Επεξεργασία Ερωτήσεων

Εισαγωγή

1. ΜΟΝΤΕΛΑ ΔΕΔΟΜΕΝΩΝ
   - Μοντέλα
   - Γλώσσες Ερωτήσεων
   - Επεξεργασία Ερωτήσεων
     - Επεξεργασία Ερωτήσεων σε Σχεσιακά ΣΔΒΔ
     - Επεξεργασία Ερωτήσεων σε Κατανεμημένα Σχεσιακά ΣΔΒΔ
     - → **Επεξεργασία Ερωτήσεων σε Ημιδομημένα Δεδομένα**
2. ΑΡΧΙΤΕΚΤΟΝΙΚΕΣ
3. ΤΡΟΠΟΙ ΜΕΤΑΔΟΣΗΣ

Εισαγωγή

1. ΜΟΝΤΕΛΑ ΔΕΔΟΜΕΝΩΝ
   - ...
   - Επεξεργασία Ερωτήσεων
     - **Επεξεργασία Ερωτήσεων σε Ημιδομημένα Δεδομένα**
       - **Αποθήκευση**
       - → **Ευρετήρια**
       - **Κατανεμημένος Υπολογισμός**
   - Συστήματα (Lore, Strudel)
2. ΑΡΧΙΤΕΚΤΟΝΙΚΕΣ
3. ΤΡΟΠΟΙ ΜΕΤΑΔΟΣΗΣ

Επεξεργασία Ερωτήσεων για Ημι-δομημένα Δεδομένα

Επεξεργασία Ερωτήσεων

Τα ίδια βασικά στάδια

1. Μετάφραση - ένα σχέδιο εκτέλεσης
2. Βελτιστοποίηση
3. Μηχανή Εκτέλεσης

- Έλλειψη σχήματος
- Κατανομή/Αρχιτεκτονικές

Αποθήκευση

- Storage back-end
- Type information

Τρόποι Αποθήκευσης

1. Κείμενο
2. Σχεσιακή Βάση Δεδομένων
3. Αντικειμενο-στραφή Βάση Δεδομένων
4. Αυτο-οργάνωση - Υβριδική αποθήκευση

Ευρετήρια

Ευρετήρια για

1. Εκφράσεις Μονοπατιών (σε δέντρα & γράφους)
2. XML documents (και γενικά tagged αρχεία)
3. Κείμενο (text files) - search engines

Ευρετήρια για εκφράσεις μονοπατιών

Ευρετήρια για εκφράσεις μονοπατιών

Σχήμα 8.8 (σελίδα 181)

Ερωτήσεις:

(R1) part.name

(R2) part.supplier.name

(R3) _*.supplier.name

(R4) part._*.subpart.name

Ευρετήρια για εκφράσεις μονοπατιών

κόμβοι που ακριβώς τα ίδια μονοπάτια από τη ρίζα (π.χ., κόμβοι n1, n3 και n12 - n2, n13 και n4 στο Σχήμα 8.8)

p1 and p2 **language-equivalent**: for any path expression query, either both p1 and p2 are in the answer or none is

Ευρετήρια για εκφράσεις μονοπατιών

Για κάθε κόμβο x

$L_x \equiv \{w \mid \exists \text{ path from the root to x labeled } w\}$

L_x

- μπορεί να είναι άπειρο αν ο γράφος έχει κύκλους
- γενικά περιγράφεται από μια κανονική έκφραση

x and y **language-equivalent**: $(x \equiv y)$ if $L_x = L_y$

Ευρετήρια για εκφράσεις μονοπατιών

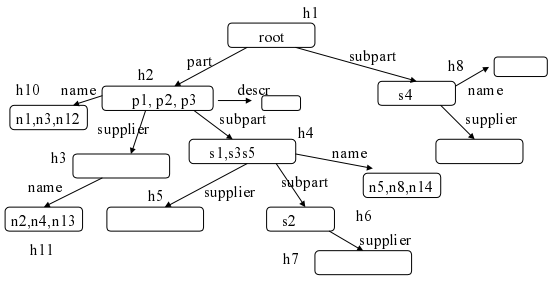
Έστω [x] η κλάση ισοδυναμίας για το x

Κατασκευή του ευρετηρίου

ένα κόμβο για κάθε κλάση ισοδυναμίας

υπάρχει ακμή από το [x] στο [y] με label a αν υπάρχουν κόμβοι x' στο [x] και y' στο [y] που συνδέονται με κάποια ακμή με label a

Ευρετήρια για εκφράσεις μονοπατιών



Ευρετήρια για εκφράσεις μονοπατιών

Ερωτήσεις:

(R1) part.name

$h1 \rightarrow h2 \rightarrow hn$

(R2) part.supplier.name

$h1 \rightarrow h2 \rightarrow h3 \rightarrow h11$

(R3) _*.supplier.name

??

