Optimal aggregation algorithms for middleware

CS856 Fall 2005 Presentation

Weihan Wang
w23wang@uwaterloo.ca

November 23, 2005

About the paper

- Ronald Fagin, IBM Research
- Amnon Lotem, Maryland
- Moni Naor, Weizmann, Israel

- In *ACM Symp. Principles of Database Systems, 2001.*

Agenda

- Background
- Fagin's algorithm
- Threshold algorithm
- θ-approximation
- NRA algorithm
- Combined algorithm

Agenda

- **Background**
- Fagin's algorithm
- Threshold algorithm
- θ-approximation
- NRA algorithm
- Combined algorithm

Motivation: top-*k* queries

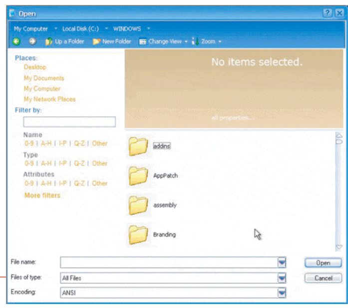
Find a girl with
long hair, brown eyes,
and sweet voice

Top-1 query with 3 attributes

Motivation: top-*k* queries

- Multimedia DB: "find 10 pictures that are funny and large in size"
- Info. retrieval: "find 100 papers that are most relevant to my research areas"
- Data stream: "find 5 users with the largest bandwidth usage"
- Live examples:
  - QBIC: wwwqbic.almaden.ibm.com
  - Flickr: www.flickr.com
  - WinFS for Windows Vista

WinFS for Windows Vista



Data model

	x_1	x_2	x_3
A	.1	.2	.7
B	.5	.3	.3
C	.1	.9	.4
D	.3	.1	.9
E	.8	.4	.6

→

L_1

E	.8
B	.5
D	.3
C	.1
A	.1

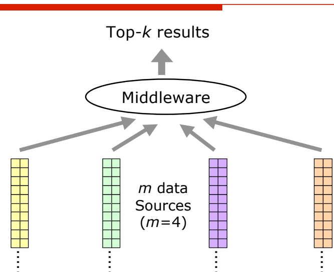
L_2

C	.9
E	.4
B	.3
A	.2
D	.1

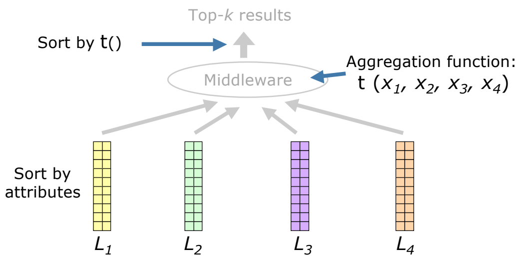
L_3

D	.9
A	.7
E	.6
C	.4
B	.3

Data integration sys. (middleware)

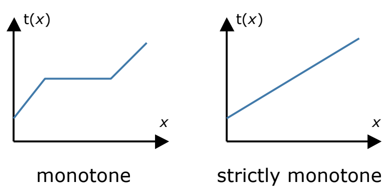


Problem definition

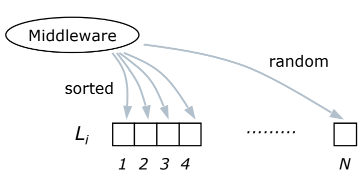


Aggregation functions

- Can be max(), min(), avg(), …



Sorted and random access

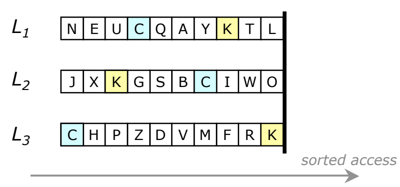


Agenda

- Background
- **Fagin's algorithm**
- Threshold algorithm
- θ-approximation
- NRA algorithm
- Combined algorithm

Fagin's algorithm (FA)

suppose m=3, k=2; objects are A, B, C, …, Z



Stop when there are k objects, such that each of them has been seen in each list.

Fagin's algorithm (FA): step 2

- For each object R has been seen:
  - Do *random access* to get all of its attributes.
  - Calculate t(R).
- Sort all these objects and output the first k objects.

