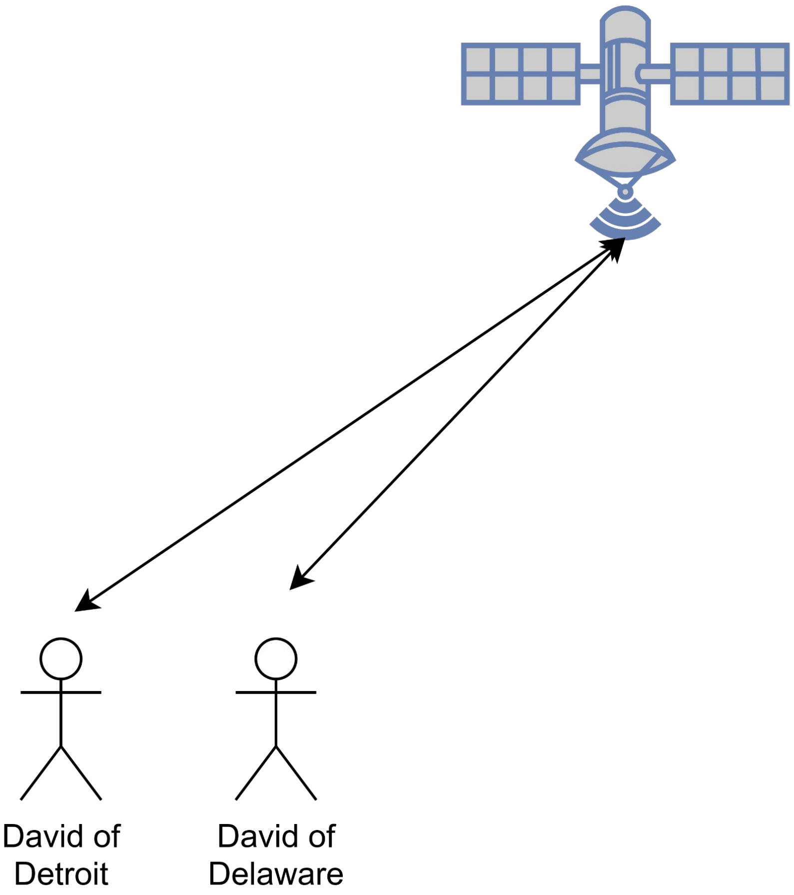
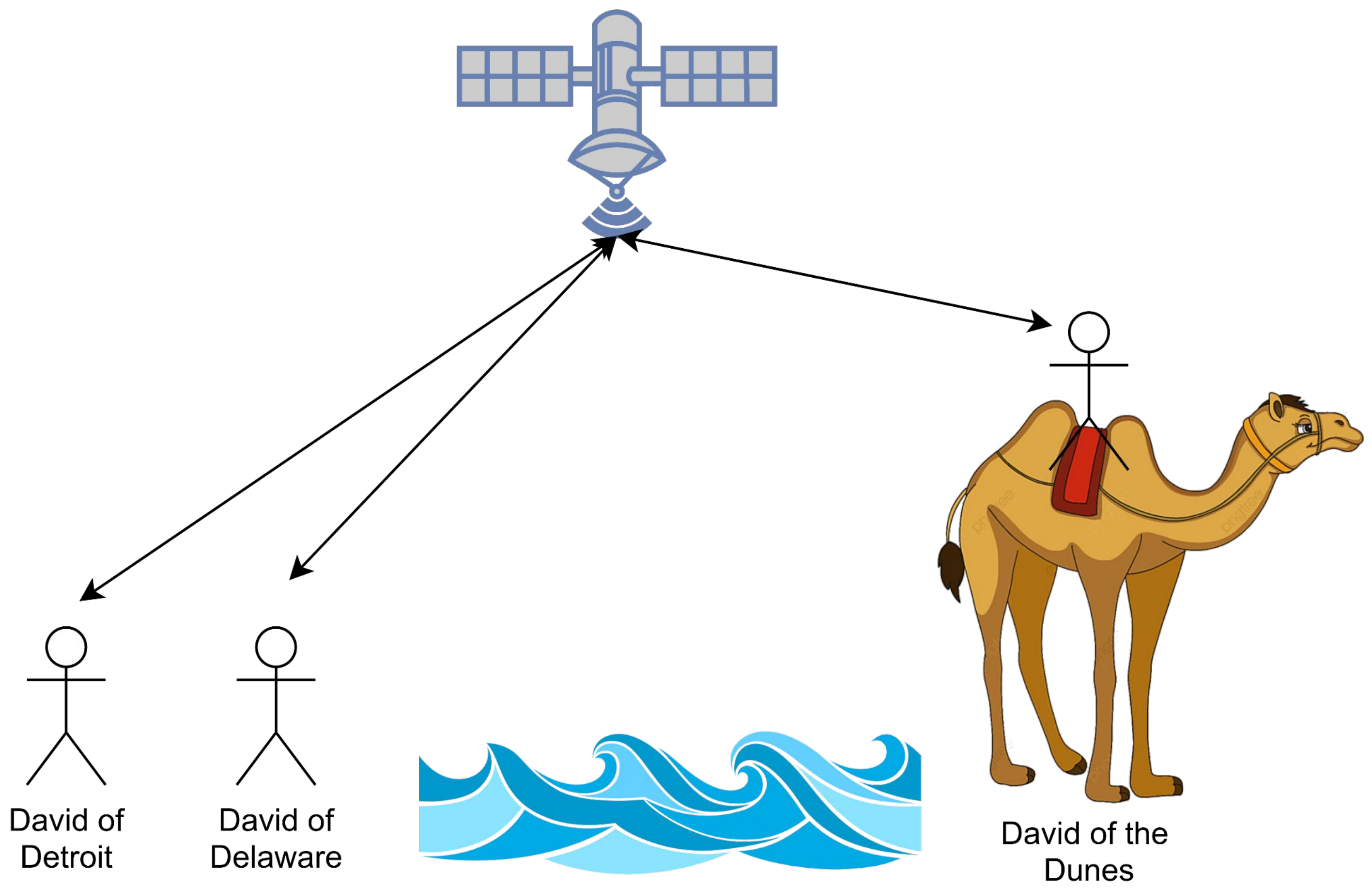
