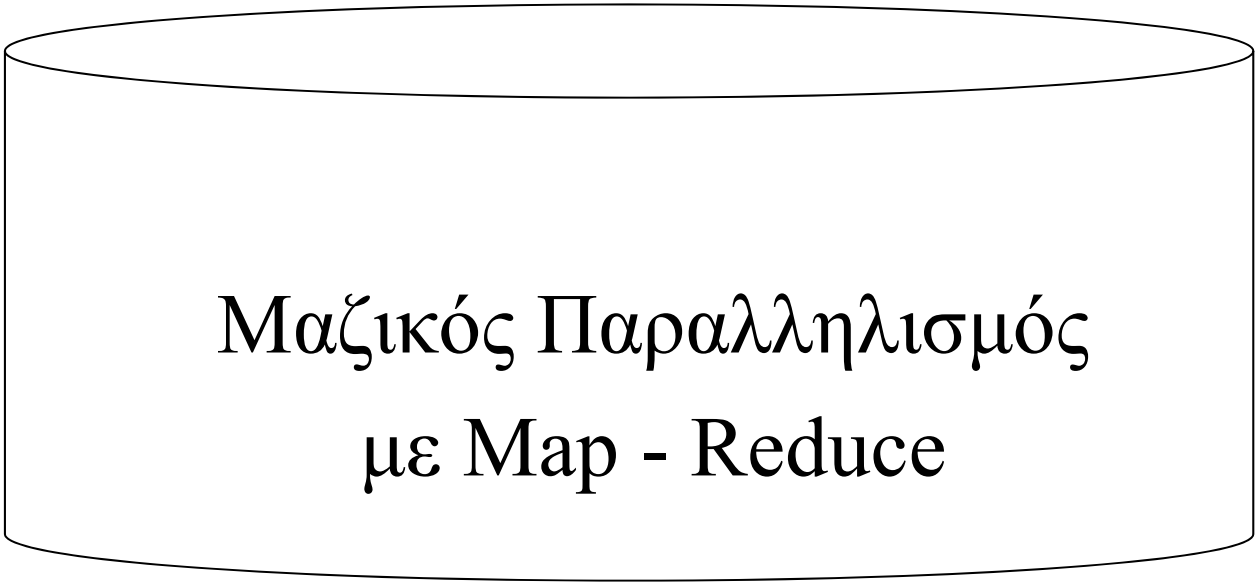


- Μοντέλο
- Θέματα υλοποίησης
- Παραδείγματα διαχείρισης δεδομένων

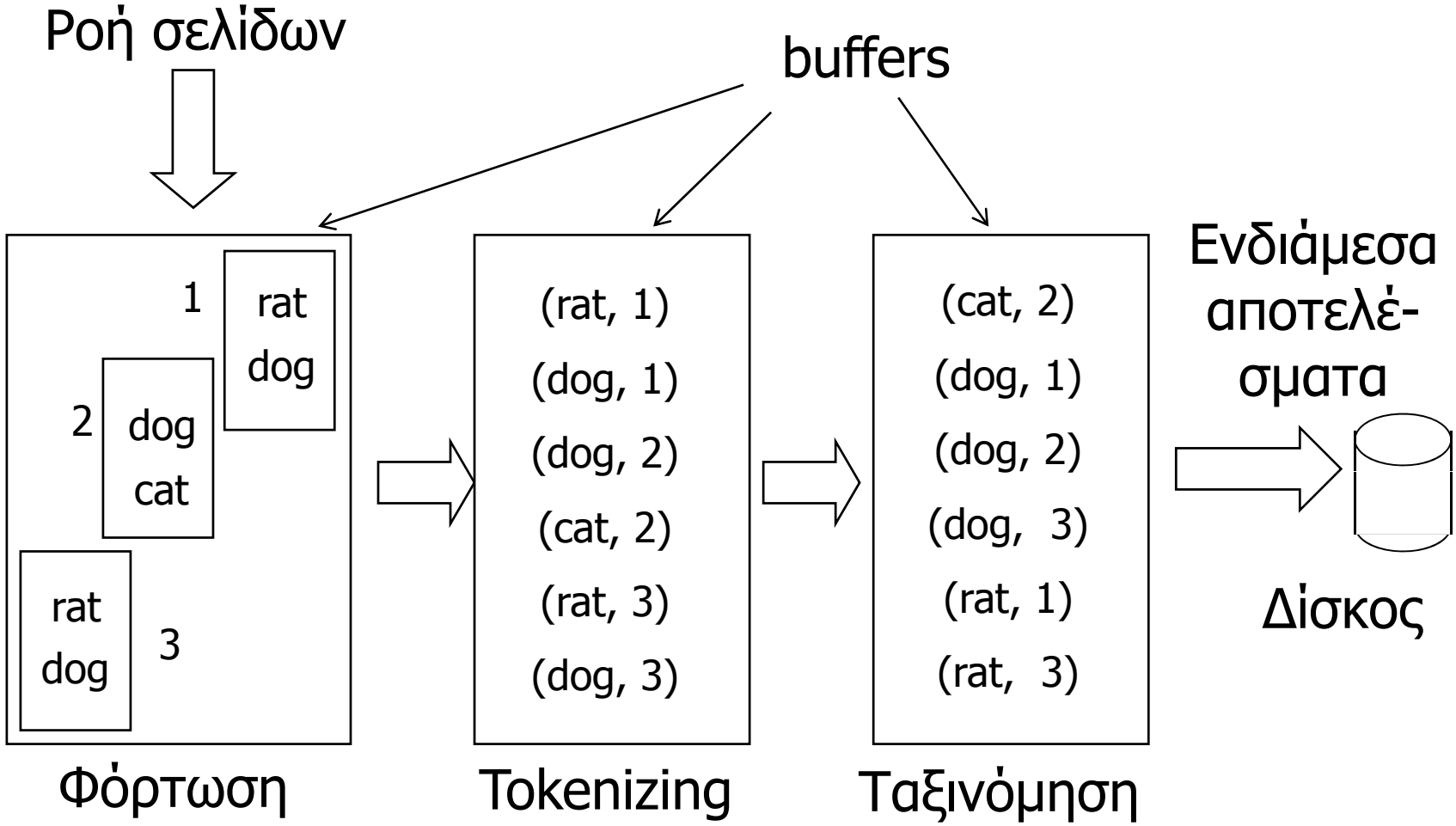
Ευχαριστίες

- Οι διαφάνειες στηρίζονται σε μεγάλο βαθμό στο υλικό που είναι διαθέσιμο από το εργαστήριο Infolab του Stanford για τα μαθήματα CS345A και CS347.
  http://infolab.stanford.edu

