CSCI4707 Lab03 Report

Replace Policy	Values10k		Values100k	
	Test1	Test2	Test1	Test2
FIFO	00:00:00.17308	00:00:00.23209	00:00:00.241469	00:00:00.28201
	00:00:00.178313	00:00:00.237079	00:00:00.216113	00:00:00.269014
	00:00:00.179175	00:00:00.236932	00:00:00.254106	00:00:00.254416
	00:00:00.172407	00:00:00.261892	00:00:00.213929	00:00:00.28384
Clock Sweep	00:00:00.180262	00:00:00.34845	00:00:00.330539	00:00:00.373271
	00:00:00.182693	00:00:00.34859	00:00:00.323551	00:00:00.412399
	00:00:00.176537	00:00:00.405331	00:00:00.323578	00:00:00.422329
	00:00:00.175072	00:00:00.378696	00:00:00.3217	00:00:00.40865

Note: Because the default buffer pool is too large, so executing our test data doesn't use replace policy. So I reset the buffer pool to 80, using the following command '*$HOME/lab03/bin/postgres –B 80 –D $HOME/lab03/data'.* And then we executed the given test case, we got the following results.

Analysis

From the above result, we can clearly find that, *FIFO* algorithm executed always faster than *clock sweep.* Firstly, we find that both test1 and test2 always try to find different data with different given value, which means replace will be called a lot of times. Because of this, we think the reason that *FIFO* runs faster than *clock sweep* is *FIFO* algorithm just return the first unpinned buffer. It is easy to find such a buffer. And *FIFO* algorithm just scans the whole buffer once. Comparing to *FIFO*, *clock sweep* attempt to find the unpinned and oldest buffer in the buffer pool. So in this algorithm, we set another variable to keep the value that how long the buffer has been used. This process may cost times. And *clock sweep* algorithm scans the whole buffer more than one times. So *clock sweep* algorithm run slowly than *FIFO*.