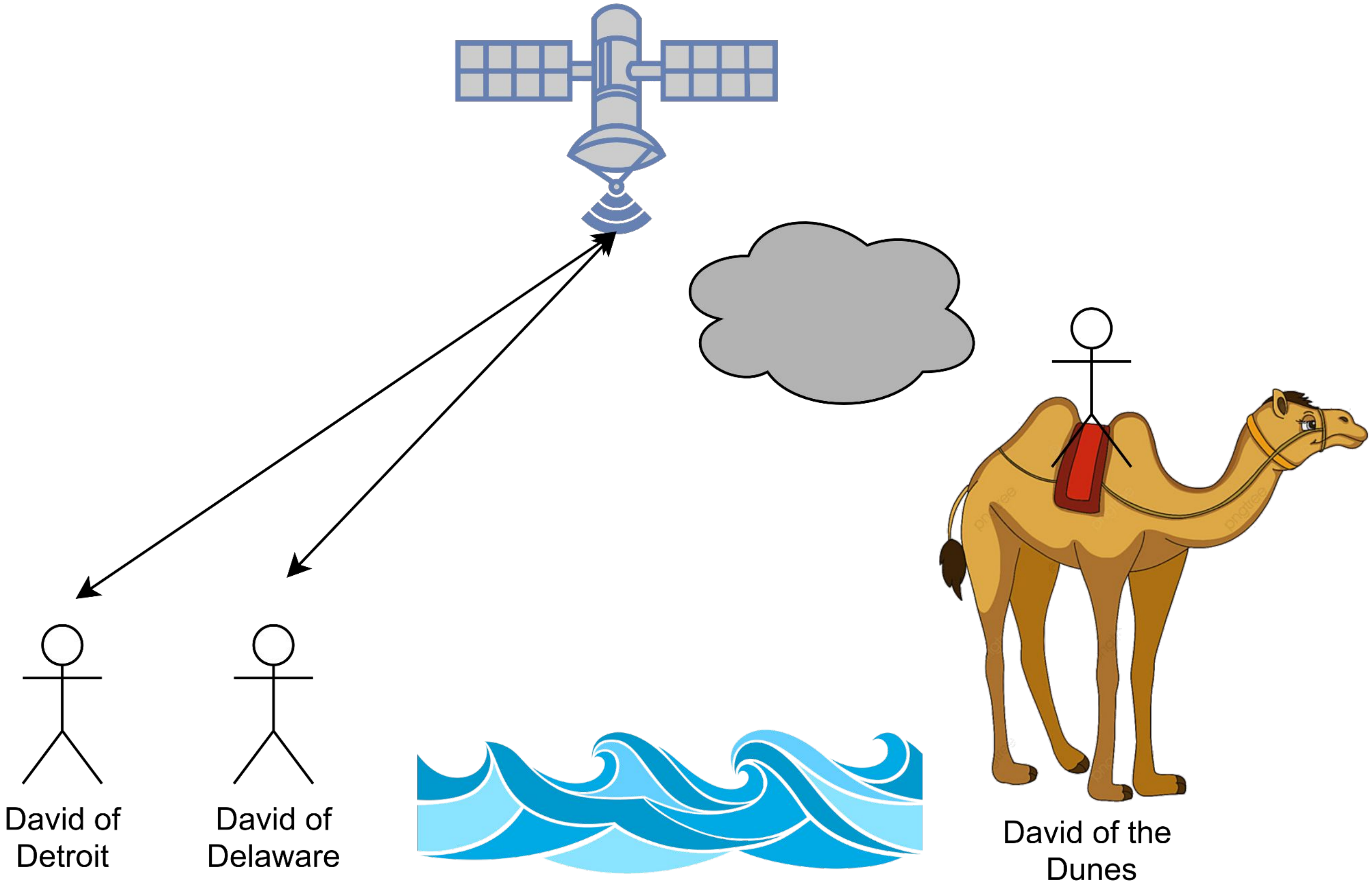
Spanner: Google vs Math