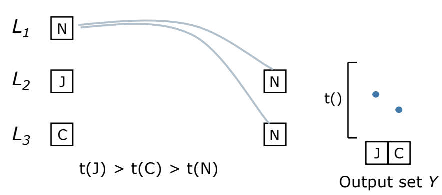
FA is *correct*, but not always *optimal*.

Agenda

- Background
- Fagin's algorithm
- **Threshold algorithm**
- θ-approximation
- NRA algorithm
- Combined algorithm

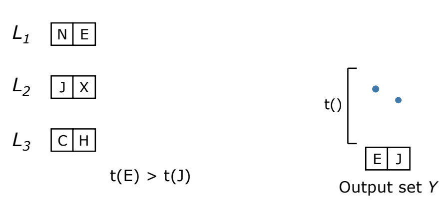
Threshold algorithm (TA)

m=3, *k*=2



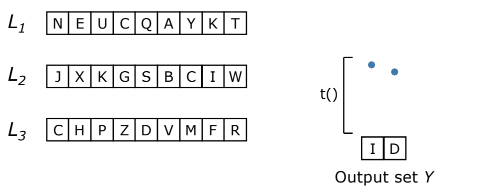
Threshold algorithm (TA)

m=3, *k*=2



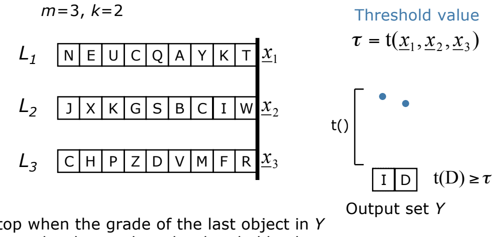
Threshold algorithm (TA)

m=3, k=2



Threshold algorithm (TA)

m=3, k=2



Stop when the grade of the last object in Y is equal or larger than the threshold value.

Middleware cost

- In this paper, we use middleware cost to measure optimality of an algorithm.
- To answer a query on database D, an algorithm A needs:
  - s sorted accesses
  - r random accesses
- The middleware cost of A on D *is*:

  cost(A,D)=sc_S+rc_R

Instance optimality

- A set of databases: **D**
- A set of (middleware) algorithms: **A**
- $B \in$ **A** is instance optimality if:

  cost(B,D) ≤ c·cost(A,D)+c'

  for every $A \in$ **A** and $D \in$ **D**

  c: optimality ratio

Instance optimality of TA

- Assumptions
  - t(): monotone
  - **D**: all
  - **A**: no wild guess
- Optimality: TA is instance optimal, with optimality ratio $m+m(m-1)c_R/c_S$

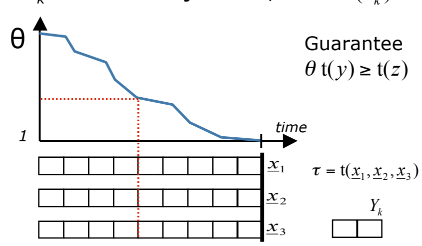
Instance optimality of TA (2)

- Assumptions
  - t(): strictly monotone
  - **D**: unique
  - **A**: all
- Optimality: TA is instance optimal

Agenda

θ-approximation

Y_k is the last object in Y, $\theta \equiv \tau / t(Y_k)$



Instance optimality of θ-approximation

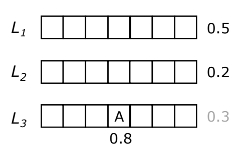
- Assumptions
  - t(): monotone
  - **D**: all
  - **A**: no wild guess
  - θ> 1
- Optimality: θ-approximation is instance optimal

Agenda

- Background
- Fagin's algorithm
- Threshold algorithm
- θ-approximation
- **No-Random-Access algorithm**
- Combined algorithm

Lower/upper bound of an object

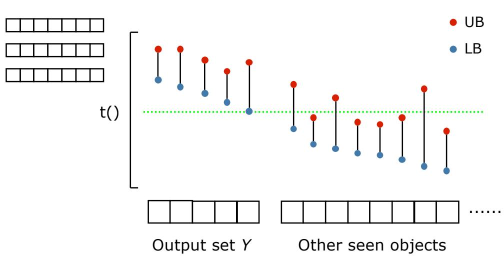
- Define Lower bound LB() as the value of t() when setting all unknown attributes to 0
- Define Upper bound UB() as the value of t() when setting all unknown attributes to $\underline{x}_i$