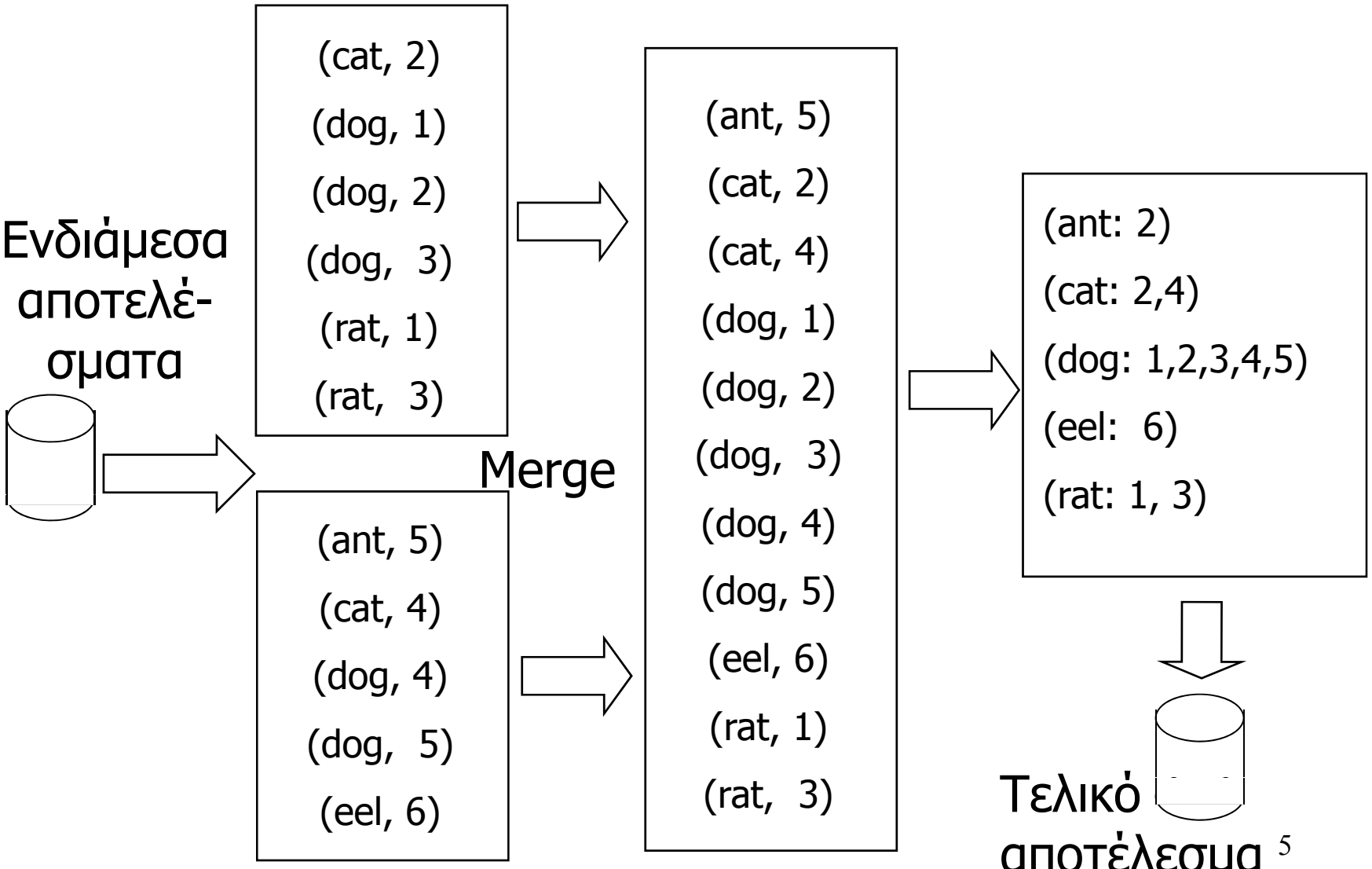
Τι είναι το MapReduce

- Πρόκειται για ένα προγραμματιστικό μοντέλο,
- που είναι κατάλληλο για επεξεργασία πολύ μεγάλων όγκων δεδομένων σε κατανεμημένους πόρους.
- Βασίζεται σε ιδέες γνωστές από δεκαετίες!
- Αρχικά αναπτύχθηκε από την Google αλλά οι περισσότεροι το χρησιμοποιούν μέσω της υλοποίησης ανοιχτού κώδικα Hadoop.