Yash Lala


David of
Detroit
David of
Delaware


David of Detroit
David of Delaware
David of the Dunes

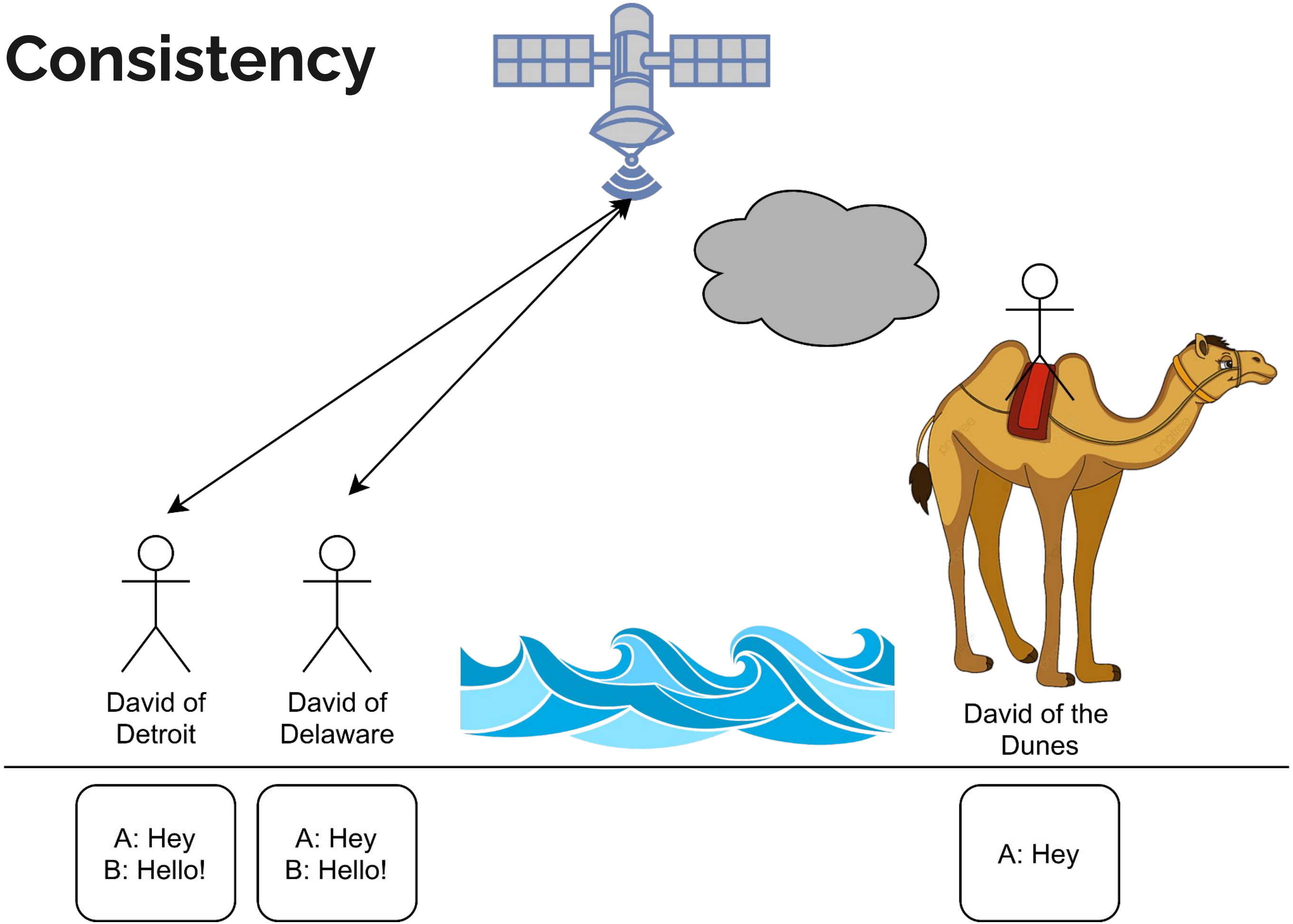

David of Detroit
David of Delaware
David of the Dunes

The CAP Theorem

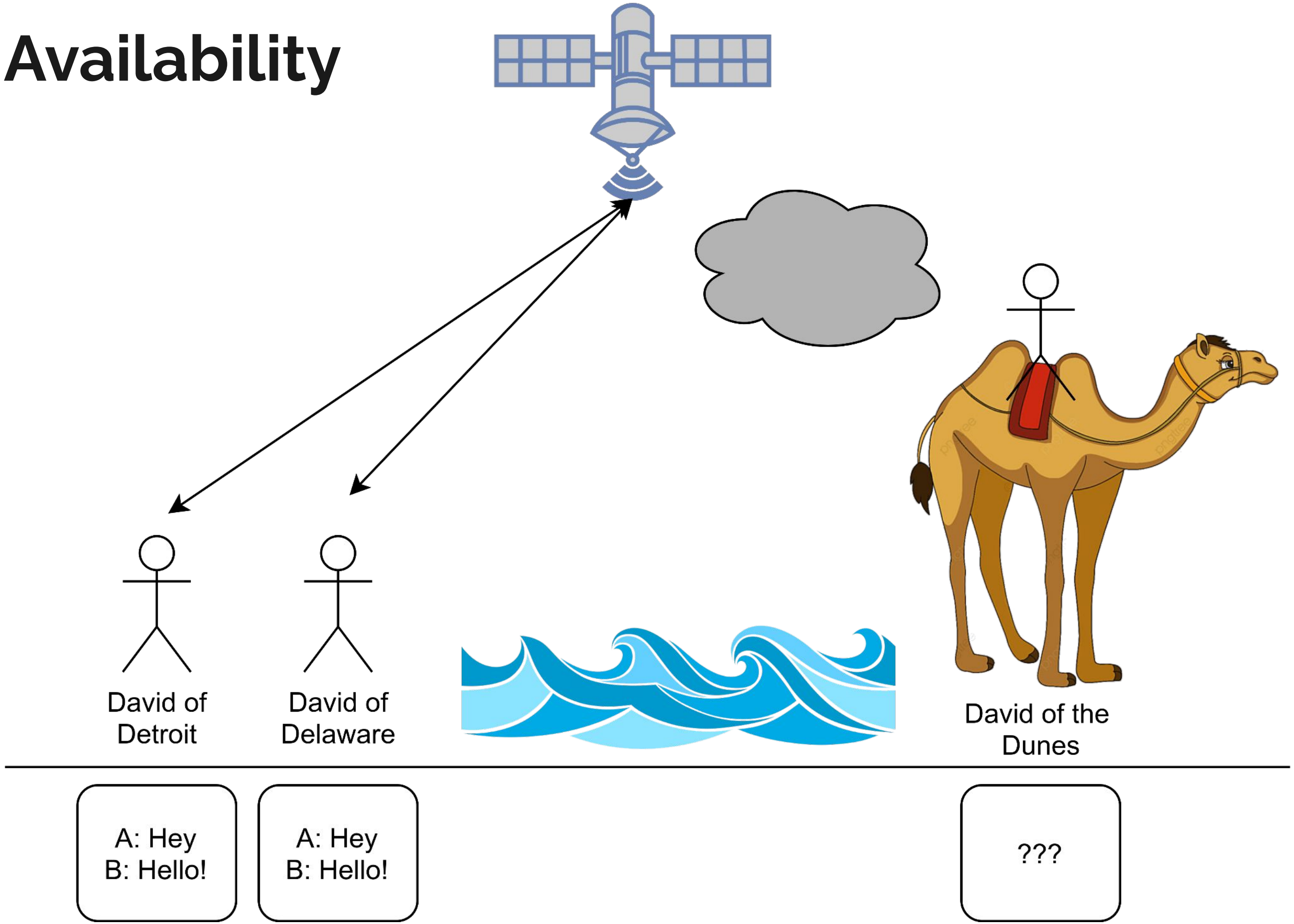
