτονικών NUMA (non-uniform memory access):
  - βασίζονται σε κοινό χώρο διευθύνσεων της κατανεμημένης μνήμης.



Που είμαστε

- Εισαγωγικά στοιχεία για παραλληλισμό και ΒΔ
- Μοντέλα και αρχιτεκτονικές παραλληλισμού
- **Διαμερισμός δεδομένων**
- Παράλληλη επεξεργασία ερωτημάτων

Τεχνικές

- Διαμερισμός και τοποθέτηση δεδομένων
  - Φυσική τοποθέτηση της ΒΔ σε πολλαπλούς κόμβους.
  - Στατικά ή Δυναμικά.
- Παράλληλη επεξεργασία
  - Εύκολα για τον τελεστή επιλογής.
  - Άλλοι τελεστές, όπως της σύνδεσης, απαιτούν μεγαλύτερη προσοχή.
- Βελτιστοποίηση παράλληλων ερωτημάτων
  - Πιο δύσκολη επιλογή πλάνου εκτέλεσης.
- Διαχείριση Συναλλαγών
- <u>Σημείωση:</u>

  Πολλά από αυτά τα θέματα είναι ίδια ή παρόμοια με θέματα που κλασικά θεωρούνται ότι ανήκουν στις *κατανεμημένες* ΒΔ.

Διαμερισμός Δεδομένων

- Κάθε σχέση διαιρείται σε *n* μέρη.
- Κάθε πλειάδα αντιστοιχείται σε ένα από αυτά τα μέρη.
- Το *n* μπορεί να είναι είτε ίσο με τον αριθμό των κόμβων (πλήρης διαμερισμός) είτε συνάρτηση μεταδεδομένων, όπως το μέγεθο της σχέσης, η συχνότητα προσπέλασης, κλπ.
- Τρόποι υλοποίησης
  - Round-robin: Π.χ. η πλειάδα *i* ανατίθεται στον επεξεργαστή (*i mod n)*
  - Διαμερισμός με κατακερματισμό: επιλέγονται κάποια πεδία, πάνω στα οποία εφαρμόζεται μία συνάρτηση κατακερματισμού για τον καθορισμό της ανάθεσης. Π.χ., hash_function: *value(A_i) mod n*
  - Διαμερισμός με διαστήματα τιμών: ταξινόμηση δεδομένων σύμφωνα με κάποια πεδία, και επιλογή διαστημάτων τιμών για τα πεδία αυτά.

Χαρακτηριστικά Τεχνικών Διαμερισμού

- Ερωτήματα που προσπελαύνουν ολόκληρη τη σχέση:
  - Ιδανικός ο round-robin, ο διαμερισμός με κατακερματισμό ή με διαστήματα τιμών επίσης αποδίδει καλά.
- Ερωτήματα που ικανοποιούν μία συνθήκη ισότητας
  - ο διαμερισμός με κατακερματισμό ή με διαστήματα τιμών υπερέχουν σαφώς του round-robin
  - Αν η συνάρτηση κατακερματισμού ή η επιλογή διαστημάτων έχει γίνει βάσει χαρακτηριστικών της συνθήκης.
- Ερωτήματα που ικανοποιούν μία γενική συνθήκη
  - ο διαμερισμός με διαστήματα τιμών υπερέχει του round-robin και του διαμερισμού με κατακερματισμό.

- Όμως ο διαμερισμός με διαστήματα τιμών (και με κατακερματισμό) είναι πιο ευάλωτος σε προβλήματα ανομοιόμορφης κατανομής.

