Discuss process synchronization, and consider problems such as Bounded-Buffer, Readers–Writers, and Dining-Philosophers Problems.

Hardware instructions that support synchronization:

* Memory barriers
  + Memory model: strongly or weekly ordered
* Hardware Instructions
* Atomic Variables

Cooperating processes are those sharing a memory address or accessing the same data; they are, therefore, affected by the execution of other processes. Cooperative processes can be susceptible to data inconsistency because several processes may want to access, read or modify the same data. Disorderly access to this data may result in race conditions. The following problems are common synchronization problems, that are often used to test synchronization techniques.

The bounded buffer, or producer-consumer problem, occurs when a consumer may attempt to consume data not yet produced. A typical producer consumer relationship is when a server (producer), produces HTML content to be requested by a client (consumer) to display a website. A typical solution to the producer consumer problem is a buffer space that is filled by the producer and emptied by the consumer. To avoid data inconsistency in a fixed-size (bounded) buffer, a consumer waits for the producer to produce if the buffer is empty and a producer waits for a consumer to consume if the buffer is full. The bounded buffer poses another problem, however. As the size is limited, a counter can be incremented by the producer and decremented by the consumer to keep track of the size. Concurrently manipulating the counter, may result in a race condition (inconsistent size).

Another problem is the read-write problem, which occurs when a writer and either another writer or reader attempt to access the same data. Variants of the reader - writer problem include the first readers – writers and the second readers – writers problem. In the first problem, readers don’t have to wait for other readers to finish, even if a writer is waiting. In the second problem, new readers may not read if a writer is waiting. The first problem may result in writer starvation, while the second may result in reader starvation.

The dining philosophers problem is a simile to multiple operators trying to access multiple items simultaneously. To elaborate, a group of philosophers are seated around a circular table with one chopstick in between every philosopher. Philosophers in this analogy only have 3 modes, thinking, hungry and eating. As a philosopher thinks, she is non-reactive, however, when she is hungry, she attempts to pick up a chopstick from either side to eat. She is only able to do so if both are free. If they are both free, she picks them up, eats, puts them back down and starts thinking again. This is to demonstrate the importance of allocating enough resources for enough processes to avoid a deadlock or starvation.

Synchronization techniques:

Windows:

* In the kernel: spinlock for short programs
* Outside Kernel: Dispatcher objects

Synchronization is the process of slicing time to allow each processes its own execution time.