No CAP

- Consistency
- Availability
- Partition Tolerance

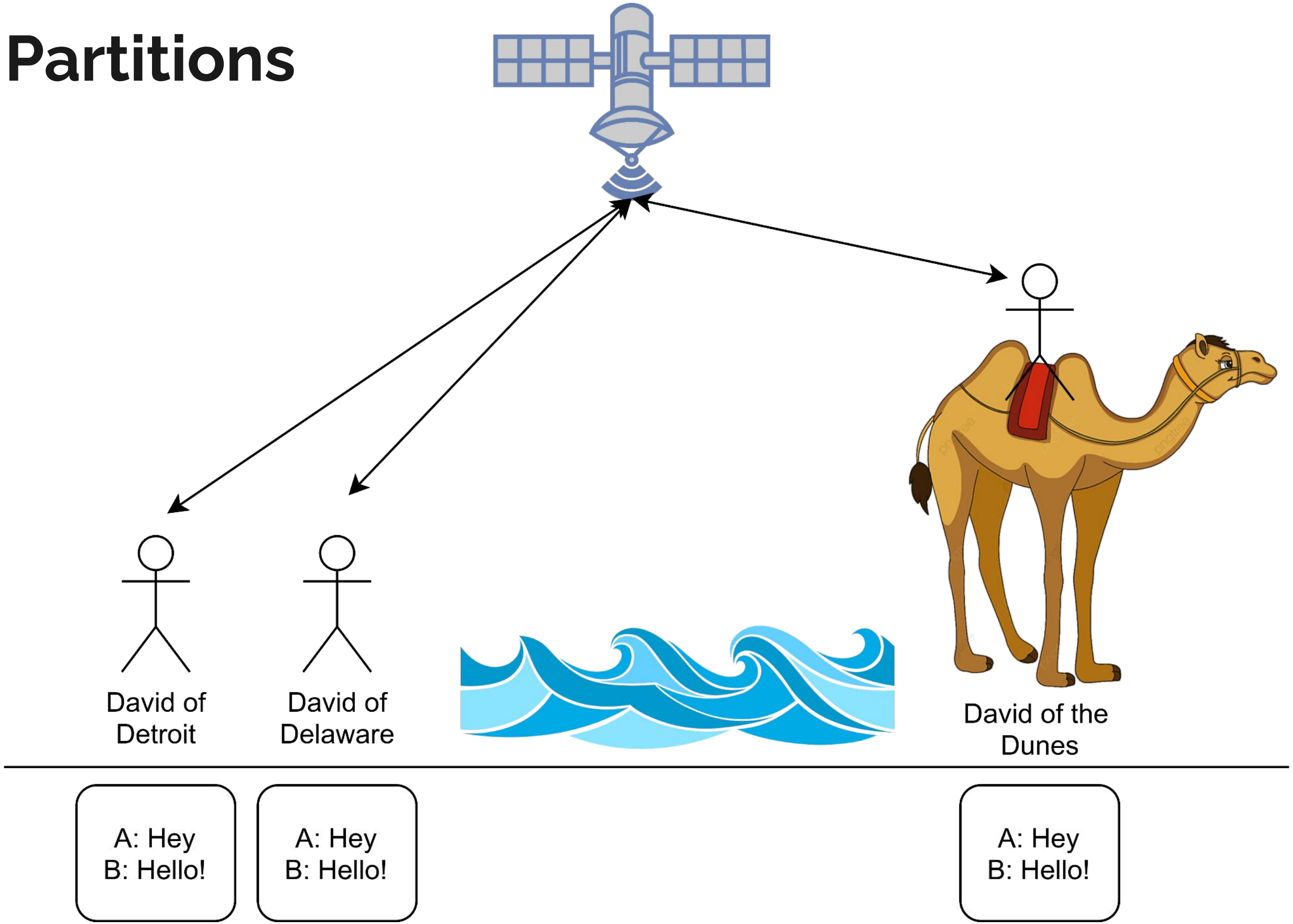
No Consistency



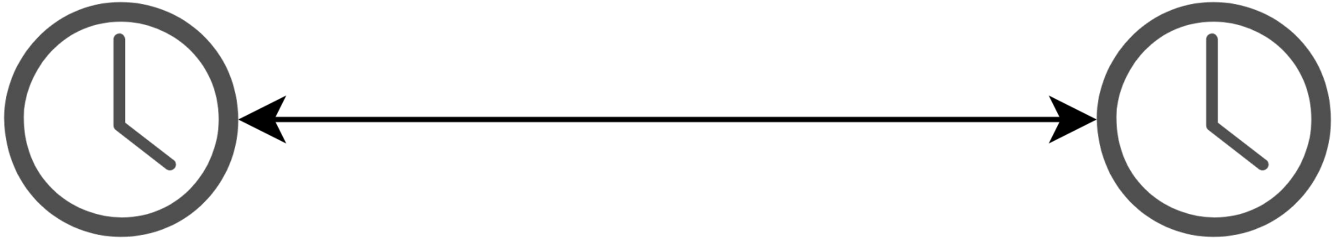
No Availability



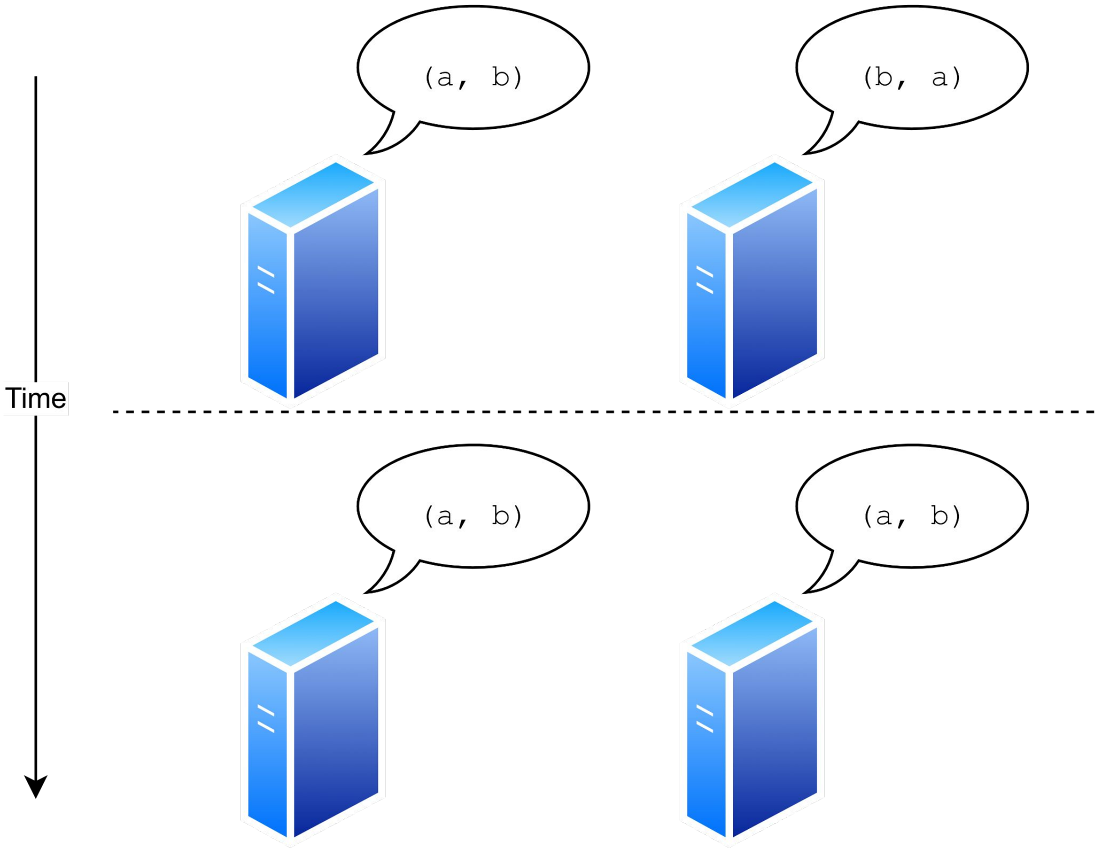
No Partitions



