

CSE 444: Database Internals

Lecture 8 Operator Algorithms (part 2)

CSE 444 - Winter 2019

1

Announcements

- Lab 2 released – new branch “lab2” contains instructions
- Lab 2 / part 1 due on Thursday
 - We will not run any tests – So bugs are OK
- Homework 2 due on Friday
- Paper review for master's due on Friday

CSE 444 - Winter 2019

2

Outline

- **Join operator algorithms**
 - One-pass algorithms (Sec. 15.2 and 15.3)
 - Index-based algorithms (Sec 15.6)
 - Two-pass algorithms (Sec 15.4 and 15.5)

CSE 444 - Winter 2019

3

Index Based Selection

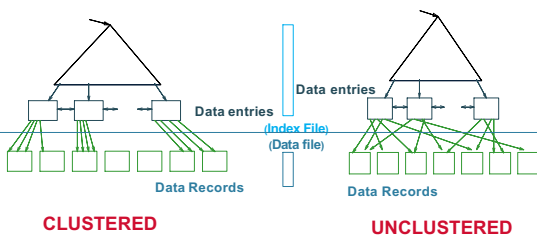
Selection on equality: $\sigma_{a=v}(R)$

- $B(R)$ = size of R in blocks
- $T(R)$ = number of tuples in R
- $V(R, a)$ = # of distinct values of attribute a

CSE 444 - Winter 2019

4

B+ Trees



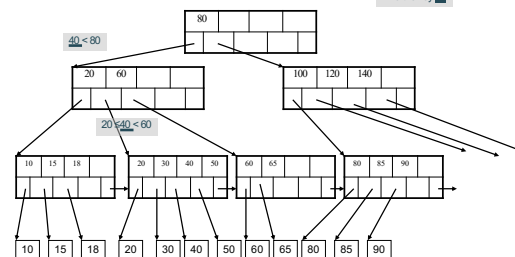
Note: can also store data records directly as data entries

CSE 444 - Winter 2019

5

B+ Tree Example

$d = 2$



CSE 444 - Winter 2019

6

Index Based Selection

Selection on equality: $\sigma_{a=v}(R)$

- $B(R)$ = size of R in blocks
- $T(R)$ = number of tuples in R
- $V(R, a)$ = # of distinct values of attribute a

What is the cost in each case?

- Clustered index on **a**:
- Unclustered index on **a**:

CSE 444 - Winter 2019 7

Index Based Selection

Selection on equality: $\sigma_{a=v}(R)$

- $B(R)$ = size of R in blocks
- $T(R)$ = number of tuples in R
- $V(R, a)$ = # of distinct values of attribute a

What is the cost in each case?

- Clustered index on **a**: $B(R)/V(R,a)$
- Unclustered index on **a**: $T(R)/V(R,a)$

CSE 444 - Winter 2019 8

Index Based Selection

Selection on equality: $\sigma_{a=v}(R)$

- $B(R)$ = size of R in blocks
- $T(R)$ = number of tuples in R
- $V(R, a)$ = # of distinct values of attribute a

What is the cost in each case?

- Clustered index on **a**: $B(R)/V(R,a)$
- Unclustered index on **a**: $T(R)/V(R,a)$

Note: we ignore I/O cost for index pages

CSE 444 - Winter 2019 9

Index Based Selection

- Example:

$B(R) = 2000$
$T(R) = 100,000$
$V(R, a) = 20$

cost of $\sigma_{a=v}(R) = ?$
- Table scan:
- Index based selection:

CSE 444 - Winter 2019 10

Index Based Selection

- Example:

$B(R) = 2000$
$T(R) = 100,000$
$V(R, a) = 20$

cost of $\sigma_{a=v}(R) = ?$
- Table scan: $B(R) = 2,000$ I/Os
- Index based selection:

CSE 444 - Winter 2019 11

Index Based Selection

- Example:

$B(R) = 2000$
$T(R) = 100,000$
$V(R, a) = 20$

cost of $\sigma_{a=v}(R) = ?$
- Table scan: $B(R) = 2,000$ I/Os
- Index based selection:
 - If index is clustered:
 - If index is unclustered:

CSE 444 - Winter 2019 12

Index Based Selection

- Example: $B(R) = 2000$
 $T(R) = 100,000$
 $V(R, a) = 20$ cost of $\sigma_{a=v}(R) = ?$
- Table scan: $B(R) = 2,000$ I/Os
- Index based selection:
 - If index is clustered: $B(R)/V(R,a) = 100$ I/Os
 - If index is unclustered:

CSE 444 - Winter 2019 13

Index Based Selection

- Example: $B(R) = 2000$
 $T(R) = 100,000$
 $V(R, a) = 20$ cost of $\sigma_{a=v}(R) = ?$
- Table scan: $B(R) = 2,000$ I/Os
- Index based selection:
 - If index is clustered: $B(R)/V(R,a) = 100$ I/Os
 - If index is unclustered: $T(R)/V(R,a) = 5,000$ I/Os

CSE 444 - Winter 2019 14

Index Based Selection

- Example: $B(R) = 2000$
 $T(R) = 100,000$
 $V(R, a) = 20$ cost of $\sigma_{a=v}(R) = ?$
- Table scan: $B(R) = 2,000$ I/Os !
- Index based selection:
 - If index is clustered: $B(R)/V(R,a) = 100$ I/Os
 - If index is unclustered: $T(R)/V(R,a) = 5,000$ I/Os !

CSE 444 - Winter 2019 15

Index Based Selection

- Example: $B(R) = 2000$
 $T(R) = 100,000$
 $V(R, a) = 20$ cost of $\sigma_{a=v}(R) = ?$
- Table scan: $B(R) = 2,000$ I/Os
- Index based selection:
 - If index is clustered: $B(R)/V(R,a) = 100$ I/Os
 - If index is unclustered: $T(R)/V(R,a) = 5,000$ I/Os

Lesson: Don't build unclustered indexes when $V(R,a)$ is small !

CSE 444 - Winter 2019 16

Index Nested Loop Join

$R \bowtie S$

- Assume S has an index on the join attribute
- Iterate over R, for each tuple fetch corresponding tuple(s) from S
- Previous nested loop join: cost
 - $B(R) + T(R) \cdot B(S)$
- **Index Nested Loop Join Cost:**
 - If index on S is clustered: $B(R) + T(R) \cdot B(S) / V(S,a)$
 - If index on S is unclustered: $B(R) + T(R) \cdot T(S) / V(S,a)$

CSE 444 - Winter 2019 17

Outline

- **Join operator algorithms**
 - One-pass algorithms (Sec. 15.2 and 15.3)
 - Index-based algorithms (Sec 15.6)
 - Two-pass algorithms (Sec 15.4 and 15.5)

CSE 444 - Winter 2019 18

Two-Pass Algorithms

- What if data does not fit in memory?
- Need to process it in multiple passes

- Two key techniques
 - Sorting
 - Hashing

CSE 444 - Winter 2019

19

Basic Terminology

- A run in a sequence is an increasing subsequence

- What are the runs?
2, 4, 99, 103, 88, 77, 3, 79, 100, 2, 50

CSE 444 - Winter 2019

20

Basic Terminology

- A run in a sequence is an increasing subsequence

- What are the runs?

2, 4, 99, 103, 88, 77, 3, 79, 100, 2, 50

CSE 444 - Winter 2019

21

External Merge-Sort: Step 1

Phase one: load M blocks in memory, sort, send to disk, repeat

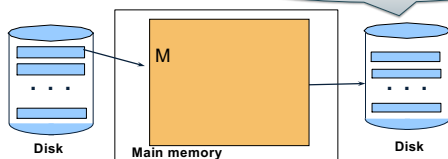
CSE 444 - Winter 2019

22

External Merge-Sort: Step 1

Phase one: load M blocks in memory, sort, send to disk, repeat

Q: How long are the runs?



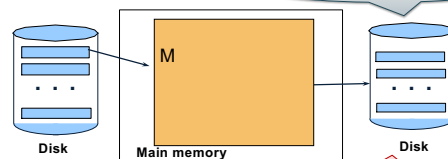
CSE 444 - Winter 2019

23

External Merge-Sort: Step 1

Phase one: load M blocks in memory, sort, send to disk, repeat

Q: How long are the runs?



CSE 444 - Winter 2019

24

External Merge-Sort: Step 1

Phase one: load M blocks in memory, sort, send to disk, repeat

Q: How long are the runs?