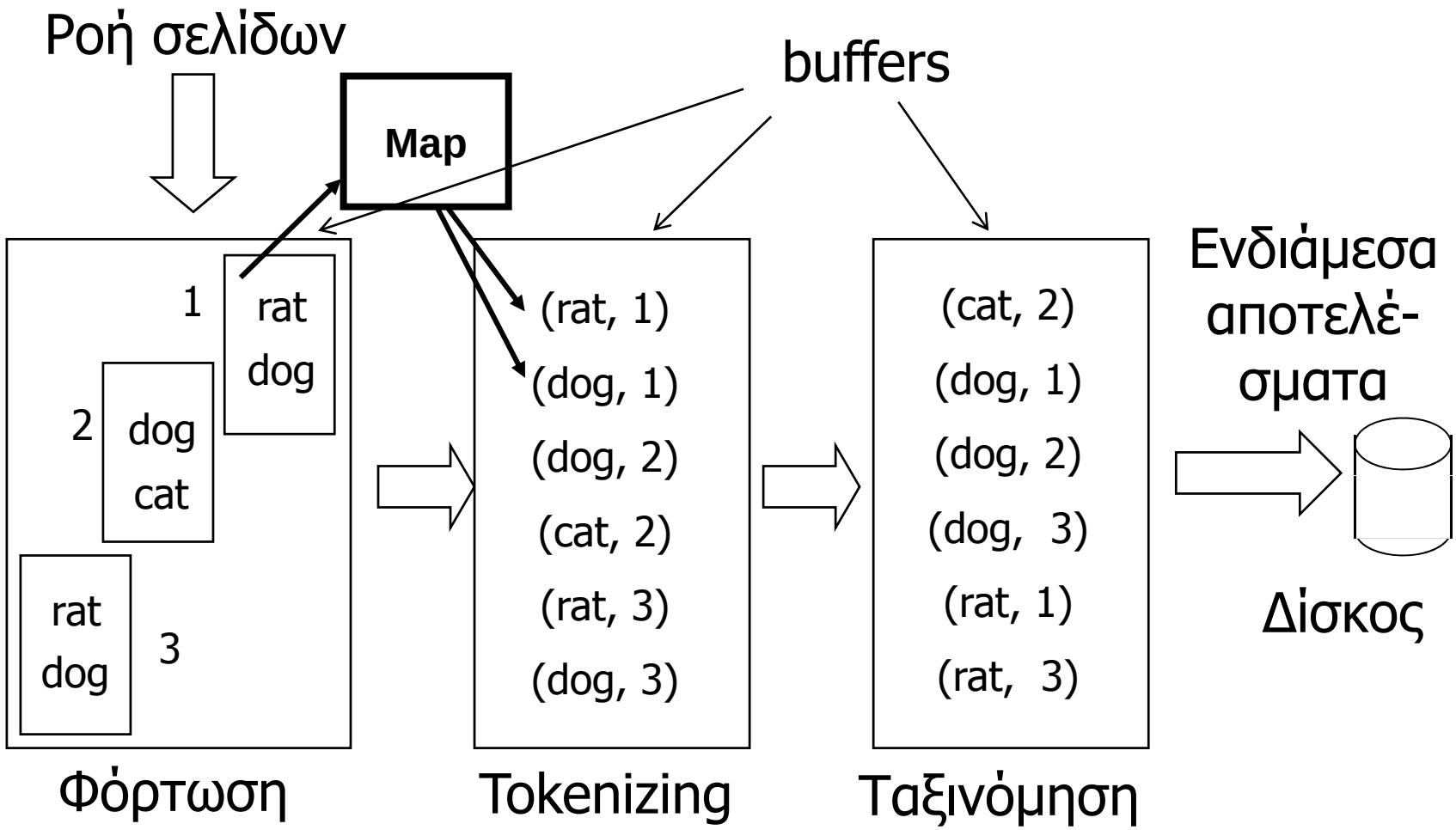
Δημιουργία Ευρετηρίου (index) κειμένου - I



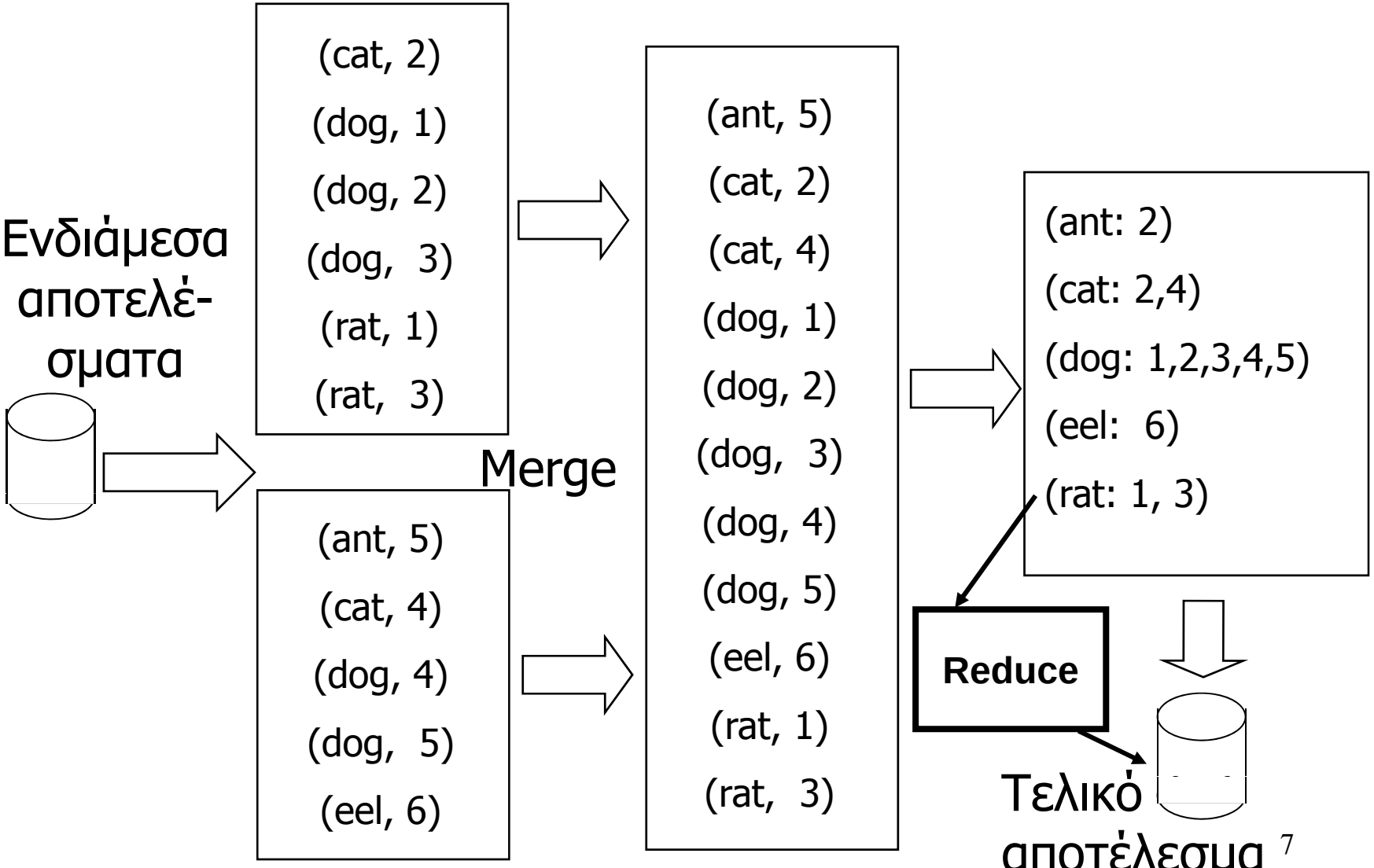
Δημιουργία Ευρετηρίου (index) κειμένου - II



Γενικό Μοντέλο επεξεργασίας: Map



Γενικό Μοντέλο επεξεργασίας: Reduce



Συνοπτική Περιγραφή

- map (κλειδί, τιμή) → λίστα από ζεύγη (κλειδί΄, τιμή΄)
- reduce (κλειδί΄, λίστα από τιμές΄) -> τελική_τιμή (list)

Ή αλλιώς:

- Map: (k1, v1) → [(k2, v2)]
- Reduce: (k2, [v2]) → [v3] //σε εκδόσεις Hadoop μπορεί να αλλάξει το κλειδί k2 σε k3.