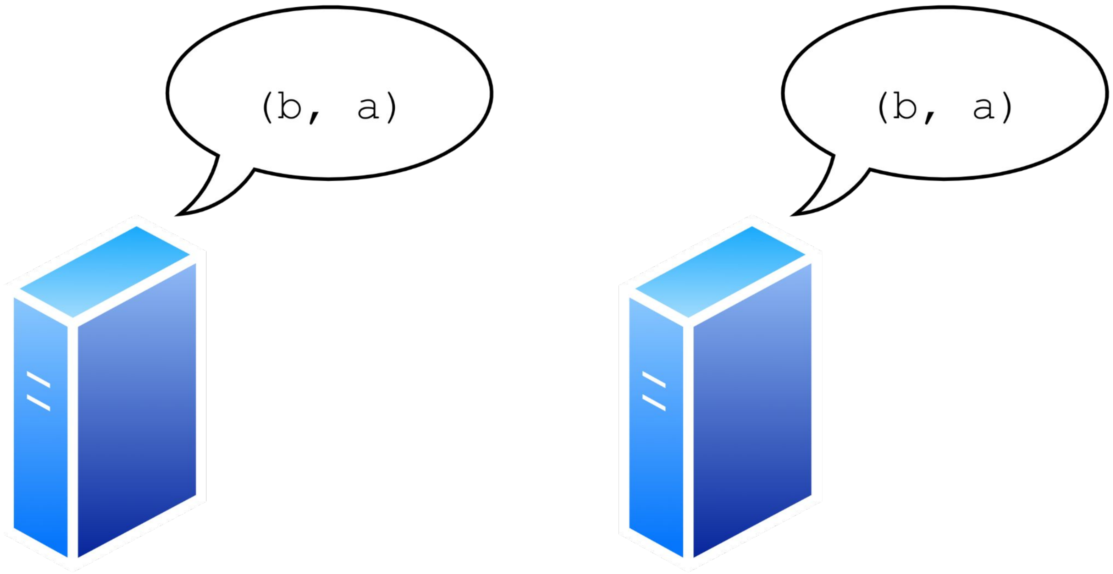
Clock Drift



Consistency Models



Eventual Consistency



Serializability

Enter Google

Strict Serializability

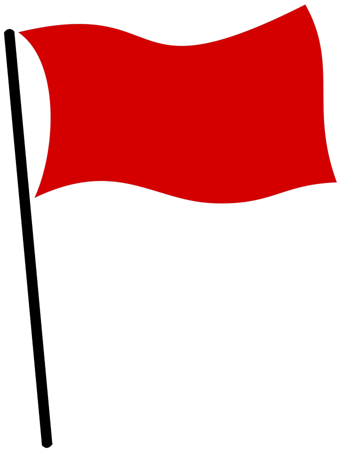
$$t_{abs}(e_1^{commit}) < t_{abs}(e_2^{start})$$