Διαμερισμός Δεδομένων με αντίγραφα

- Για επίτευξη υψηλής διαθεσιμότητας χρειάζονται αντίγραφα (data replication).
- Απλές λύσεις
  - Πρωτεύον και δευτερεύον αντίγραφο δίσκου (mirrored disks)
  - Όταν ένας κόμβος αποτύχει, αυξάνεται ο φόρτος σε κάποιον άλλο κόμβο.
- Πιο σύνθετες λύσεις διαμερισμού:
  - Interleaved: το δευτερεύον αντίγραφο διαμερίζεται μεταξύ όλων των υπολοίπων κόμβων. Υπάρχει όμως σημαντικό πρόβλημα αν δύο κόμβοι αποτύχουν.
  - Αλυσιδωτός (chained): το δευτερεύον αντίγραφο τοποθετείται σε διπλανό κόμβο του πρωτεύοντος, επειδή είναι λιγότερο πιθανό δύο διπλανοί κόμβοι να αποτύχουν απ' ότι δύο οποιοιδήποτε κόμβοι να αποτύχουν.

Διαμερισμός Δεδομένων με αντίγραφα

Interleaved (πάνω) και αλυσιδωτός (κάτω) διαμερισμός

Node	1	2	3	4
Primary copy	R1	R2	R3	R4
Backup copy	r 2.3 r 3.2 r 4.1	r 1.1 r 3.3 r 4.2	r 1.2 r 2.1 r4.3	r 1.3 r 2.2 r 3.1

Node	1	2	3	4
Primary copy	R1	R2	R3	R4
Backup copy	r4	r1	r2	r3

Παραλληλισμός τελεστή επιλογής

- Ας υποθέσουμε ότι υπάρχουν p επεξεργαστές, ο καθένας από τους οποίους έχει τον δικό του δίσκο.
- Η αρχιτεκτονική είναι χωρίς κοινόχρηστους πόρους.
- Για κάθε σχέση R, όλες οι πλειάδες διαμερίζονται σε όλους τους επεξεργαστές.
- Θέλουμε να εκτελέσουμε την πράξη $\sigma_C (R)$, στοχεύοντας στην ελαχιστοποίηση του χρόνου εκτέλεσης.
- Αυτό επιτυγχάνεται αν η σχέση έχει διαμεριστεί σε ίσα κομμάτια και σε όλους τους επεξεργαστές (τα αποτελέσματα αποθηκεύονται τοπικά).
- Κάθε κόμβος εκτελεί τοπικά τον τελεστή επιλογής πάνω στα δεδομένα του δίσκου του.
- Παρόμοια εκτελείται και η πράξη $\pi_L (R)$.

Παραλληλισμός επιλογής - αποτελέσματα

- Η πράξη σ_C *(R),* σε αντίθεση με την π_L *(R)* μπορεί να οδηγήσει σε αλλαγή της κατανομής των πλειάδων στους επεξεργαστές.
- Αυτό είναι σημαντικό αν υπάρχουν και άλλοι τελεστές στο πλάνο εκτέλεσης μετά την επιλογή.
- Π.χ., έστω ότι *C: attribute_A=10.*
- Αν ο διαμερισμός έχει γίνει με κατακερματισμό ή με διαστήματα τιμών βάσει του *attribute_A,* τότε όλο το αποτέλεσμα θα είναι σε ένα μόνο από τους *p* επεξεργαστές.
- Αυτό το πρόβλημα αποφεύγεται αν οι τέτοιου τύπου διαμερισμοί χρησιμοποιούν όλα τα πεδία μίας σχέσης.

Παραλληλισμός διαγραφής διπλών πλειάδων

- Έστω η πράξη διαγραφής διπλών τιμών *δ(R)*.
- Αν έχουμε εφαρμόσει διαμερισμό με κατακερματισμό πάνω σε όλα τα πεδία, τότε όλες οι διπλές πλειάδες βρίσκονται στον ίδιο κόμβο.
- Συνεπώς, μπορούμε να εφαρμόσουμε τοπικά τον αλγόριθμο διαγραφής διπλών πλειάδων και το συνολικό αποτέλεσμα θα είναι σωστό.