A: Length = M blocks

Sidenote: Can increase to length 2M using "replacement selection" (details in book)

25

External Merge-Sort: Step 2

Phase two: merge M runs into a bigger run

- Merge M - 1 runs into a new run
- Result: runs of length M (M - 1) ≈ M²

CSE 444 - Winter 2019

26

Example

- Merging three runs to produce a longer run:

0, 14, 33, 88, 92, 192, 322
 2, 4, 7, 43, 78, 103, 523
 1, 6, 9, 12, 33, 52, 88, 320

Output:
 0

CSE 444 - Winter 2019

27

Example

- Merging three runs to produce a longer run:

0, 14, 33, 88, 92, 192, 322
 2, 4, 7, 43, 78, 103, 523
 1, 6, 9, 12, 33, 52, 88, 320

Output:
 0, ?

CSE 444 - Winter 2019

28

Example

- Merging three runs to produce a longer run:

0, 14, 33, 88, 92, 192, 322
 2, 4, 7, 43, 78, 103, 523
 1, 6, 9, 12, 33, 52, 88, 320

Output:
 0, 1, ?

CSE 444 - Winter 2019

29

Example

- Merging three runs to produce a longer run:

0, 14, 33, 88, 92, 192, 322
 2, 4, 7, 43, 78, 103, 523
 1, 6, 9, 12, 33, 52, 88, 320

Output:
 0, 1, 2, 4, 6, 7, ?

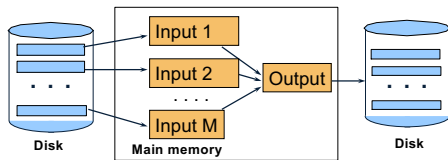
CSE 444 - Winter 2019

30

External Merge-Sort: Step 2

Phase two: merge M runs into a bigger run

- Merge $M - 1$ runs into a new run
- Result: runs of length M ($M - 1$) $\approx M^2$



If approx. $B \leq M^2$ then we are done 31

Cost of External Merge Sort

- Assumption: $B(R) \leq M^2$
- Read+write+read = $3B(R)$

CSE 444 - Winter 2019

32

Discussion

- What does $B(R) \leq M^2$ mean?
- How large can R be?

CSE 444 - Winter 2019

33

Discussion

- What does $B(R) \leq M^2$ mean?
- How large can R be?
- Example:
 - Page size = 32KB
 - Memory size 32GB: $M = 10^6$ -pages

CSE 444 - Winter 2019

34

Discussion

- What does $B(R) \leq M^2$ mean?
- How large can R be?
- Example:
 - Page size = 32KB
 - Memory size 32GB: $M = 10^6$ pages
- R can be as large as 10^{12} pages
 - 32×10^{15} Bytes = 32 PB

CSE 444 - Winter 2019

35

Merge-Join

- Join $R \bowtie S$
- How?....

CSE 444 - Winter 2019

36

Merge-Join

Join $R \bowtie S$

- Step 1a: generate initial runs for R
- Step 1b: generate initial runs for S
- Step 2: merge and join
 - Either merge first and then join
 - Or merge & join at the same time

CSE 444 - Winter 2019

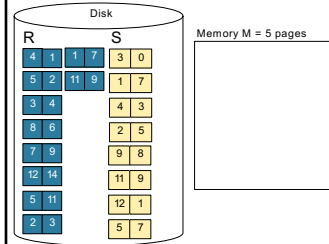
37

Merge-Join Example

Setup: Want to join R and S

- Relation R has 10 pages with 2 tuples per page
- Relation S has 8 pages with 2 tuples per page

