Cost Estimation of Relational Operations

- Cost estimation is important to improve the performance of relational operations
- There are two primary factors affecting the performance of relational operations:
  - Computation costs in CPU
  - I/O costs
- Commonly, I/O costs are dominant
  - 'ms' order
    - HDD: seek + rotational latency + data transfer
  - During a disk access, millions steps of a program can be executed in a recent CPU

Notations for the Cost Estimation

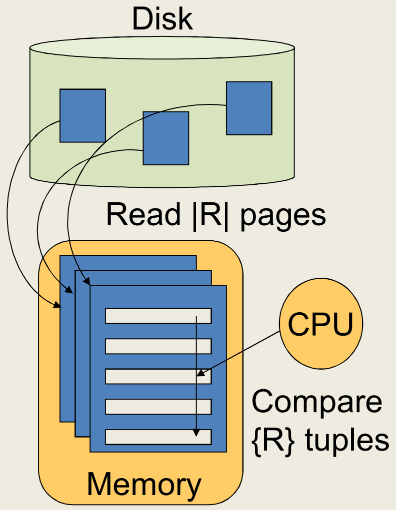
- We estimate the CPU costs by calculating the number of tuples to be handled and the disk access costs by calculating the number of disk pages to be accessed
- We use two notations
  - {*R*} : The number of tuples in the relation *R* (Cardinality)
    - To measure CPU costs
  - |*R*| : The number of disk pages for storing the relation *R*
    - To measure disk access costs

Relational Algebra Operations

- Dedicated Relational Algebra Operations
  - Selection ($\sigma_{conditions}\ R$)
  - Projection ($\pi_{attribute\text{-}list}\ R$)
  - Join (θ -Join: $R \bowtie_{r\theta s} S$, eq-Join: $R \bowtie_{r=s} S$)
  - Division ($R \div S$)
- Set Operations
  - Union ($R \cup S$)
  - Intersection ($R \cap S$)
  - Difference ($R - S$)
  - Cartesian Product (R x S)

Execution of a Selection Operation

- Selection ($\sigma_{conditions}\ R$)
  - Assuming to select one tuple satisfying the specified conditions
- If there is no index, nor the target attribute is not a key
  - Comparisons: {R}
  - Disk I/Os: |R|



Pseudo Code for Selection

```
for (i = 1; i<|R|; i++) {
  read i-th page from disks into a buffer;
  f := true;
  while (f) {
    get a tuple t from the buffer;
    if t is empty then f := false;
    else {
      get the target attribute value v from t;
      if v satisfies the condition then output t;
    }
  }
}
```

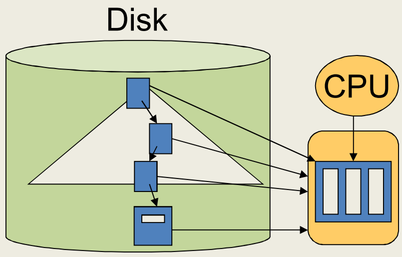
Read |*R*| pages in total

Compare {*R*} tuples in total

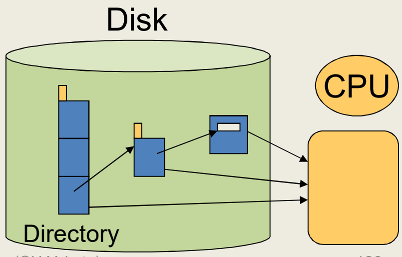
2020/7/13 Advance Data Engineering (©H.Yokota) 125

Selection with Indices

- If there is a index for the target attribute
- B⁺-tree
  - Disk I/Os: Height of the Tree + 1
  - Comparisons: Height of the Tree (*H*) x Search in a node by binary search ($\log_2 F^*$)



- Extensible Hashing
  - Disk I/Os: Directory + Local hash table + 1
  - Comparisons: constant



2020/7/13 Advance Data Engineering (©H.Yokota) 126

Range Queries

- In many cases, we need tuples satisfying conditions expressed as a range, such as 10<x<20
- B^{+}-tree is suitable for the range queries
  - Links between leaves are used for traversing adjacent values
    - At first, search for a tuple much with the lower bound (10), then traverse links to search for another tuple much with the higher bound (20)
      - Tuples between these bounds are the results
- Hash based indices are not effective for the range queries
  - Hash functions cannot keep adjacent information

Question (6-1)

- There have been proposed a number of methods to handle multi-attribute indices. One of them is a composite key index using B+-tree structure. For a case of two attribute, we have two types of orders for attribute X and Y:
  - (X, Y): {(a, 10) < (a, 50) < (b, 30) < (c, 20) < (d, 40)}
  - (Y, X) : {(10, a) < (20, c) < (30, b) < (40, d) < (50, a)}
- Both orders are effective for the condition of a < X < d, and 10 < Y < 50.
- Consider factors for deciding the order.

