Database Query Processing using Finite Cursor Machines

Jan Van den Bussche
Hasselt University

joint work with Martin Grohe, Yuri Gurevich, Dirk Leinders, Nicole Schweikardt, Jerzy Tyszkiewicz

Streaming/Sequential access to data

E.g. 2-pass database query processing:

1. sort the relations
2. do relational algebra by synchronized scans

E.g. information retrieval:

- inverted files
- do AND, OR, NOT by synchronized scans

$\Rightarrow$ relational algebra by information retrieval?

E.g. data stream model of computation:

- sequential access only
- limited # of passes
- limited memory
- sorting

Finite cursor machines (FCM)

Works on relational database (lists, not sets)

Fixed # of cursors on each relation

Cursors are 1-way

Fixed # of registers, store bitstrings

Built-in bitstring functions on data elements & bitstrings

Finite state control

Abstract State Machine (ASM)

Example

Sliding window join $R \bowtie_\theta S$

Window on $R = 50$, window on $S = 30$

Use 50 cursors on R, 30 on S

θ can be arbitrary

Computational completeness and restrictions

- Use bitstring functions for encoding data elements, concatenation

- Single scan loads entire DB in one bitstring

- Arbitrary computable bitstring function at the end

$\Rightarrow$ impose limitations on length of bitstrings in registers:

- $O(1)$-machines: do not store anything in registers

- $o(n)$-machines: registers cannot store entire DB

Positive results: $O(1)$-machines;
Negative results: $o(n)$-machines

Relational algebra

σ, π, $\cup$ are easy

$\bowtie$ in general impossible: quadratic size, but linear time

Even checking $R \cap S \neq \emptyset$ is impossible

Proof for $O(1)$-machines: $a_1 < a'_1 < a_2 < a'_2 < \cdots < a_n < a'_n$

- Ramsey's theorem to reduce built-in predicates to $<$ only
- $R = \{a_1, \ldots, a_n\}$, $S = \{a'_n, \ldots, a'_1\}$
- Fooling argument (can check only constant # of pairs)

Difference operator also impossible

Proof for $o(n)$-machines

For $I \subset \{1, \ldots, n\}$ define

$$A^I := \{a_i \mid i \in I\} \cup \{a'_i \mid i \notin I\}$$

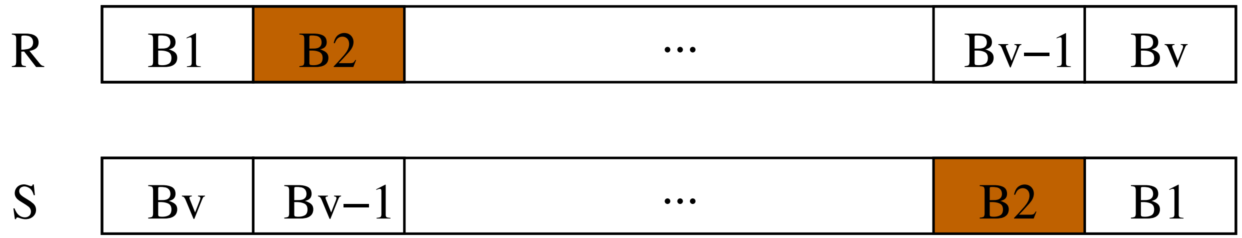
Then $A^I \cap A^J = \emptyset \Leftrightarrow J = \text{co}I$.

$\Rightarrow$ instance $D(I)$: $(R = A^I, S = A^{\text{co}I})$
with R sorted ascending, S sorted descending

There are 2^n such instances

Machine has k cursors, set $v := \binom{k}{2} + 1$

Machine can check at most $v - 1$ blocks



$2^n/v$ instances do not check some fixed block

$2^n/(v \cdot 2^{n-n/v})$ of those are equal outside that block

$2^n/(v \cdot 2^{n-n/v} \cdot (n^k \cdot 2^{r \cdot o(n)})^k)$ of those are in same state each time a cursor leaves the block in R or S

$\Rightarrow$ take I and J out of those

Machine cannot distinguish instance $(R = A^I, S = A^{\text{co}J})$ from instances $D(I)$ and $D(J)$

Sorted inputs

Difference operator, testing emptiness of $\bowtie$, become easy

Semijoin $\ltimes$ avoids quadratic output problem

Every **semijoin algebra query** can be computed by a **query plan** composed of **FCM's** and **sorting** operations

$\Rightarrow$ problem of avoiding **intermediate sorting**

Intermediate sorting

$$(R(A,B) - S(A,B)) \ltimes T(B,C)$$

- Stupid:

$$\mathsf{sort}_B(\mathsf{sort}_{A,B}(R) - \mathsf{sort}_{A,B}(S)) \ltimes \mathsf{sort}_B(T)$$

- Smarter:

$$(\mathsf{sort}_{B,A}(R) - \mathsf{sort}_{B,A}(S)) \ltimes \mathsf{sort}_B(T)$$

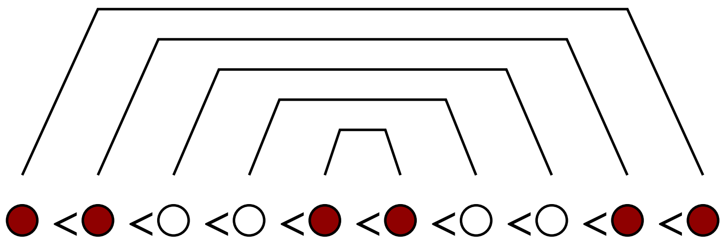
Can intermediate sorting always be avoided?

Note: FCM's are closed under composition

$\Rightarrow$ Is every semijoin algebra query computable by a single FCM on sorted inputs?

Ascending order only, $O(1)$-machines

Palindrome problem: given a word structure w over $\{0, 1\}$, equipped with a fully nested matching, is w a palindrome?



Expressible in semijoin algebra

Ascending order only: **not** solvable by $O(1)$-machine
[Proof: multihead finite automata]

Ascending & descending order: solvable by $O(1)$-machine

Strongest negative result

"RST-query" $= R(A) \ltimes (S(A,B) \ltimes T(B))$

Nonemptiness of RST-query is not solvable by an $o(n)$-machine on sorted inputs in ascending & descending orders

- built-in functions arbitrary

- "simplest possible counterexample"

Proof: similar to checking $R \cap S \neq \emptyset$

Further remarks

FCM's on sorted inputs can do:

- more relational algebra than just semijoin algebra
- more queries than relational algebra with counting

Open problem: Can query plans composed of FCM's and sorting operators compute **all** boolean relational algebra (= first-order logic) queries?

We conjecture **no**, and for $O(1)$-case, nonuniform parameterized complexity theory seems to agree with us

Unlikely that FO is in time $n \log n$

Conclusion

Theoretical computation model inspired by classical database query processing

New twist on streaming model

Fixed # of cursors, registers, can be relaxed

Semijoin algebra as natural “linear” fragment of relational algebra