Values shown are values of join attribute for each given tuple

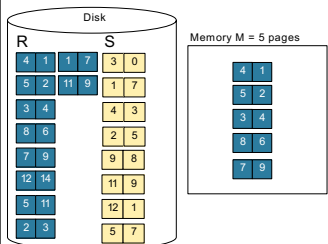


CSE 444 - Winter 2019

38

Merge-Join Example

Step 1: Read M pages of R and sort in memory

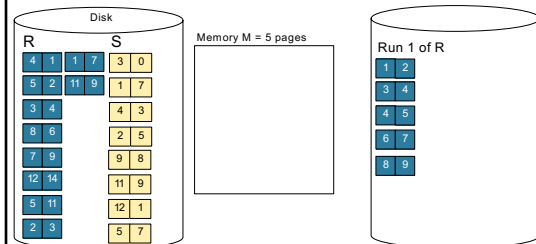


CSE 444 - Winter 2019

39

Merge-Join Example

Step 1: Read M pages of R and sort in memory, then write to disk

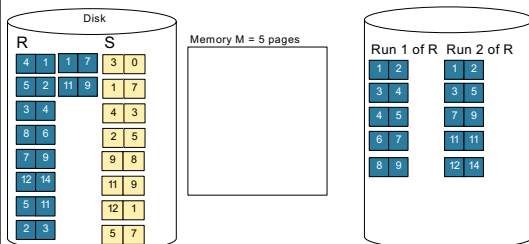


CSE 444 - Winter 2019

40

Merge-Join Example

Step 1: Repeat for next M pages until all R is processed

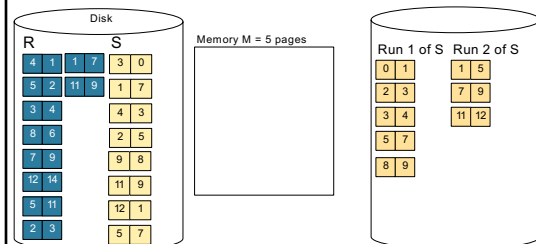


CSE 444 - Winter 2019

41

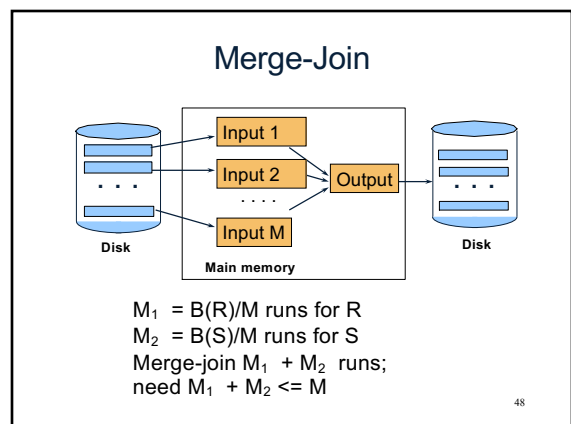
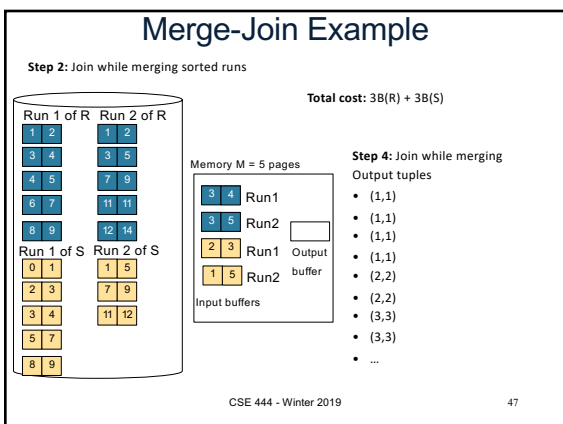
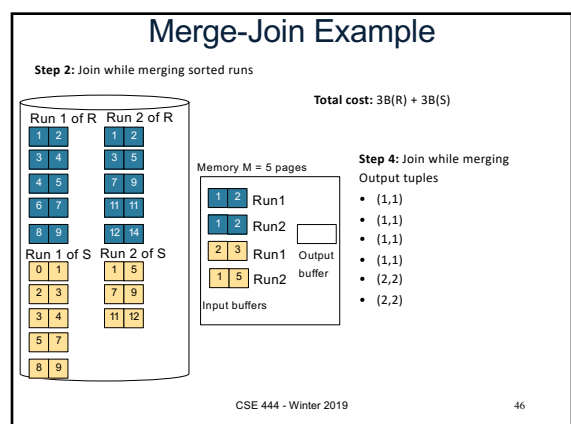
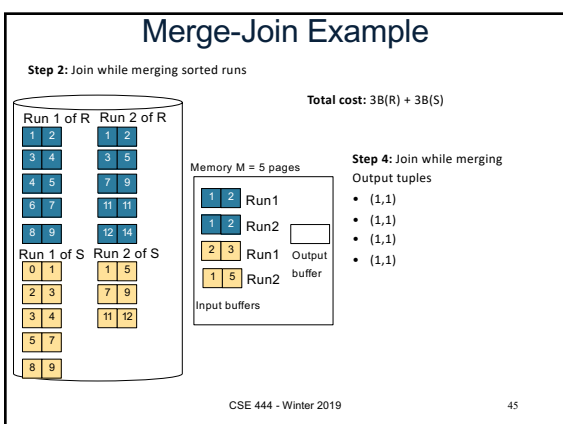
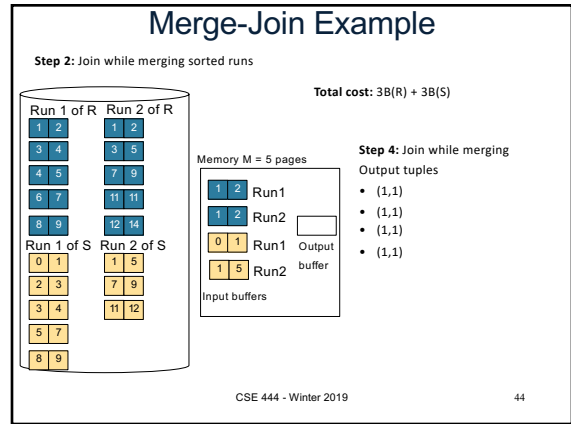
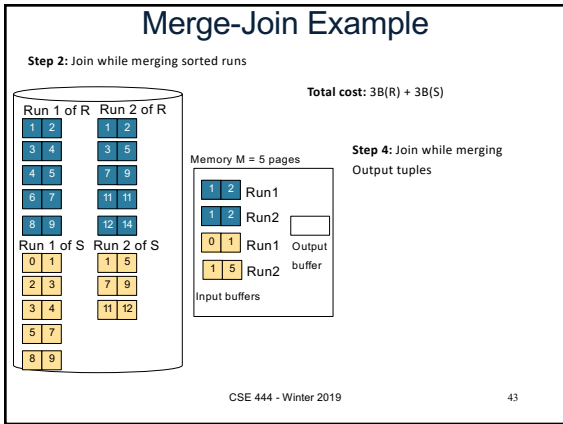
Merge-Join Example

