Assignment 2

Name: Fubang ZHAO
Date: 23/10/2017

Question 1: Join Algorithms

Relation R contains 10,000 tuples and has 10 tuples per page. Relation S contains 4,000 tuples and also has 10 tuples per page. Attribute b of relation S is the primary key for S. Both relations are stored as simple heap files. Neither relation has any indexes built on it. 41 buffer pages are available.

(1) $L = min(M, n) = 400$. In this question, we have $L/(B-1) = 400/(41-1) \leq B-2 = 39$, so we get the cost:

$$3 \times (1000 + 400) = 4200 I/Os \quad (1)$$

To remain the cost unchanged, we need to satisfy $L/(B-1) = B-2$, which means:

$$B > \sqrt{L} + 1 = 21 \quad (2)$$

So **the minimum number of buffer pages is 22.**

(2) The best way is to use Block Nested Loops. We set S as the outer, so the cost of Block Nested Loops is $400 + \lceil 400/(B-2) \rceil * 1000$. Let $\mathbf{B = 402}$, the cost is $\mathbf{1000 + 400 = 1400}$

(3) (a). Yes. Assuming that we partition R into k buckets, we need to satisfy: $k - 1 + \lceil \frac{400}{k} \rceil + 1 + 1 \leq B$, we get $\mathbf{k = 20}$. And it satisfies that $k < B - 2$. So we have enough memory to perform the hybrid hash join.

(b). For R, each bucket has $\lceil \frac{400}{k} \rceil = 20$ pages. For S, each bucket has $\lceil \frac{1000}{k} \rceil = 50$. So there are $20 + 50 = 70$ pages which have no need to write and reread. So the total I/O cost is:

$$3 \times (1000 + 400 - 70) + 70 = \mathbf{4060} I/Os \quad (3)$$

(4) The idea is quite similar to the last question. However, we don't need a buffer to store S to do the merge. so we need to just satisfy $k - 1 + \lceil \frac{400}{k} \rceil + 1 \leq B$, which means$k = 16$. There are 25 pages which belong to S_0 and have no need to write and reread. So the total cost is:

$$3 \times (400 - 25) + 25 = \mathbf{1150} I/Os \quad (4)$$

Question 2

(1) **Pass 0**: Find the best access path to $R, S_1, S_2, ..., S_{n-1}$ with n computations.

Pass 1: Find the best plans for all 2-relations sets. We need to spend $2(n-1)$ computations because we need to consider each relation to another. In the end, $n-1$ best plans are stored.

Pass 2: Find the best plan for all 3-relation sets with $(n-1)*(n-2)$ computations and C_{n-1}^2 plans are stored.

...

As shown in the table:

Pass Number	Nb of computations	Nb of plans stored
0	n	n
1	$2(n-1)$	C_{n-1}^1
2	$(n-2)*C_{n-1}^1$	C_{n-1}^2
...	...	...
k	$(n-k)C_{n-1}^{k-1} = kC_{n-1}^k$	C_{n-1}^k

So the total number of computations is:

$$n+2(n-1)+\sum_{k=2}^{n-1} kC_{n-1}^k = 2n-1+\sum_{k=1}^{n-1} kC_{n-1}^k = 2n-1+(n-1)2^{n-2} \quad (5)$$

So the complexity of dynamic programming for finding an optimal plan is: $\mathbf{O(n2^{n-2})}$

(2) As explained in last question, for Pass k, we need to store C_{n-1}^k. So the maximum number of plans is $C_{n-1}^{\lfloor \frac{n-1}{2} \rfloor}$.

Question 3

(1) As shown in the query plan:

```
--------------------------------------------------------------------------
---------------------------------------------------------
 Bitmap Heap Scan on authors  (cost=7918.93..22522.87 rows=321355 width=19)
 (actual time=52.804..162.421 rows=322803 loops=1)
   Recheck Cond: ((name)::text < 'David J. DeWitt'::text)
   Heap Blocks: exact=10587
   ->  Bitmap Index Scan on authors_name  (cost=0.00..7838.59 rows=321355 w
idth=0) (actual time=50.527..50.527 rows=322803 loops=1)
         Index Cond: ((name)::text < 'David J. DeWitt'::text)
 Planning time: 0.090 ms
 Execution time: 214.746 ms
(7 rows)
```

(a) PostgreSQL uses the Bitmap Heap Scan on authors and Bitmap Index Scan on authors for this query.

(b) The estimated number of rows is **321355** and the actual number of rows is **322803**.

(c) As shown in the picture below:

```
dblp=# select relname, reltuples from pg_class where relname='authors_name'
;
   relname    |  reltuples
--------------+-------------
 authors_name | 1.69134e+06
(1 row)
```

The estimation is based on the histogram and the estimated number of rows (tuples) before the bucket to which $'DavidJ.DeWitt'$ belongs. As shown in the picture above, the total number of tuples is $1.69134e + 06$.

According to the pg_stats, we can get $'DavidJ.DeWitt'$ belongs to the 20th bucket out of 100 buckets. So the estimated Nb of rows is $(1.69134e + 06) \times 19/100 \approx 321355$

(2) As shown in the query plan:

```
---------------------------------------------------------------------------
--------------------------------------------
 Index Scan using authors_name on authors  (cost=0.43..8.45 rows=1 width=19
) (actual time=0.025..0.026 rows=1 loops=1)
   Index Cond: ((name)::text = 'David J. DeWitt'::text)
 Planning time: 1.856 ms
 Execution time: 0.049 ms
(4 rows)
```

(a) PostgreSQL uses **Index Scan** for this query.