(R4) part._*.subpart.name

search all nodes in the subtree rooted at h2

Ευρετήρια για εκφράσεις μονοπατιών

Πως θα υπολογίσουμε αν $x \equiv y$;

Reverse graph

ισχύει

$\forall x,y,\ x \approx y \Rightarrow x \equiv y$

Ευρετήρια για XML text

Ευρετήρια για XML text

• **περιοχή (region)**: ένα συνεχόμενο (contiguous) τμήμα κειμένου στο αρχείο

• **σύνολο περιοχών (region set)** : ένα σύνολο περιοχών τέτοιο ώστε δυο οποιεσδήποτε περιοχές του συνόλου είτε είναι ξένες (disjoint) είτε η μία περιέχεται στην άλλη

Στη δεντρική αναπαράσταση:

κάθε κόμβος ορίζει μια περιοχή (π.χ , ο κόμβος p2 αντιστοιχεί στο κείμενο κάτω από τον κόμβο p2)

σύνολο περιοχών: σύνολο κόμβων

Ευρετήρια για XML text

Ιδιαίτερο ενδιαφέρον - σύνολα περιοχών που αντιστοιχούν σε XML tags

π.χ , part ορίζει το σύνολα περιοχών {p1, p2, p3}

subpart ορίζει το σύνολα περιοχών {s1, s2, s3, s4, s5}

Ευρετήρια για XML text

Περιοχή

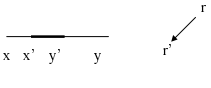
ζεύγος (x, y) : start and end position of the region in the text file

Σύνολο Περιοχών

ordered tree: κάθε κόμβος μια περιοχή

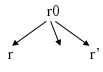
• r = (x, y) *ancestor* of r' = (x', y') (r' ⊆ r)

if $x \le x' \le y' \le y$



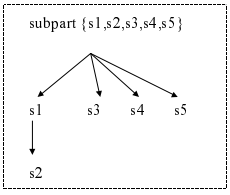
• r to the *left* of r'

if $x \le y \le x' \le y'$



Ευρετήρια για XML text

Παράδειγμα



Ευρετήρια για XML text

Region Algebra (άλγεβρα για περιοχές)

- πάνω σε σύνολα περιοχών
- τελεστές (op) που δίνουν ως αποτέλεσμα σύνολα περιοχών: s1 op s2

νέα σύνολα περιοχών -- όχι νέες περιοχές: αυτές θεωρούνται προκαθορισμένες (αντιστοιχούν στους κόμβους)

Ευρετήρια για XML text

Παραδείγματα τελεστών μιας άλγεβρας περιοχών

s1 **intersect** s2 ≡ {r | r ∈ s1, r ∈ s2}

s1 **included** s2 ≡ {r | r ∈ s1, ∃ r ' ∈ s2, r ⊆ r' }

s1 **including** s2 ≡ {r | r ∈ s1, ∃ r ' ∈ s2, r ⊇ r' }

s1 **parent** s2 ≡ {r | r ∈ s1, ∃ r ' ∈ s2, r is a parent of r' }

s1 **child** s2 ≡ {r | r ∈ s1, ∃ r ' ∈ s2, r is a child of r' }

Ευρετήρια για XML text

Παραδείγματα

- subpart {s1, s2, s3, s4, s5} part {p1, p2, p3}

  subpart included part {s1, s2, s3, s5}

  part including subpart {p2, p3}

- name {n1, n2, …, n12} part {p1, p2, p3}

  name child part {n1, n3, n12}

  name included part {n1, n2, .., n9, n12}

Ευρετήρια για XML text

Υπολογισμός Τελεστών

s1 op s2

traverse the tree representations of s1 and s2 simultaneously similar to a merge join

