CHAPTER 20

Database-System Architectures

Practice Exercises

20.1 Is a multiuser system necessarily a parallel system? Why or why not?

Answer:
No. A single processor with only one core can run multiple processes to manage mutiple users. Most modern systems are parallel, however.

20.2 Atomic instructions such as compare-and-swap and test-and-set also execute a memory fence as part of the instruction on many architectures. Explain what is the motivation for executing the memory fence, from the viewpoint of data in shared memory that is protected by a mutex implemented by the atomic instruction. Also explain what a process should do before releasing a mutex.

Answer:
FILL IN MORE
The memory fence ensures that the process that gets the mutex will see all updates that happened before the instruction, as long as processes execute a fence before releasing the mutex. Thus, even if the data was updated on a different core, the process that acquires the mutex is guaranteed to see the latest value of the data.

20.3 Instead of storing shared structures in shared memory, an alternative architecture would be to store them in the local memory of a special process and access the shared data by interprocess communication with the process. What would be the drawback of such an architecture?

Answer:
The drawbacks would be that two interprocess messages would be required to acquire locks, one for the request and one to confirm grant. Interprocess communication is much more expensive than memory access, so the cost of locking would increase. The process storing the shared structures could also become a bottleneck.

The benefit of this alternative is that the lock table is protected better from erroneous updates since only one process can access it.

20.4 Explain the distinction between a *latch* and a *lock* as used for transactional concurrency control.

Answer:
Latches are short-duration locks that manage access to internal system data structures. Locks taken by transactions are taken on database data items and are often held for a substantial fraction of the duration of the transaction. Latch acquisition and release are not covered by the two-phase locking protocol.

20.5 Suppose a transaction is written in C with embedded SQL, and about 80 percent of the time is spent in the SQL code, with the remaining 20 percent spent in C code. How much speedup can one hope to attain if parallelism is used only for the SQL code? Explain.

Answer:
Since the part which cannot be parallelized takes 20% of the total running time, the best speedup we can hope for is 5. In Amdahl's law: $\frac{1}{(1-p)+(p/n)}$, $p = 4/5$ and n is arbitrarily large. So, $1 - p = 1/5$ and p/n aproaches zero.

20.6 Consider a pair of processes in a shared memory system such that process A updates a data structure, and then sets a flag to indicate that the update is completed. Process B monitors the flag, and starts processing the data structure only after it finds the flag is set.

Explain the problems that could arise in a memory architecture where writes may be reordered, and explain how the sfence and lfence instructions can be used to ensure the problem does not occur.

Answer:
The goal here is that the consumer process B should see the data structure state after all updates have been completed. But out of order writes to main memory can result in the consumer process seeing some but not all the updates to the data structure, even after the flag has been set.

To avoid this problem, the producer process A should issue an sfence after the updates, but before setting the flag. It can optionally issue an sfence after setting the flag, to push the update to memory with minimum delay. The consumer process B should correspondingly issue an lfence after the flag has been found to be set, before accessing the datastructure.

20.7 In a shared-memory architecture, why might the time to access a memory location vary depending on the memory location being accessed?

Answer:

In a NUMA architecture, a processor can access its own memory faster than it can access shared memory associated with another processor due to the time taken to transfer data between processors.

20.8 Most operating systems for parallel machines (i) allocate memory in a local memory area when a process requests memory, and (ii) avoid moving a process from one core to another. Why are these optimizations important with a NUMA architecture?

Answer:
In a NUMA architecture, a processor can access its own memory faster that it can access shared memory associated with another processor due to the time taken to transfer data between processors. Thus, if the data of a process resides in local memory, the process execution would be faster than if the memory is non-local.

Further, if a process moves from one core to another, it may lose the benefits of local allocation of memory, and be forced to carry out many memory accesses from other cores. To avoid this problem, most operating systems avoid moving a process from one core to another wherever possible.

20.9 Some database operations such as joins can see a significant difference in speed when data (e.g., one of the relations involved in a join) fits in memory as compared to the situation where the data do not fit in memory. Show how this fact can explain the phenomenon of **superlinear speedup**, where an application sees a speedup greater than the amount of resources allocated to it.

Answer:
We illustrate this by an example. Suppose we double the amount of main memory and that as a result, one of the relations now fits entirely in main memory. We can now use a nested-loop join with the inner-loop relation entirely in main memory and incur disk accesses for reading the input relations only one time. With the original amount of main memory, the best join strategy may have had to read a relation in from disk more than once.

20.10 What is the key distinction between homogeneous and federated distributed database systems?

Answer:
The key diference is the degree of cooperation among the systems and the degree of centralized control. Homogeneous systems share a global schema, run the same database-system software and actively cooperate on query processing. Federated systems may have distinct schemas and software, and may cooperate in only a limited manner.

20.11 Why might a client choose to subscribe only to the basic infrastructure-as-a-service model rather than to the services offered by other cloud service models?

Answer:
A client may wish to control its own applications and thus may not wish to subscribe to a software-as-a-service model; or the client might wish further to be able to choose and manage its own database system and thus not wish to subscribe to a platform-as-a-service model.

20.12 Why do cloud-computing services support traditional database systems best by using a virtual machine, instead of running directly on the service provider's actual machine, assuming that data is on external storage?

Answer:
By using a virtual machine, if a physical machine fails, virtual machines running on that physical machine can be restarted quickly on one or more other physical machines, improving availability. (Assuming of course that data remains accessible, either by storing multiple copies of data, or by storing data in an highly available external storage system.)