Exercise Session 7

Further Concurrency Control Techniques

7.1 Data Contention, Resource Contention and Thrashing

a) Explain the difference between **data contention** and **resource contention** and the way they could lead to data thrashing.

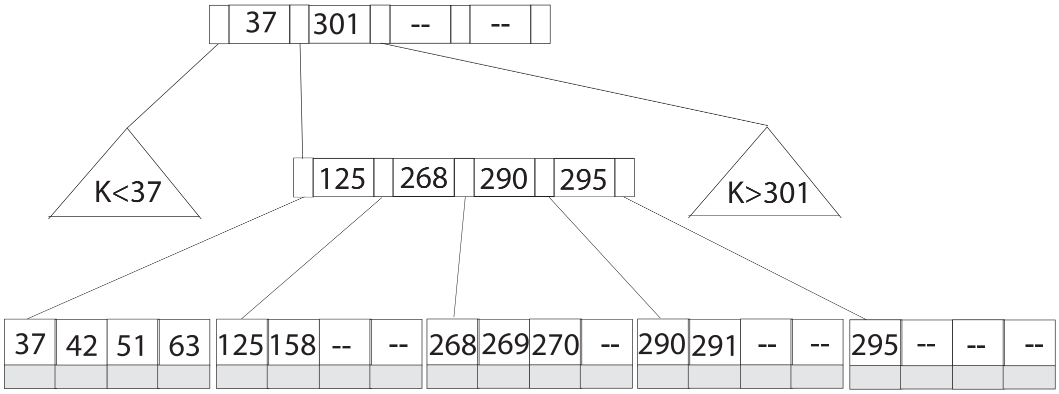
b) What could be done to reduce data contention and achieve throughput beyond the thrashing point?

7.2 Tree Locking (TL) Protocols

a) Explain briefly the main principles of TL protocols.

b) Explain how TL protocols can be used to synchronize access to B+-Trees.

c) Given is the following B+-Tree:



- Which nodes are safe (unsafe) for the insert/delete operation?

- Show how the operations Insert(271) and Insert(272) would be executed if the Bayer/Schkolnick algorithm is used.
- To what extent are exclusive locks unnecessarily set in the Bayer/Schkolnick algorithm? How can this be avoided?

7.3 Timestamp Ordering Protocols

a) Explain why the TO Rule guarantees serializability.

b) Describe a possible implementation of a TO-based Scheduler.

- What data structures are used?
- How are the Basic TO Rules enforced?

c) Given are the transactions T1 (timestamp=1), T5 (timestamp=5) and T10 (timestamp=10). T1, T2 and T3 send the following operations to the scheduler:

Point in Time	**T1**	**T2**	**T3**
1	BOT		
5		BOT	
10	w1[x]		BOT
11		r5[x]	
12			r10[x]
13		w5[x]	
14		c5	
15			w10[x]
16	c1		
17			c10

We assume that if an operation is sent to the Data Manager at point in time t, the scheduler receives confirmation from the Data Manager at point in time t+1 for read operations and t+2 for write operations. Parallel read operations are possible. Under these assumptions, describe in detail the sequence of actions performed by the scheduler. Show how the TO data structures are used.

Time	BTO Test	w-max-sched[x]	r-max-sched[x]	w-in-transit[x]	r-in-transit[x]	queue[x]
...						

If a transaction T sets o_max_scheduled[x] to TS(T) and is subsequently aborted can o_max_scheduled[x] be restored to its before image with respect to T?

d) Discuss the following issues in the context of the Basic TO Method:

- Deadlocks
- Livelocks, Starvation
- Strictness

7.4 Optimistic Concurrency Control

a) Explain the different paradigms of pessimistic and optimistic CC methods.

b) Describe the typical phases in the execution of transactions by an optimistic CC scheduler.

c) Describe a possible implementation of a locking-based optimistic CC protocol.

- What criteria are used for validation?
- Which of these criteria for backwards validation would be sufficient, if additional information about the relative order of transaction's phases is available? When should transaction Tn be backwards validated with respect to Tm, if:

  1.) end_write_phase(Tm) < start_read_phase(Tn)

  2.) end_write_phase(Tm) < start_write_phase(Tn)

  3.) end_read_phase(Tm) < end_read_phase(Tn)

- Which transactions must the scheduler backwards-validate against? When can the protocol data (read/write sets) for a transaction be deleted?