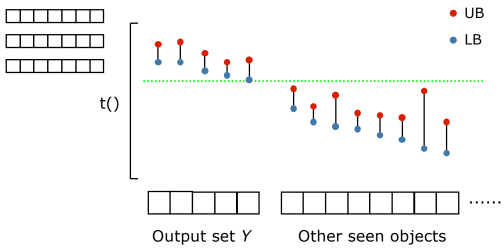
LB(A) =t(0, 0, 0.8)

UB(*A*)=t(0.5, 0.2, 0.8)

NRA algorithm



NRA algorithm



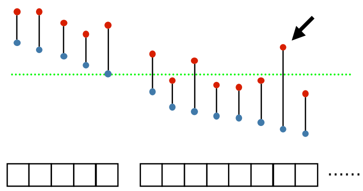
Instance optimality of NRA

- Assumptions
  - t(): monotone
  - **D**: all
  - **A**: no random access
- Optimality: NRA is instance optimal

Agenda

- Background
- Fagin's algorithm
- Threshold algorithm
- θ-approximation
- NRA algorithm
- **Combined algorithm**

Combined algorithm (CA)



For every $\lfloor c_R / c_S \rfloor$ step, obtain all unknown attr. of the object with the largest UB.

Instance optimality of CA

- Assumptions
  - t(): strictly monotone in each argument
  - **D**: unique
  - **A**: all
- Optimality: CA is instance optimal

Agenda

- Background
- Fagin's algorithm
- Threshold algorithm
- θ-approximation
- NRA algorithm
- Combined algorithm

Conclusion

- TA is instance optimal in most cases
- θ-approx: early stop
- NRA: random access is not allowed
- CA: random access is costly
- Future work
  - *Tightly* instance optimal
  - More efficient structure of NRA
  - Compare CA vs. TA

Discussion

- Object caching of TA
- Grades output of NRA
- Other metrics for algorithm optimality
- Assumptions on databases

Backup slides

I.O. in other fields

- Competitive analysis
- Approximation algorithms
- The mean of Monte Carlo estimation (Dagum et al.)
- Operations on sorted sets (Demaine et al.)

Memory overhead

- FA: need to remember t() for all objects that have been seen.
- TA: only need to remember t() for objects in *Y*.
- NRA: similar to FA.

Wild guess example

L_1
$(1,1)$
$(2,1)$
$(3,1)$
$\ldots$
$(n+1,1)$
$(n+2,0)$
$(n+3,0)$
$\ldots$
$(2n+1,0)$

L_2
$(2n+1,1)$
$(2n,1)$
$(2n-1,1)$
$\ldots$
$(n+1,1)$
$(n,0)$
$(n-1,0)$
$\ldots$
$(1,0)$

Instance Optimality w/ wild guess

- Assumptions
  - t(): min(x_1,x_2)
  - **D**: all
  - **A**: all
- Optimality: no algorithms is instance optimal

Wild guess example w/ θ

L_1
$(1,\cdot)$
$(2,\cdot)$
$\ldots$
$(n+1,\frac{1}{\theta})$
$(n+2,\frac{1}{2\theta^2})$
$(n+3,\cdot)$
$\ldots$
$(2n+1,\cdot)$

L_2
$(2n+1,\cdot)$
$(2n,\cdot)$
$\ldots$
$(n+1,\frac{1}{\theta})$
$(n,\frac{1}{2\theta^2})$
$(n-1,\cdot)$
$\ldots$
$(1,\cdot)$

Instance Optimality w/ wild guess

- Assumptions
  - t(): min(x_1,x_2)
  - **D**: unique
  - **A**: all
  - θ> 1
- Optimality: no algorithms is instance optimal

Costs for top-1 & top-2 for NRA

L_1
$(R, 1)$
$(\cdot, \frac{1}{3})$
$\dots$
$(\cdot, \frac{1}{3})$

L_2
$(\cdot, \frac{1}{3})$
$\dots$
$(\cdot, \frac{1}{3})$
$(R, 0)$

Instance optimality of CA (2)

- Assumptions
  - t(): min()
  - **D**: unique
  - **A**: all
- Optimality: CA is instance optimal

Instance optimality dependency

- Assumptions
  - t(): min()
  - **D**: all
  - **A**: no wild guess
- Optimality: no algorithm has instance optimality independent of C_R/C_S