Database cracking

Stratos Idreos, Martin Kersten and Stefan Manegold
CWI Amsterdam, The Netherlands
The open problem

**Dynamic** environments

What **kind** of indices should be used, **when** and on **which** data?

Database experts or special tools monitor the system

More difficult in databases with **huge** datasets
Database cracking

We explore self-organization

Each query triggers physical re-organization of the database

We designed and implemented a DBMS using database cracking

We work on top of MonetDB, a column oriented database system
select A>5 and A<10
Cracking a database

select A > 5 and A < 10
Cracking a database

select A>5 and A<10

17 13 12 15 4 6 2 8 3 4

3 8 6
select A > 5 and A < 10
Cracking a database

select A>5 and A<10

[Diagram showing selection criteria]

- <= 5: 3, 4, 2
- > 5: 6, 8
- >= 10: 12, 15, 17, 13, 15
Cracking a database

select A > 5 and A < 10

Improve data access for future queries

<= 5

> 5

>= 10
Cracking a database

select A > 5 and A < 10

17
3
8
6
2
12
13
4
15

select A > 2 and A < 15

<= 5

3
4
2
8
6
12
15
17
13

> 5

>= 10

Improve data access for future queries
Cracking a database

- Improve data access for future queries

- Select A > 5 and A < 10
- Select A > 2 and A < 15

- Values
  - <= 5: 3, 4, 2
  - > 5: 8, 6
  - >= 10: 12, 15, 17, 13
Cracking a database

**Improve data access for future queries**

- **Select A > 5 and A < 10**
  - <= 5
  - > 5
  - >= 10
- **Select A > 2 and A < 15**
  - 2
  - 4
  - 3
Cracking a database

select A>5 and A<10

select A>2 and A<15

Improve data access for future queries
Cracking a database

- Improve data access for future queries

1. Select A > 5 and A < 10
2. Select A > 2 and A < 15

17  15  13  12  11  10  9  8  7  6  5  4  3  2

17  15  13  12  11  10  9  8  7  6  5  4  3  2

17  15  13  12  11  10  9  8  7  6  5  4  3  2

17  15  13  12  11  10  9  8  7  6  5  4  3  2
Cracking a database

Improve data access for future queries

select A>5 and A<10

select A>2 and A<15

< 5

>= 10

<= 2

> 5

> 2

>= 10

>= 15
Cracking a database

Improve data access for future queries

select A>5 and A<10

select A>2 and A<15
Cracking a database

**Improve data access for future queries**

- Select A > 5 and A < 10
  - > 5
  - >= 10

- Select A > 2 and A < 15
  - <= 2
  - > 2
  - > 5
  - >= 10
  - >= 15

**The more we crack the more we learn**

- The more we improve data access for future queries, the more we learn.

Cracking algorithms

There are two types of cracking algorithms:

- Split a piece in **two** new pieces
- Split a piece in **three** new pieces
The **first** time a range query is posed on an attribute $A$, a cracking DBMS makes a **copy** of column $A$, called the **cracker column** of $A$.

A cracker column is **continuously** physically re-organized based on queries that **need** to touch attribute such as the result is in a contiguous space.

For each cracker column, there is a **cracker index**.
The cracker select operator

- The simple select operator:
  1. Scans the column
  2. Return a new column that contains qualifying values

- The crackers select operator:
  1. Searches the cracker index
  2. Physically re-organizes pieces found
  3. Update the cracker index
  4. Return a slice of the cracker column as result

- More steps but faster because we analyze less data
Testing the select operator
Research and opportunities for cracking

- Optimization
  - optimal piece size / granularity / index (avl) depth
- Exploit cracking for join queries, aggregate queries etc.
- Concurrency issues
- Cracking histograms
- Distributed cracking
- A priori cracking
- ...

...
Research and opportunities for cracking

- Optimization
  - optimal piece size / granularity / index (avl) depth

- Exploit cracking for join queries, aggregate queries etc.

- Concurrency issues

- Cracking histograms

- Distributed cracking

- A priori cracking

THANK YOU!
How cracking compares to a sorting strategy?
- Sort data upfront and then use binary search

For a sorting strategy we have to make an investment upfront

Sorting needs prior knowledge of query workload

Similar arguments stand for indices
Impact on query plan

select R.c from R where 5 < R.a < 10 and 9 < R.b < 20
Impact on query plan

`select R.c from R where 5 < R.a < 10 and 9 < R.b < 20`

crackers.select(Ra, 5, 10)  
crackers.select(Rb, 9, 20)

`algebra.OIDintersect(Ra1, Rb1)`

`algebra.fetch(Ra2, Rc)`
Impact on query plan

`select R.c from R where 5 < R.a < 10 and 9 < R.b < 20`
Impact on query plan

select R.c from R where 5 < R.a < 10 and 9 < R.b < 20

Ra1

`crackers.select(Ra, 5, 10)`

Ra2

`algebra.joinselect(Ra1, Rb, 9, 20)`

`algebra.fetch(Ra2, Rc)`
Scalability
TPC-H query 6