-Το κλειδί δεν έχει εδώ την αυστηρή έννοια των ΒΔ, δηλ. μπορούν να υπάρχουν περισσότερες ενδιάμεσες τιμές με το ίδιο κλειδί.

Παράδειγμα: καταμέτρηση εμφανίσεων λέξεων

```
map(String doc, String value);
  // doc: όνομα εγγράφου
  // value: περιεχόμενο εγγράφου
  for each word w in value:
     EmitIntermediate(w, “1”);
```

Παράδειγμα:

map(doc, “cat dog cat bat dog”) emits
  [cat 1], [dog 1], [cat 1], [bat 1], [dog 1]

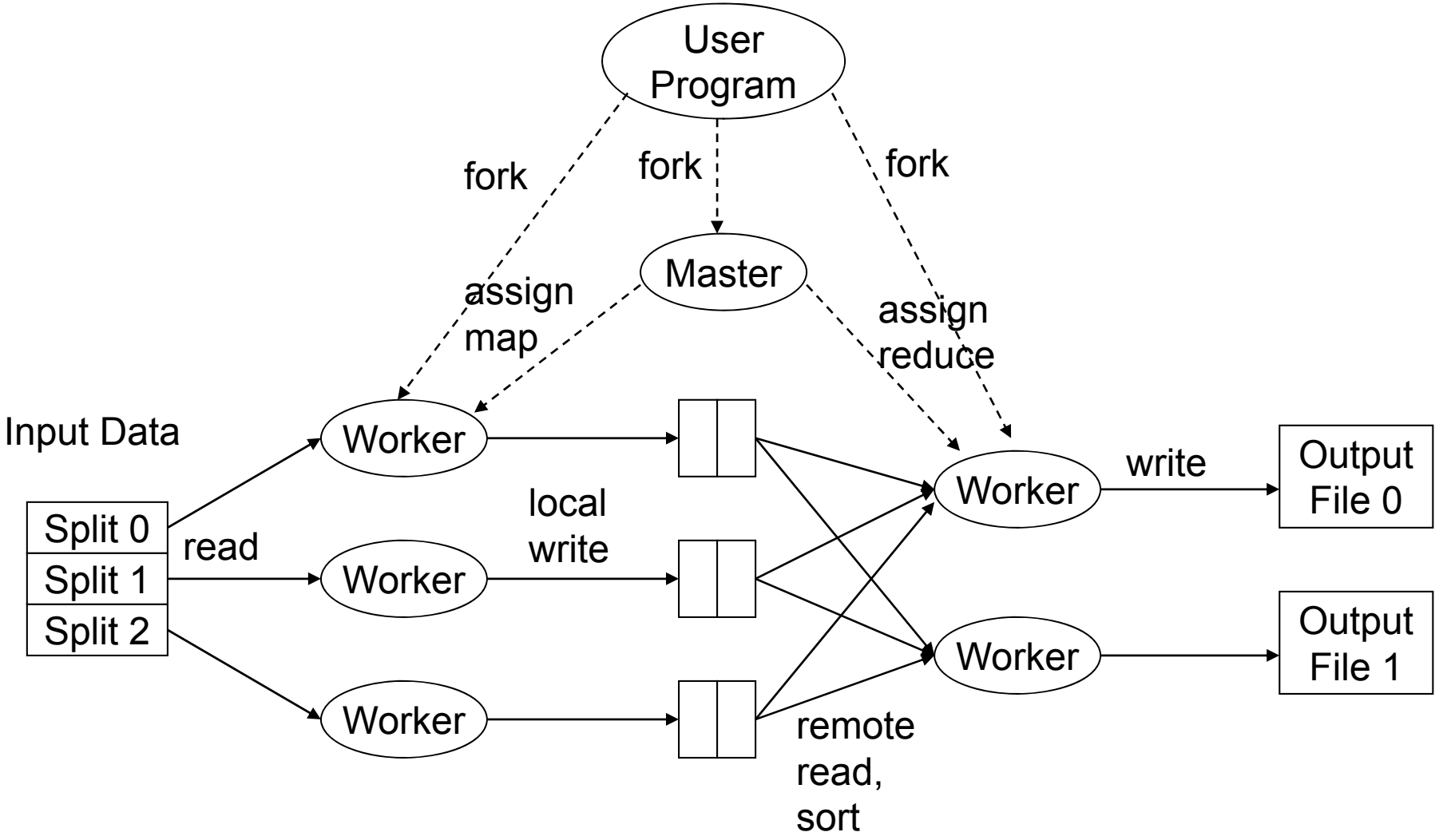
Παράδειγμα: καταμέτρηση εμφανίσεων λέξεων

```
reduce(String key, Iterator values);
  // key: λέξη
  // values: λίστα με καταμετρήσεις
  int result = 0;
  for each v in values:
     result += ParseInt(v)
  Emit(AsString(result));
```

Παράδειγμα:

reduce(“dog”, “1 1 1 1”) emits “4”

Σύνοψη παράλληλης εκτέλεσης



Μοντέλο επεξεργασίας - Map

- Η είσοδος στη συνάρτηση map (π.χ., γραμμές από αρχείο, εγγραφές από ΒΔ, κλπ.) έχουν τη μορφή (κλειδί, τιμή).
- Η συνάρτηση map μετατρέπει κάθε ένα τέτοιο ζευγάρι σε ένα άλλο ζευγάρι (κλειδί΄, τιμή΄).
  - Επεξεργάζεται κάθε στοιχείο της εισόδου ξεχωριστά (γι αυτό και παραλληλίζεται εύκολα).
  - Αντιστοιχεί στο ενδιάμεσο αποτέλεσμα (δηλ. την τιμή΄) ένα κλειδί.
- Ο βαθμός παραλληλισμού καθορίζεται από το σύστημα.