Step 1: Do the same with S



CSE 444 - Winter 2019

42



Partitioned Hash Algorithms

- Partition R into k buckets:
 $R_1, R_2, R_3, \dots, R_k$

CSE 444 - Winter 2019

49

Partitioned Hash Algorithms

- Partition R into k buckets:
 $R_1, R_2, R_3, \dots, R_k$
- Assuming $B(R_1)=B(R_2)=\dots=B(R_k)$, we have
 $B(R_i) = B(R)/k$, for all i

CSE 444 - Winter 2019

50

Partitioned Hash Algorithms

- Partition R into k buckets:
 $R_1, R_2, R_3, \dots, R_k$
- Assuming $B(R_1)=B(R_2)=\dots=B(R_k)$, we have
 $B(R_i) = B(R)/k$, for all i
- Goal: each R_i should fit in main memory:
 $B(R_i) \leq M$

CSE 444 - Winter 2019

51

Partitioned Hash Algorithms

- Partition R into k buckets:
 $R_1, R_2, R_3, \dots, R_k$
- Assuming $B(R_1)=B(R_2)=\dots=B(R_k)$, we have
 $B(R_i) = B(R)/k$, for all i
- Goal: each R_i should fit in main memory:
 $B(R_i) \leq M$

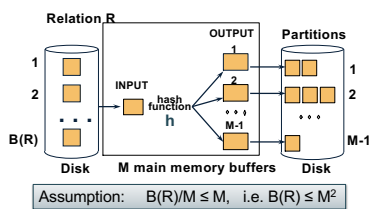
How do we choose k?