(b) 1. This query has an equality condition and 'name' differs a lot so that there are not many rows.

2. Compared to this query, the last question has a quite large number of rows(because the condition is '<', so it uses the bitmap index scan.

(3) (a) As shown in the query plan:

```
-----------------------------------------------------------------------------
----------------------------------------------------------------
 Nested Loop  (cost=8.89..163545.70 rows=5 width=159) (actual time=24.203..
4013.110 rows=186 loops=1)
   ->  Hash Join  (cost=8.46..163543.23 rows=5 width=27) (actual time=24.16
4..4007.708 rows=186 loops=1)
         Hash Cond: (pa.authid = a.id)
         ->  Seq Scan on paperauths pa  (cost=0.00..129792.71 rows=8997871
width=8) (actual time=0.017..1944.746 rows=8997871 loops=1)
         ->  Hash  (cost=8.45..8.45 rows=1 width=19) (actual time=0.016..0.
016 rows=1 loops=1)
               Buckets: 1024  Batches: 1  Memory Usage: 9kB
               ->  Index Scan using authors_name on authors a  (cost=0.43..
8.45 rows=1 width=19) (actual time=0.012..0.013 rows=1 loops=1)
                     Index Cond: ((name)::text = 'David J. DeWitt'::text)
   ->  Index Scan using papers_pkey on papers p  (cost=0.43..0.48 rows=1 wi
dth=132) (actual time=0.024..0.024 rows=1 loops=186)
         Index Cond: (id = pa.paperid)
 Planning time: 0.943 ms
 Execution time: 4013.249 ms
(12 rows)
```

(b) 1. $a.name =' DavidJ.DeWitt'$. Implemented by the Index Scan using $authors_{name}$ on authors a.

2. $pa.authid = a.id$. Implemented by the Hash Join which includes two operations: the Seq Scan on $pa.authid$ and Hash on $a.id$.

3. $pa.paperid = p.id$. Implemented by the Nested Loop Join which includes the Hash Join and Index Scan above.

(c) The estimated number of rows is **5** and teh actual number of wors is **186**

(d) Because that $pa.authid$ is a foreign key and $a.id$ is a primary key. So the estimated Nb of rows is:

$$N_{tuple_{pa}} \times \frac{1}{N_{tuple_a}} = \frac{8997871}{1.69134e+06} \approx 5 \tag{6}$$

(4) (a) As shown in the query plan:

```
----------------------------------------------------------------------------
----------------------------------------------------------------------------
 Hash Join  (cost=225078.04..590553.72 rows=1709598 width=159) (actual time
=2866.383..15699.360 rows=1633612 loops=1)
   Hash Cond: (pa.paperid = p.id)
   ->  Hash Join  (cost=28422.80..281232.51 rows=1709598 width=27) (actual
time=271.311..7764.730 rows=1633612 loops=1)
         Hash Cond: (pa.authid = a.id)
         ->  Seq Scan on paperauths pa  (cost=0.00..129792.71 rows=8997871
width=8) (actual time=0.031..2093.764 rows=8997871 loops=1)
         ->  Hash  (cost=22522.87..22522.87 rows=321355 width=19) (actual t
ime=270.835..270.835 rows=322803 loops=1)
               Buckets: 65536  Batches: 8  Memory Usage: 2561kB
               ->  Bitmap Heap Scan on authors a  (cost=7918.93..22522.87 r
ows=321355 width=19) (actual time=53.042..162.779 rows=322803 loops=1)
                     Recheck Cond: ((name)::text < 'David J. DeWitt'::text)
                     Heap Blocks: exact=10587
                     ->  Bitmap Index Scan on authors_name  (cost=0.00..783
8.59 rows=321355 width=0) (actual time=50.774..50.774 rows=322803 loops=1)
                           Index Cond: ((name)::text < 'David J. DeWitt'::t
ext)
   ->  Hash  (cost=95777.44..95777.44 rows=3149344 width=132) (actual time=
2592.526..2592.526 rows=3150923 loops=1)
         Buckets: 32768  Batches: 256  Memory Usage: 2228kB
         ->  Seq Scan on papers p  (cost=0.00..95777.44 rows=3149344 width=
132) (actual time=0.004..925.594 rows=3150923 loops=1)
 Planning time: 0.562 ms
 Execution time: 15953.088 ms
(17 rows)
```

(b) When we use Nested Loop Join, the cost is $M + M * N$. In the case of '=', M is very small, so it is efficient to use Nested Loop Join. However, in the case of '¡', M is large so we choose to use Hash Join which cost is $M + k * N$(k depends on the size of memory, we can make it equal to 1 if we have enough memory).

(c) For the Hash Join on $pa.authid = a.id$, the estimated number of rows is computed by: $Size(Paperauths) \times RF_{a.name})$, where:

$$RF_{a.name} = \frac{N_{estimated.rows}}{N_{tuples_a}} = \frac{321355}{1.69134e + 06} \tag{7}$$

$$Size(Paperauths) = 8997871 \tag{8}$$

So the estimated number of Rows is $8997871 * \frac{312355}{1691340} \approx 1709615$