Μοντέλο επεξεργασίας - Reduce

- Η επεξεργασία της συνάρτησης reduce γίνεται αφού ολοκληρωθεί η επεξεργασία της map.
- Για κάθε ξεχωριστό κλειδί΄, δημιουργείται μία λίστα που περιέχει όλες τις αντίστοιχες τιμές΄.
- Η συνάρτηση reduce υπολογίζει μία τελική τιμή για το κλειδί΄ επεξεργάζοντας τη λίστα με τις τιμές που παρήγαγε η map (το κλειδί΄ δεν αλλάζει).
- Οι τιμές για κάθε κλειδί΄ μπορούν να υπολογίζονται παράλληλα.

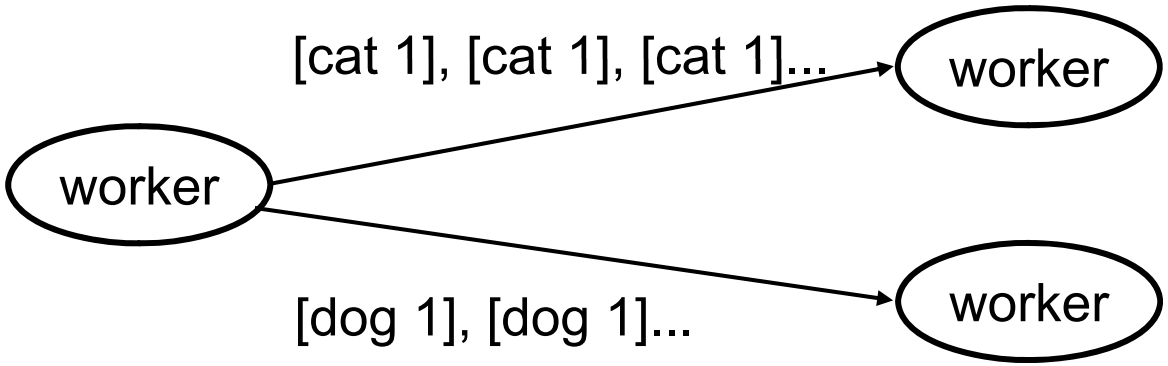
Συντονισμός – κόμβος master

- Ο κόμβος master
  - Ελέγχει αν μια εργασία (task) δεν έχει εκτελεστεί (idle), εκτελείται (in-progress) ή έχει ολοκληρωθεί (completed).
  - Οι εργασίες χρονοδρομολογούνται μόλις ελευθερώνονται κόμβοι.
  - Όταν ολοκληρώνεται μία map εργασία, ο master ενημερώνεται για τη θέση και το μέγεθος των ενδιάμεσων αποτελεσμάτων, που είναι σε τόσα αρχεία, όσα και οι κόμβοι για τη reduce.
  - Κατόπιν ενημερώνονται οι κόμβοι για τη reducer.
- Επίσης ο master ελέγχει περιοδικά (με pings) τους κόμβους για τον εντοπισμό σφαλμάτων.

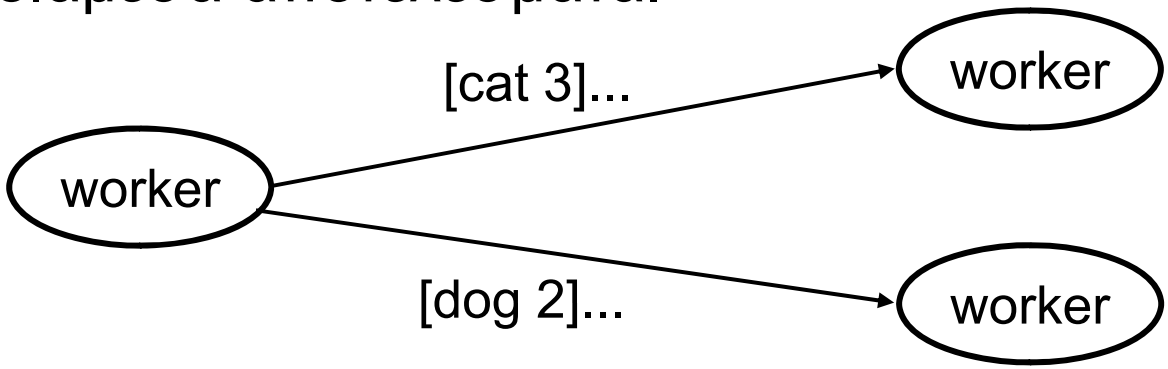
Θέματα υλοποίησης

- Συναρτήσεις συνδυασμού
- Σύστημα Αρχείων
- Διαμερισμός εισόδου και κλειδιών
- Αποτυχίες
- Εργασίες Backup
- Ταξινόμηση αποτελεσμάτων

Συναρτήσεις συνδυασμού (combine)



Παρόμοιο με την εκτέλεση ενός τοπικού *reduce* στα ενδιάμεσα αποτελέσματα.



Κατανεμημένο Σύστημα Αρχείων

Όλες οι μετακινήσεις δεδομένων γίνονται μέσω ενός κατανεμημένου συστήματος αρχείων.

Ο κάθε worker μπορεί να προσπελαύνει κάθε μέρος της εισόδου

Ο κάθε reduce worker έχει πρόσβαση και στους τοπικούς δίσκους

κάθε worker μπορεί να γράφει στο δίσκο το κομμάτι της απάντησης που συνεισφέρει. Η απάντηση είναι σ' ένα κατανεμημένο αρχείο