Ευρετήρια για XML text

Παράδειγμα : s1 **included** s2

(υποθέτουμε ότι τα s1 και s2 είναι σύνολα ξένων περιοχών)

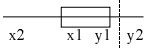
Αρχικά (x1, y1) το πρώτο στοιχείο του s1 και (x2, y2) το πρώτο στοιχείο του s2

Μέχρι το τέλος της λίστας s1 ή s2

Αν x1 < x2, advance s1

Αν y1 > y2, advance s2

Αλλιώς, πρόσθεσε το (x1, y1) στο αποτέλεσμα, advance s1



Ευρετήρια για XML text

Ερωτήσεις:

(R1) part.name

name child (part child root)

(R2) part.supplier.name

name child (supplier child (part child root))

(R3) _*.supplier.name

name child supplier

(R4) part._*.subpart.name

name child (subpart included (part child root))

Ευρετήρια για XML text

(R5)

```
select X
from _*.subpart: {name: X, _*.supplier.address: "Philadelphia"
```

name child (subpart includes (supplier parent (address intersect "Philadelphia")))

Ευρετήρια για XML text

• μόνο ένα περιορισμένο αριθμό από κανονικές εκφράσεις, συγκεκριμένα για τις εκφράσεις

$R_1 . R_2 . \ldots R_n$

όπου R_i label constant or the Kleene closure (_*)

• μόνο για ordered trees

Ευρετήρια για κείμενο

Ευρετήρια για κείμενο

ένας σημαντικός τύπος ερώτησης: **keyword search**

• ο πιο συνηθισμένος τύπος ερώτησης στις μηχανές αναζήτησης

• συχνά κρατείται και μια λίστα με *συνώνυμα* (π.χ, ερώτηση για car - επίσης, automobile)

• πιο περίπλοκες ερωτήσεις που περιλαμβάνουν and, or, not

Ευρετήρια για κείμενο

Δυο βασικοί τύποι ερωτήσεων:

- boolean
- ranked query

1. Boolean query (conjuctive normal form)

$$(t_{11} \vee t_{12} \vee \ldots t_{1i1}) \wedge \ldots \wedge (t_{j1} \vee t_{j2} \vee \ldots \vee t_{jij})$$

όπου t_{ij} είναι ανεξάρτητα query terms ή keywords

j conjucts (που αντιστοιχούν σε διαφορετικές έννοιες (concepts)) - καθεμία από πολλά disjuncts (που αντιστοιχούν σε διαφορετικούς όρους για την ίδια έννοια)

Ευρετήρια για κείμενο

2. Ranked query

Αποτέλεσμα

σύνολο από documents που επιπρόσθετα είναι **ταξινομημένα με βάση τη σχετικότητά τους** (ranked by their relevance)

Information retrieval

Δύο κριτήρια

• **precision (ακρίβεια)**: ποσοστό των ανακαλούμενων (retrieved) documents που είναι σχετικά με την ερώτηση

• **recall**: ποσοστό των σχετικών documents της βάσης δεδομένων που ανακαλούνται ως απάντηση στην ερώτηση

Ευρετήρια

Ευρετήριο

ζεύγη <keyword, documentid>

με πιθανά επιπρόσθετα πεδία όπως πόσες φορές εμφανίζεται το keyword στο document

• Μια *μηχανή αναζήτησης* δημιουργεί ένα κεντρικό ευρετήριο για documents που είναι αποθηκευμένα σε διάφορα sites

Inverted files

- Για κάθε όρο (term) μια ταξινομημένη λίστα (inverted list) από τα ids των documents που περιέχουν αυτόν το όρο
- Επιπρόσθετα, όλοι οι πιθανοί όροι τοποθετούνται σε ένα δευτερεύον ευρετήριο (π.χ., B+-δέντρο) -- ευρετήριο λεξιλογίου

Conjunction -- ξεκινώντας από τη συντομότερη λίστα

Inverted files

Rid	Document
1	agent James Bond
2	agent mobile computer
3	James Madison movie
4	James Bond movie

Word	Inverted List
agent	<1,2>
Bond	<1,4>
computer	<2>
James	<1,3,4>
Madison	<3>
mobile	<2>
movie	<3,4>

Signature files

Μια εγγραφή ευρετηρίου **(signature - υπογραφή)** για κάθε document

Κάθε υπογραφή έχει *σταθερό* μέγεθος b bits - το b ονομάζεται **πλάτος της υπογραφής (signature width)**