CSE 444 - Winter 2019

52

Partitioned Hash Algorithms

- We choose $k = M-1$. Each bucket has size approx.
 $B(R)/(M-1) \approx B(R)/M$



CSE 444 - Winter 2019

53

Grace-Join

$R \bowtie S$

Note: grace-join is also called partitioned hash-join

CSE 444 - Winter 2019

54

Grace-Join

$R \bowtie S$

- Step 1:
 - Hash S into M buckets
 - Send all buckets to disk
- Step 2
 - Hash R into M buckets
 - Send all buckets to disk
- Step 3
 - Join every pair of buckets

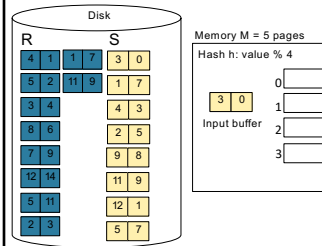
Note: grace-join is also called *partitioned hash-join*

CSE 444 - Winter 2019

55

Partitioned Hash-Join Example

Step 1: Read relation S one page at a time and hash into M-1 (=4 buckets)

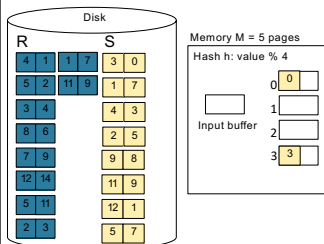


CSE 444 - Winter 2019

56

Partitioned Hash-Join Example

Step 1: Read relation S one page at a time and hash into the 4 buckets

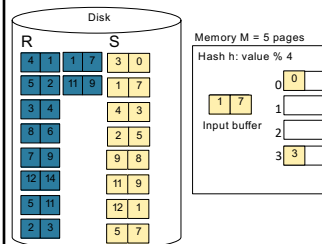


CSE 444 - Winter 2019

57

Partitioned Hash-Join Example

Step 1: Read relation S one page at a time and hash into the 4 buckets

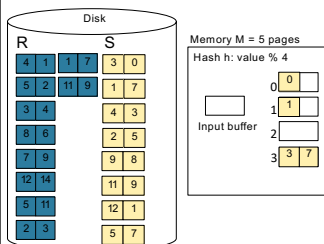


CSE 444 - Winter 2019

58

Partitioned Hash-Join Example

Step 1: Read relation S one page at a time and hash into the 4 buckets

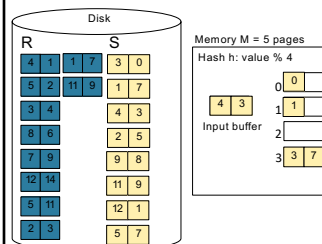


CSE 444 - Winter 2019

59

Partitioned Hash-Join Example

Step 1: Read relation S one page at a time and hash into the 4 buckets

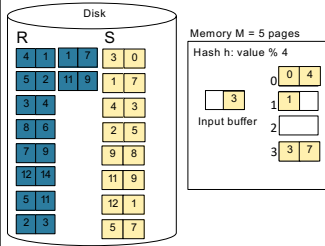


CSE 444 - Winter 2019

60

Partitioned Hash-Join Example

Step 1: Read relation S one page at a time and hash into the 4 buckets
When a bucket fills up, flush it to disk

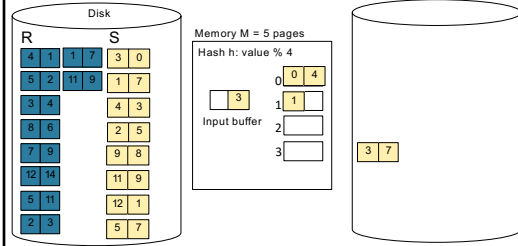


CSE 444 - Winter 2019

61

Partitioned Hash-Join Example

Step 1: Read relation S one page at a time and hash into the 4 buckets
When a bucket fills up, flush it to disk

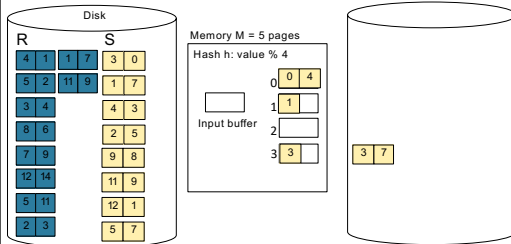


CSE 444 - Winter 2019

62

Partitioned Hash-Join Example

Step 1: Read relation S one page at a time and hash into the 4 buckets
When a bucket fills up, flush it to disk