$$\Rightarrow \quad s_1 < s_2$$

Strict Serializability: The Invariant

Google Spanner

1. Externally Consistent
2. Available everywhere
3. Tolerates data center failures



$$t_{abs}(e_1^{commit}) < t_{abs}(e_2^{start})$$

$$\Rightarrow \quad s_1 < s_2$$

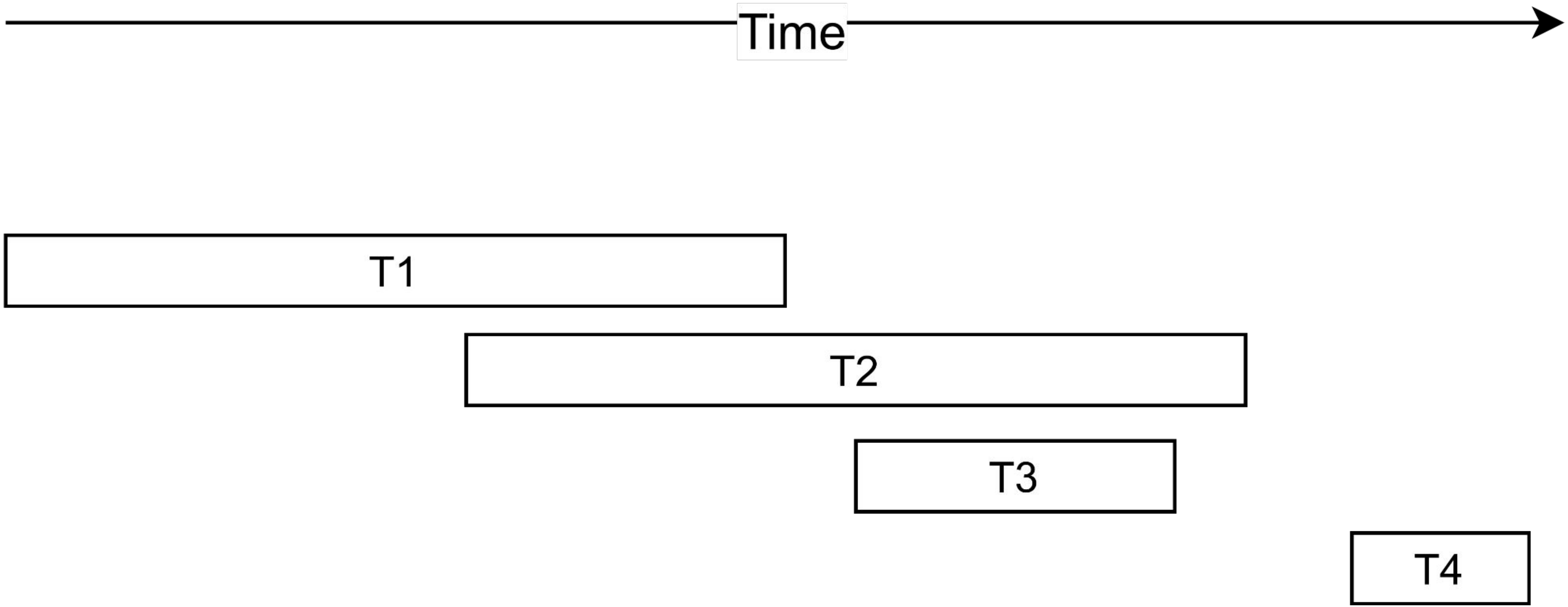
Strict Serializability: The Invariant

Hardware Improvements > CAP

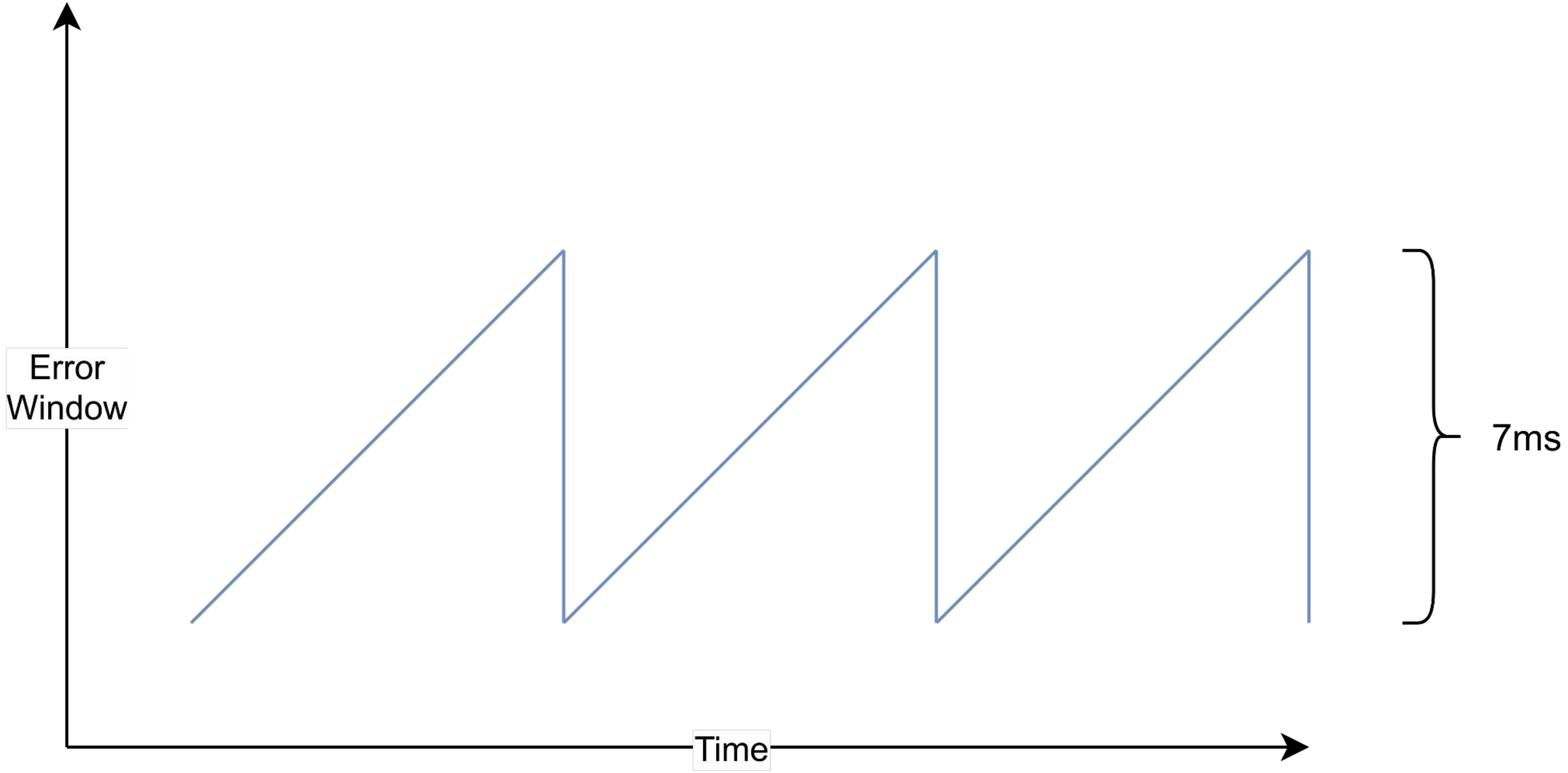
Bounding Absolute Time

TrueTime

Method	Returns
$TT.now()$	$TTinterval$: $[earliest, latest]$
$TT.after(t)$	true if t has definitely passed
$TT.before(t)$	true if t has definitely not arrived