Ποια bits της υπογραφής ενός κειμένου τίθονται ίσα με 1 εξαρτάται από το ποιες λέξεις εμφανίζονται στο κείμενο

- εφάρμοσε μια συνάρτηση κατακερματισμού σε κάθε λέξη που εμφανίζεται στο κείμενο
- θέσε τα bits που εμφανίζονται στο αποτέλεσμα της συνάρτησης

ένα bit μπορεί να γίνεται 1 πολλαπλές φορές από διαφορετικές λέξεις

Rid	Document	Signature
1	agent James Bond	1100
2	agent mobile computer	1101
3	James Madison movie	1011
4	James Bond movie	1110

Πλάτος υπογραφής 4

Word	Hash
agent	1000
Bond	0100
computer	0100
James	1000
Madison	0001
mobile	0001
movie	0010

Signature files

- Μια υπογραφή S_1 **ταιριάζει (matches)** μια άλλη υπογραφή S_2 αν όλα τα bits που έχουν τεθεί στην S_2 έχουν επίσης τεθεί και στην S_1
- Αν μια υπογραφή S_1 ταιριάζει μια υπογραφή S_2, τότε η υπογραφή S_1 έχει τουλάχιστον τόσα bits όσα και η υπογραφή S_2

Rid	Document	Signature
1	agent James Bond	**1100 S2**
2	agent mobile computer	**1101 S1**
3	James Madison movie	1011
4	James Bond movie	1110

Ευρετήρια για κείμενο

Signature files

Conjuction (Σύζευξη)

1. Δημιούργησε την υπογραφή της ερώτησης εφαρμόζοντας την συνάρτηση κατακερματισμού σε κάθε λέξη στην ερώτηση

2. Scan το αρχείο των υπογραφών και ανακάλεσε (retrieve) όλα τα documents των οποίων οι υπογραφές ταιριάζουν με την υπογραφή της ερώτησης

3. Ανακάλεσε (retrieve) κάθε πιθανό ταίριασμα και έλεγξε αν πραγματικά περιέχει τις λέξεις της ερώτησης

false positive

Ευρετήρια για κείμενο

Rid	Document	Signature
1	agent James Bond	1100
2	agent mobile computer	1101
3	James Madison movie	1011
4	James Bond movie	1110

Word	Hash
agent	1000
Bond	0100
computer	0100
James	1000
Madison	0001
mobile	0001
movie	0010

Πλάτος υπογραφής 4

Ερωτήσεις: "James", "James" and "Bond", "movie" and "Madison"

Ευρετήρια για κείμενο

Signature files

Disjuction (διάζευξη)

1. Δημιούργησε μια λίστα από υπογραφές για την ερώτηση μία για κάνε λέξη της ερώτησης

2. Scan το αρχείο των υπογραφών αρχείων και ανακάλεσε όλα τα documents των οποίων οι υπογραφές ταιριάζουν τουλάχιστον με μια υπογραφή στη λίστα των υπογραφών της ερώτησης

Ευρετήρια για κείμενο

Rid	Document	Signature
1	agent James Bond	1100
2	agent mobile computer	1101
3	James Madison movie	1011
4	James Bond movie	1110

Word	Hash
agent	1000
Bond	0100
computer	0100
James	1000
Madison	0001
mobile	0001
movie	0010

Ερωτήσεις: "James" or "Bond", "Bond" or "agent"

Ευρετήρια για κείμενο

scan the *complete signature file*

Κάθετος διαμερισμός του αρχείου με τις υπογραφές σε σύνολα από b bit slices -- bit sliced signature file

Για μια ερώτηση με q bits - retrieve only q bit slices

Ερωτήσεις στο www

Διάκριση σελίδων σε

- authorities (αυθεντίες) και
- hubs

Αυθεντία (authority) : μια σελίδα που είναι πολύ σχετική σε ένα συγκεκριμένο θέμα και αναγνωρίζεται από τις άλλες σελίδες ως έγκυρη (authoritative) στο θέμα

Ερωτήσεις στο www

Οι άλλες σελίδες (**hubs**) έχουν συνήθως ένα μεγάλο αριθμό αναφορών (hyperlinks) στις αυθεντίες αν και οι ίδιες δεν είναι ιδιαίτερα γνωστές και το περιεχόμενο τους δεν είναι απαραίτητα πολύ σχετικό με το θέμα