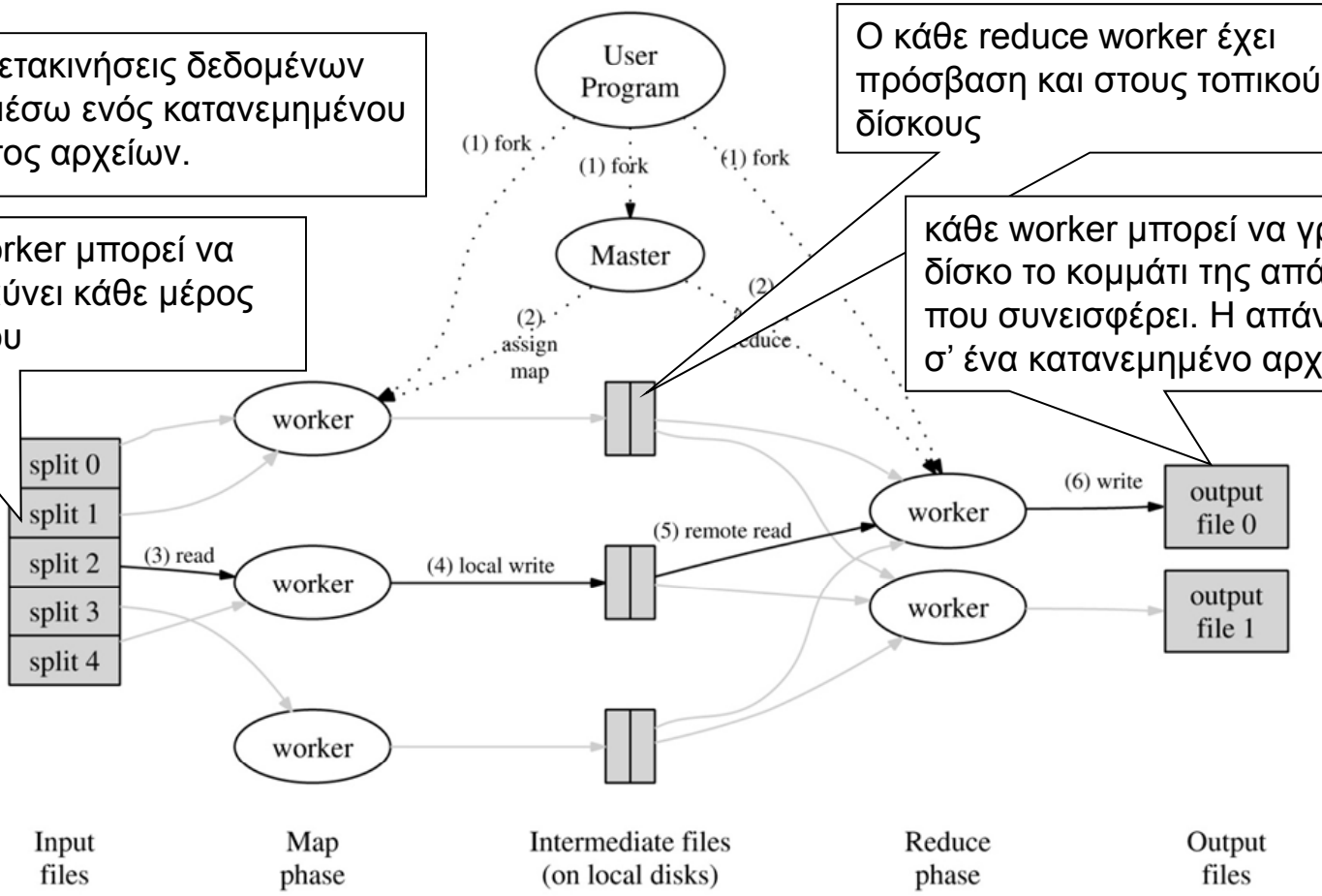


Figure 1: Execution overview

Διαμερισμός εισόδου και κλειδιών

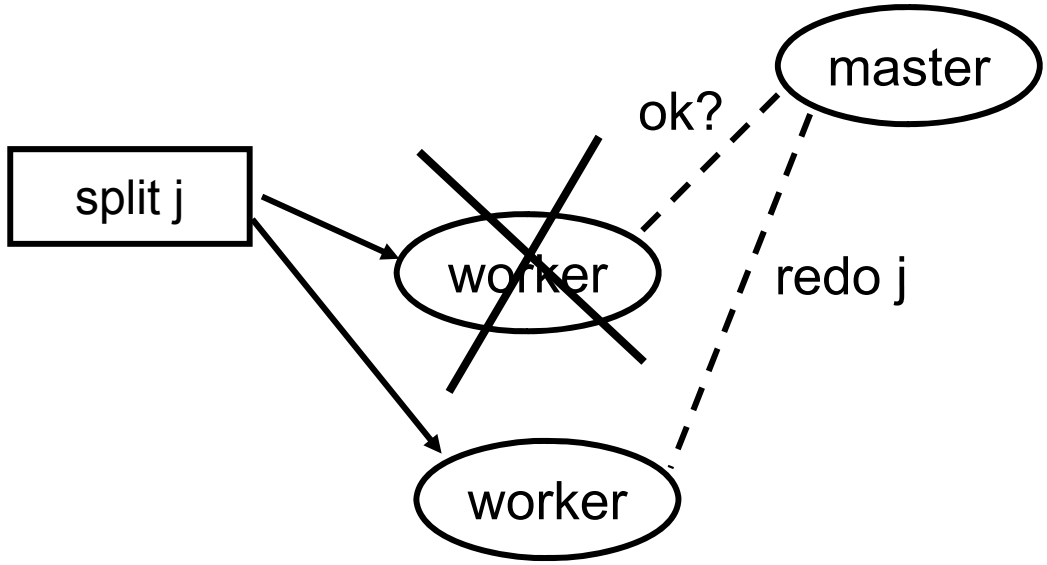
- Πόσοι θα είναι κόμβοι worker;
  - Καλύτερα να υπάρχουν πολλά κομμάτια εισόδου ανά κόμβο, για καλύτερη εξισορρόπηση φόρτου και για καλύτερη αντιμετώπιση αποτυχιών.

- Σε πόσα κομμάτια θα διαμεριστεί η είσοδος;
  - Πιο πολλά από τον αριθμό των κόμβων,
  - Που είναι ήδη μεγάλος.
  - Στα τυπικά παραδείγματα της google ο διαμερισμός είναι σε κομμάτια των 64MB.

- Η ανάθεση γίνεται έτσι ώστε να ανατίθονται κομμάτια της εισόδου στους πιο κοντινούς κόμβους.
  - Όταν αυτό είναι εφικτό

Αποτυχίες

- Ο κόμβος Master εντοπίζει αποτυχίες και ξανα αναθέτει την εργασία του κόμβου που απέτυχε σε άλλο κόμβο worker.
- Αν οι αποτυχίες οφείλονται σε λάθη στα δεδομένα εισόδου, τότε το αντίστοιχο αρχικό κομμάτι δεδομένων μπορεί να αφαιρεθεί.