Παραλληλισμός ένωσης (μαζί με διαγραφή διπλοτύπων)

- Αν οι δύο σχέσεις που ενώνονται είναι διαμερισμένες με την ίδια συνάρτηση κατακερματισμού, τότε η ένωση εκτελείται από κάθε επεξεργαστή παράλληλα πάνω στα τοπικά δεδομένα.
- Αν δεν είναι, τότε πρέπει να αντιγραφούν τα δεδομένα και να διαμεριστούν (έστω και προσωρινά) σύμφωνα με μία κοινή συνάρτηση κατακερματισμού.
- Έτσι ο τελεστής ένωσης μπορεί να εφαρμοστεί τοπικά σε όλους τους επεξεργαστές.
- Επίσης με μία καλή συνάρτηση κατακερματισμού αποφεύγονται στις περισσότερες περιπτώσεις προβλήματα ανισορροπίας φόρτου.
- Ακριβώς με τον ίδιο τρόπο εκτελούνται οι πράξεις διαφοράς και τομής.

Παραλληλισμός group-by

- Εφαρμόζουμε μία συνάρτηση κατακερματισμού πάνω στις ιδιότητες ομαδοποίησης, έτσι ώστε να προκύψει ένας κάδος για κάθε επεξεργαστή.
  →Κάθε κάδος περιέχει πολλές ομάδες.
  →Αλλά κάθε ομάδα είναι μόνο σε ένα κάδο.

- Αφού αποσταλούν οι πλειάδες ενός κάδου από όλους τους επεξεργαστές στον τελικό επεξεργαστή, εφαρμόζουμε όποιον αλγόριθμο ομαδοποίησης επιθυμούμε.

Παραλληλισμός σύνδεσης ισότητας

- $R(X,Y) \bowtie S(Y,Z)$ με τυχαίο αρχικό διαμερισμό.
- Εφαρμόζουμε μία συνάρτηση κατακερματισμού πάνω στο Y για όλες τις πλειάδες, έτσι ώστε να προκύψει ένας κάδος για κάθε επεξεργαστή.
- Αφού αποσταλούν οι πλειάδες ενός κάδου από όλους τους επεξεργαστές στον τελικό επεξεργαστή, εφαρμόζουμε όποιον αλγόριθμο σύνδεσης επιθυμούμε.
- Αν η συνθήκη σύνδεσης δεν είναι ισότητα, τότε η παραπάνω μέθοδος δεν δίνει σωστά αποτελέσματα.