Παραδείγματα hubs

συλλογή από πηγές για ένα θέμα σε μια επαγγελματική σελίδα

λίστα από σελίδες σχετικές με τα hobbies ενός συγκεκριμένου χρήστη ή και

τμήμα των bookmarks ενός συγκεκριμένου χρήστη

Ερωτήσεις στο www

• το βασικό χαρακτηριστικό των hubs είναι ότι έχουν πολλά links προς σχετικές σελίδες

• ενώ μπορεί να υπάρχουν πολύ λίγα links που δείχνουν σε ένα hub

Ερωτήσεις στο www

Ο αλγόριθμος HITS

Είσοδο: ερώτηση χρήστη με έναν αριθμό όρων (terms)

Αποτέλεσμα: ένα σύνολο από καλές αυθεντίες και hubs

www :: κατευθυνόμενο γράφο (κόμβοι: σελίδες, ακμές: hyperlink)

Ο αλγόριθμος προχωρά σε δύο βήματα:

- Βήμα Δειγματοληψίας (sampling step): ένα σύνολο σελίδων που καλείται το βασικό σύνολο
- Δεύτερο Βήμα : ποιες από τις σελίδες στο βασικό σύνολο είναι καλές αυθεντίες και ποιες καλά hubs

Ερωτήσεις στο www

Βήμα Δειγματοληψίας

1. Retrieve ένα σύνολο web σελίδων που περιέχουν τους όρους της ερώτησης - το σύνολο αυτό καλείται **σύνολο ρίζα (root set)**

Πως;

για παράδειγμα υπολογίζοντας την ερώτηση ως μια boolean keyword query

Ερωτήσεις στο www

Αρκεί;

Link page: μια σελίδα που είτε περιέχει κάποιο hyperlink σε σελίδα του συνόλου ρίζα ή μια σελίδα του συνόλου ρίζα έχει κάποιο hyperlink σε αυτήν

2. Επέκταση του συνόλου ρίζα με όλες τις link σελίδες → **βασικό σύνολο**

Βασική σελίδα: σελίδα του βασικού συνόλου

Ερωτήσεις στο www

Δεύτερο βήμα

Συσχετίζουμε με κάθε σελίδα δύο βάρη:

- hub weight
- authority weight

Υπολογίζουμε τα βάρη με βάση την υπόθεση

• ότι μια σελίδα είναι καλή αυθεντία αν πολλά καλά hubs δείχνουν σε αυτήν

• ενώ ένα hub είναι ένα καλό hub αν έχει πολλά links σε καλές αυθεντίες

Ερωτήσεις στο www

Αρχικά και τα δύο βάρη είναι ίσα με 1

Έστω μια βασική σελίδα p με hub weight h_p και authority weight a_p

Σε κάθε βήμα

αυξάνουμε το a_p ώστε να είναι ίσο με το άθροισμα των hub weights h_q των σελίδων q που δείχνουν στη p

αυξάνουμε το h_p ώστε να είναι ίσο με το άθροισμα των authority weights a_q όλων των σελίδων q στις οποίες δείχνει η p

Ερωτήσεις στο www

Στο βιβλίο (cow book, σελ. 670),

αποδοτική υλοποίηση βασισμένοι σε πίνακες

Ερωτήσεις στο www

Αποτέλεσμα του HITS με ερώτηση Gates -

οι καλύτερες (highest rank) αυθεντίες

http://www.roadahead.com/

http:// www.microsoft.com/

http:// www.microsoft.com/corpinfo/bill-g.htm

Κατανεμημένος Υπολογισμός

- Γνώση του σχήματος (πχ. που βρίσκεται η σχετική πληροφορία) -- τεχνικές από κατανεμημένο υπολογισμό ερωτήσεων
- Μη γνώση του σχήματος

Κατανεμημένος Υπολογισμός με γνώση του Σχήματος

Θεωρείστε το σχήμα των σελίδων 191--192

Υπάρχει αντιγραφή της πληροφορίας τοπικά (S1) εκτός από την περίπτωση του <rating> που μπορεί να περιέχει link (href) στα πραγματικά δεδομένα στο site S2: www.nhtsa.dot.gov