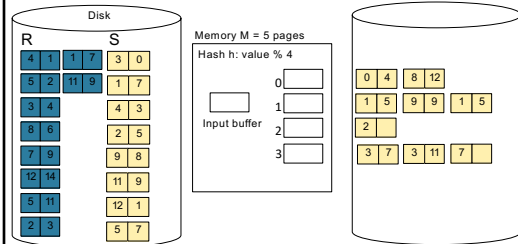


CSE 444 - Winter 2019

63

Partitioned Hash-Join Example

Step 1: Read relation S one page at a time and hash into the 4 buckets
At the end, we get relation S back on disk split into 4 buckets

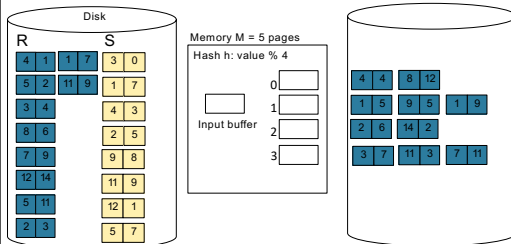


CSE 444 - Winter 2019

64

Partitioned Hash-Join Example

Step 2: Read relation R one page at a time and hash into same 4 buckets

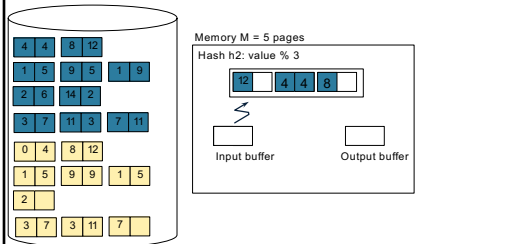


CSE 444 - Winter 2019

65

Partitioned Hash-Join Example

Step 3: Read one partition of R and create hash table in memory using a different hash function



CSE 444 - Winter 2019

66

Partitioned Hash-Join Example

Step 4: Scan matching partition of S and probe the hash table
 Step 5: Repeat for all the buckets

Total cost: $3B(R) + 3B(S)$

Memory $M = 5$ pages
 Hash h_2 : value % 3

Input buffer: [12, 4, 4, 8]
 Output buffer: []

CSE 444 - Winter 2019 67

Grace-Join

- Partition both relations using hash fn h : R tuples in partition i will only match S tuples in partition i .

Original Relation
 INPUT
 hash function h
 OUTPUT
 Partitions
 Disk B main memory buffers Disk

CSE 444 - Winter 2019 68

Grace-Join

- Partition both relations using hash fn h : R tuples in partition i will only match S tuples in partition i .
- Read in a partition of R, hash it using h_2 ($\neq h_1$). Scan matching partition of S, search for matches.

Original Relation
 INPUT
 hash function h
 OUTPUT
 Partitions
 Disk B main memory buffers Disk

Partitions of R & S
 hash fn h_2
 Hash table for partition S_i ($< M-1$ pages)
 INPUT
 hash h_2
 OUTPUT
 Disk B main memory buffers Disk

CSE 444 - Winter 2019 70

Grace Join

- Cost: $3B(R) + 3B(S)$
- Assumption: $\min(B(R), B(S)) \leq M^2$

CSE 444 - Winter 2019 70

Hybrid Hash Join Algorithm

- Partition S into k buckets
 t buckets S_1, \dots, S_t stay in memory
 $k-t$ buckets S_{t+1}, \dots, S_k to disk
- Partition R into k buckets
 - First t buckets join immediately with S
 - Rest $k-t$ buckets go to disk
- Finally, join $k-t$ pairs of buckets:
 $(R_{t+1}, S_{t+1}), (R_{t+2}, S_{t+2}), \dots, (R_k, S_k)$

CSE 444 - Winter 2019 71

Hybrid Hash Join Algorithm

Original Relation
 INPUT
 hash function h
 OUTPUT
 Partitions
 Disk B main memory buffers Disk

CSE 444 - Winter 2019 72

Hybrid Join Algorithm

- How to choose k and t?

CSE 444 - Winter 2019

73

Hybrid Join Algorithm

- How to choose k and t?
 - Choose k large but s.t. $k \leq M$

CSE 444 - Winter 2019

74

Hybrid Join Algorithm

- How to choose k and t?
 - Choose k large but s.t.