Απόδοση μη παραλληλοποιημένης σύνδεσης

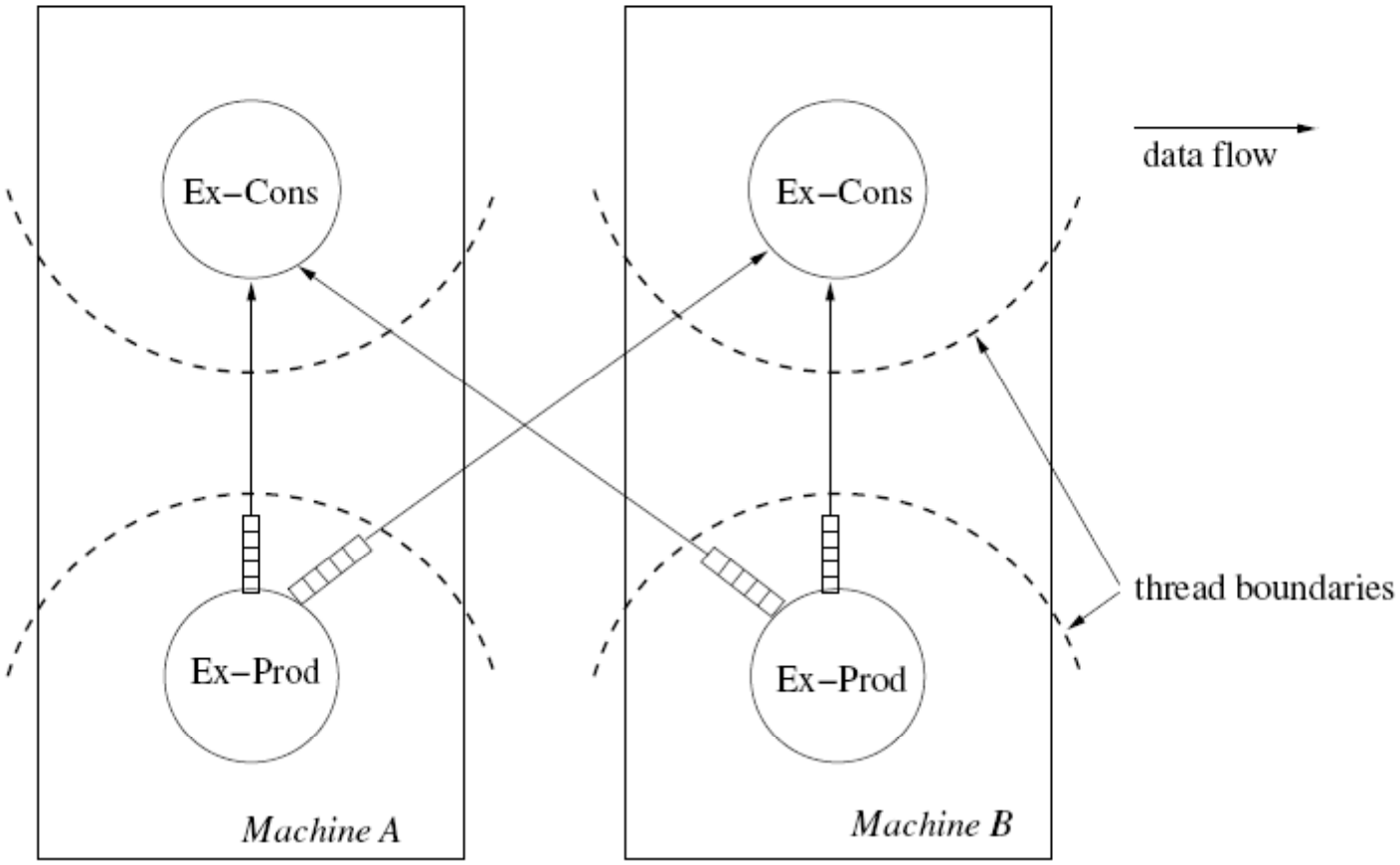
- **Μέθοδος κατακερματισμού – $X \bowtie Y$**
- Υποθέτουμε ότι δεν συμβαίνει υπερχείλιση σε κάποιο κάδο και ότι ο πίνακας κατακερματισμού αποθηκεύεται στην κύρια μνήμη.
- Το κόστος επεξεργασίας περιλαμβάνει:
  1. ανάγνωση των γραμμών των πινάκων,
  2. δημιουργία και αποθήκευση των κάδων στο δίσκο, και
  3. ανάγνωση των κάδων.
- Σύμφωνα με τα προηγούμενα, το κόστος είναι:

$$C_{J_5} = 3 \cdot (b_X + b_Y)$$

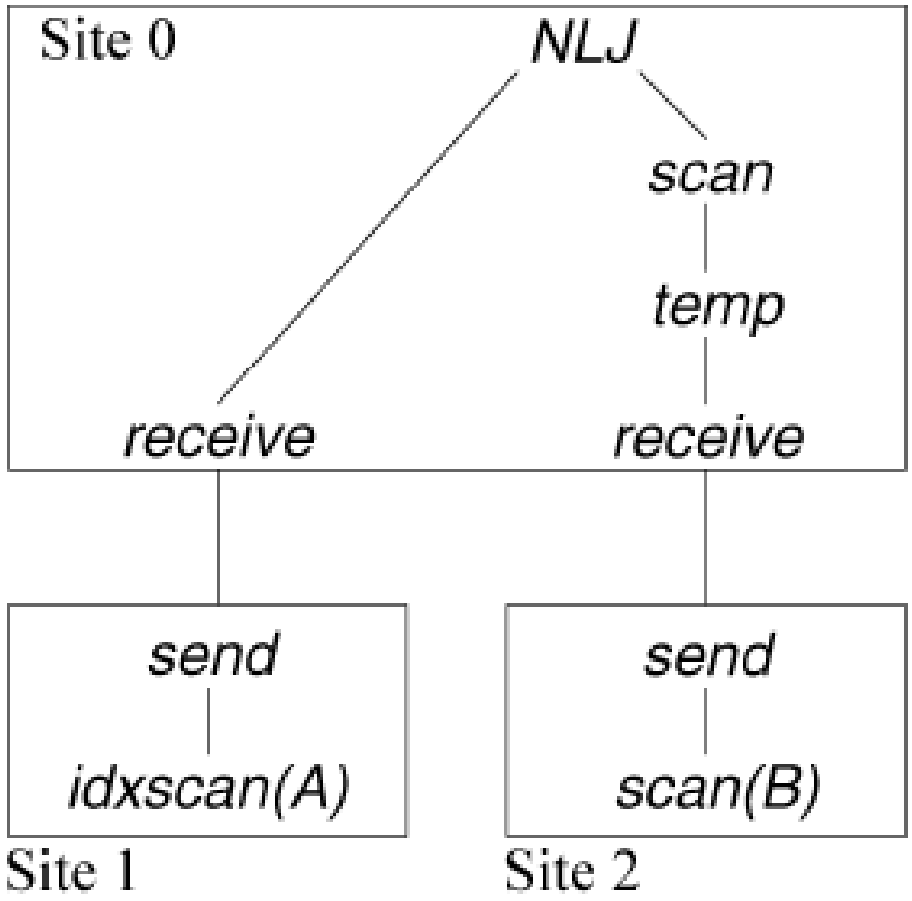
Απόδοση παραλληλισμού

- Έστω ότι το βασικό κόστος είναι το κόστος I/O.
- Το κόστος σε κάθε επεξεργαστή να διαβάσει τα τοπικά δεδομένα (για να εφαρμόσει τη συνάρτηση κατακερματισμού) είναι $(1/p)\ (B_X + B_Y)$.
- Άλλο τόσο είναι το κόστος ενός επεξεργαστή να αποθηκεύσει τοπικά τα δεδομένα που αποστέλλουν οι υπόλοιποι επεξεργαστές.
- Εφαρμόζοντας τοπικά τον αλγόριθμο σύνδεσης σε $(1/p)$ο κάθε φορά των δεδομένων, το συνολικό κόστος για ένα επεξεργαστή είναι:
  $(1/p)\ (B_X + B_Y) + (1/p)\ (B_X + B_Y) + 3\ (1/p)\ (B_X + B_Y)$
  $= 5\ (1/p)\ (B_X + B_Y)$
- Περαιτέρω μείωση του κόστους αν ο αλγόριθμος εκτελεστεί σε ένα πέρασμα και δεν χρειαστεί προσωρινή αποθήκευση.
- Προσοχή στην επαναχρησιμοποίηση των ίδιων συναρτήσεων κατακερματισμού
- Τα συνολικά δεδομένα που αποστέλλει ένας επεξεργαστής προς τους άλλους είναι $((p-1)/p)\ (B_X + B_Y)$.

Ο τελεστής ανταλλαγής (exchange)



Εναλλακτική μέθοδος



Θέματα βελτιστοποίησης

- Μοντέλα κόστους που λαμβάνουν υπόψιν τον παραλληλισμό.
  - Δεν εκτελούνται όλες οι πράξεις σειριακά.
- Μικρότερο κόστος δεν σημαίνει και συντομότερες απαντήσεις.
- Μεγαλύτερος χώρος αναζήτησης
  - Συμπεριλαμβάνονται και τα θαμνώδη δένδρα
- Πρόβλημα ανάθεσης και καθορισμού του βαθμού του παραλληλισμού.