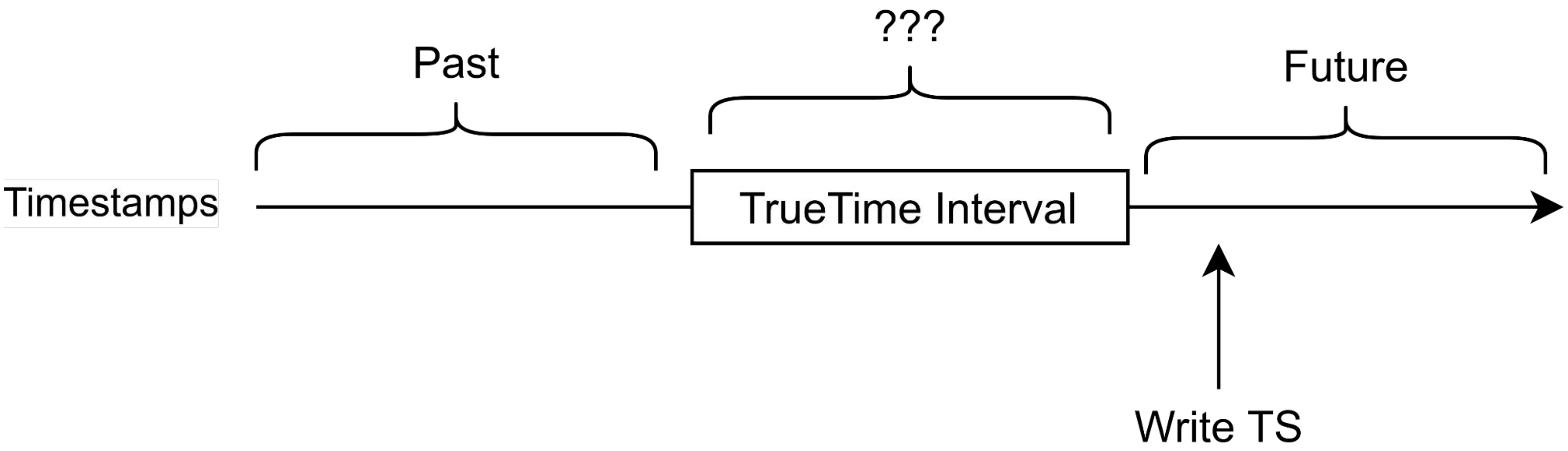


Marzullo’s Algorithm

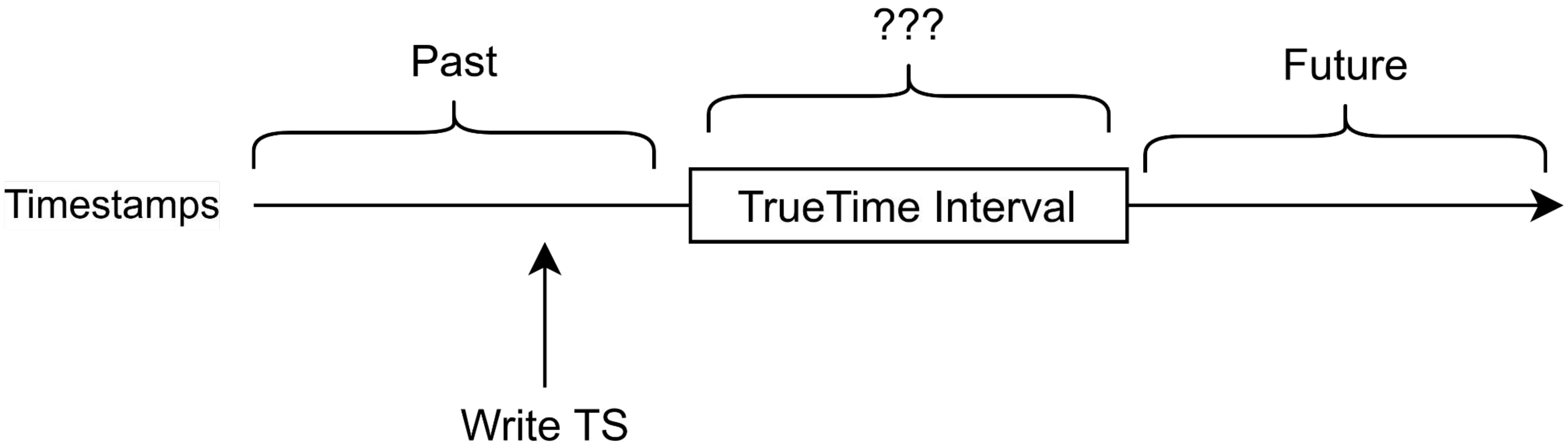


Clock Error

Making our Bounds Useful



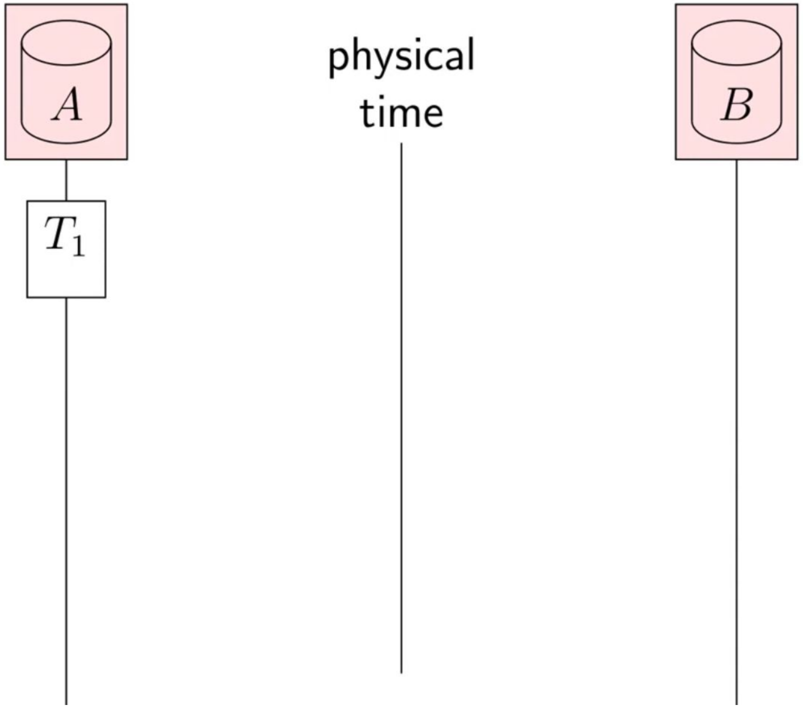
Assigning a Write Timestamp



Strategic Waiting

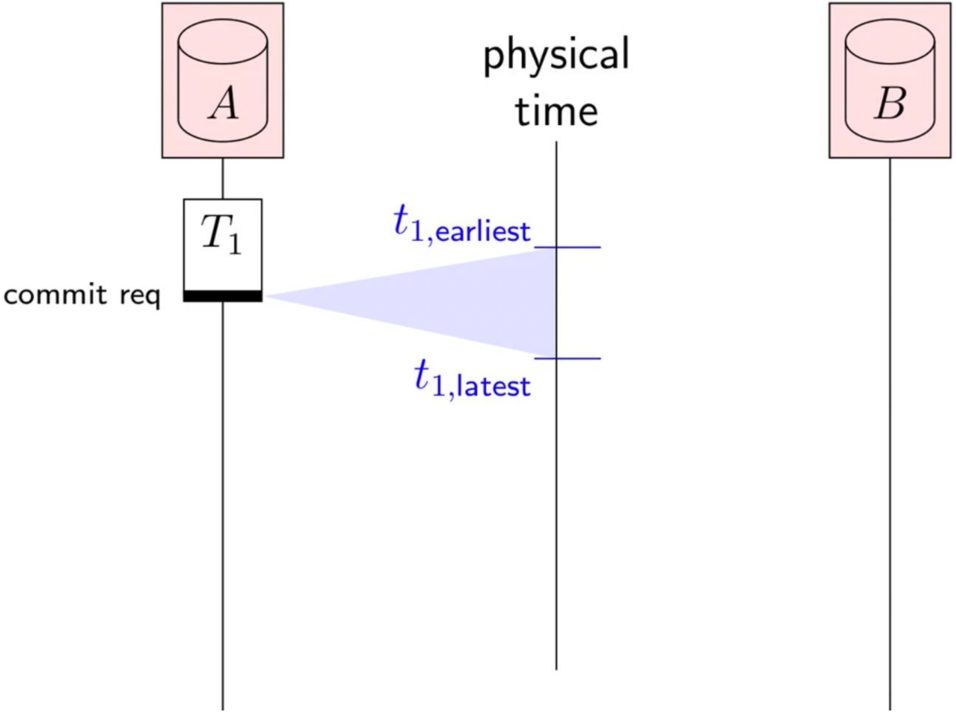
Example Transaction

TrueTime: explicit physical clock uncertainty

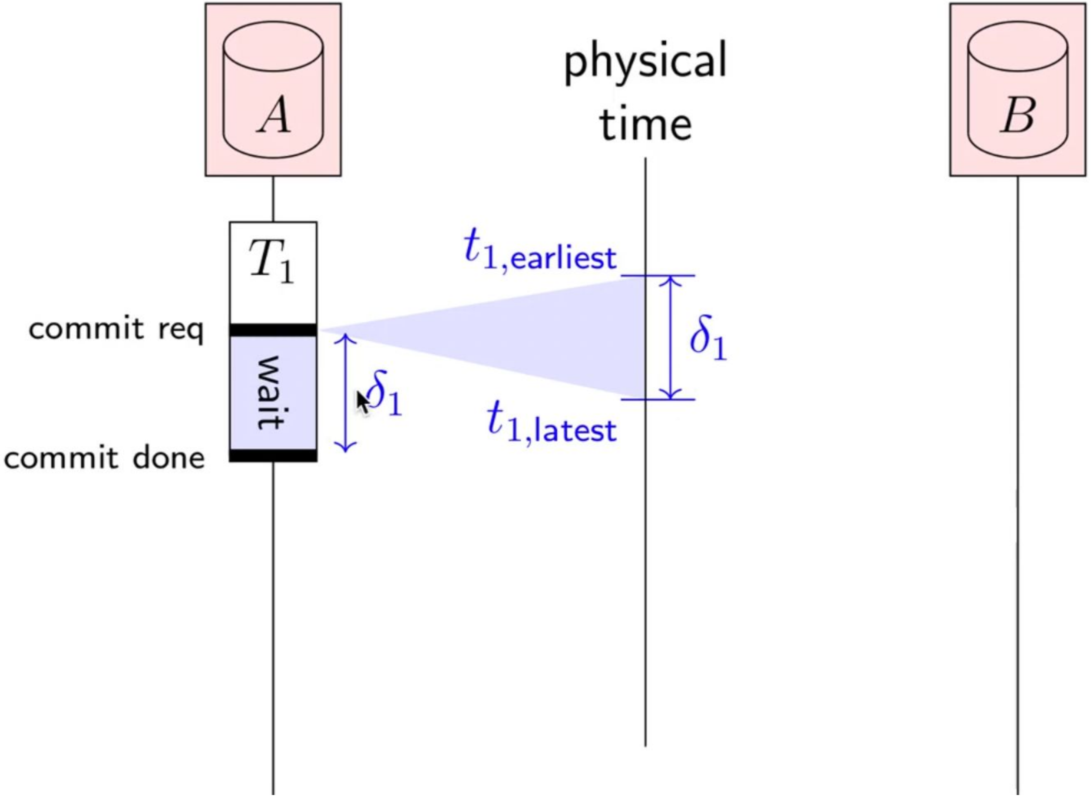


TrueTime: explicit physical clock uncertainty

Spanner's TrueTime clock returns $[t_{\text{earliest}}, t_{\text{latest}}]$.
True physical timestamp must lie within that range.