Κατανεμημένος Υπολογισμός με γνώση του Σχήματος

Θεωρείστε την ερώτηση

```
Q = select X.name X.address, Y.make, Y. Model
    where dealer X,
          X.address.city = "Springfield"
          X.car Y,
          Y.Year >= 1996
          Y._*.ratings R,
          R._*.driver >= 4, R._*.passenger >= 4
```

Κατανεμημένος Υπολογισμός με γνώση του Σχήματος

Ένωση δυο ερωτήσεων Q1 και Q2 όπου η Q1 μπορεί να εκτελεστεί τοπικά

Q1 = select X.name X.address, Y.make, Y. Model
where dealer X,
X.address.city = "Springfield"
X.car Y,
Y.Year >= 1996
Y.**[^href]***.ratings R,
R._*.driver >= 4, R._*.passenger >= 4

Κατανεμημένος Υπολογισμός με γνώση του Σχήματος

Q2 = select entry: {dealer:X, car:Y, ref:Y.ref}
where dealer X,
X.address.city = "Springfield"
X.car Y,
Y.Year >= 1996
indomain(Y.ref, "www.nhtsa.dot.gov")

Έστω **R** το αποτέλεσμα

περιέχει τα X, Y που είναι τοπικά (S1) και το Y.ref που βρίσκεται σε άλλο site (S2)

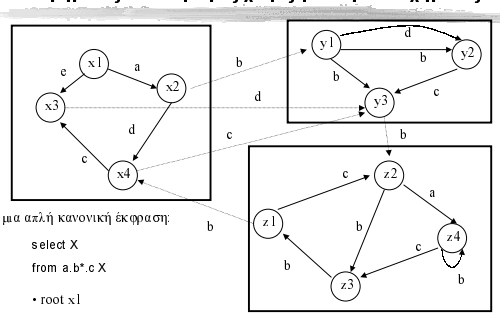
Πως θα γίνει ο υπολογισμός (distributed join!)

Κατανεμημένος Υπολογισμός με γνώση του Σχήματος

Semijoin!

1. Αποστολή του R στο site S2
2. Εκτέλεση στο site S2 της ερώτησης
   select nhtsa: {dealer:X, car:Y}
   from R.entry E, E.dealer X, E.car Y, E.href H
   where H.rating Z, Z._*.driver >= 4, Z._*.passenger >=4
3. Αποστολή του αποτελέσματος στο site S1
4. Υπολογισμός στο S1 της ερώτησης
   select X.name, X.addr, Y.make, Y.model
   from H.nhtsa N, N.dealer X, N.car Y

Κατανεμημένος Υπολογισμός χωρίς γνώση του Σχήματος



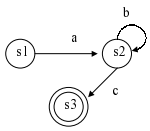
Έστω μια απλή κανονική έκφραση:

select X
from a.b*.c X

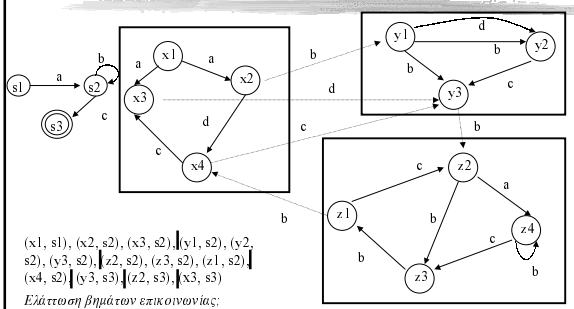
• root x1

Κατανεμημένος Υπολογισμός χωρίς ΓΣ

Κατασκευάζουμε το αυτόματο για την κανονική έκφραση a.b*.c -- Προχωράμε κατασκευάζοντας την Closure



Κατανεμημένος Υπολογισμός χωρίς ΓΣ



(x1, s1), (x2, s2), (x3, s2), (y1, s2), (y2, s2), (y3, s2), (z2, s2), (z3, s2), (z1, s2), (x4, s2), (y3, s3), (z2, s3), (x3, s3)

Ελάττωση βημάτων επικοινωνίας;

