Jumping ahead

- Now we're on to chapters 11 and 12.
- Chapter 11: Time and Global States
- Chapter 12: Coordination and Agreement

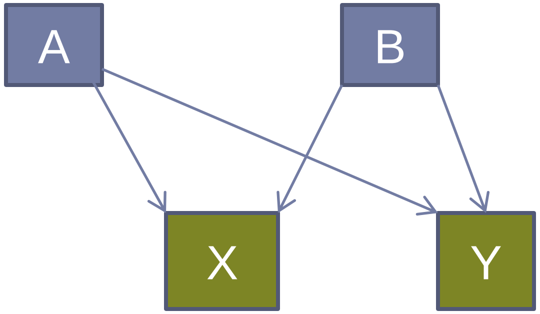
Time 

- Time plays an important role in distributed systems.
  - Maintaining consistency of distributed data.
    - E.g.: Timestamps to serialize transactions.
  - Authentication protocols.
  - Elimination of duplicate updates.

- Typically we are concerned with the order of operations that occur on different nodes relative to each other.
  - How do we do this if there is no shared, universal clock that all nodes can refer to?
  - Even worse, how do we synchronize with each other if messaging times and clock behaviors are variable?

Relativity

- The relative ordering of events is intimately tied to the observer and what is being observed.
  - For example, we have two event producers A and B, and two event consumers, X and Y and a fixed speed of light.
    - Assume as an outside-of-the-universe observer, we can determine for certain that A and B produce their events at the same time.
    - To X, equidistant from both, they appear simultaneous. To Y, they see B happen before A.



Relativity

- So, because the speed at which information can propagate has a fixed, finite upper bound, two different observers can come to two different conclusions about the same sequence of events.

- How does this impact distributed systems?
  - Ability to know order of two events on different nodes.
  - Ability to know if two distributed events occurred simultaneously.
    - This is an issue with fast-paced games.

Clocks

- Let t be “real time”.
- A clock is a physical device attached to a computer that oscillates at a certain frequency counting these oscillations.
- Let $H_i(t)$ be the clock value on node i at time t.
  - This could be a cycle counter, tick counter.
- The software clock translates this into an approximation of real time:

$$C_i(t) = \alpha H_i(t) + \beta$$

  - Scale and offset.
- The clock resolution is the amount of real time that elapses between two adjacent clock ticks.
  - Successive events are guaranteed to get a unique time if they occur at times further apart than a single clock tick.

Clocks: Drift and Skew

- Skew: The instantaneous difference between two clocks.
- Drift: Two clocks that count at different rates will have a growing skew. They drift relative to each other.
  - Imprecision in clock manufacturing can cause this.
  - Changes in the environment can affect oscillation times (e.g.: temperature fluctuations).
  - Relativistic effects. Time dilation occurs in the presence of a gravitational field.
    - GPS requires this sort of correction because the GPS satellites are in a different part of the Earth's gravitational field than receivers.
- Quartz crystals are common oscillators in computers and other devices. They have a drift rate of 10^-6 seconds per day. That's one second for every 1,000,000 seconds, or 11.6 days.

Synchronization

- There are two types of time synchronization.
  - **External**: Synchronization with an external authority that is connected directly to an external time source, such as an atomic clock or radio receiver.
  - **Internal**: Synchronization of a set of clocks with each other to some known accuracy. Allows us to observe event intervals amongst the set of participants, but not outsiders or real time.

External synchronization

- Given some synchronization bound D>0 and a source S of UTC.
- Condition: For all participants i = 1…n and all real times t over some interval:

$$|S(t) - C_i(t)| < D$$

- If this condition holds, then we can say that the clocks $C_i(t)$ are accurate to within the bound D.

Internal synchronization

- Given some synchronization bound D>0 and a set of N interacting nodes.
- Condition: For all nodes i and j:

$$\left| C_i(t) - C_j(t) \right| < D$$

  - for all real times t over some interval.
- If this condition holds, then we can say that the clocks C_i agree within a bound D.

Internal vs. external synchronization

- Internal synchronization does not imply external synchronization because of inherent drift from the external source.
- On the other hand, if all N nodes are synchronized with bounds D to the source S, then all N nodes are internally synchronized with bound 2D.
  - Why 2D? Each node is within D of the source S. A node can be +D or –D. So, in the extreme case one node is +D and the other is –D. That is 2D apart.

Drift rates and intervals

- Given a bounded drift rate ρ, we can say that the error in measuring the interval between real times t and t’ (t’ > t) is bounded by:

$$(1-\rho)(t'-t) \leq H(t') - H(t) \leq (1+\rho)(t'-t)$$

- This forbids large jumps in the clock under normal operating conditions (where the drift rate bound holds).

Monotonicity

- t‘ > t implies C(t’) > C(t)
  - I would make this weaker, where t’ > t implies C(t’) >= C(t) if t’-t is less than the clock resolution.

- Monotonicity simply states that given two points in real time, the order relationship between them will be preserved by the software clock.
  - Interestingly, monotonicity can be violated when a clock ‘flips over’ as the finite representation of the clock overflows.
  - Fortunately, this is rare, and we can detect it.