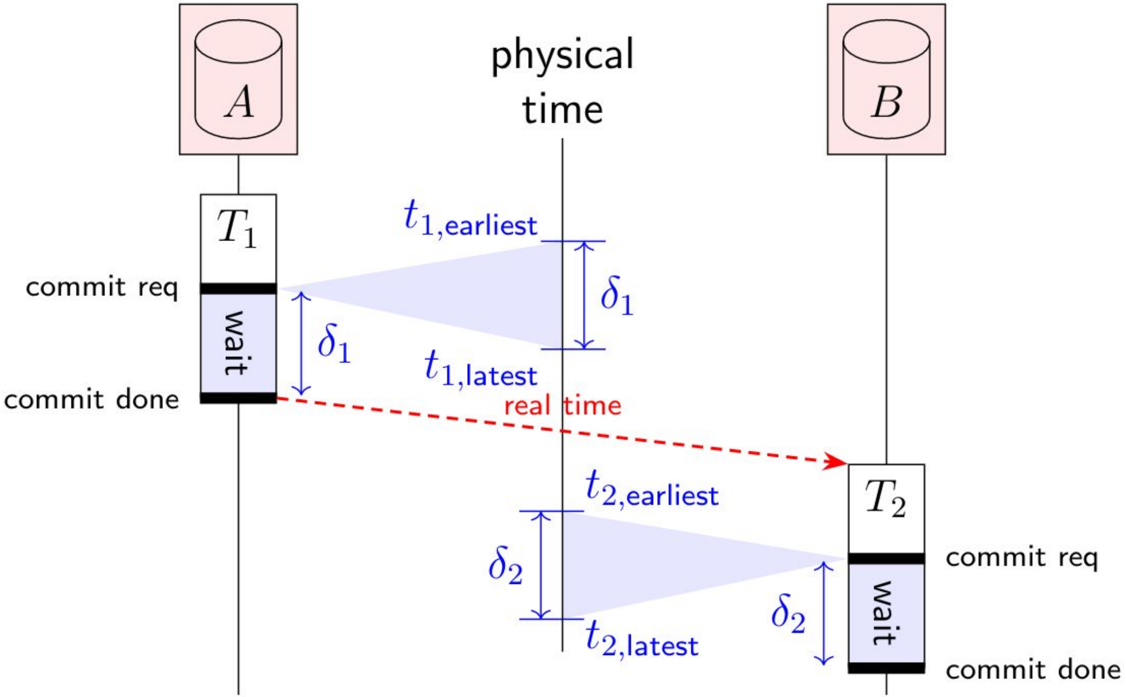


Spanner's TrueTime clock returns $[t_{\text{earliest}}, t_{\text{latest}}]$.
True physical timestamp must lie within that range.
On commit, wait for uncertainty $\delta_i = t_{i,\text{latest}} - t_{i,\text{earliest}}$.



TrueTime: explicit physical clock uncertainty

Spanner's TrueTime clock returns $[t_{\text{earliest}}, t_{\text{latest}}]$.
True physical timestamp must lie within that range.
On commit, wait for uncertainty $\delta_i = t_{i,\text{latest}} - t_{i,\text{earliest}}$.



Problem Solved

Takeaways

- $$$ > Theorems
- Understand your Systems