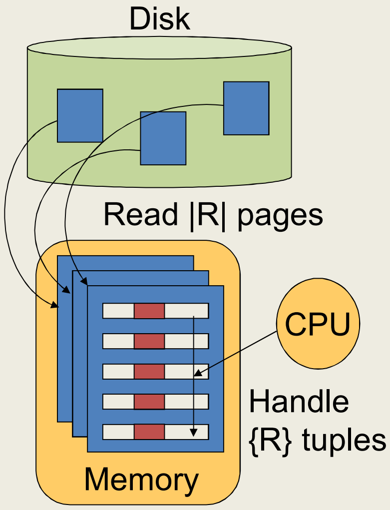
Projection Operations

- Projection ($\pi_{attribute\text{-}list}\ R$)
  - At first, we focus on the Delta Projection to concentrate on the cost for attribute extraction
    - We can ignore the cost for eliminating duplication
  - Then we consider the execution of Duplication Elimination for Projection
    - Three types of algorithms: Nested Loop, Sort, Hash
- Delta Projection
  - All pages should be read, and all tuples in those pages should be handled to extract the target attributes

2020/7/13 Advance Data Engineering (©H.Yokota) 129

Delta Projection

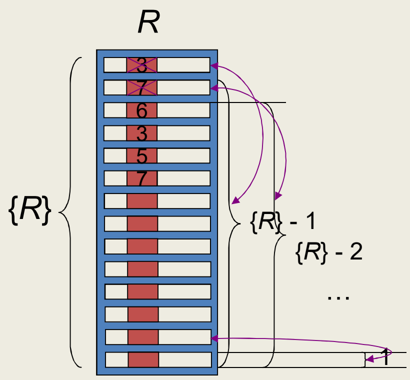
- Costs:
  - Almost same as it for the selection except comparisons
  - Disk I/Os: $|R|$
  - Handled tuples: $\{R\}$
- Index has no effect for Projection operations
  - All tuples has to be accessed to extract desired attributes



2020/7/13 Advance Data Engineering (©H.Yokota) 130

Nested Loop base Duplication Elimination (NLDE)

- At first, we concentrate on a straightforward algorithm ignoring disk I/Os
- Search the same value from the top by scanning rest tuples
- Comparisons:
  - ({*R*} - 1) + ({*R*} - 2) +…+ 1
  - ({*R*} x ({*R*} - 1))/ 2



2020/7/13 Advance Data Engineering (©H.Yokota) 131

Pseudo Code for NLDE

```
for (i = 1; i ≤ {R}; i++) {          ← Outer Loop
  f := true;
  get i-th tuple t_x and attribute value v_x in t_x;
  for (j = i +1: j ≤ {R}; j++) {     ← Inner Loop
    get j-th tuple t_y and attribute value v_y in t_y;
    if v_x == v_y then f := false;
  }
  if f then output t_x
}
```

- This is a very straightforward algorithm and you can apply many optimizations on it

2020/7/13 Advance Data Engineering (©H.Yokota) 132

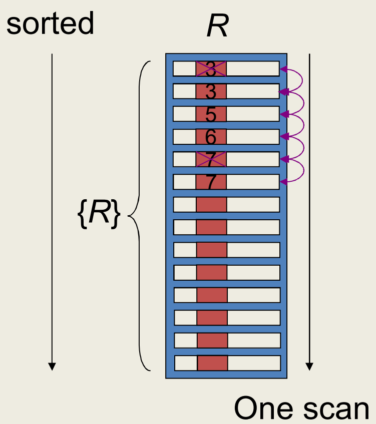
Costs for NLDE with Disk I/Os

- The costs depend on the size of buffer
- If the buffer size is enough to keep all disk pages into the memory:
  - Disk I/Os: $|R|$
- If the buffer size is just the same as a disk page:
  - We have to apply the nested loop algorithm for disk I/Os
  - Disk I/Os : $(|R| \times (|R| + 1)) / 2$