Backup Εργασίες

- Υπάρχει περίπτωση ένα μηχάνημα να έχει μείνει πίσω στην εκτέλεση των εργασιών (straggler) λόγω π.χ., κάποιου τεχνικού προβλήματος, όπως χαλασμένος δίσκος.
- Εξαιτίας αυτού του μηχανήματος, όλη η επεξεργασία καθυστερεί.
  - Όταν πολλές διεργασίες εκτελούνται παράλληλα, η πιο αργή καθορίζει και τον χρόνο ολοκλήρωσης.
- Λύση: όταν πλησιάζει η συνολική εργασία προς το τέλος, τότε ο συντονιστής χρονοδρομολογεί πολλαπλές φορές τις map/reduce υποεργασίες που απομένουν.
  - Με το σκεπτικό ότι τουλάχιστον μία θα ολοκληρωθεί στον προβλεπόμενο χρόνο.
  - Υπάρχουν επιπλέον μηχανισμοί για την διαγραφή περιττών (διπλών) αποτελεσμάτων όταν όλες οι υποεργασίες τελειώνουν κανονικά.

Ταξινόμηση αποτελεσμάτων

- Γίνεται σε κάθε κόμβο βάσει κλειδιού.

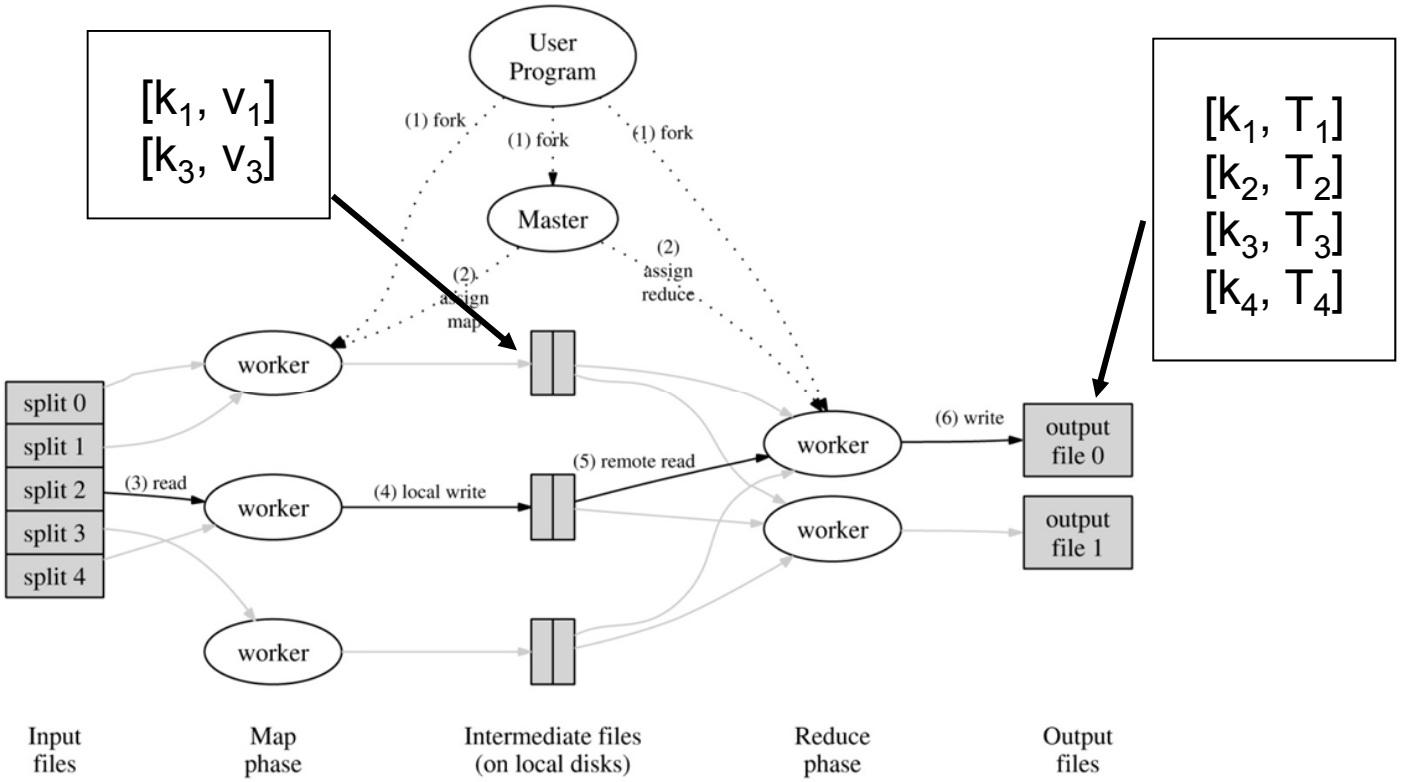
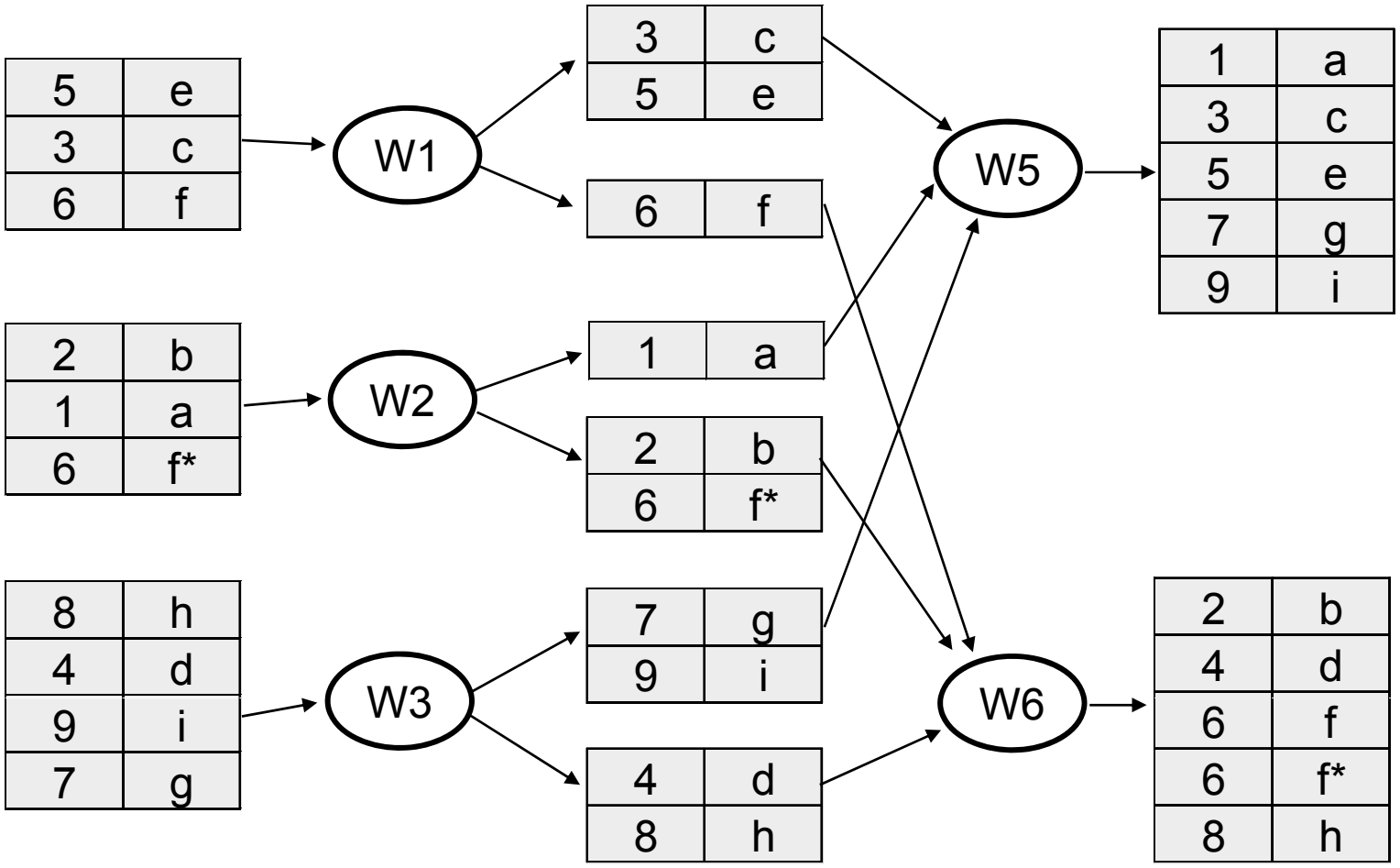


