CS186 Discussion Section

Buffer Pool Replacement Policies, Data Page Layouts

1. “Numbers everyone should know” by Jeff Dean (Google)
   - L1 cache reference 0.5 ns
   - L2 cache reference 7 ns
   - Main memory reference 100 ns
   - Read 1 MB sequentially from memory 250,000 ns
   - Disk seek 10,000,000 ns
   - Read 1 MB sequentially from disk 30,000,000 ns

2. Why use a buffer pool?

3. Why can’t DBMS rely on the memory/file management of the OS?

4. What is sequential flooding? What are the ways to mitigate the problem?

5. Buffer Replacement Policies - 4 Buffer pages

Access Pattern: A B C D A F A D G D G E D F

- LRU
  - 1 :
  - 2 :
  - 3 :
  - 4 :
  - hits:
  - misses:
- MRU
  - 1 :
  - 2 :
  - 3 :
  - 4 :
  - hits:
  - misses:
- CLOCK
  - 1 :
  - 2 :
  - 3 :
  - 4 :
  - hits:
  - misses:
- (Bonus) CLOCK (with a ref count max of 2)
  - 1 :
  - 2 :
  - 3 :
  - 4 :
  - hits:
  - misses:

6. Two alternative formats for variable lengths records

CS186 Discussion Section 1 (part 2)

Disk I/O

5. Consider two tables:
Students(sid, name, year, department), 200 pages, 1,000 tuples
Enrolled(sid, course, grade), 500 pages, 6,000 tuples
Query: for each student, list all his/her class grades:

```
SELECT name, course, grade
FROM Students, Enrolled
WHERE Students.sid = Enrolled.sid
```

Assume that we only have 1 disk, and that we do **not** have to write the resultant tuples back to disk.
Consider the join of Student and Enrolled in a nested loop (the naïve nested loops algorithm) with Student as the outer.
Also assume that we **don't** cache any pages in our buffer pool.

1. What is the total number of I/Os this join will require?

2. Of the total number of I/Os, how many are **sequential** I/Os? (Assume that the data for each relation is located in a continuous clump, but the two relations are located in different places.)

3. Of the total number of I/Os, how many are **random** I/Os?