2020/7/13 Advance Data Engineering (©H.Yokota) 133

Sort based Duplication Elimination (SDE)

- If the tuples are sorted by the target attribute, it becomes very easy to check duplications
- Here, we first ignore disk I/Os again
- Compare continuous two tuples
- Duplications can be eliminated by one scan



2020/7/13 Advance Data Engineering (©H.Yokota) 134

Pseudo Code for SDE

for (i = 1; $i \leq \{R\}$; i++) { — One scan

get i-th tuple t_x and attribute value v_x in t_x;

if (i < {R}) then {

get (i+1)-th tupe t_y and attribute value v_Y in t_y;

if $v_x \neq v_y$ then output t_x;}

else output t_x;}

- Costs to eliminate duplications: $\{R\}$
- If the sorted tuples are stored into disk pages, the number of disk I/Os is $|R|$
- But, it does not contain costs for sorting tuples

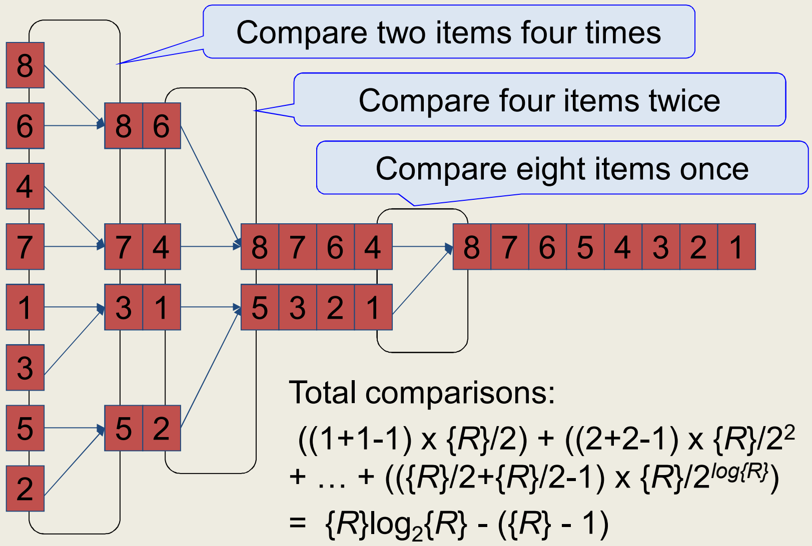
2020/7/13 Advance Data Engineering (©H.Yokota) 135

Sort Algorithms for DB Operations

- There are many sort algorithms
  - Bubble Sort, Bitonic Sort, Quick Sort, N-Way Merge Sort, and so on
- The Quick Sort is not suitable for database operations containing disk I/Os, while the N-Way Merge Sort is commonly used for them
  - Why? (Order of both algorithm is identical: O(n*log(n))

2020/7/13 Advance Data Engineering (©H.Yokota) 136

2-Way Merge Sort



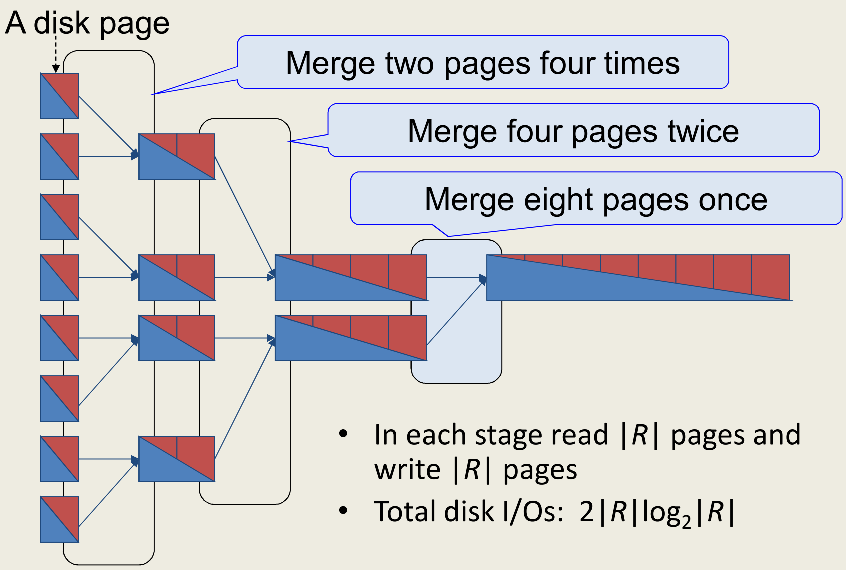
Total comparisons:

$$((1+1-1) \times \{R\}/2) + ((2+2-1) \times \{R\}/2^2 + \ldots + ((\{R\}/2+\{R\}/2-1) \times \{R\}/2^{\log\{R\}})$$

$$= \{R\}\log_2\{R\} - (\{R\} - 1)$$

2020/7/13 Advance Data Engineering (©H.Yokota) 137

2-Way Page Merge Sort



- In each stage read $|R|$ pages and write $|R|$ pages
- Total disk I/Os: $2|R|\log_2|R|$

2020/7/13 Advance Data Engineering (©H.Yokota) 138

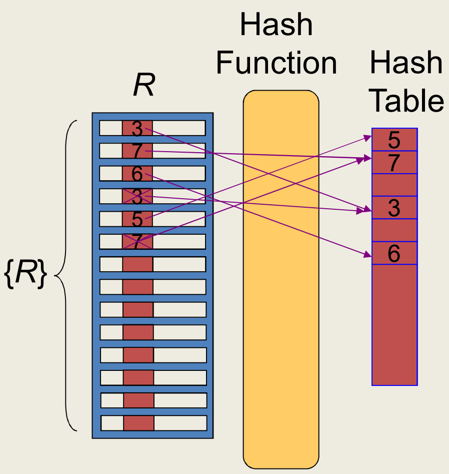
Costs of SDE with Disk I/Os

- At first, sort all pages with the 2-way merge sort, then eliminate duplications
  - Comparisons :
    - Merge sort: $\{R\}\log_2\{R\}$ - ($\{R\}$ - 1)
    - Duplication Elimination: $\{R\}$
    - Total: $\{R\}\log_2\{R\}$ +1
  - Disk I/Os :
    - Merge Sort: $2|R|\log_2|R|$
    - Duplication Elimination: $|R|$
      - Which can be omitted by on the fly processing
    - Write the result into disk: $|R|$
    - Total: $2|R|(\log_2|R|+1)$

2020/7/13 Advance Data Engineering (©H.Yokota) 139

Hash based Duplication Elimination (HDE)

- Apply a hash function for each tuple
- If there is no hash collision, results can be derived by one scan (optimistic)
  - Comparison : $\{R\}$
  - Disk I/O : $|R|$
- When the size of hash table is small, many collisions occur
- When the size of hash table is large, it cannot be stored on memory
- The costs for generating large extensible hashing is high



2020/7/13 Advance Data Engineering (©H.Yokota) 140

Pseudo Code of Hash base DE

```
for (i = 1; i ≤ {R}; i++) {
 get i-th tuple t and attribute value v in t;
 x = h(v);                      Applying the hash function
 if ht[x] is empty then {
   ht[x] := t;                  Insert tuple in the hash table
   output t;
 }
}
```

* Strictly, we have to consider the treatment of Hash collisions

Question (6-2)

- To handle hash collision, explain which portion in the pseudo code should be modified into what type of codes.

Costs for a Projection Operation

- Costs for duplication elimination (DE)
  + costs for extracting attributes (EA)

	Disk I/Os	Comparisons
NLDE + EA (large buffer)	$2\|R\|$	$(\{R\} \times (\{R\} - 1))/ 2$
NLDE + EA (a page size buffer)	$(\|R\| \times (\|R\| + 1)) / 2 + \|R\|$	$(\{R\} \times (\{R\} - 1))/ 2$
SDE + EA	$\|R\|(2\log_2\|R\|+3)$	$\{R\}\log_2\{R\} +\{R\} - 1$
HDE + EA (Optimistic)	$2\|R\|$	$\{R\}$