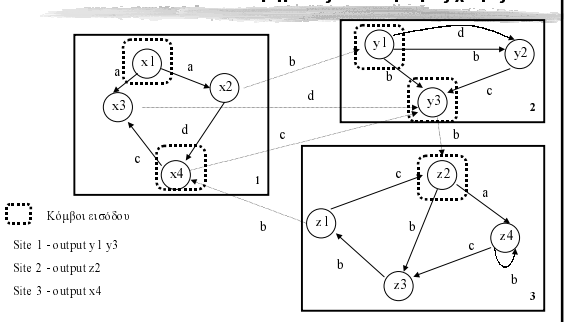
Κατανεμημένος Υπολογισμός χωρίς ΓΣ

• Κάθε site αναγνωρίζει τις αναφορές σε μη τοπικούς κόμβους -- αντίγραφα αυτών των κόμβων προστίθενται σε κάθε site: **output nodes**

• Επίσης, κάθε site αναγνωρίζει τους τοπικούς κόμβους που είναι στόχος εξωτερικών αναφορών: **input nodes**

• Η ρίζα του γράφου προστίθεται στους κόμβους εισόδου για το site 1

Κατανεμημένος Υπολογισμός χωρίς ΓΣ



Κατανεμημένος Υπολογισμός χωρίς ΓΣ

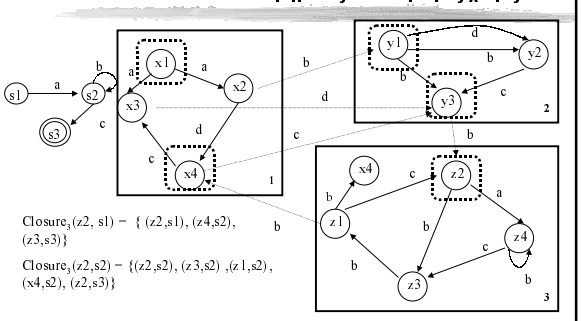
• Κατασκευάζουμε το αυτόματο και το στέλνουμε σε κάθε site

• Αρχίζουμε την ίδια διαδικασία σε κάθε site.

  • Η διαδικασία "τρέχει" το αυτόματο ξεκινώντας από κάθε κόμβο εισόδου του site

  • Έστω n ένας κόμβος εισόδου στο site i και s η κατάσταση στη αυτόματο, υπολογίζουμε την $Closure_i(n, s)$ μέχρι να μην αλλάζει

Κατανεμημένος Υπολογισμός χωρίς ΓΣ



$Closure_3(z2, s1) = \{ (z2,s1), (z4,s2), (z3,s3)\}$

$Closure_3(z2,s2) = \{(z2,s2), (z3,s2) ,(z1,s2), (x4,s2), (z2,s3)\}$

Κατανεμημένος Υπολογισμός χωρίς ΓΣ

• Από το $Closure_i(n, s)$ για κάθε n υπολογίζουμε δύο σύνολα:

$Stop_i(n, s)$: το σύνολο των ζευγών (n', s') που ανήκουν στο $Closure_i(n, s)$ και το n' είναι output node

$Result_i(n, s)$: το σύνολο των κόμβων στο $Closure_i(n, s)$ που είναι ζεύγος με τελική κατάσταση

$Closure_3(z2, s1) = \{ (z2,s1), (z4,s2), (z3,s3)\}$ $Closure_3(z2,s2) = \{(z2,s2), (z3,s2) ,(z1,s2), (x4,s2), (z2,s3)\}$

$Stop_3(z2,s1) = \{\}$ $Stop_3(z2,s2) = \{(x4,s2)\}$

$Result_3(z2,s1) = \{z3\}$, $Result_3(z2,s2) = \{z2\}$

Κατανεμημένος Υπολογισμός χωρίς ΓΣ

• Σε κάθε site κατασκευάζουμε δυο δυαδικές σχέσεις για το $Stop_i$ και το $Result_i$

((n,s), (n', s')) για (n', s') ∈ $Stop_i$(n,s)

((n,s), n') για n' ∈ $Result_i$(n,s)

$Stop_3(z2,s1) = \{\}$ $Stop_3(z2,s2) = \{(x4,s2)\}$

$Result_3(z2,s1) = \{z3\}$, $Result_3(z2,s2) = \{z2\}$

Start	Stop
(z2,s2)	(x4,s2)

Start	Result
(z2,s1)	z3
(z2,s2)	z2

site 3

Κατανεμημένος Υπολογισμός χωρίς ΓΣ

- Κάθε site στέλνει τις δύο σχέσεις σε ένα κεντρικό site όπου υπολογίζεται η ένωση τους
- Το κεντρικό site υπολογίζει η transitive Closure των σχέσεων Start/Stop και βρίσκει όλα τα Stop ζεύγη που είναι προσπελάσιμα από το (x1, s1).
- Από αυτά τα ζεύγη χρησιμοποιεί τη σχέση Start/Result για να βρεί τους κόμβους που είναι προσπελάσιμη από τη ρίζα (x1, s1).