Figure 1: Execution overview

Παράδειγμα: ταξινόμηση εγγραφών



Map: extract k, output [k, record]

Reduce: Τίποτα!

Παραδείγματα - Α

- **Κατανεμημένο Grep:**

- map: παράγει μία γραμμή αν η συγκεκριμένη γραμμή ταυτίζεται με το πρότυπο.
- reduce: απλή αντιγραφή των ενδιάμεσων αποτελεσμάτων στην έξοδο.

Παραδείγματα - Β

- **Καταμέτρηση της συχνότητας προσπέλασης μιας διεύθυνσης URL**:

- Map: επεξεργασία των αρχείων log και παραγωγή ενδιάμεσων αποτελεσμάτων της μορφής <URL, 1> για κάθε προσπέλαση.

- Reduce: πρόσθεση όλων των τιμών για κάποιο συγκεκριμένο URL και παραγωγή ζευγών <URL, total count> .

Παραδείγματα - Γ

- **Αντίστροφος Γράφος Web-Link**:

- Map: παραγωγή ζευγών <target, source> για κάθε σύνδεσμο target προς μια σελίδα από την σελίδα source που εκείνη τη στιγμή επεξεργάζεται.

- Reduce: απλή αντιγραφή στην έξοδο των αποτελεσμάτων που έχουν τη μορφή ζευγών <target, list(source)>.

Παραδείγματα ερωτημάτων

- Έστω η σχέση Employees(id, dno, salary).
- θέλουμε να υπολογίσουμε το ερώτημα:

```
Select dno, SUM(salary)
from employees
where salary>1000
group by dno
```

- Map: για κάθε πλειάδα με `salary>1000` δημιουργείται ένα ζεύγος `(dno,salary)`
- Reduce: για κάθε τιμή του *dno,* η είσοδος είναι όλες οι αντίστοιχες τιμές `salary.` Η συνάρτηση απλώς τις προσθέτει.

Παραδείγματα σχεσιακών τελεστών

- Τελεστές όπως η επιλογή σ_C ή η προβολή π_A απαιτούν μόνο κατάλληλη συνάρτηση map.
- Έστω η σύνδεση $R(X,Y) \bowtie S(Y,Z)$.
  - Map:
    - Είσοδος: (όνομα σχέσης *R* ή *S,* μία πλειάδα *t)*
    - Έξοδος: λίστα (τιμή του *Y*,

      λίστα από (όνομα σχέσης, υπόλοιπο πλειάδας *X* ή *Z*)).
  - Reduce: για κάθε τιμή του *Y,* δημιουργεί όλα τα ζευγάρια *XZ*.

Πλεονεκτήματα

- Το μοντέλο Map-Reduce κρύβει τις τεχνικές λεπτομέρειες της παραλληλοποίησης και παρέχει built-in μηχανισμούς ανάκαμψης από σφάλματα.
- Πολλά προβλήματα μπορούν να εκφραστούν με βάση το μοντέλο Map-Reduce.
- Οι εφαρμογές κλιμακώνονται σε χιλιάδες μηχανήματα.
  - Αλλά παρόμοια κλιμάκωση επιτυγχανεται και με παράλληλη επεξεργασία ερωτημάτων σε αρχιτεκτονικές shared nothing.

Μειονεκτήματα

- Το μοντέλο επεξεργασίας μοναδικής εισόδου και δύο σταδίων δεν είναι ευέλικτο ώστε να προσαρμόζεται σε όλα τα επιθυμητά σενάρια.
- Απαιτείται η συγγραφή κώδικα ακόμη και για τους πιο συνηθισμένους τελεστές, όπως επιλογής και προβολής.
  - Δήλωση εργασιών σε πολύ χαμηλότερο επίπεδο σε σχέση με την SQL.
- Ο κώδικας μέσα στις συναρτήσεις map και reduce δεν είναι φανερός → δυσκολία στην βελτιστοποίηση.

Υλικό για μελέτη

- MapReduce: Simplified Data Processing on Large Clusters, *Jeffrey Dean and Sanjay Ghemawat,*

  http://labs.google.com/papers/mapreduce-osdi04.pdf
- Hadoop: υλοποίηση ανοιχτού κώδικα
- Pig Latin: A Not-So-Foreign Language for Data Processing, *Christopher Olston, Benjamin Reedy, Utkarsh Srivastavava, Ravi Kumar, Andrew Tomkins,* http://wiki.apache.org/pig/