One block/bucket in memory
 $k \leq M$

CSE 444 - Winter 2019

75

Hybrid Join Algorithm

- How to choose k and t?
 - Choose k large but s.t.
 - Choose t/k large but s.t. $t/k * B(S) \leq M$

One block/bucket in memory
 $k \leq M$

CSE 444 - Winter 2019

76

Hybrid Join Algorithm

- How to choose k and t?
 - Choose k large but s.t.
 - Choose t/k large but s.t.

One block/bucket in memory
 $k \leq M$
First t buckets in memory
 $t/k * B(S) \leq M$

CSE 444 - Winter 2019

77

Hybrid Join Algorithm

- How to choose k and t?
 - Choose k large but s.t.
 - Choose t/k large but s.t.
 - Together:

One block/bucket in memory
 $k \leq M$
First t buckets in memory
 $t/k * B(S) \leq M$

$t/k * B(S) + k - t \leq M$

CSE 444 - Winter 2019

78

Hybrid Join Algorithm

- How to choose k and t?
 - Choose k large but s.t. $k \leq M$

One block/bucket in memory
 - Choose t/k large but s.t. $t/k * B(S) \leq M$

First t buckets in memory
 - Together: $t/k * B(S) + k - t \leq M$
- Assuming $t/k * B(S) \gg k - t$: $t/k = M/B(S)$

CSE 444 - Winter 2019 79

Hybrid Join Algorithm

- How to choose k and t?
 - Choose k large but s.t. $k \leq M$

One block/bucket in memory
 - Choose t/k large but s.t. $t/k * B(S) \leq M$

First t buckets in memory
 - Together: $t/k * B(S) + k - t \leq M$
- Assuming $t/k * B(S) \gg k - t$: $t/k = M/B(S)$

Total size of first t buckets

CSE 444 - Winter 2019 80

Hybrid Join Algorithm

- How to choose k and t?
 - Choose k large but s.t. $k \leq M$

One block/bucket in memory
 - Choose t/k large but s.t. $t/k * B(S) \leq M$

First t buckets in memory
 - Together: $t/k * B(S) + k - t \leq M$
- Assuming $t/k * B(S) \gg k - t$: $t/k = M/B(S)$

Total size of first t buckets

Number of remaining buckets

CSE 444 - Winter 2019 81

Hybrid Join Algorithm

Even better: adjust t dynamically

- Start with $t = k$: all buckets are in main memory
- Read blocks from S, insert tuples into buckets
- When out of memory:
 - Send one bucket to disk
 - $t := t - 1$
- Worst case:
 - All buckets are sent to disk ($t=0$)
 - Hybrid join becomes grace join

CSE 444 - Winter 2019 82

Hybrid Join Algorithm

Cost of Hybrid Join:

- Grace join: $3B(R) + 3B(S)$
- Hybrid join:
 - Saves 2 I/Os for t/k fraction of buckets
 - Saves $2t/k(B(R) + B(S))$ I/Os
 - Cost: $(3 - 2t/k)(B(R) + B(S)) = (3 - 2M/B(S))(B(R) + B(S))$

CSE 444 - Winter 2019 83

Hybrid Join Algorithm

- What is the advantage of the hybrid algorithm?

CSE 444 - Winter 2019 84

Hybrid Join Algorithm

- What is the advantage of the hybrid algorithm?

It degrades gracefully when S larger than M:

- When $B(S) \leq M$
 - Main memory hash-join has cost $B(R) + B(S)$
- When $B(S) > M$
 - Grace-join has cost $3B(R) + 3B(S)$
 - Hybrid join has cost $(3-2t/k)(B(R) + B(S))$

CSE 444 - Winter 2019

85

Summary of External Join Algorithms

- Block Nested Loop: $B(S) + B(R) \cdot B(S) / (M-1)$
- Index Join: $B(R) + T(R)B(S) / V(S, a)$
(unclustered)
- Partitioned Hash: $3B(R) + 3B(S)$;
– $\min(B(R), B(S)) \leq M^2$
- Merge Join: $3B(R) + 3B(S)$
– $B(R) + B(S) \leq M^2$

CSE 444 - Winter 2019

86

Summary of Query Execution

- For each logical query plan
 - There exist many physical query plans
 - Each plan has a different cost
 - Cost depends on the data
- Additionally, for each query
 - There exist several logical plans
- Next lecture: query optimization
 - How to compute the cost of a complete plan?
 - How to pick a good query plan for a query?

CSE 444 - Winter 2019

87