κόστος επικοινωνίας;

Κατανεμημένος Υπολογισμός χωρίς ΓΣ

Start	Stop
(x1,s1)	(y1,s2)
(x4,s2)	(y3,s3)
(y1,s2)	(z2,s2)
(y3,s2)	(z2,s2)
(z2,s2)	(x4,s2)

transitive closure

(x1,s1), (y1,s2), (z2,s2),(x4,s2),(y3,s3)

Start	Result
(x1,s3)	x1
(x4,s2)	x3
(x4,s3)	x4
(y1,s2)	y3
(y1,s3)	y1
(z2,s1)	z3
(z2,s2)	z2
(z2,s3)	z2

Result
y3, z2, x3

Κατανεμημένα ΣΔΒΔ

Είδη

- **Ομοιογένεια**
- **Ετερογένεια**

gateway protocols: API that exposes DBMS functionality to external applications (e.g., ODBC, JDBC)

Κατανεμημένα ΣΔΒΔ

Αρχιτεκτονικές

- **Client/server**

clients (user-interface issues)

servers (manage data and execute transactions)

(client cache)

Κατανεμημένα ΣΔΒΔ

Αρχιτεκτονικές

- **Collaborating server systems**

a single query spans multiple servers

a collection of database servers, each capable of running transactions against local data which *cooperatively* execute transactions spanning multiple servers

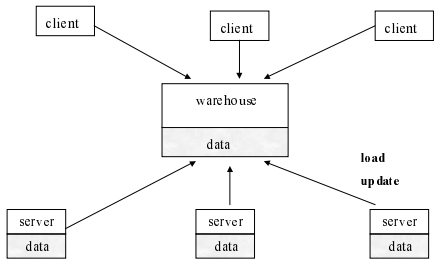
Κατανεμημένα ΣΔΒΔ

Αρχιτεκτονικές

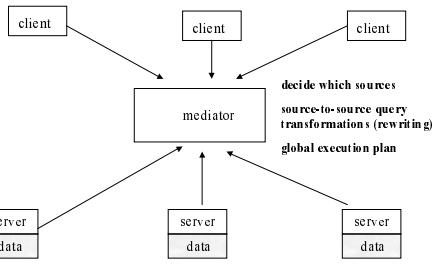
- **Middleware systems**

just one special database server (layer of software) that coordinates the execution of queries and transactions across one or more independent database servers

Αρχιτεκτονικές



Αρχιτεκτονικές



Mediators

1. **Data conversion** (μετατροπή δεδομένων μεταξύ διαφορετικών μοντέλων) -- ή rewrite τις ερωτήσεις σε διαφορετικά μοντέλα

2. **Data intergration** (συγχώνευση δεδομένων από διαφορετικές πηγές σε μια κοινή όψη) -- ή rewrite (decompose) μια ερώτηση σε έναν αριθμό από ερωτήσεις που η κάθε μία θα εκτελεστεί σε ένα site

Mediators

Data integration

1. Όταν οι πληροφορία σε κάθε site είναι disjoint

Import τα σχήματα των sources

Μετάφραση της ερώτησης σε ερωτήσεις που θα εκτελεστούν σε ένα μόνο site (παράδειγμα στο βιβλίο)

Mediators

Data integration

2. Όταν οι πληροφορία σε κάθε site δεν είναι disjoint (data fusion)

Χρήση Skolem functions

όταν το σύστημα προσπαθήσει να δημιουργήσει ένα αντικείμενο με την ίδια τιμή Skolem οι ακμές προστίθενται στο ήδη υπάρχον αντικείμενο

Mediators

Data integration -- information manifold

Global as view vs Local as view

Τα sources θεωρούνται views over the integrated data

παράδειγμα: ως projection

Mediators -- Incremental Maintenance

Αποφυγή υπολογισμού της όψης από την αρχή!

Μια αλλαγή θα επηρεάσει ή όχι την όψη